

MINISTERUL EDUCAȚIEI



**UNIVERSITATEA TEHNICĂ**

DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

# REȚEA NEURONALĂ DE DETECTARE A PROCENTULUI DE GRĂSIME CORPORALĂ DIN IMAGINI

PROIECT DE DIPLOMĂ

Autor: **Andrei GERMAN**

Conducător științific: **Prof. dr. ing. Dan Ioan GOȚA**

**2024**

MINISTERUL EDUCAȚIEI



---

**UNIVERSITATEA TEHNICĂ**

DIN CLUJ-NAPOCA

**FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE**



**UNIVERSITATEA TEHNICĂ**

DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

Vizat,

DECAN

**Prof. dr. ing. Vlad Muresan**

DIRECTOR DEPARTAMENT AUTOMATICĂ

**Prof. dr. ing. Honoriu VĂLEAN**

Autor: **Andrei GERMAN**

Rețea neuronală de detectare a procentului de grăsime corporală din  
imagini

1. **Enunțul temei:** *Crearea unei rețele neuronale capabile să calculeze și să prezică procentul de grăsime corporală pe baza imaginilor*
2. **Conținutul proiectului:** *Pagina de prezentare, Declarație privind autenticitatea proiectului, Sinteza proiectului, Cuprins, Introducere, Studiu Bibliografic, Analiză, Proiectare și Implementare, Concluzii, Bibliografie.*
3. **Locul documentării:** *Universitatea Tehnică din Cluj-Napoca*
4. **Consultanți:**
5. **Data emiterii temei:** 19.12.2023
6. **Data predării:** 02.09.2024

Semnătura autorului

Semnătura conducătorului științific

**UNIVERSITATEA TEHNICĂ**

DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

**Declarație pe proprie răspundere privind  
autenticitatea proiectului de diplomă**

Subsemnatul(a) **Andrei GERMAN**,  
legitimat(ă) cu CI seria ZS nr. 154327, CNP 5020725260051,  
autorul lucrării:

Rețea neuronală de detectare a procentului de grăsime corporală din imagini

elaborată în vederea susținerii examenului de finalizare a studiilor de licență la **Facultatea de Automatică și Calculatoare**, specializarea Choose an item., din cadrul Universității Tehnice din Cluj-Napoca, sesiunea Choose an item. a anului universitar 2023-2024, declar pe proprie răspundere, că această lucrare este rezultatul propriei activități intelectuale, pe baza cercetărilor mele și pe baza informațiilor obținute din surse care au fost citate, în textul lucrării, și în bibliografie.

Declar, că această lucrare nu conține porțiuni plagiate, iar sursele bibliografice au fost folosite cu respectarea legislației române și a convențiilor internaționale privind drepturile de autor.

Declar, de asemenea, că această lucrare nu a mai fost prezentată în fața unei alte comisii de examen de licență.

În cazul constatării ulterioare a unor declarații false, voi suporta sancțiunile administrative, respectiv, *anularea examenului de licență*.

Data

02.09.2024

Prenume NUME

(semnătura)



## SINTEZA

proiectului de diplomă cu titlul:

Rețea neuronală de detectare a procentului de grăsime corporală din  
imagini

Autor: **Andrei GERMAN**

Conducător științific: **Prof. dr. ing. Dan Ioan Goța**

1. Cerințele temei: Crearea, antrenarea și validarea unei rețele neuronale capabile să calculeze și să prezică procentul de grăsime corporală pe baza imaginilor
2. Soluții alese: Crearea unui model în Python, care să fie antrenat pe un set de date bazate pe metode de măsurare a procentului de grăsime corporală deja existente
3. Rezultate obținute: Două modele antrenate, cu acuratețe de antrenare de 100%, respectiv 97%, și acuratețe de validare de 39%, respectiv 57%
4. Testări și verificări: Modelele au fost testate pe date noi care sunt menite să relice condiții reale
5. Contribuții personale: Printre contribuțiile personale se numără crearea setului de date din surse multiple, deoarece nu există niciun alt set de imagini suficient de complex pentru antrenarea unui model., precum și crearea celor două modele folosite în antrenarea rețelei
6. Surse de documentare: Toate studiile citate și utilizate în crearea acestei lucrări sunt citate în bibliografie



**UNIVERSITATEA TEHNICĂ**

DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

Semnătura autorului

Semnătura conducătorului științific

# Cuprins

<b>1</b>	<b>INTRODUCERE.....</b>	<b>2</b>
1.1	CONTEXT GENERAL .....	2
1.2	OBIECTIVE.....	2
1.3	SPECIFICAȚII .....	5
<b>2</b>	<b>STUDIU BIBLIOGRAFIC.....</b>	<b>7</b>
<b>3</b>	<b>ANALIZĂ, PROIECTARE, IMPLEMENTARE.....</b>	<b>9</b>
<b>4</b>	<b>CONCLUZII.....</b>	<b>40</b>
4.1	REZULTATE OBȚINUTE.....	40
4.2	DIRECȚII DE DEZVOLTARE.....	40
<b>5</b>	<b>BIBLIOGRAFIE.....</b>	<b>42</b>

# 1 Introducere

## 1.1 Context general

În ultimele două decenii, procentul adulților clasificați ca fiind obezi a crescut alarmant la nivel global, trecând de la 7% la 16%, în timp ce, în rândul copiilor, procentul s-a cvadruplat, de la 2% la 8%, conform datelor furnizate de Organizația Mondială a Sănătății. Această creștere rapidă și îngrijorătoare a obezității a devenit o problemă majoră de sănătate publică la nivel mondial, având un impact semnificativ asupra calității vieții și asupra riscului de apariție a unor boli cronice, cum ar fi diabetul de tip 2, bolile cardiovasculare și anumite tipuri de cancer.

Dacă ne îndreptăm atenția către țările dezvoltate, unde populația are, în general, acces stabil la alimente și un stil de viață mai sedentar, statisticile devin și mai alarmante. În aceste țări, între 30 și 40% dintre adulți sunt clasificați ca fiind obezi, iar între 7 și 9% dintre aceștia suferă de obezitate severă. Aceste cifre reflectă o tendință periculoasă care necesită intervenții urgente pentru a preveni escaladarea și mai mare a problemei. Creșterea prevalenței obezității a determinat apariția și dezvoltarea unor metode avansate de monitorizare și evaluare a compoziției corporale, esențiale pentru identificarea și gestionarea acestor afecțiuni.

Printre cele mai frecvent utilizate metode de evaluare a compoziției corporale se numără Indicele de Masă Corporală (IMC) și procentul de grăsime corporală. IMC este o valoare numerică derivată dintr-o formulă simplă care utilizează înălțimea și greutatea unei persoane pentru a clasifica indivizii în diferite categorii de greutate: subponderal, normoponderal, supraponderal și obez. Deși IMC este un instrument utilizat pe scară largă datorită simplității sale, are limitări semnificative. Acesta nu face distincția între masa grasă și masa musculară, ceea ce poate duce la clasificări eronate. De exemplu, persoanele cu o masă musculară mare, cum ar fi atleții, pot avea un IMC ridicat, similar cu cel al unei persoane supraponderale, deși compoziția lor corporală este sănătoasă.

În contrast, procentul de grăsime corporală oferă o măsurătoare mult mai precisă și complexă a compoziției corporale, reflectând în mod direct cantitatea de grăsime din corp. Acest parametru este crucial pentru evaluarea stării de sănătate a unui individ, fiind un indicator mult mai bun al riscului de obezitate și al complicațiilor asociate acesteia, precum și un predictor al riscului de apariție a unor boli metabolice, cum ar fi diabetul de tip 2. Studiile au arătat că diferențele în procentul de grăsime corporală între femei și bărbați sunt determinate de factori biologici și fiziologici. În general, femeile au un procent mai mare de grăsime corporală comparativ cu bărbații, acest aspect fiind esențial pentru funcțiile vitale, cum ar fi echilibrul hormonal și sănătatea reproductivă.

Un procent sănătos de grăsime corporală este esențial pentru funcționarea optimă a organismului. În mod paradoxal, atât un procent prea mare, cât și un procent prea mic de grăsime corporală pot avea consecințe negative asupra sănătății. De exemplu, un procent prea scăzut de grăsime corporală, întâlnit frecvent în rândul sportivilor de



performanță, poate afecta sănătatea hormonală și poate compromite funcțiile vitale, mai ales la femei. În sporturi, procentul de grăsime corporală joacă un rol crucial, influențând performanța, viteza, puterea și capacitatea de recuperare a atleților. Prin urmare, monitorizarea și menținerea unui procent optim de grăsime corporală este esențială atât pentru sănătatea generală, cât și pentru performanța sportivă.

Cu toate acestea, din cauza complexității măsurătorii procentului de grăsime corporală, metodele tradiționale de evaluare necesită echipamente și personal specializat. Printre cele mai precise metode se numără Absorbțimetria duală cu raze X (DEXA scan), cântărirea hidrodensitometrică (sub apă), Pletismografia cu deplasare de aer (ADP) și Imagistica prin rezonanță magnetică (IRM/MRI). Deși aceste metode sunt extrem de precise, ele sunt și costisitoare și implică utilizarea unor echipamente specializate care nu sunt accesibile tuturor.

Ca alternativă, măsurarea pliurilor cutanate este o metodă mai accesibilă pentru evaluarea procentului de grăsime corporală. Totuși, această metodă are limitări importante, inclusiv dependența de abilitatea și experiența persoanei care efectuează măsurătorile, precum și variabilitatea rezultatelor în funcție de formula utilizată pentru calculul estimărilor. Aceste limitări subliniază necesitatea dezvoltării unor metode non-invazive, accesibile, precise și ușor de utilizat pentru evaluarea compoziției corporale, metode care să poată fi utilizate atât în cercetare, cât și în practica clinică sau pentru uz personal.

În acest context, rețelele neuronale convoluționale (RNC) au demonstrat un potențial enorm în procesarea și analiza imaginilor, inclusiv a celor medicale. Capacitatea acestor rețele de a învăța și de a extrage caracteristici complexe din imagini le face candidați ideali pentru aplicații de clasificare și predicție în domeniul sănătății. Studiul de față își propune să investigheze posibilitatea utilizării rețelelor neuronale convoluționale pentru determinarea procentului de grăsime corporală pe baza analizării imaginilor. Prin dezvoltarea și validarea unui model de RNC capabil să estimeze procentul de grăsime corporală cu o precizie comparabilă sau chiar superioară metodelor actuale, s-ar putea ajunge la o metodă non-invazivă de evaluare a compoziției corporale. Aceasta ar putea fi utilizată nu doar în cercetare și practica clinică, ci și în viața de zi cu zi, oferind astfel un instrument valoros pentru monitorizarea sănătății și prevenirea obezității și a bolilor asociate.

## **1.2 Obiective**

Obiectivul principal al acestui studiu este de a dezvolta și implementa o Rețea Neuronală Convoluțională (RNC) care să fie capabilă să proceseze imagini ale corpului uman și să realizeze predicții precise cu privire la sexul și procentul de grăsime corporală al persoanei respective. Acest obiectiv implică o serie de pași esențiali, fiecare contribuind la realizarea unei soluții robuste și fiabile pentru evaluarea compoziției corporale într-un mod non-invaziv și accesibil.

Pasi pentru realizarea obiectivului principal:

a. Colectarea Datelor

Primul pas crucial în realizarea acestui obiectiv este colectarea unui set de date extins și diversificat, care să includă imagini ale corpului uman împreună cu procentul de grăsime corporală corespunzător fiecărui individ. Este esențial ca aceste imagini să fie obținute din surse de măsurare recunoscute și precise, cum ar fi scanările DEXA, care oferă date de înaltă calitate și exactitate. Setul de date trebuie să fie suficient de cuprinzător pentru a permite rețelei neuronale să învețe și să generalizeze detaliile esențiale, reflectând corect procentul de grăsime corporală. În plus, setul de date trebuie să includă o diversitate largă de sexe, vârste, și etnii pentru a asigura aplicabilitatea și echitatea modelului la o populație cât mai variată, evitând astfel riscurile de discriminare și asigurând o acuratețe ridicată pentru toate categoriile demografice.

#### b. Preprocesarea datelor

Odată ce datele sunt colectate, următorul pas este preprocesarea acestora, un proces esențial pentru pregătirea eficientă a datelor în vederea antrenării modelului. Preprocesarea implică mai multe etape, inclusiv redimensionarea imaginilor la dimensiuni standard, pentru a asigura uniformitatea și compatibilitatea datelor în timpul antrenării. În plus, normalizarea imaginilor este crucială pentru a elimina variațiile neesențiale din date și pentru a îmbunătăți performanța modelului. Augmentarea datelor este un alt pas important în acest proces, care implică generarea de versiuni ușor modificate ale imaginilor originale prin rotații, scalări sau schimbări de contrast. Aceste tehnici ajută la creșterea diversității setului de date și la îmbunătățirea capacității modelului de a generaliza, reducând astfel riscul de supraadaptare (overfitting), unde modelul învață prea specific pe datele de antrenament și nu reușește să performeze bine pe date noi.

#### c. Proiectarea arhitecturii Rețelei Neuronale Convoluționale

Un pas critic în dezvoltarea acestui proiect este proiectarea arhitecturii RNC, care implică selecția și configurarea optimă a numărului de nivele interioare, filtre, și alți parametri relevanți care să contribuie la maximizarea performanței modelului. Această etapă necesită o înțelegere profundă a conceptelor de deep learning și a modului în care diferite structuri ale rețelelor neuronale pot influența capacitatea acestora de a învăța și de a face predicții precise. De asemenea, este importantă ajustarea hiperparametrilor, cum ar fi rata de învățare, numărul de epoci de antrenament și dimensiunea batch-ului, pentru a optimiza procesul de antrenare și a obține cele mai bune rezultate posibile.

#### d. Antrenarea Modelului

Antrenarea modelului este o etapă fundamentală în acest proiect, care implică utilizarea tehnicilor avansate de deep learning pentru a instrui RNC pe setul de date preprocesat. În timpul acestui proces, modelul va învăța să recunoască și să extragă caracteristicile relevante din imagini, care sunt asociate cu sexul și procentul de grăsime corporală. De asemenea, optimizarea performanței modelului prin ajustarea continuă a parametrilor este esențială pentru a îmbunătăți acuratețea și fiabilitatea predicțiilor. Aceasta presupune utilizarea unor tehnici precum regularizarea, dropout-ul, și alte metode de prevenire a supraadaptării, asigurând astfel că modelul se va generaliza bine pe date noi și necunoscute.

Obiectivul secundar al studiului: Validarea și testarea modelului

Obiectivul secundar al studiului este validarea și testarea RNC dezvoltate, pentru a evalua acuratețea și fiabilitatea acestora într-un context cât mai apropiat de condițiile reale de utilizare. Această etapă este esențială pentru a asigura că modelul poate fi utilizat cu succes de către utilizatori, atât în cercetare, cât și în practică, fără a fi afectat semnificativ de variațiile din seturile de date noi.

Pasi pentru realizarea obiectivului secundar:

a. Testarea modelului pe un set de date nou

Primul pas în realizarea acestui obiectiv secundar este testarea modelului pe un set de date care nu a fost utilizat în procesul de antrenare sau validare. Acest lucru permite evaluarea capacității modelului de a generaliza și de a face predicții precise pe date noi, simulând astfel condițiile reale de utilizare. Este esențial ca acest set de date să fie diversificat și să reflecte variabilitatea întâlnită în viața reală pentru a obține o evaluare corectă a performanței modelului.

b. Compararea rezultatelor modelului RNC cu metode consacrate

După testarea modelului, următorul pas este compararea rezultatelor obținute de RNC cu valorile obținute prin metode consacrate, precum DEXA scan sau MRI. Această comparație este crucială pentru a valida acuratețea modelului și pentru a demonstra că acesta poate oferi predicții comparabile cu cele ale metodelor tradiționale, de înaltă precizie.

Prin realizarea acestor obiective, proiectul își propune să contribuie la creșterea accesibilității metodelor de evaluare a obezității și a compoziției corporale pentru un număr cât mai mare de persoane. Este important de subliniat că acest proiect nu intenționează să înlocuiască metodele clasice, specialiștii din domeniu sau consultațiile la nutriționist, ci să ofere un instrument suplimentar care să sprijine centrele specializate existente și să fie util pentru uz personal, contribuind astfel la îmbunătățirea sănătății publice prin monitorizarea compoziției corporale. Răspândirea unor metode non-invazive, accesibile și precise poate avea un impact semnificativ asupra prevenției și gestionării obezității, facilitând astfel o mai bună înțelegere și gestionare a acestei afecțiuni complexe.

### **1.3 Specificații**

În cadrul acestei lucrări, s-a dezvoltat o rețea neuronală care are ca scop preluarea și clasificarea imaginilor corpului uman în funcție de procentajul de grăsime corporală. Procesul de dezvoltare a acestui model implică mai multe etape esențiale, fiecare contribuind la asigurarea unei performanțe ridicate și a unei precizii în predicțiile realizate de rețea.

Prima etapă în dezvoltarea modelului a constat în colectarea unui set extins de imagini ale corpului uman, care să fie reprezentative pentru o gamă largă de sexe, etnii și

tipuri corporale. Aceste imagini au fost selectate pentru a include o diversitate suficientă, ceea ce permite rețelei neuronale să învețe și să generalizeze mai bine detaliile specifice fiecărui procentaj de grăsime corporală. Odată ce imaginile au fost colectate, ele au fost redimensionate la o dimensiune standard de 180p x 180p. Această redimensionare este esențială pentru a asigura uniformitatea datelor de intrare, permițând astfel o prelucrare mai eficientă și un antrenament mai consistent al modelului.

Următoarea etapă importantă a fost preprocesarea imaginilor, un proces esențial care asigură calitatea și relevanța datelor înainte de a fi introduse în modelul de rețea neuronală. Imaginile standardizate au fost supuse unui proces de augmentare, care a inclus operațiuni precum rotații, flipuri, ajustări ale contrastului și luminozității, precum și adăugarea de zgomot controlat. Aceste tehnici de augmentare au fost aplicate pentru a crea varietate și pentru a reduce zgomotul din datele inițiale, ceea ce ajută la îmbunătățirea capacității modelului de a generaliza informația și de a învăța mai eficient. Această etapă este crucială pentru a preveni supraadaptarea (overfitting), asigurând astfel că modelul va funcționa bine pe date noi, necunoscute.

După finalizarea procesului de preprocesare, datele pregătite au fost introduse într-un model de rețea neuronală convoluțională (RNC), care utilizează diverse tehnici de machine learning pentru a analiza în detaliu imaginile. Modelul a fost proiectat pentru a extrage caracteristici esențiale din imagini, cum ar fi contururile, texturile și nuanțele, care sunt relevante pentru estimarea procentajului de grăsime corporală. Folosind aceste caracteristici, modelul învață să generalizeze și să facă predicții precise privind procentajul de grăsime corporală al subiectului.

În final, modelul dezvoltat este capabil să afișeze o imagine împreună cu predicția făcută, oferind utilizatorului o estimare vizuală și numerică a procentului de grăsime corporală. Validarea modelului s-a realizat prin compararea rezultatelor obținute cu date reale și măsurători realizate prin metode consacrate, cum ar fi DEXA scan și MRI, pentru a asigura acuratețea și fiabilitatea predicțiilor.

Acest model de detectare a procentului de grăsime corporală a fost dezvoltat utilizând limbajul de programare Python și mediul de dezvoltare Jupyter Notebook, care au permis o flexibilitate și o rapiditate sporită în experimentare și dezvoltare. Printre uneltele esențiale utilizate se numără TensorFlow și Keras, care au fost folosite pentru construirea și antrenarea rețelelor neuronale. De asemenea, OpenCV a fost utilizat pentru procesarea imaginilor, în timp ce librăriile Pandas și Numpy au jucat un rol crucial în manipularea și gestionarea datelor. Matplotlib a fost folosită pentru vizualizarea rezultatelor, permițând o analiză detaliată a performanței modelului și facilitând identificarea eventualelor zone care necesită îmbunătățiri.

## 2 Studiu bibliografic

Cel mai vechi studiu care a propus problema posibilității de a prezice procentul de grăsime corporala a fost "A novel prediction method for body fat by using Choquet integral with respect to L-measure and Gamma-support" în 2009 și s-a folosit de regresia integrala Choquet bazat pe L-measure și Gamma-support

Regresia integrală Choquet este o metodă de modelare non-liniară care utilizează integralul Choquet pentru a combina informațiile provenite din diferite surse sau variabile. Aceasta metodă este potrivită pentru situații în care relațiile dintre variabile sunt complexe și interdependente.

Integralul Choquet este un concept matematic folosit pentru a agrega informații, ținând cont nu doar de valorile individuale ale variabilelor, dar și de interdependențele dintre ele. Acesta permite o combinație flexibilă și ponderată a variabilelor, oferind o alternativă la metodele tradiționale de regresie liniară.

L-measure este o funcție de măsură care cuantifică importanța sau relevanța fiecărei variabile individuale și a subseturilor de variabile în procesul de predicție. Aceasta funcție de măsură ajută la determinarea modului în care fiecare subset de variabile contribuie la valoarea finală a predicției. În contextul regresiei integrale Choquet, L-measure este utilizată pentru a pondera variabilele și subseturile acestora, asigurându-se că cele mai relevante informații sunt cele care au cel mai mare impact asupra rezultatului final.

Gamma-support este un concept utilizat pentru a determina nivelul de suport sau relevanță al diferitelor subseturi de variabile în cadrul modelului de regresie. Gamma-support este folosit pentru a identifica acele subseturi de variabile care au o contribuție semnificativă la predicție. Astfel, se poate reduce dimensionalitatea problemei, concentrându-se doar pe cele mai relevante combinații de variabile.

Următorul stagi al predicției procentului de grăsime corporala a început în 2015 cu studii precum "Intelligent feature subset selection with unspecified number for body fat prediction based on binary-GA and Fuzzy-Binary-GA" și "Research on human body composition prediction model based on Akaike Information Criterion and improved entropy method" în care se folosesc algoritmi genetici.

Binary Genetic Algorithm (Binary-GA) este un tip de algoritm genetic în care soluțiile posibile (indivizii) sunt reprezentate prin cromozomi binari. Fiecare cromozom este o secvență de biți (0 și 1), care codifică o soluție pentru problema de optimizare.

Fuzzy-Binary Genetic Algorithm (Fuzzy-Binary-GA) combină conceptele de logică fuzzy cu algoritmi genetici binari pentru a îmbunătăți performanța în problemele de optimizare. În această abordare, se introduce un mecanism fuzzy pentru a gestiona incertitudinea și variația în procesul de selecție și adaptare a operatorilor genetici.

Atât Binary-GA cât și Fuzzy-Binary-GA sunt metode puternice de optimizare, fiecare cu propriile sale avantaje. Binary-GA este simplu și robust, potrivit pentru o gamă largă de probleme. Fuzzy-Binary-GA, pe de altă parte, îmbunătățește performanța prin utilizarea logicii fuzzy, făcându-l mai adaptabil și capabil să gestioneze incertitudinea și variația în procesul de optimizare.

Akaike Information Criterion (AIC) este un criteriu utilizat pentru a selecta un model statistic dintre un set de modele candidate. AIC este utilizat în contextul modelelor statistice care sunt estimate prin metoda de maximum verosimilitate și oferă o măsură de calitate relativă a acestor modele. AIC ajută la echilibrarea complexității modelului și a capacității acestuia de a se potrivi datelor. Akaike Information Criterion (AIC) este un instrument valoros pentru selecția modelului în statistică și învățare automată, oferind o modalitate simplă și eficientă de a evalua și compara modelele candidate. Deși are limitele sale, AIC rămâne o metodă populară datorită echilibrului pe care îl oferă între complexitatea modelului și adecvarea acestuia la date.

Din 2018 până în prezent studiile și metodele de predicție folosite au început să folosească machine learning și rețele neuronale

Machine Learning (ML) este un subdomeniu al inteligenței artificiale care se concentrează pe dezvoltarea algoritmilor și modelelor ce permit calculatoarelor să învețe din date și să facă predicții sau decizii fără a fi explicit programate pentru fiecare sarcină în parte. ML se bazează pe detectarea tiparelor și relațiilor din date, permițând astfel automatizarea și îmbunătățirea performanței în diverse aplicații.

Convolutional Neural Network (CNN) este un tip specializat de rețea neuronală artificială, proiectată pentru a procesa date care au o grilă topologică, cum ar fi imaginile. CNN-urile sunt utilizate în principal pentru recunoașterea și clasificarea imaginilor datorită capacității lor de a capta caracteristici spațiale și de a gestiona date de înaltă dimensiune.

Studii precum "Exploring Regression Models and Optimization Techniques for Accurate Body Fat Prediction: A Comparative Approach" explorează predicția procentului de grăsime corporală prin modelele de regresie. Alegerea algoritmilor de regresie dezvăluie un echilibru delicat între robustețe și precizie, cu regresia Ridge emergând ca alegerea superioară datorită preciziei sale predictive și robusteții în prezența valorilor aberante. Acest rezultat extinde tehnicile de predicție a grăsimii corporale, în special în contextul datelor zgomotoase sau neregulate. Studiul subliniază, de asemenea, importanța preprocesării caracteristicilor, evidențiind rolul crucial al standardizării în îmbunătățirea performanței predictive. O descoperire importantă este existența unei asocieri pozitive semnificative între vârstă, greutatea corporală și procentul de grăsime corporală. Integrarea vârstei ca variabilă predictivă sporește precizia predicției grăsimii corporale, oferind un potențial considerabil pentru îmbunătățirea rezultatelor în domeniile sănătății și fitness-ului, prin evaluări și intervenții specifice vârstei.

## 3 Analiză, proiectare, implementare

### 3.1. Analiza problemei

Analiza datelor joacă un rol crucial în diverse industrii, unde identificarea și interpretarea modelelor și tendințelor din date pot duce la îmbunătățirea bunurilor și serviciilor. Pentru a putea efectua analize complexe și pentru a implementa eficient învățarea automată, este esențial ca datele să fie organizate și structurate într-un mod coerent și accesibil. O practică standard în acest sens este împărțirea datelor în două seturi mari: unul dedicat antrenării și altul pentru validare. Această organizare facilitează nu doar procesul de antrenare a algoritmilor, ci și evaluarea riguroasă a acestora, asigurându-se că modelele rezultate pot fi aplicate în mod fiabil pe date noi.

În cadrul procesului de dezvoltare a algoritmilor de învățare automată, folderul de antrenare este utilizat pentru a învăța și extrage tendințele din datele disponibile, în timp ce folderul de validare conține datele care sunt utilizate pentru a testa și evalua performanța modelului creat. Această separare este esențială pentru a verifica capacitatea algoritmului de a generaliza, evitând astfel supraspecializarea pe datele de antrenament. Astfel, structura datelor în foldere distincte pentru antrenare și validare reprezintă o parte fundamentală a procesului de dezvoltare a unui model robust, capabil să fie aplicat în contexte reale de producție.

Rețelele neuronale convoluționale (RNC) necesită de obicei un volum mare de date pentru a instrui eficient algoritmi de învățare automată. După finalizarea etapei de antrenare, folderul de validare devine esențial pentru a evalua performanța modelului pe un set de date suplimentar, asigurându-se că modelul poate generaliza și performa bine pe date noi. În general, se recomandă ca setul de validare să conțină cel puțin 20% din datele disponibile, pentru a oferi o evaluare echilibrată și precisă a modelului. În cadrul acestui studiu specific, raportul dintre setul de antrenare și cel de validare a fost stabilit la 65%-35%. Alegerea de a mări proporția setului de validare a fost făcută strategic pentru a maximiza performanța modelului, oferindu-i acestuia mai multe date pentru evaluarea și ajustarea sa înainte de implementarea finală.

Este important de menționat că întregul set de date utilizat în acest studiu a fost construit și împărțit manual, deoarece nu există un set de date preexistent care să corespundă în totalitate cerințelor și specificațiilor acestui proiect. Deși există similitudini între acest studiu și altele anterioare, natura unică a datelor utilizate aici necesită o atenție specială în crearea și organizarea acestora. Setul de date include doar imagini bazate pe surse verificate, ceea ce explică și numărul limitat de imagini disponibile. Această abordare, deși poate limita cantitatea de date, garantează calitatea și fiabilitatea modelului final, asigurându-se că predicțiile realizate sunt bazate pe date de înaltă precizie.

Număr date pentru setul de antrenare:	Număr date pentru setul de validare:	Număr total de imagini folosite în crearea modelului:
503 imagini	258 imagini	761 imagini

Tabel 3.1. Numarul Datelor folosite in crearea Retelei

 Training		6/28/2024 9:08 PM	File folder
 Validation		6/29/2024 7:52 PM	File folder

Figura 3.1. Structura de impartire a datelor





































 f10-12%	 f46-48%	 m19-20%
 f13-15%	 f49-51%	 m21-22%
 f16-18%	 f52-54%	 m23-24%
 f19-21%	 m3%	 m25-26%
 f22-24%	 m4%	 m27-28%
 f25-27%	 m5-6%	 m29-30%
 f28-30%	 m7-8%	 m31-32%
 f31-33%	 m9-10%	 m33-34%
 f34-36%	 m11-12%	 m35-36%
 f37-39%	 m13-14%	 m37-38%
 f40-42%	 m15-16%	 m39-40%
 f43-45%	 m17-18%	 m41-42%

Figura 3.2. Structura seturilor de antrenare si validare



Fișierele de date sunt structurate pentru a reflecta incremental variațiile procentuale de grăsime corporală, cu diferențiere clară în funcție de sex, având în vedere specificitățile biologice și fiziologice ale acestora. Pentru a asigura o acuratețe cât mai mare în predicțiile modelului, s-au implementat incremente de 2% sau 3% pentru fiecare categorie. Femeile, în mod natural, au un procent de grăsime corporală mai mare decât bărbații, cu o diferență medie cuprinsă între 9% și 11%. Din acest motiv, pentru clasificarea datelor referitoare la femei, s-au folosit incremente de 3%, permițând astfel o mai bună reprezentare a variațiilor și o mai mare precizie în estimările făcute de rețeaua neuronală.

În ceea ce privește bărbații, datorită capacității lor de a atinge și menține un procent de grăsime corporală mult mai scăzut decât femeile, structura fișierelor a fost ajustată pentru a include și incremente mai mici. Astfel, pentru cazurile rare în care bărbații ating niveluri extrem de scăzute de grăsime corporală, de 3% sau 4%, au fost create fișiere suplimentare cu incremente de 1%. Aceste niveluri de grăsime corporală, deși posibile, sunt extrem de dificil de menținut pe termen lung din punct de vedere fiziologic, însă includerea lor este esențială pentru a acoperi întregul spectru al potențialelor scenarii întâlnite în rândul populației masculine. Această abordare asigură faptul că modelul dezvoltat poate face predicții precise chiar și în cazuri mai puțin comune, reflectând astfel realitatea diversității umane și a diferitelor categorii de compoziție corporală.

### 3.2. Implementarea în Python

Preprocesarea datelor reprezintă un set complex de procese și tehnici care sunt esențiale pentru a transforma datele brute într-un format care poate fi utilizat eficient în diverse aplicații, cum ar fi învățarea automată, analiza statistică și vizualizarea datelor. Această etapă este crucială pentru a asigura că datele sunt curate, coerente și într-o formă optimă pentru a genera rezultate precise și relevante. Prin preprocesare, se urmărește îmbunătățirea calității datelor, eliminarea erorilor și asigurarea uniformității, ceea ce contribuie la obținerea unor performanțe mai bune în modelele de învățare automată și la o interpretare corectă a rezultatelor.

În acest caz specific, am rescalat toate imaginile din setul de date la dimensiunea standardizată de 180x180 pixeli, pentru a asigura consistența în procesul de antrenare a modelului. După rescalare, s-a utilizat ImageDataGenerator din biblioteca Keras pentru a realiza augmentarea imaginilor de input. ImageDataGenerator este un instrument extrem de util și puternic pentru augmentarea datelor în timp real, care permite generarea de loturi de date cu augmentări aplicate în mod dinamic pe parcursul antrenamentului. Aceasta înseamnă că fiecare lot de date care intră în model este modificat și diversificat în timp real, ceea ce ajută la creșterea capacității modelului de a generaliza și de a performa bine pe date necunoscute. Astfel, augmentarea în timp real poate îmbunătăți semnificativ performanța și robustețea modelului de învățare automată, reducând riscul de supraadaptare și contribuind la dezvoltarea unui model mai fiabil și mai precis. Această abordare nu doar că sporește diversitatea setului de date, dar și optimizează procesul de învățare, rezultând într-un model capabil să facă predicții mai precise și mai generalizabile.

```
data_train_gen = ImageDataGenerator(
    rescale=1.0/255,
    rotation_range=15,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest'
)
```

Figura 3.5. Setarea parametrilor de augmentare a datelor

Acesta folosește diferiți parametri de augmentare cum ar fi:

Rotation\_range - intervalul de grade pentru rotații aleatorii ale imaginii;

Width\_shift\_range - fracțiunea din lățimea totală pentru deplasări orizontale aleatorii;

Height\_shift\_range - fracțiunea din înălțimea totală pentru deplasări verticale aleatorii;

Shear\_range - intensitatea tăierii (unghiul de tăiere în grade)

Zoom\_range - intervalul pentru zoom aleatoriu

Horizontal\_flip - răsturnarea orizontală aleatorie a imaginilor

Acești parametri se aplică aleatoriu asupra imaginilor pentru a crea diferite versiuni ale acestora. În acest fel rețeaua învață să recunoască elementele din imagine de la distanțe diferite, din unghiuri și poziții diferite. Augmentarea imaginilor este esențială pentru evitarea fenomenului de supraînvățare.

```
data_train = data_train_gen.flow_from_directory(
    data_train_path,
    shuffle=True,
    target_size=(img_height, img_width),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32
)
```

Figura 3.6. Augmentarea setului de antrenare

Imaginile din setul de validare nu sunt supuse procesului de augmentare, deoarece acestea trebuie să reflecte cât mai fidel condițiile reale în care modelul va fi aplicat, fără nicio modificare artificială.

Crearea modelului: După finalizarea etapelor de preprocesare, datele prelucrate sunt pregătite pentru a fi transferate către algoritmul de învățare automată. Modelul implementat în acest studiu este o rețea neuronală convoluțională, configurată cu un singur punct de intrare (input), un singur punct de ieșire (output), și 7 nivele interne (straturi) care contribuie la procesarea datelor. Această arhitectură a fost aleasă pentru a

asigura un echilibru optim între complexitate și performanță, fiind suficient de simplă pentru a fi eficientă, dar totuși capabilă să captureze caracteristicile esențiale din imagini.

Pentru implementarea modelului în Python, am utilizat clasa *Sequential* din biblioteca Keras, care este extrem de versatilă și potrivită pentru construirea de rețele neuronale strat cu strat. Prin utilizarea acestei clase, straturile sunt adăugate în mod secvențial, de la primul strat de intrare până la ultimul strat de ieșire, ceea ce simplifică semnificativ procesul de construire a modelului.

Fiecare strat din model preia ca input ieșirea stratului anterior, ceea ce asigură o propagare eficientă a datelor prin rețea. Această metodă secvențială este extrem de convenabilă pentru modelele liniare, deoarece nu necesită definirea explicită a legăturilor dintre straturi, totul fiind automatizat și intuitiv. De asemenea, această abordare permite adăugarea de noi straturi în model fără complicații suplimentare, ceea ce facilitează modificările și optimizările ulterioare ale arhitecturii rețelei neuronale. În acest model, am implementat un total de 8 blocuri, dintre care 4 sunt blocuri convoluționale, având ca scop principal extragerea și procesarea caracteristicilor esențiale din imaginile de intrare. Aceste blocuri convoluționale sunt esențiale pentru a identifica și a învăța trăsături complexe din datele vizuale, cum ar fi contururile, texturile și alte detalii importante care pot ajuta modelul să facă predicții precise ale procentului de grăsime corporală.

```
model.add(Input(shape=(180, 180, 3)))
model.add(Conv2D(64,(3,3),padding='same', activation='relu'))
model.add(layers.BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.4))

model.add(Conv2D(128,(5,5),padding='same', activation='relu'))
model.add(layers.BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.4))

model.add(Conv2D(256,(3,3),padding='same', activation='relu'))
model.add(layers.BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.4))

model.add(Conv2D(512,(3,3),padding='same', activation='relu'))
model.add(layers.BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.4))
```

Figura 3.7. Blocurile Convoluționale

Primul bloc are un rol dublu: nu doar că îndeplinește funcția de strat de intrare, dar asigură și că modelul nu acceptă ca intrare decât imagini care au dimensiunile standardizate de 180x180 pixeli și care sunt compuse din trei canale de culoare (RGB). Acest control inițial al dimensiunilor și formatului imaginii este esențial pentru a asigura coerența datelor pe parcursul procesului de prelucrare și antrenare.

După acest prim strat de intrare, stratul convolutional Conv2D intră în acțiune. Acesta aplică un număr specific de filtre care au ca scop extragerea unor informații detaliate din imagini. În cazul primului bloc convolutional, s-au setat 64 de filtre de ieșire, fiecare având dimensiunile 3x3 pixeli. Aceste filtre sunt esențiale pentru identificarea caracteristicilor de bază din imagini, cum ar fi marginile și colțurile. Parametrul *padding* este utilizat pentru a asigura că harta de caracteristici de ieșire are aceleași dimensiuni spațiale (înălțime și lățime) ca și intrarea, ceea ce se realizează prin completarea marginii imaginii cu zerouri, dacă este necesar. Această tehnică este esențială pentru a păstra informațiile la margini și pentru a preveni reducerea dimensiunii imaginii pe măsură ce aceasta trece prin straturile convoluționale.

Funcția de activare folosită în acest model este ReLU (Rectified Linear Unit), o funcție extrem de populară și eficientă în modelele de învățare automată, în special în rețelele neuronale convoluționale. Funcția ReLU este definită matematic prin formula:

$$\text{ReLU}(x) = \max(0, x)$$

Aceasta înseamnă că funcția returnează valoarea de intrare  $x$  dacă  $x$  este mai mare decât 0; în caz contrar, returnează 0. Funcția ReLU introduce non-linearitate în model, un aspect crucial care permite rețelei să învețe și să capteze tipare mult mai complexe și diverse din date. Fără utilizarea funcțiilor de activare non-lineare precum ReLU, chiar și o rețea neuronală cu mai multe straturi s-ar comporta similar unui model liniar simplu, fără a putea capta relații complexe între datele de intrare. Un alt beneficiu major al utilizării funcției ReLU este că aceasta setează toate valorile negative la zero, rezultând o reprezentare a datelor mai rară. Această trunchiere la zero poate fi benefică deoarece reduce dependența modelului de valorile specifice ale intrării, făcând rețeaua mai robustă și, în multe cazuri, reducând riscul de suprainvătărare (overfitting). De asemenea, ReLU este extrem de eficientă din punct de vedere computațional, deoarece operația de trunchiere la zero este foarte simplă și rapidă, făcând-o mai eficientă în comparație cu alte funcții de activare mai complexe, cum ar fi sigmoid sau tanh, care necesită calcule mai complexe și timp de procesare mai mare. Astfel, ReLU nu doar că îmbunătățește performanța modelului, dar contribuie și la optimizarea resurselor de calcul.

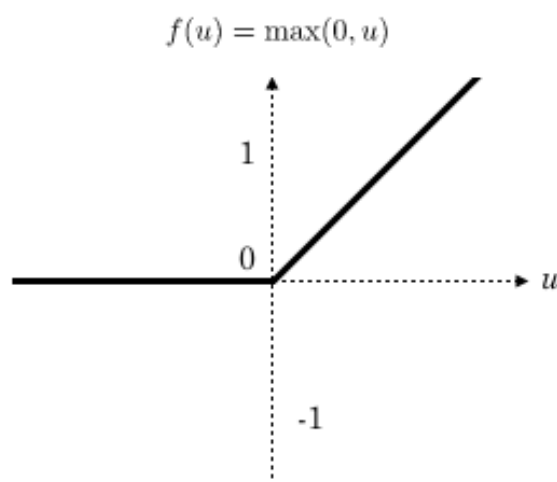


Figura 3.8. Graficul funcției ReLU

Stratul BatchNormalization normalizează activările stratului anterior pentru fiecare mini-batch în timpul antrenamentului. Acest lucru ajută la stabilizarea și accelerarea procesului de antrenare. Normalizează ieșirea stratului convoluțional pentru a avea media zero și varianța unitară. După normalizare, aplică o transformare de scalare și deplasare. Aceasta este controlată de parametri care pot fi învățați și care sunt actualizați în timpul antrenamentului.

Stratul MaxPooling2D reduce dimensiunile spațiale (înălțimea și lățimea) ale hărților de caracteristici, ceea ce ajută la reducerea numărului de parametri și a încărcării computaționale. Dimensiunea ferestrei de pooling este de 2x2 pixeli. Acest lucru înseamnă că va lua valoarea maximă din fiecare bloc de 2x2 pixeli din harta de caracteristici. Implicit, pasul (stride) este același cu dimensiunea ferestrei de pooling, astfel încât fereastra se deplasează cu 2 pixeli la un moment dat. Acest lucru reduce efectiv înălțimea și lățimea hărților de caracteristici la jumătate.

Stratul de Dropout este implementat ca metoda de regularizare a rețelei neuronale.

Prin dezactivarea aleatorie a unei fracțiuni de neuroni (în cazul de față 40%) în timpul antrenamentului, modelul este încurajat să învețe reprezentări mai robuste și să generalizeze mai bine pe date noi.

Regularizarea în cadrul rețelelor neuronale este o tehnică utilizată pentru a preveni supra învățarea și pentru a îmbunătăți capacitatea modelului de a generaliza pe date noi, nevăzute.

Supra învățarea (overfitting) este un fenomen în învățarea automată și statistică unde un model învață foarte bine detaliile și zgomotul din setul de date de antrenament la un nivel care îi afectează negativ performanța pe date noi, nevăzute. În esență, modelul devine prea complex și specific pentru setul de date de antrenament, capturând particularitățile și zgomotul acestuia în loc să învețe pattern-uri generale aplicabile și altor date. Supra învățarea apare atunci când modelul învață prea bine detaliile și zgomotul din setul de date de antrenament, performând slab pe date noi. Regularizarea ajută la controlul acestui fenomen, impunând restricții asupra modelului și reducând complexitatea acestuia.

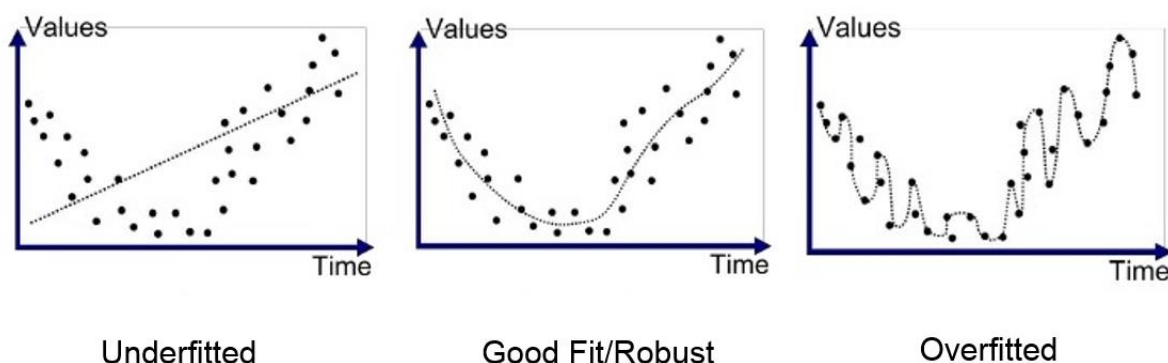


Figura 3.9. Diferența dintre Overfitting și Underfitting

Regularizarea este esențială pentru a construi modele de învățare automată eficiente și robuste, capabile să generalizeze bine pe date noi. Prin utilizarea tehnicilor de regularizare, cum ar fi L1, L2, dropout și altele, putem controla complexitatea modelului și preveni supraînvățarea, asigurând performanțe bune pe seturi de date nevăzute.

Celelalte 3 blocuri convoluționale sunt similare primului. Singurele diferențe fiind faptul că acestea preiau ca intrare ieșirea stratului precedent, și numărul de filtre aplicate în stratul Conv2D. Acestea fiind 128, 256, respectiv 512.

Următorul bloc are un singur strat și anume stratul Flatten, care este utilizat pentru a transforma un tensor multidimensional într-un vector unidimensional. Este frecvent utilizat în arhitecturile de rețele neuronale convoluționale înainte de a trece la straturi dense. Straturile convoluționale și de pooling produc ieșiri 3D (înălțime, lățime, canale). Straturile dense așteaptă, de obicei, intrări 1D (un vector). Stratul Flatten este utilizat pentru a realiza această tranziție.

```
model.add(Flatten())
```

Figura 3.10. Stratul Flatten

Următoarele două blocuri sunt straturi complet conectate (dense). Acestea joacă un rol esențial în procesarea finală a caracteristicilor extrase de straturile convoluționale și în realizarea clasificării sau a regresiei. Într-un strat dens, fiecare neuron este conectat la fiecare neuron din stratul anterior. Aceasta înseamnă că fiecare ieșire a neuronului anterior contribuie la fiecare ieșire a neuronului curent. Matematic, această operație este o multiplicare matrice-vector urmată de adăugarea unui vector de bias. Stratul dens efectuează o transformare liniară a intrărilor. Fiecare ieșire a neuronului este calculată ca o combinație liniară a intrărilor sale ponderate:

$$y=W \cdot x+b$$

Formula 3.2. Calcularea ieșirilor în funcție de greutăți și bias

Unde  $W$  este matricea de greutăți,  $x$  este vectorul de intrare și  $b$  este vectorul de bias.

După transformarea liniară, o funcție de activare ReLU, este aplicată pentru a introduce non-liniarități în model. Acest lucru permite modelului să învețe relații complexe. Straturile dense sunt esențiale pentru a conecta straturile convoluționale, care extrag caracteristici, cu ieșirea modelului, care efectuează sarcina finală de clasificare.

```

model.add(Dense(128))
model.add(layers.BatchNormalization())
model.add(layers.Activation('relu'))
model.add(Dropout(0.4))

model.add(Dense(256, kernel_regularizer=l2(0.03)))
model.add(layers.BatchNormalization())
model.add(layers.Activation('relu'))
model.add(Dropout(0.4))

```

Figura 3.11 Blocurile Dense

Stratul dense este un strat complet conectat, unde fiecare neuron este conectat la toți neuronii din stratul anterior. Primul bloc are 128 de neuroni.

Stratul BatchNormalization normalizează activările stratului anterior pentru fiecare mini-batch în timpul antrenamentului. Acesta ajustează activările pentru a avea media zero și varianța unitară. După normalizare, stratul aplică o transformare de scalare și deplasare (parametri de scalare și bias sunt învățați în timpul antrenamentului). Aceasta ajută la stabilizarea și accelerarea procesului de antrenament.

Cel de-al doilea bloc are 256 de neuroni și folosește regularizare de tip L2 înainte de ultimul strat. Regularizarea L2, cunoscută și sub numele de Ridge Regularization, este o tehnică utilizată pentru a preveni supra-învățarea în modelele de învățare automată. Funcționează prin adăugarea unei penalizări la funcția de pierdere a modelului, penalizare proporțională cu suma pătratelor greutăților (coeficienților) modelului. Aceasta are efectul de a încuraja modelul să mențină greutățile cât mai mici posibil, ceea ce duce la un model mai simplu și mai generalizabil.

Regularizarea L2 adaugă un termen suplimentar la funcția de pierdere originală. Funcția de pierdere  $J(\theta)$  cu regularizare L2 este definită astfel:

$$J(\theta) = J_0(\theta) + \lambda \sum_{i=1}^n \theta_i^2$$

Formula 3.3. calcularea funcției de regularizare L2

Aici,  $J_0(\theta)$  este funcția de pierdere originală (de exemplu, eroarea medie pătratică pentru regresie sau entropia încrucișată pentru clasificare),  $\theta$  reprezintă greutățile modelului, iar  $\lambda$  este hiperparametrul de regularizare care controlează cât de mare este penalizarea. Termenul de regularizare penalizează valorile mari ale greutăților. Aceasta are ca rezultat scăderea valorilor greutăților, reducând astfel complexitatea modelului. Greutățile mici ajută la prevenirea supraînvățării, deoarece modelul nu va depinde foarte

mult de nicio caracteristică specifică a datelor de antrenament. În timpul antrenamentului, algoritmul de optimizare va actualiza greutatele modelului nu doar pentru a minimiza eroarea de predicție, ci și pentru a minimiza suma pătratelor greutăților. Constrângerea greutăților să fie mici face modelul mai stabil și mai puțin susceptibil la variații mici în datele de intrare. Greutățile mici conduc la un model mai simplu, care este mai ușor de interpretat și de generalizat.

Stratul de output este tot un strat Dense care folosește funcția de activare Softmax.

Funcția de activare Softmax este utilizată în mod obișnuit în rețelele neuronale pentru probleme de clasificare multi-clasă. Aceasta transformă ieșirile unei rețele neuronale într-o distribuție de probabilități, unde suma probabilităților pentru toate clasele este egală cu 1. Funcția de activare Softmax este deosebit de utilă în stratul final al unei rețele neuronale pentru clasificare, deoarece permite modelului să prezică probabilitatea ca o intrare să aparțină fiecărei clase posibile.

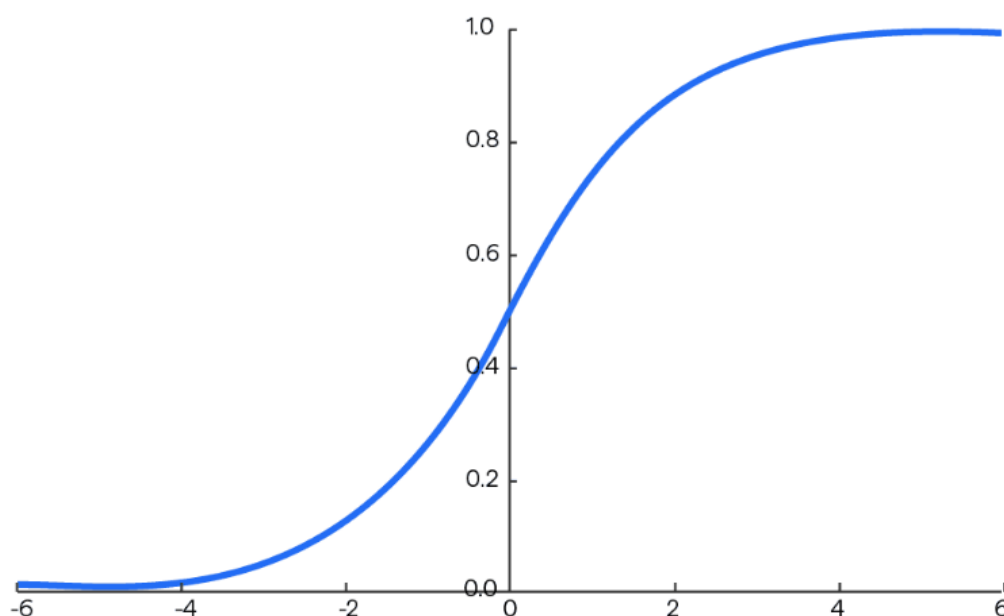


Figura 3.12. Graficul funcției Softmax

Pentru un vector de intrări  $z=[z_1, z_2, \dots, z_n]$ , ieșirea Softmax pentru fiecare element  $z_i$  este dată de formula:

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^n e^{z_j}}$$

Formula 3.4. calcularea funcției de activare Softmax

Fiecare element al vectorului de ieșire va fi o valoare între 0 și 1, reprezentând probabilitatea prezisă pentru clasa respectivă. Suma tuturor elementelor vectorului de ieșire va fi egală cu 1, ceea ce permite interpretarea ieșirilor ca probabilități. Transformă



logiturile brute (ieșirile stratului dens fără funcție de activare) în valori normalizate, care pot fi interpretate ca probabilități.

Funcția de activare *Softmax* joacă un rol crucial în rețelele neuronale, în special în problemele de clasificare pe clase multiple. Unul dintre principalele avantaje ale utilizării *Softmax* este că poate contribui semnificativ la stabilitatea numerică în timpul antrenamentului rețelelor neuronale, prin reducerea riscului de sub- sau supra-flux numeric, care poate apărea datorită calculelor exponențiale implicate. Pentru a evita aceste probleme, o tehnică comună de stabilizare utilizată în aplicarea funcției *Softmax* este scăderea valorii maxime a vectorului de intrare înainte de aplicarea exponențierii. Această abordare simplă, dar eficientă, previne ca valorile mari să devină și mai mari după exponențiere, ceea ce ar putea duce la instabilități numerice și la probleme de performanță.

Funcția *Softmax* normalizează ieșirile stratului final al rețelei neuronale, transformându-le într-o distribuție de probabilități. Acest lucru permite interpretarea directă a acestor ieșiri ca probabilități de clasă, fiecare valoare reprezentând probabilitatea ca observația de intrare să aparțină unei anumite clase. Această caracteristică face ca *Softmax* să fie esențială pentru problemele de clasificare, unde este important nu doar să se facă o predicție, ci și să se cuantifice încrederea în acea predicție.

Prin normalizarea ieșirilor, *Softmax* nu doar că îmbunătățește stabilitatea numerică a rețelei, dar și contribuie la o mai bună performanță a modelului în ansamblu. Această funcție este un instrument fundamental în învățarea automată, fiind larg utilizată în modele care necesită o clasificare precisă și fiabilă. De asemenea, *Softmax* ajută la interpretarea rezultatelor modelului, facilitând procesul de luare a deciziilor pe baza ieșirilor modelului, făcându-l astfel indispensabil în numeroase aplicații de clasificare.

**Total params: 9,659,684 (36.85 MB)**

**Trainable params: 9,656,996 (36.84 MB)**

**Non-trainable params: 2,688 (10.50 KB)**

Figura 3.13 Numarul total de parametrii folositi in primul model

Layer (type)	Output Shape	Param #
conv2d_20 (Conv2D)	(None, 180, 180, 64)	1,792
batch_normalization_30 (BatchNormalization)	(None, 180, 180, 64)	256
max_pooling2d_20 (MaxPooling2D)	(None, 90, 90, 64)	0
dropout_30 (Dropout)	(None, 90, 90, 64)	0
conv2d_21 (Conv2D)	(None, 90, 90, 128)	204,928
batch_normalization_31 (BatchNormalization)	(None, 90, 90, 128)	512
max_pooling2d_21 (MaxPooling2D)	(None, 45, 45, 128)	0
dropout_31 (Dropout)	(None, 45, 45, 128)	0
conv2d_22 (Conv2D)	(None, 45, 45, 256)	295,168
batch_normalization_32 (BatchNormalization)	(None, 45, 45, 256)	1,024
max_pooling2d_22 (MaxPooling2D)	(None, 22, 22, 256)	0
dropout_32 (Dropout)	(None, 22, 22, 256)	0
conv2d_23 (Conv2D)	(None, 22, 22, 512)	1,180,160
batch_normalization_33 (BatchNormalization)	(None, 22, 22, 512)	2,048
max_pooling2d_23 (MaxPooling2D)	(None, 11, 11, 512)	0
dropout_33 (Dropout)	(None, 11, 11, 512)	0
flatten_5 (Flatten)	(None, 61952)	0
dense_15 (Dense)	(None, 128)	7,929,984
batch_normalization_34 (BatchNormalization)	(None, 128)	512
activation_10 (Activation)	(None, 128)	0
dropout_34 (Dropout)	(None, 128)	0
dense_16 (Dense)	(None, 256)	33,024
batch_normalization_35 (BatchNormalization)	(None, 256)	1,024
activation_11 (Activation)	(None, 256)	0
dropout_35 (Dropout)	(None, 256)	0
dense_17 (Dense)	(None, 36)	9,252

Figura 3.14.1 si 3.14.2 Rezumatul primului model

Pentru compilarea modelului am folosit optimizatorul Adam.

```
from tensorflow.keras.optimizers import Adam
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

Figura 3.15. Compilarea modelului

Un optimizator este un algoritm care ajustează greutatea rețelei neuronale pentru a minimiza funcția de pierdere.

Adam (Adaptive Moment Estimation) este un optimizator avansat care combină avantajele optimizatorilor AdaGrad și RMSProp. Este popular datorită convergenței rapide și stabilității.

Hiperparametri folosiți de Adam:

learning\_rate: Rata de învățare, default este 0.001.

beta\_1: Coeficientul pentru estimarea primului moment (media), default este 0.9.

beta\_2: Coeficientul pentru estimarea celui de-al doilea moment (varianța), default este 0.999.

epsilon: Un termen mic pentru stabilitate numerică, default este 1e-7.

Pentru calcularea funcției de pierdere am folosit Categorical Crossentropy (entropie încrucișată categorisită) este folosită pentru probleme de clasificare multi-clasă. Aceasta măsoară diferența dintre distribuția probabilităților prezise de model și distribuția reală.

Formula pentru entropia încrucișată categorisită este:

$$L = - \sum_{i=1}^N y_i \log(p_i)$$

Formula 3.5. calcularea funcției de pierdere

unde  $y_i$  este eticheta reală (valoarea adevărată) și  $p_i$  este probabilitatea prezisă pentru clasa respectivă.

Metrica folosită pentru a evalua performanța modelului este acuratetea. În acest caz, este folosită acuratețea, care reprezintă proporția de predicții corecte din totalul de predicții.

Înainte de antrenarea efectivă a modelului am introdus două clase: EarlyStopping și ModelCheckpoint. EarlyStopping și ModelCheckpoint sunt două clase din Keras care oferă funcționalități pentru monitorizarea și controlul antrenamentului modelului.

EarlyStopping este un callback care oprește antrenamentul modelului dacă performanța pe setul de validare nu se îmbunătățește după un anumit număr de epoci. Acest lucru previne supra antrenarea modelului și economisește timp și resurse.

Acesta monitorizează parametrul `val_accuracy`, iar dacă acesta nu se îmbunătățește după perioada de răbdare de 20 de epoci, antrenamentul se va opri timpuriu. `val_accuracy` este acuratețea calculată pe setul de date de validare la sfârșitul fiecărei epoci de antrenament.

La final modelul va restabili greutatea din epoca cu cea mai bună performanță pe setul de validare. Acest lucru asigură că modelul final, după antrenare, are cele mai bune greutăți în funcție de performanța de validare.

```
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint
early_stopping = EarlyStopping(monitor='val_accuracy', patience=20, restore_best_weights=True)
checkpoint = ModelCheckpoint('best_model.keras', monitor='val_accuracy', save_best_only=True)
```

Figura 3.16. Metode pentru îmbunătățirea performanței și economisirea resurselor

ModelCheckpoint este un callback care salvează modelul după fiecare epocă de antrenament. Acesta poate fi configurat să salveze doar cel mai bun model, în funcție de o metrică specificată.

`Best_model.keras` este numele fișierului în care va fi salvat modelul cu cea mai mare acuratețe pe setul de validare.

Callback-ul `save_best_only` va salva doar modelul care are cea mai bună performanță (cea mai mare acuratețe pe setul de validare).

Acest lucru economisește spațiu de stocare și asigură că doar cel mai performant model este salvat.

Pentru algoritmul de antrenare am folosit un număr mare de epoci pentru a permite rețelei să își dezvolte capacitatea de a înțelege imaginile, dar am folosit EarlyStopping pentru a asigura că odată ce am ajuns la cea mai bună valoare, aceasta nu continuă să epuizeze resurse fără a oferi rezultate.

```
ep = 200
history = model.fit(data_train, validation_data=data_val, epochs=ep, batch_size=32, verbose=1, callbacks=[early_stopping, checkp
```

Figura 3.17.

Numărul de epoci de antrenament este setat la 200, ceea ce înseamnă că întregul set de date de antrenament va fi procesat de model de 200 de ori. O epocă reprezintă o trecere completă prin toate datele de antrenament, permițând modelului să învețe și să ajusteze greutatea interne pe baza erorilor acumulate. Acest număr de epoci este crucial pentru a asigura că modelul are suficient timp pentru a învăța și a converge spre o soluție optimă, maximizând performanța pe setul de date de antrenament.

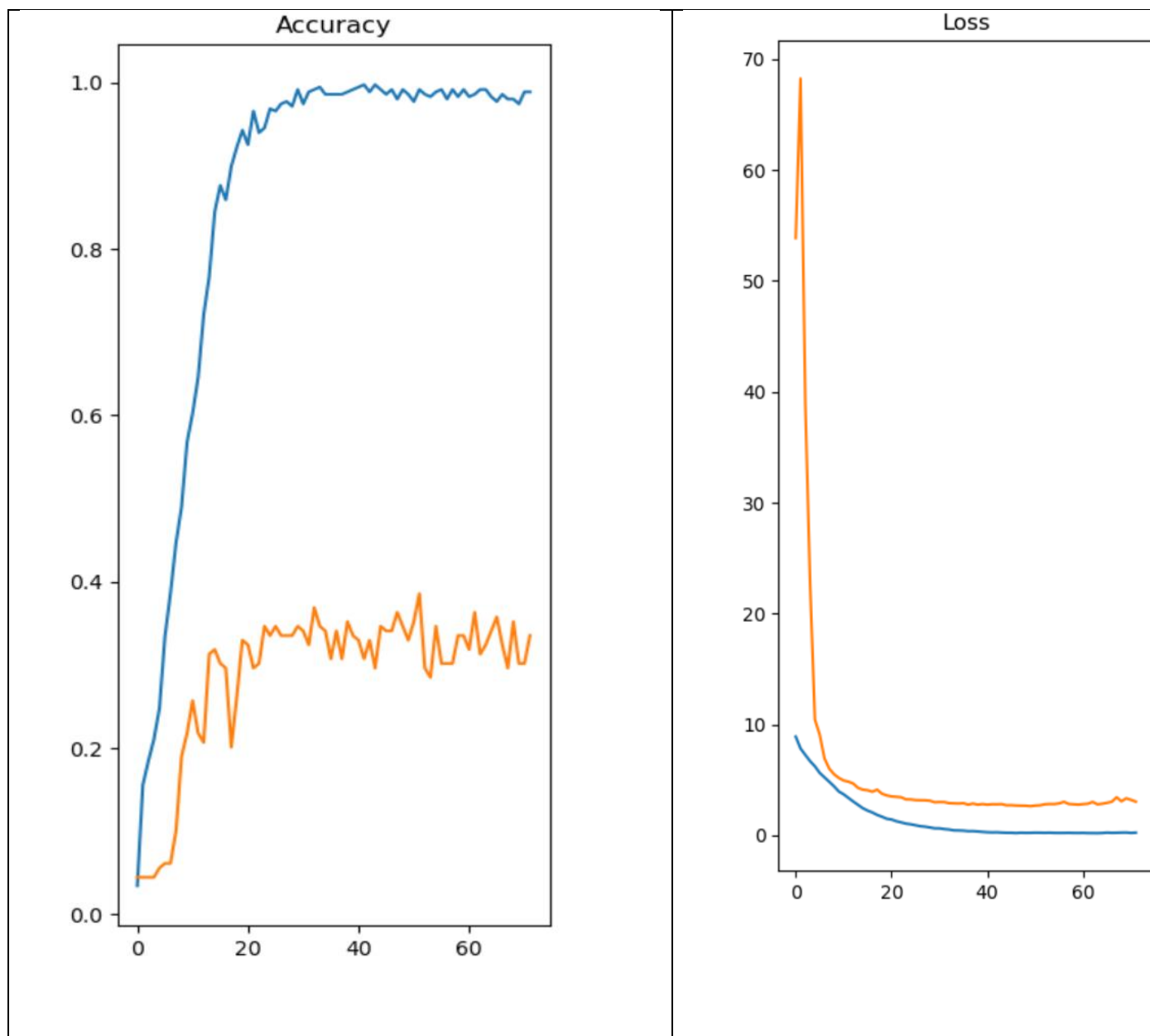
În procesul de antrenament, metoda *fit* este utilizată pentru a instrui modelul pe datele de antrenament și pentru a evalua performanța acestuia pe un set de date de validare, care nu a fost utilizat în timpul antrenamentului. Această metodă returnează un obiect de tip *History*, care conține date detaliate despre performanța modelului pentru fiecare epocă de antrenament, inclusiv pierderea (loss) și acuratețea pentru ambele seturi de date. Aceste informații sunt extrem de valoroase pentru monitorizarea progresului modelului și pentru ajustarea parametrilor în timp real, dacă este necesar.

În acest caz, datele de antrenament vor fi procesate în loturi de 32 de exemple, cunoscute și sub denumirea de *batch-uri*. Alegerea dimensiunii batch-ului este un factor important care influențează performanța și viteza de antrenare a modelului. Un batch mai mic necesită mai puțină memorie, ceea ce poate fi avantajos atunci când se lucrează cu resurse limitate, cum ar fi în cazul GPU-urilor cu memorie restrânsă. Cu toate acestea, batch-urile mai mici pot necesita un număr mai mare de epoci pentru a atinge convergența, deoarece modelul vede mai puține date în fiecare trecere, ceea ce poate duce la un proces de învățare mai lent.

Modelul a fost configurat cu atenție pentru a maximiza performanța pe datele de validare, asigurându-se astfel că modelul final nu doar că învață bine pe datele de antrenament, dar și că generalizează eficient pe datele noi. Acuratețea de antrenare a modelului ajunge la 100%, ceea ce indică faptul că modelul a reușit să învețe foarte bine din setul de date de antrenament. Cu toate acestea, acuratețea pe setul de validare atinge o valoare maximă de 39.11%, sugerând că există o posibilă discrepanță între performanța pe setul de antrenament și capacitatea modelului de a generaliza pe date noi. Acest lucru poate indica fenomenul de supraadaptare (overfitting), unde modelul devine foarte bun la a recunoaște tiparele din setul de antrenament, dar nu se descurcă la fel de bine cu datele necunoscute.

Pentru a facilita o vizualizare mai ușoară și mai clară a datelor și a procesului de antrenare, vom folosi un grafic (plot). Acest grafic va afișa evoluția pierderii și acurateței atât pentru setul de antrenament, cât și pentru cel de validare, pe parcursul celor 200 de epoci. Vizualizarea acestor date într-un format grafic permite identificarea rapidă a trendurilor și a potențialelor probleme, cum ar fi supraadaptarea, și oferă o bază solidă pentru ajustarea ulterioară a hiperparametrilor modelului.

În graficele următoare se poate observa evoluția hiperparametrilor modelului, acestea sunt reprezentate pe culori: albastru reprezentând acuratețea și pierderea pentru setul de antrenare, iar cu portocaliu pentru setul de validare.



Figurile 3.18, 3.19 acuratetea si functia de pierdere a modelului

În cazul funcției de pierdere, se poate observa o scădere semnificativă a ambelor valori, atât pentru setul de antrenare, cât și pentru cel de validare, pe măsură ce modelul progresează în timpul antrenamentului. Este de remarcat faptul că valoarea funcției de pierdere în cazul setului de antrenare se apropie de zero într-un ritm mai rapid comparativ cu cea din setul de validare, acest lucru fiind atribuit diferenței de acuratețe dintre cele două seturi de date. Pe măsură ce modelul învață, ambele valori ale funcției de pierdere scad considerabil față de valorile lor inițiale, indicând o îmbunătățire a performanței modelului. Această scădere continuă sugerează că modelul devine din ce în ce mai precis în predicțiile sale pe setul de antrenament, deși menținerea unei performanțe bune și pe setul de validare este esențială pentru a asigura capacitatea de generalizare a modelului.

Ultimul pas pentru finalizarea modelului este crearea algoritmului de predicție.

```
image='me5.jpg'
image=tf.keras.utils.load_img(image, target_size=(img_height,img_width))
img_arr=tf.keras.utils.array_to_img(image)
img_bat=tf.expand_dims(img_arr,0)

predict=model.predict(img_bat)

1/1 ————— 0s 33ms/step

score=tf.nn.softmax(predict)

plt.imshow(image)
print('Person in image is {} with accuracy of {:.2f}'.format(data_categories[np.argmax(score)],np.max(score)*100))

Person in image is m25-26% with accuracy of 3.56
```

Figura 3.20. Algoritmul de predicție

Imaginea este inițial convertită într-un array NumPy, ceea ce reprezintă un pas esențial în pregătirea imaginii pentru prelucrare ulterioară. Acest array NumPy servește drept format standard pentru manipularea și analiza datelor în Python, permițând efectuarea eficientă a operațiilor matematice și algoritmice necesare.

Ulterior, folosind funcția `tf.expand_dims`, se adaugă o dimensiune suplimentară la array-ul imaginii, transformând astfel imaginea individuală într-un lot (batch) care conține o singură imagine. Acest pas este crucial deoarece modelul de rețea neuronală este antrenat și configurat pentru a primi ca intrare un lot de imagini, nu doar o singură imagine. Prin adăugarea acestei dimensiuni suplimentare, se asigură compatibilitatea formatului imaginii cu cerințele de intrare ale modelului, permițând astfel prelucrarea corectă a datelor în timpul predicției.

După ce imaginea este pregătită corespunzător, metoda `predict` este utilizată pentru a aplica modelul pre-antrenat asupra imaginii transformate. Această metodă generează predicții sub forma unui array care conține probabilitățile asociate fiecărei clase posibile. Pentru a interpreta rezultatele, sunt utilizate funcții auxiliare care facilitează identificarea etichetelor claselor corespunzătoare fiecărei probabilități calculate.

Ulterior, funcția `score` aplică funcția de activare `softmax` la rezultatele brute ale predicției modelului. Funcția `softmax` este esențială deoarece convertește scorurile brute ale predicției în probabilități, asigurându-se că valorile se însumează la 1 și sunt ușor de interpretat. Clasa cu cea mai mare probabilitate este determinată folosind funcția `np.argmax`, care returnează indicele clasei cu probabilitatea maximă.

După ce predicția este realizată și interpretată, rezultatul este printat alături de imaginea originală la care se face referire. Acest proces complet permite nu doar efectuarea unei predicții precise, ci și vizualizarea rezultatului într-un context ușor de înțeles, facilitând analiza și evaluarea performanței modelului în raport cu imaginea testată. Acest lanț de operații, de la pregătirea imaginii până la afișarea rezultatului, ilustrează clar modul în care un model de rețea neuronală convoluțională poate fi utilizat pentru a prezice cu precizie clasa unei imagini.

## Implementarea celui de al doilea model:

Pentru a îmbunătăți performanțele și rezultatele primului model am decis implementarea unui al doilea model. Acesta este asemănător, dar cu câteva schimbări definitorii: augmentările aplicate prin intermediul ImageDataGenerator sunt mai agresive, și sunt adăugate două noi metode de a îmbunătăți acuratețea, cross validation și class weights.

Cross-Validation este o tehnică esențială în evaluarea performanței unui model de învățare automată, deoarece permite o estimare mai robustă și fiabilă a capacității modelului de a generaliza pe date noi, care nu au fost utilizate în timpul antrenamentului. Această tehnică implică împărțirea setului de date într-un număr de sub-seturi, ceea ce asigură o evaluare mai detaliată și diversificată a modelului. Scopul principal al cross-validation este să verifice cât de bine va funcționa modelul atunci când este aplicat pe date noi, neprocesate anterior, prevenind astfel riscul de supraadaptare (overfitting).

Una dintre cele mai utilizate metode de cross-validation este K-Fold Cross-Validation. În acest caz, setul de date este împărțit în K sub-seturi, cunoscute și sub numele de "folds". Modelul este antrenat K ori, de fiecare dată folosind K-1 fold-uri pentru antrenament și un fold diferit pentru validare. Aceasta înseamnă că fiecare fold este utilizat o singură dată ca set de validare, în timp ce restul fold-urilor sunt folosite pentru antrenare. Astfel, întregul set de date este folosit atât pentru antrenament, cât și pentru validare, ceea ce duce la o evaluare mai precisă și completă a modelului.

La finalul procesului de cross-validation, performanța modelului este evaluată pentru fiecare dintre cele K fold-uri, iar rezultatele sunt combinate pentru a calcula o medie a performanțelor. Această medie oferă o estimare mult mai robustă și fiabilă a capacității modelului de a generaliza, spre deosebire de evaluarea tradițională, care utilizează un singur set de validare. În acest studiu, KFold a fost configurat cu un număr de 5 fold-uri (`n_splits=5`), ceea ce înseamnă că setul de date a fost împărțit în 5 sub-seturi egale. Modelul a fost antrenat de 5 ori, utilizând de fiecare dată 4 fold-uri pentru antrenament și unul diferit pentru validare.

În fiecare iterație a procesului, seturile de antrenament și validare sunt generate folosind metoda `flow_from_directory`, care permite crearea de imagini augmentate, crescând astfel diversitatea setului de date și îmbunătățind capacitatea modelului de a generaliza.

După ce modelul este antrenat pe fold-ul curent, acesta este validat pe setul de validare corespunzător. În timpul fiecărei iterații, se evaluează acuratețea modelului pe setul de validare, iar modelul care obține cea mai bună performanță pe acest set este salvat pentru a fi utilizat ulterior în predicții.



```

train_gen = data_train_gen.flow_from_directory(
    data_train_path,
    target_size=(img_height, img_width),
    batch_size=batch_size,
    class_mode='categorical',
    shuffle=True
)
val_gen = data_val_gen.flow_from_directory(
    data_val_path,
    target_size=(img_height, img_width),
    batch_size=batch_size,
    class_mode='categorical',
    shuffle=True
)

```

Figura 3.21. crearea si generarea foldurilor

După finalizarea procesului de cross-validation, se calculează media performanțelor pentru toate fold-urile, oferind astfel o estimare finală a performanței modelului pe date nevăzute. Această medie nu doar că ajută la determinarea cât de bine va performa modelul pe date complet noi, ci și reduce riscul de supraadaptare, deoarece modelul este validat pe sub-seturi diferite ale datelor, evitându-se astfel dependența excesivă de un singur set de date. În concluzie, cross-validation oferă o evaluare robustă a modelului, asigurând rezultate mai stabile și mai puțin influențate de o împărțire specifică a datelor. Această abordare permite obținerea unui model care este nu doar precis, ci și capabil să performeze bine în situații reale, pe date noi și variate.

Class weights sunt esențiale în antrenarea modelelor de învățare automată, mai ales atunci când există un dezechilibru semnificativ între clase în setul de date. În situațiile în care anumite clase sunt mult mai puțin frecvente decât altele, modelul poate dezvolta o tendință de a favoriza clasele dominante sau mai frecvente, neglijând astfel clasele mai rare. Această tendință poate duce la o performanță slabă pe clasele sub-reprezentate, ceea ce poate afecta negativ capacitatea modelului de a generaliza și de a face predicții corecte pentru toate categoriile.

Pentru a contracara acest dezechilibru, se folosesc ponderi specifice pentru fiecare clasă, cunoscute sub numele de class weights. Prin atribuirea unor ponderi diferite fiecărei clase, modelul este „forțat” să acorde mai multă atenție greșelilor făcute pe clasele sub-reprezentate. Astfel, greșelile pe aceste clase sunt penalizate mai sever, ceea ce determină modelul să învețe mai eficient din exemplele mai rare și să îmbunătățească acuratețea pe aceste clase.

În cadrul procesului de cross-validation, ponderile claselor sunt calculate separat pentru fiecare fold, pe baza distribuției claselor din setul de date de antrenament. Funcția `compute_class_weight` din biblioteca `sklearn.utils.class_weight` este utilizată pentru a calcula aceste ponderi. Când se folosește parametrul `class_weight='balanced'`, ponderile sunt calculate astfel încât clasele mai rare să aibă o pondere mai mare, iar clasele mai frecvente să aibă o pondere mai mică. Aceasta înseamnă că modelul va fi mai puțin predispus să ignore clasele sub-reprezentate și va acorda o atenție proporțională fiecărei clase în parte.

```
labels = train_gen.classes
class_weights = compute_class_weight(class_weight='balanced', classes=np.unique(labels), y=labels)
class_weights = dict(enumerate(class_weights))
```

Figura 3.22. calcularea greutatilor pentru fiecare clasa

Aceste ponderi calculate sunt apoi aplicate în timpul antrenamentului modelului, utilizând parametrul `class_weight` în funcția `fit`. Prin aplicarea acestor ponderi, modelul este instruit să trateze cu mai multă importanță greșelile făcute pe clasele sub-reprezentate, reducând riscul de a dezvolta un bias puternic în favoarea claselor dominante. Acest lucru este deosebit de important în situațiile în care există un dezechilibru pronunțat între clase, cum ar fi într-un set de date de clasificare binară în care 90% dintre exemple aparțin unei clase și doar 10% celelalte.

Prin aplicarea ponderilor, modelul este „forțat” să învețe din exemplele sub-reprezentate, ceea ce duce la o îmbunătățire a performanței pe aceste clase. Această tehnică nu doar că ajută la obținerea unui model mai echilibrat, dar contribuie și la o performanță globală mai bună. În acest fel, se evită bias-ul în favoarea claselor frecvente, asigurându-se că modelul ia în considerare importanța fiecărei clase, inclusiv a celor mai puțin frecvente.

Un alt avantaj al utilizării ponderilor pentru clase este creșterea recall-ului pentru clasele rare. Deoarece modelul este încurajat să fie mai atent la aceste exemple, probabilitatea ca acestea să fie clasificate corect crește, îmbunătățind astfel performanța modelului pe aceste clase. În concluzie, utilizarea ponderilor pentru clase în procesul de antrenare este o strategie esențială pentru a obține un model de învățare automată care să fie nu doar precis, dar și echilibrat și capabil să generalizeze corect pe toate clasele, indiferent de frecvența acestora în setul de date.

```
data_train_gen = ImageDataGenerator(
    rescale=1.0/255,
    rotation_range=10,
    zoom_range=0.2,
    horizontal_flip=True,
    brightness_range=[0.8, 1.2],
)
```

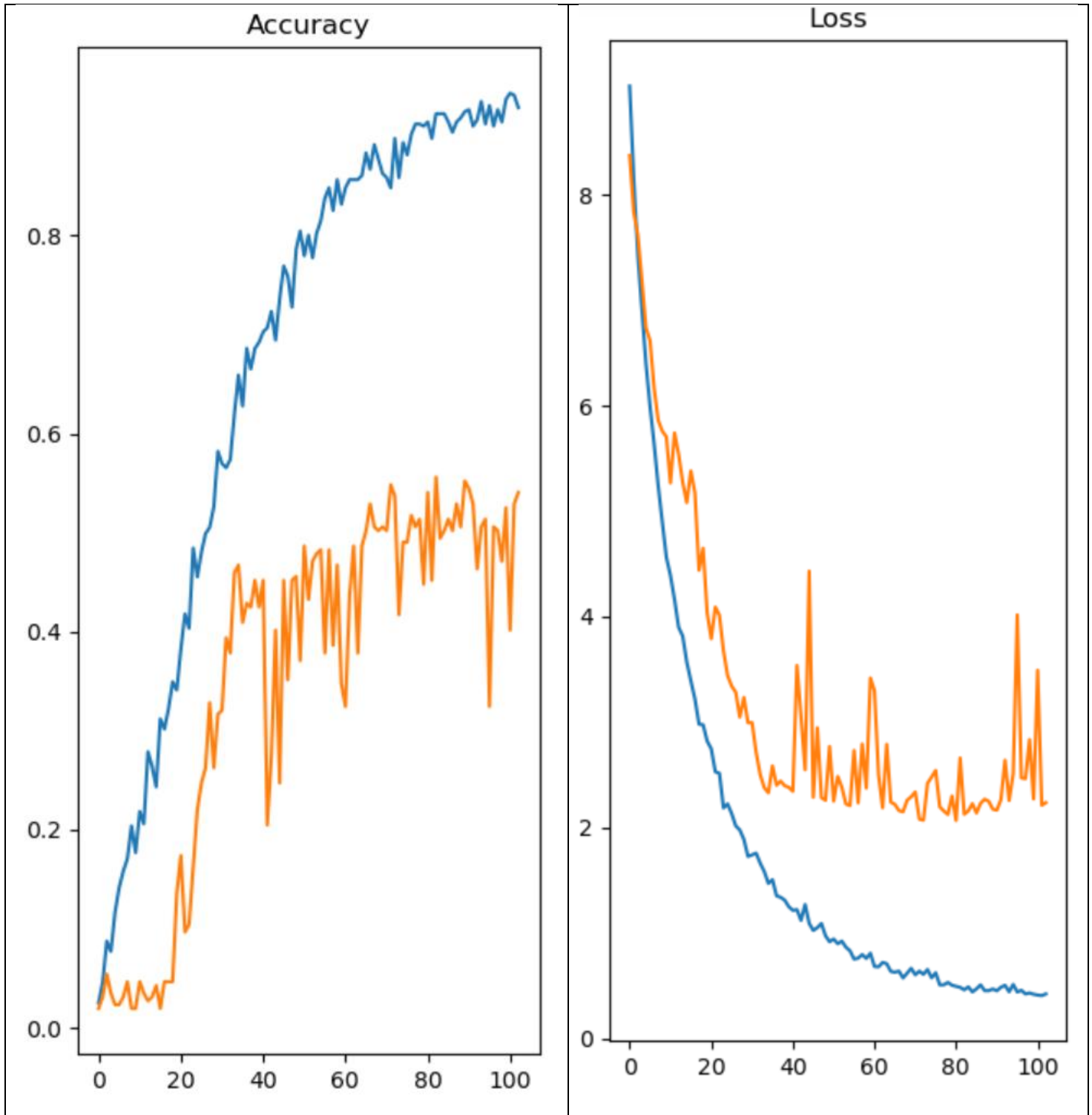
Figura 3.23. implementarea de noi augmentari

Acest set de augmentări oferă un echilibru mai adecvat între diversificarea datelor și păstrarea caracteristicilor esențiale ale imaginilor, ceea ce poate duce la o performanță mai bună a modelului. Prin limitarea `rotation_range` la 10 grade și adăugarea `brightness_range`, setul de parametri generează variații subtile, dar realiste, care imită condițiile de iluminare și poziționare ușor diferite, fără a distorsiona semnificativ structura originală a imaginii. Acest aspect este crucial pentru modele care depind de detalii fine, cum ar fi detectarea procentului grăsimii corporale.

În contrast, setul original de augmentări include schimbări mai drastice de poziție (`width_shift_range` și `height_shift_range`) și de unghi (`rotation_range` de 15 grade), ceea ce poate introduce distorsiuni nenaturale care ar putea confunda modelul, mai ales dacă

detaliile spațiale sunt critice pentru sarcina de clasificare. În plus, `fill_mode='nearest'` poate genera artefacte vizuale la marginile imaginii, reducând astfel acuratețea predicțiilor. În concluzie, primul set de augmentări este preferabil pentru a menține integritatea vizuală esențială și pentru a îmbunătăți capacitatea modelului de a generaliza.

Pentru vizualizarea rezultatelor celui de al doilea model vom folosi din nou un grafic. Cu albastru putem observa graficele corespunzătoare setului de antrenare, iar cu portocaliu cele corespunzătoare setului de validare



Se pot observa cu usurinta schimbarile fata de primul model. In timp ce acuratetea setului de antrenare ajunge la valori asemanatoare cu cele precedente, atingand o valoare de 97%, acuratetea setului de validare a crescut semnificativ. De la 39.11% am ajuns la valoarea de 56.78%, ceea ce indica o crestere a acuratetii de validare de 18%.

In cazul graficului functiilor de pierdere se pot observa din nou imbunatatiri fata de modelul initial. In primul rand graficul corespunzator setului de antrenare atinge valori aproape de zero, iar cel corespunzator setului de validare nu mai are spikeul initial, ci porneste din valori asemanatoare celui de validare. Acesta de asemenea scade in valori semnificativ fata de cel precedent, de la 5 la 2.1.

Figurile 3.24 si 3.25 graficele pentru performantele celui de al doilea system

Rezumatul celui de al doilea model:

Layer (type)	Output Shape	Param #
conv2d_60 (Conv2D)	(None, 180, 180, 64)	1,792
batch_normalization_90 (BatchNormalization)	(None, 180, 180, 64)	256
max_pooling2d_60 (MaxPooling2D)	(None, 90, 90, 64)	0
conv2d_61 (Conv2D)	(None, 90, 90, 128)	204,928
batch_normalization_91 (BatchNormalization)	(None, 90, 90, 128)	512
max_pooling2d_61 (MaxPooling2D)	(None, 45, 45, 128)	0
conv2d_62 (Conv2D)	(None, 45, 45, 256)	295,168
batch_normalization_92 (BatchNormalization)	(None, 45, 45, 256)	1,024
max_pooling2d_62 (MaxPooling2D)	(None, 22, 22, 256)	0
conv2d_63 (Conv2D)	(None, 22, 22, 512)	1,180,160
batch_normalization_93 (BatchNormalization)	(None, 22, 22, 512)	2,048
max_pooling2d_63 (MaxPooling2D)	(None, 11, 11, 512)	0
flatten_15 (Flatten)	(None, 61952)	0
dense_45 (Dense)	(None, 128)	7,929,984
batch_normalization_94 (BatchNormalization)	(None, 128)	512
activation_30 (Activation)	(None, 128)	0
dropout_30 (Dropout)	(None, 128)	0
dense_46 (Dense)	(None, 256)	33,024
batch_normalization_95 (BatchNormalization)	(None, 256)	1,024

activation_31 (Activation)	(None, 256)	0
dropout_31 (Dropout)	(None, 256)	0
dense_47 (Dense)	(None, 36)	9,252

Total params: 28,973,678 (110.53 MB)

Trainable params: 9,656,996 (36.84 MB)



Non-trainable params: 2,688 (10.50 KB)

Optimizer params: 19,313,994 (73.68 MB)

Figura 3.26. Rezumatul si parametrii totali ai celui de al doilea model

Se poate observa diferența de mărime si complexitate a modelului. Numărul total de parametri este triplu față de primul model. Aici au început să apară limitări de hardware. Timpul de antrenare al rețelei pentru un singur ciclu depășeste durata de 3 ore.

### 3.3. Validarea si testarea modelelor implementate

<p>Model 1</p> <p>Persoana 1:</p> <p>Sex: Masculin</p> <p>Procent de grasime: 21%</p> <p>Predictie:</p> <p>Person in image is m19-20% with accuracy of 7.21</p> 	<p>Model 1</p> <p>Persoana 2:</p> <p>Sex: Feminin</p> <p>Procent de grasime: 22%</p> <p>Predictie:</p> <p>Person in image is f22-24% with accuracy of 5.00</p> 
<p>Predicția are ca marja de eroare de grăsime 1%. Modelul performează bine , având in vedere ca odată cu creșterea procentului de grăsime corporala, diferența dintre procente este din ce in ce mai mica.</p>	<p>Predicția are marja de eroare practica de 0%. Modelul performează foarte bine. Acest lucru se datorează si calității luminii si a poziției direct spre camera din imagine.</p>

Model 1

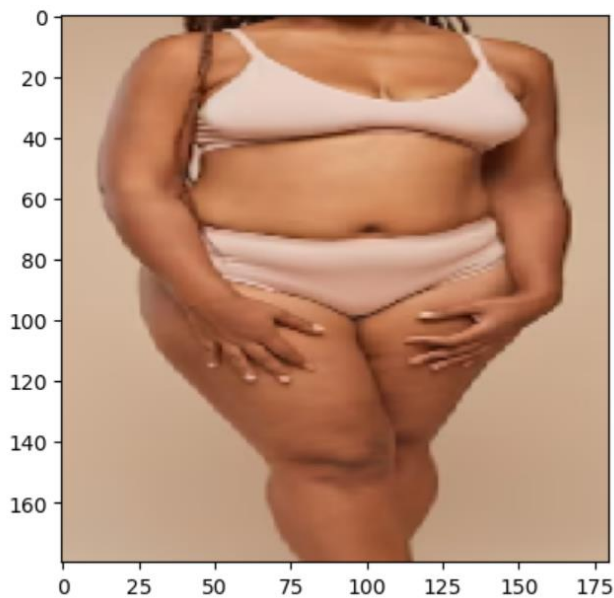
Persoana 3:

Sex: Feminin

Procent de grasime: 44%

Predictie:

Person in image is f40-42% with accuracy of 3.23



Model 1

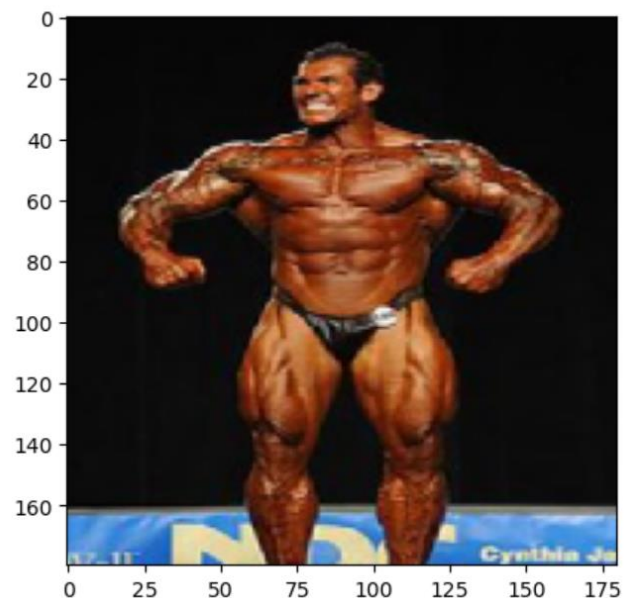
Persoana 4:

Sex: Masculin

Procent de grasime: 5%

Predictie:

Person in image is m5-6% with accuracy of 4.47



Predictia are marja de eroare de 2%. Modelul performeaza bine considerand procentul de grasime relativ ridicat al persoanei. Fapt care reduce din acuratețe

Predictia are marja de eroare practic 0%. Modelul performeaza foarte bine



Model 1

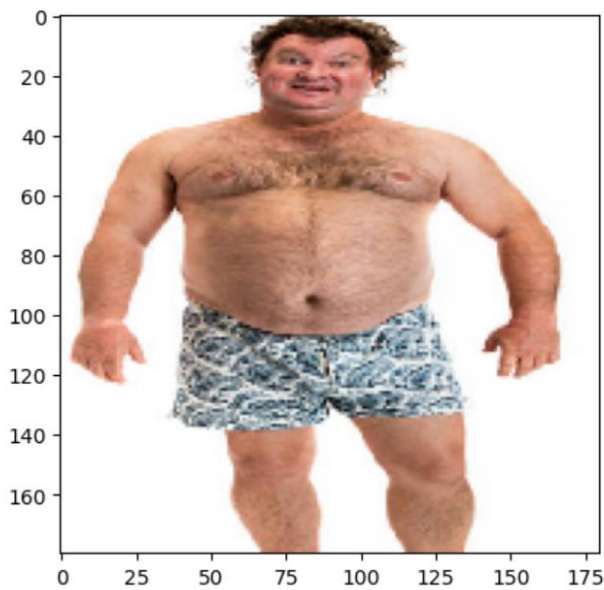
Persoana 5:

Sex: Masculin

Procent de grăsime: 38%

Predicție:

Person in image is m33-34% with accuracy of 3.80



Predicția are marja de eroare de aproximativ 4%. Modelul performează bine luând in considerare procentul de grăsime relativ ridicat. O parte din acuratețea predicției se datorează luminozității bune din imagine.

Model 1

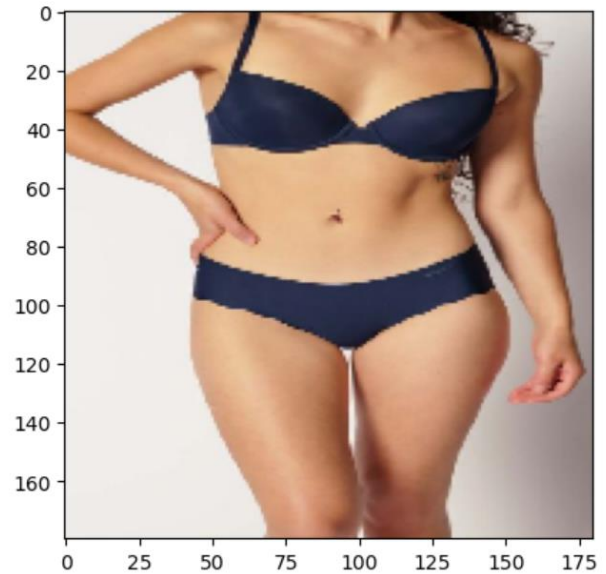
Persoana 6:

Sex: Feminin

Procent de grăsime: 20%

Predicție:

Person in image is f22-24% with accuracy of 4.35



Predicția are marja de eroare de aproximativ 2%. Modelul performează foarte bine.



Model 1

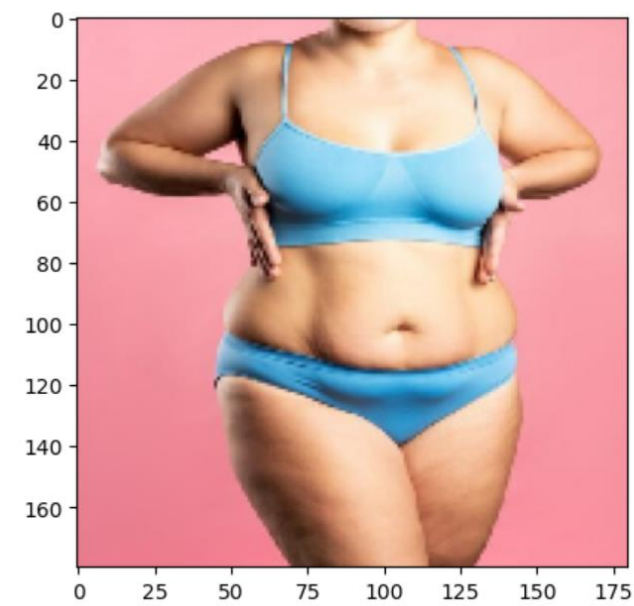
Persoana 7:

Sex: Feminin

Procent de grăsime: 45%

Predicție:

Person in image is f40-42% with accuracy of 4.58



Predicția are marja de eroare de aproximativ 3%. Modelul performează destul de bine

Model 1

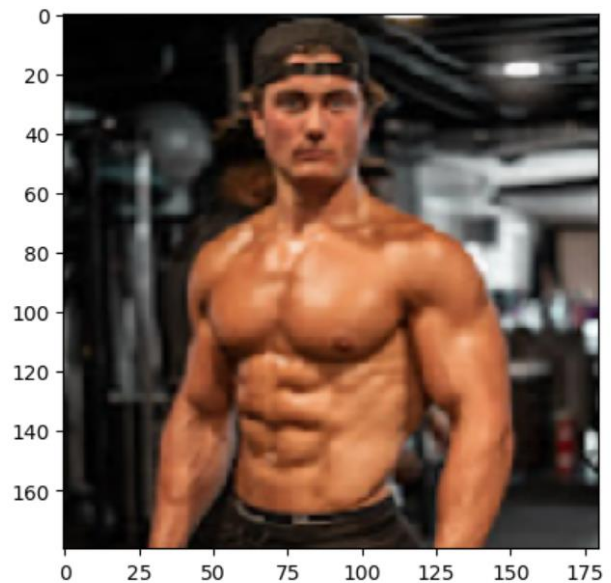
Persoana 8:

Sex: Masculin

Procent de grăsime: 9%

Predicție:

Person in image is m7-8% with accuracy of 4.33



Predicția are marja de eroare de aproximativ 1%. Performanța modelului este bună, dar considerând că diferențele dintre procente sunt semnificativ mai mari printre procentele scăzute, observăm că marja de eroare de 1% de aici corespunde proporțional marjei de eroare de 3% care apare la procentele ridicate.

Model 1

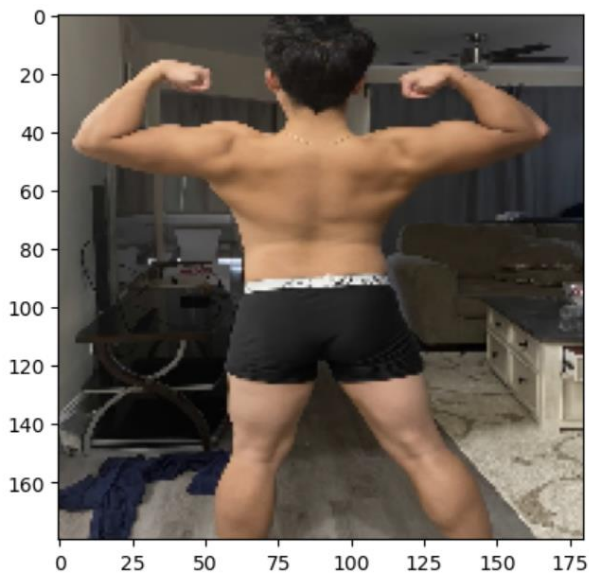
Persoana 9:

Sex: Masculin

Procent de grăsime: 19%

Predicție:

Person in image is m21-22% with accuracy of 4.24



Marja de eroare a predicției este de aproximativ 2%. Putem observa ca modelul performa destul de bine si in recunoașterea de la spate.

Model 1

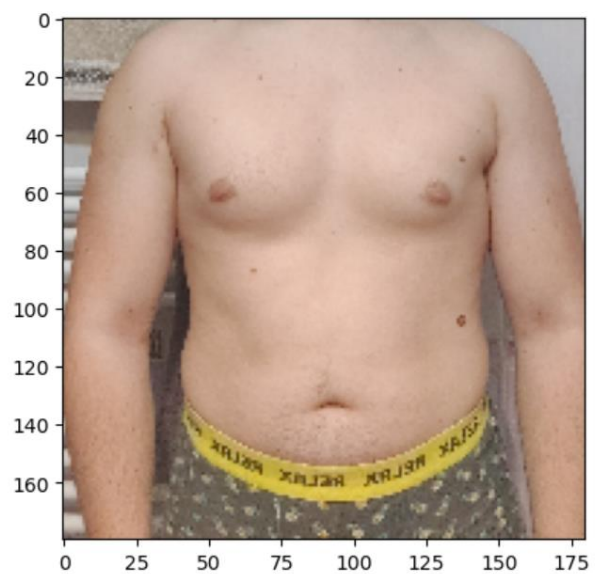
Persoana 10:

Sex: Masculin

Procent de grăsime : 27%

Predicție:

Person in image is m25-26% with accuracy of 3.45



Predicția este buna, având o marja de eroare de aproximativ 1%. Modelul performează foarte bine si din cauza luminii potrivite din imagine si a poziției orientate spre camera.

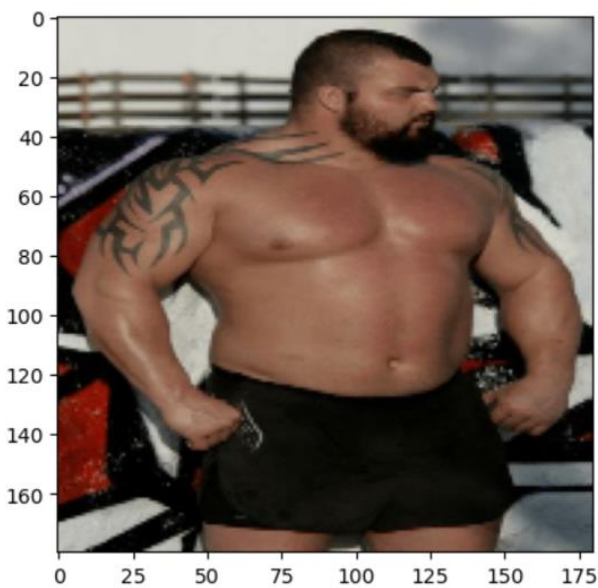
Model 1

Persoana 11

Procent de grasime: 28%

Predictie:

Person in image is m33-34% with accuracy of 3.23



Performanta modelului este decenta, marja de eroare fiind de aproximativ 5%. Considerând faptul ca odată cu creșterea procentului de grăsime corporala, diferențele vizuale dintre procente sunt din ce in ce mai mici.

Model 2

Persoana 11

Procent de grasime: 28%

Predictie:

The predicted class for the image is: ('m27-28%', 38.19549083709717)



Performanta acestui model este mai buna, având marja de eroare de sub 1%.

Model 1

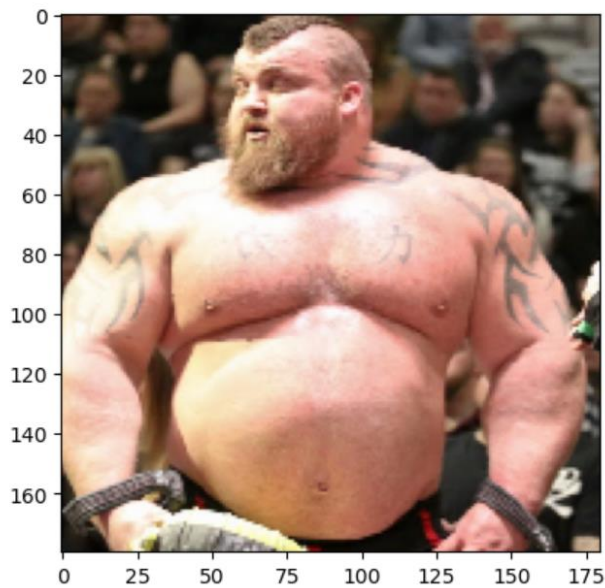
Persoana 12:

Sex: Masculin

Procent de grăsime : 44%

Predicție:

Person in image is m3% with accuracy of 3.90



In acest caz modelul este indus in eroare de zgomotul de pe fundal, si returnează o clasa întâmplătoare. Marja de eroare este de 40%, deci modelul nu performează bine de loc

Model 2

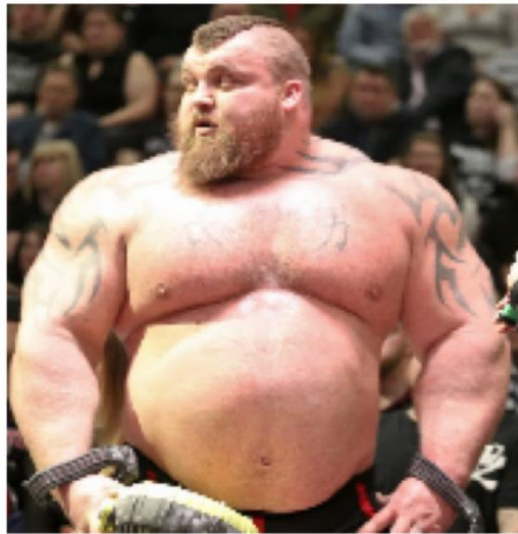
Persoana 12:

Sex: Masculin



Procent de grăsime : 44%

Predicție:

The predicted class for the image is: ('m41-42%', 83.34503769874573)



In cazul celui de al doilea model, marja de eroare este de 2%. Modelul performează foarte bine

<p>Model 1</p> <p>Persoana 13:</p> <p>Sex: Masculin</p> <p>Procent de grăsime : 10%</p> <p>Predicție:</p> <p>Person in image is m33-34% with accuracy of 7.21</p> 	<p>Model 2</p> <p>Persoana 13:</p> <p>Sex: Masculin</p> <p>Procent de grăsime : 10%</p> <p>Predicție:</p> <p>The predicted class for the image is: ('m11-12%', 51.3471782207489)</p> 
<p>Modelul are o marja de eroare de 23%, ceea ce din nou nu este bine de loc. Majoritatea pozelor cu persoane de sex masculin si 10% grăsime din setul de date sunt cu persoane cu mai multa masa musculara, iar modelul nu generalizează la fel de bine pe cum ar trebui.</p>	<p>Modelul are o marja de eroare de aproximativ 1%, ceea ce este foarte bine. Cel de al doilea model se descurca mai bine la interpretarea claselor subreprezentate datorita folosirii class weights</p>

## 4 Concluzii

### 4.1 Rezultate obținute

În această lucrare, am explorat dezvoltarea și implementarea unei rețele neuronale convoluționale pentru detectarea procentului de grăsime corporală din imagini. Am demonstrat că utilizarea rețelelor neuronale pentru estimarea grăsimii corporale poate oferi o metodă non-invazivă, comparabilă cu metodele tradiționale, precum măsurătorile antropometrice sau evaluările bioelectrice. Aceste metode, deși precise, pot fi costisitoare, incomode și dificil de accesat pentru toți utilizatorii, în timp ce soluția propusă poate fi integrată în aplicații accesibile și ușor de utilizat.

Pe parcursul experimentelor, am dezvoltat și testat două modele distincte. Primul model a atins o acuratețe de validare de 39.11%, în timp ce al doilea model a reușit să obțină o acuratețe de validare semnificativ îmbunătățită, de 57%. Această diferență între cele două modele subliniază importanța ajustării arhitecturii rețelei și a parametrilor de antrenare pentru a optimiza performanța modelului. De asemenea, arată că, deși primul model a fost capabil să generalizeze și să învețe din datele de antrenament, a prezentat limitări semnificative atunci când a fost testat pe date noi, în timp ce al doilea model a beneficiat de ajustări și optimizări care i-au permis să obțină rezultate mai bune.

Rețeaua neuronală dezvoltată a fost capabilă, pe parcursul a multiple teste, să demonstreze abilitățile sale de a învăța și de a generaliza pe imagini noi. În general, s-a observat o marjă de eroare de 1-3%, ceea ce indică o capacitate solidă de predicție. Totuși, modelul a întâmpinat dificultăți în prezența anumitor factori perturbatori, cum ar fi zgomotul mare de fundal, diferențele mari de luminozitate între valorile RGB, sau imaginile în care persoanele sunt poziționate nenatural sau nu sunt orientate corect spre cameră.

Acuratețea de validare relativ scăzută a primului model poate fi atribuită în mare parte dimensiunii reduse a setului de date. S-au încercat diverse metode de prevenire a supraînvățării, inclusiv Regularizarea L2, Dropout, EarlyStopping și augmentarea imaginilor din setul de antrenament. Deși aceste metode au reușit să crească acuratețea de validare și să reducă funcția de pierdere, nu au fost suficiente pentru a reduce diferența semnificativă dintre acuratețea de antrenare și cea de validare. Această problemă ar putea fi remediată în viitor prin extinderea setului de date, oferind astfel modelului mai multe exemple din care să învețe.

În ultimul test, chiar dacă rețeaua a fost indusă în eroare, ea a demonstrat capacitatea de a diferenția aspectul fizic al unei persoane, chiar și în condiții nefavorabile. Această caracteristică are implicații semnificative pentru utilizarea acestui model în aplicații destinate pierderii în greutate și monitorizării progresului într-un mod non-invaziv și intuitiv, fiind deosebit de utilă pentru persoanele care se confruntă cu obezitatea. Astfel, se poate observa potențialul considerabil al rețelelor neuronale și al învățării automate în dezvoltarea de soluții inovatoare pentru prevenirea obezității, diabetului și altor boli asociate.

### 4.2 Direcții de dezvoltare

Îmbunătățirea Setului de Date: Extinderea setului de date cu imagini care acoperă o gamă mai largă de diversitate demografică și fiziologică poate avea un impact semnificativ asupra

robusteții și capacității modelului de a generaliza la populații variate. În prezent, unul dintre principalele obstacole în dezvoltarea rețelelor neuronale pentru estimarea grăsimii corporale este lipsa de date suficiente și diversificate. O soluție viabilă ar fi extinderea setului de date pentru a include persoane din diferite grupe de vârstă, sexe, etnii și tipuri corporale. Acest lucru nu numai că ar îmbunătăți performanța modelului pe o gamă mai largă de imagini, dar ar reduce și riscul de supraînvățare, în care modelul se adaptează prea strâns la specificitățile unui set de date limitat. În plus, utilizarea unor seturi de date mai mari și mai variate ar facilita dezvoltarea unor modele care sunt capabile să facă predicții mai precise și mai fiabile, chiar și în condiții de variabilitate mare.

**Integrarea cu alte metode de învățare:** O altă direcție promițătoare este combinarea predicțiilor obținute de rețelele neuronale convoluționale (CNN) cu rezultate din alte metode de evaluare non-invazive, cum ar fi impedanța bioelectrică sau scanările DEXA. Integrarea acestor surse de date ar putea crește acuratețea estimărilor și ar permite o evaluare mai holistică a compoziției corporale. De exemplu, combinarea datelor vizuale cu măsurători bioelectrice ar putea compensa limitările fiecărei metode individuale, oferind astfel un model hibrid care ar putea fi mai robust și mai precis.

**Transfer Learning:** Explorarea tehnicilor de transfer learning este o altă modalitate prin care performanțele modelului pot fi îmbunătățite, mai ales atunci când seturile de date sunt limitate. Transferul de învățare implică utilizarea unor rețele neuronale pre-antrenate pe seturi de date mari, cum ar fi ImageNet, și ajustarea acestora pentru sarcina specifică de estimare a grăsimii corporale. Această abordare permite modelului să profite de cunoștințele acumulate din sarcini anterioare, ceea ce poate duce la îmbunătățirea rapidă a performanțelor, chiar și cu un set de date relativ mic pentru antrenare. Prin reutilizarea și ajustarea parametrilor învățați dintr-un set de date mare și diversificat, modelul poate capta mai eficient trăsăturile relevante pentru estimarea procentului de grăsime corporală.

**Combinarea datelor vizuale cu alte date relevante:** O altă oportunitate de îmbunătățire a estimării compoziției corporale constă în combinarea datelor vizuale cu alte surse de date, cum ar fi informații de la senzori de fitness sau date din istoricul medical. Utilizarea unor rețele neuronale multimodale, care pot procesa și integra multiple tipuri de date, ar permite o evaluare mai cuprinzătoare și precisă a sănătății corporale. De exemplu, datele de la senzori de fitness, cum ar fi ritmul cardiac, activitatea fizică sau somnul, combinate cu imagini ale corpului, ar putea oferi o perspectivă mai detaliată asupra stării generale de sănătate și ar putea îmbunătăți predicțiile modelului.

**Concluzie:** Studiul a demonstrat că rețelele neuronale convoluționale pot fi utilizate eficient pentru estimarea procentului de grăsime corporală din imagini. Cu toate acestea, există multiple direcții de dezvoltare care pot îmbunătăți și extinde această tehnologie. Prin îmbunătățirea seturilor de date, integrarea cu alte metode de măsurare și utilizarea tehnologiilor avansate, precum transfer learning și rețelele neuronale multimodale, putem continua să avansăm în domeniul evaluării non-invazive a sănătății corporale. Aceste eforturi ar putea contribui semnificativ la monitorizarea sănătății și prevenirea bolilor asociate cu compoziția corporală, oferind soluții mai accesibile și mai precise pentru o populație globală diversificată.



## 5 Bibliografie

[1] I-Ju Chen, Ming-Jung Lee, Bai-Cheng Jeng and Der-Bang Wu, "A novel prediction method for body fat by using Choquet integral with respect to L-measure and Gamma-support" 2009 International Conference on Machine Learning and Cybernetics, Baoding, China, 2009, pp. 3172-3176, doi: 10.1109/ICMLC.2009.5212802.

[2] F. Keivanian and N. Mehrshad, "Intelligent feature subset selection with unspecified number for body fat prediction based on binary-GA and Fuzzy-Binary-GA" 2015 2nd International Conference on Pattern Recognition and Image Analysis (IPRIA), Rasht, Iran, 2015, pp. 1-7, doi: 10.1109/PRIA.2015.7161651.

[3] L. Pradhan et al., "Feature Extraction from 2D Images for Body Composition Analysis," 2015 IEEE International Symposium on Multimedia (ISM), Miami, FL, USA, 2015, pp. 45-52, doi: 10.1109/ISM.2015.117.

[4] B. Chen, X. Gao, Q. Zheng and J. Wu, "Research on human body composition prediction model based on Akaike Information Criterion and improved entropy method" 2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), Datong, China, 2016, pp. 1882-1886, doi: 10.1109/CISP-BMEI.2016.7853024.

[5] Y. Lu, S. McQuade and J. K. Hahn, "3D Shape-based Body Composition Prediction Model Using Machine Learning" 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Honolulu, HI, USA, 2018, pp. 3999-4002, doi: 10.1109/EMBC.2018.8513261.

[6] F. Keivanian, R. Chiong and Z. Hu, "A Fuzzy Adaptive Binary Global Learning Colonization-MLP model for Body Fat Prediction" 2019 3rd International Conference on Bio-engineering for Smart Technologies (BioSMART), Paris, France, 2019, pp. 1-4, doi: 10.1109/BIOSMART.2019.8734215.

[7] Q. Wang, Y. Lu, X. Zhang and J. K. Hahn, "A Novel Hybrid Model for Visceral Adipose Tissue Prediction using Shape Descriptors" 2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Berlin, Germany, 2019, pp. 1729-1732, doi: 10.1109/EMBC.2019.8857092.

[8] D. H. S. Guerrero, C. S. Lapada, R. R. R. Necesito, J. B. Lazaro, A. N. Yumang and E. D. Dimaunahan, "Bioelectrical Impedance Analysis for the Prediction of Human Body Composition Using Wenner Algorithm" 2019 IEEE 11th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM), Laoag, Philippines, 2019, pp. 1-4, doi: 10.1109/HNICEM48295.2019.9073411.



- [9] E. Severeyn, S. Wong, H. Herrera, A. L. Cruz, J. Velásquez and M. Huerta, "Prediction of Abnormal Body Fat Percentage by Anthropometrics Parameters Using Receiver Operating Characteristic Curve" 2020 IEEE ANDESCON, Quito, Ecuador, 2020, pp. 1-6, doi: 10.1109/ANDESCON50619.2020.9272149.
- [10] Y. Lu, J. K. Hahn and X. Zhang, "3D Shape-Based Body Composition Inference Model Using a Bayesian Network" in IEEE Journal of Biomedical and Health Informatics, vol. 24, no. 1, pp. 205-213, Jan. 2020, doi: 10.1109/JBHI.2019.2903190.
- [11] G. Nandakumar, G. Srinivasan, H. Kim and J. Pi, "Comprehensive End-to-End Workflow for Visceral Adipose Tissue and Subcutaneous Adipose Tissue quantification: Use Case to improve MRI accessibility" 2020 IEEE 20th International Conference on Bioinformatics and Bioengineering (BIBE), Cincinnati, OH, USA, 2020, pp. 1060-1064, doi: 10.1109/BIBE50027.2020.00179.
- [12] L. Li, W. Huang and T. Zhang, "Prediction of visceral fat area of bioelectrical impedance based on ensemble learning" 2021 IEEE International Conference on Electronic Technology, Communication and Information (ICETCI), Changchun, China, 2021, pp. 438-443, doi: 10.1109/ICETCI53161.2021.9563573.
- [13] B. M. Rashed and N. Popescu, "Machine Learning Techniques for Medical Image Processing," 2021 International Conference on e-Health and Bioengineering (EHB), Iasi, Romania, 2021, pp. 1-4, doi: 10.1109/EHB52898.2021.9657673.
- [14] T. Lioner, D. E. Herwindiati and J. Hendryli, "Type of Training Recommendation Based on Body Fat Prediction Using LASSO Regression" 2022 IEEE 7th International Conference on Information Technology and Digital Applications (ICITDA), Yogyakarta, Indonesia, 2022, pp. 1-6, doi: 10.1109/ICITDA55840.2022.9971376.
- [15] J. Liu and H. Liu, "Research on Intelligent Abnormal Body Weight Monitoring Algorithm Based on Fat Depth Learning" 2022 Global Reliability and Prognostics and Health Management (PHM-Yantai), Yantai, China, 2022, pp. 1-7, doi: 10.1109/PHM-Yantai55411.2022.9942037.
- [16] N. Mahesh, P. B. Pati, K. Deepa and S. Yanan, "Body Fat Prediction using Various Regression Techniques" 2023 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI), Chennai, India, 2023, pp. 1-5, doi: 10.1109/ACCAI58221.2023.10200647.
- [17] M. D. Grady and T. M. Weller, "A Hybrid Skin, Fat, and Muscle Human Body Tissue-Mimicking Biological Phantom and Antenna Testbed" 2023 IEEE Wireless and Microwave Technology Conference (WAMICON), Melbourne, FL, USA, 2023, pp. 117-120, doi: 10.1109/WAMICON57636.2023.10124896.
- [18] E. D, U. K, A. S, G. R. A, M. M and G. P, "An Innovative Non-Invasive Approach to Personalized Nutrition and Metabolism Monitoring Using Wearable Sensors" 2023 International Conference on Intelligent Technologies for Sustainable Electric and Communications Systems (iTech SECOM), Coimbatore, India, 2023, pp. 54-58, doi: 10.1109/iTechSECOM59882.2023.10435320.

[19] A. Kunwar, S. Pandey, A. Pandey, Kemee and S. A. Farooq, "Exploring Regression Models and Optimization Techniques for Accurate Body Fat Prediction: A Comparative Approach" 2024 4th International Conference on Innovative Practices in Technology and Management (ICIPTM), Noida, India, 2024, pp. 1-6, doi: 10.1109/ICIPTM59628.2024.10563603.

[20] V. Mattsson et al., "Machine Learning-Powered Microwave Device for Local Body Composition Assessment" in *IEEE Sensors Journal*, vol. 24, no. 5, pp. 7030-7041, 1 March 2024, doi: 10.1109/JSEN.2023.3344581.

[21] G. Zhang and H. Lai, "QTN-MLP: Quantum Tensor Network-enhanced MLP for Medical Image Classification," 2024 27th International Conference on Computer Supported Cooperative Work in Design (CSCWD), Tianjin, China, 2024, pp. 2134-2139, doi: 10.1109/CSCWD61410.2024.10580373.