

**Universitatea
Transilvania
din Brașov**

**FACULTATEA DE INGINERIE ELECTRICĂ
ȘI ȘTIINȚA CALCULATOARELOR**

PROIECT DE DIPLOMĂ

Conducător științific:

Șef lucr. Dr. Ing Dănilă Adrian

Absolvent:

Cosferenț Roxana-Adelina

BRAȘOV, 2025

Departamentul Automatică și Tehnologia informației
Programul de studii: Automatică și informatică aplicată

COSFERENȚ Roxana-Adelina

Procesarea imaginilor în ecosistemul Python pentru analiza operelor de artă

Conducător științific:

Șef lucr. Dr. Ing. Dănilă Adrian


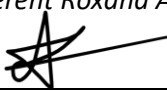
Brașov, 2025

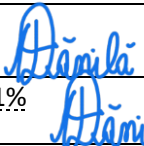
Cuprins

Lista de figuri și coduri sursă	7
Lista de acronime	8
1 Introducere	9
1.1 Stadiul actual al problemei abordate	9
1.2 Motivația alegerii proiectului	9
1.3 Scopul și obiectivele proiectului	9
1.4 Domeniul de aplicabilitate	10
1.5 Structura pe capitole.....	10
2 Noțiuni teoretice	11
2.1 Introducerea în procesarea imaginilor.....	11
2.2 Rețele neuronale convoluționale (CNN).....	11
2.3 Clasificarea imaginilor în funcție de stil	12
2.4 Tehnici in restaurarea imaginilor	13
2.4.1 Filtrarea spațială liniară și neliniară	14
2.4.2 Restaurarea în domeniul frecvenței.....	14
2.4.3 Tehnici de inpainting (completarea regiilor lipsă)	14
2.4.4 Eliminarea zgomotului cu modele de învățare automată	14
2.4.5 Restaurarea imaginilor utilizând tehnici de învățare profundă	15
2.5 Baze de date în aplicații cu imagini	16
2.6 Framework-uri utilizate.....	17
2.6.1 KERAS.....	17
2.6.2 OPEN CV.....	18
2.6.3 FLASK	18
2.6.4 SQLite	18
3 Clasificarea imaginilor artistice	19
3.1 Prelucrarea setului de date	19
3.2 Alegerea arhitecturii cnn.....	20
3.3 Încărcarea și pregătirea datelor	22
3.4 Antrenarea si validarea modelului	24
3.5 Evaluarea performanței	25
4 Restaurarea digitală și interfața web	29
4.1 Crearea bazei de date	29
4.2 Implementarea aplicației flask	30
4.3 Funcționalități disponibile.....	31
4.3.1 Vizualizarea imaginilor	33
4.3.2 Aplicarea de filtre.....	34

4.3.3	Afișarea stilului artistic	35
4.3.4	Salvarea/restaurarea imaginilor.....	37
4.3.5	Încărcarea unei imagini proprii	37
4.4	Elemente de interfață	38
4.5	Testare funcțională.....	39
5	Concluzii si direcții viitoare.....	43
5.1	Concluzii generale.....	43
5.2	Dificultăți întâmpinate	43
5.3	Posibile îmbunătățiri și extinderi ale proiectului	44
6	Bibliografie.....	46

FIȘA PROIECTULUI DE DIPLOMĂ

Universitatea Transilvania din Brașov	Proiect de diplomă nr.
Facultatea de Inginerie Electrică și Știința Calculatoarelor	
Departamentul de Automatică și informatică aplicată	Viza facultății
Programul de studii: Licenta	Anul universitar: 2024 – 2025
Candidat: Cosferenț Roxana-Adelina	Promoția: 2025
Conducător științific: Șef lucr. Dr. Ing. Dănilă Adrian	
PROIECT DE DIPLOMĂ	
Titlul lucrării: <i>Procesarea imaginilor în ecosistemul python pentru analiza operelor de artă</i>	
Problemele principale tratate: 1. Clasificarea stilurilor artistice utilizând rețele neuronale convoluționale. 2. Restaurarea digitală a picturilor (denoising, inpainting, super-resolution) cu OpenCV. 3. Dezvoltarea unei aplicații web cu Flask pentru procesarea și vizualizarea imaginilor. 4. Integrarea unei baze de date SQLite pentru gestionarea informațiilor despre picturi	
Locul și durata practicii: VIII13-01.10.2024 -17.06.2025	
Bibliografie: [1] Chollet, F. <i>Deep Learning with Python</i> , 2nd ed., Manning, 2021. [2] Gonzalez, R. C., Woods, R. E. <i>Digital Image Processing</i> , 4th ed., Pearson, 2018. [3] Documentation: Flask, OpenCV, TensorFlow, SQLite.	
Aspecte particulare: Aplicație web interactivă cu design personalizat inspirat din site-urile de prezentare artistică; Clasificare automată a stilurilor și aplicarea de filtre personalizabile pentru restaurarea imaginilor. <i>(desene si aplicații practice vor fi înserate în textul lucrării)</i>	
Primit tema la data de: 15.10.2024	
Data predării lucrării: 17.06.2025	
Director departament, <i>Prof. dr. ing. Sorin Aurel MORARU</i>	Conducător științific, <i>șef lucr. dr. ing. Adrian DĂNILĂ</i> 
Candidat, <i>Cosferenț Roxana Adelina</i> 	

PROIECT DE DIPLOMĂ – VIZE		
Data vizei	Capitole/ problemele analizate	Semnătura conducătorului științific
16.12.2024	Studiu privind stadiul actual în domeniul procesării imaginilor operelor de artă	
7.03.2025	Noțiuni teoretice privind procesarea imaginilor, rețele neuronale convoluționale și clasificarea imaginilor	
25.04.2025	Antrenarea unui model convoluțional pentru clasificarea imaginilor artistice	
03.06.2025	Aplicație software pentru restaurare digitală și interfața web	
17.06.2025	Verificare procentaj similaritate Turnitin (total ≤15% și o singură sursă ≤5%)	5% / 1% 
APRECIEREA ȘI AVIZUL CONDUCĂTORULUI ȘTIINȚIFIC		
<p>1. Nivelul competențelor teoretice în domeniul proiectului dobândite pe durata elaborării temei: (100%) Candidatul a efectuat individual activitățile de cercetare documentară și a acumulat competențe suplimentare în domeniul temei proiectului.</p> <p>2. Calitatea gestionării timpului alocat pregătirii temei de proiectare: (100%) Candidatul a gestionat foarte bine timpul alocat pregătirii temei de proiectare și a participat la toate activitățile programate.</p> <p>3. Gradul de implicare în rezolvarea temei de proiectare: (100%) Candidatul s-a implicat activ în activitatea de rezolvare a temei de proiectare.</p> <p>4. Gradul îndeplinirii sarcinilor prevăzute în programul de realizare al proiectului: (95%) Candidatul a îndeplinit în totalitate sarcinile prevăzute în program</p> <p>5. Calitatea redactării proiectului de diploma: la redactarea proiectului de diploma candidatul a respectat toate cerințele din instrucțiunile de redactare elaborate de Facultatea IESC.</p> <p><i>Confirm efectuarea stagiului de 90 de ore de practică pentru elaborarea proiectului de licență.</i></p>		
Data: 17.06.2025	ADMIS pentru susținere/ RESPINS	Conducător științific șef lucr. dr. ing. Adrian DĂNILĂ 
AVIZUL DIRECTORULUI DE DEPARTAMENT		
Data:	ADMIS pentru susținere/ RESPINS	Director departament, Prof. dr. ing. Sorin Aurel MORARU
SUSȚINEREA PROIECTULUI DE DIPLOMĂ		
Sesiunea: Iunie 2025		
Rezultatul susținerii	PROMOVAT cu media:	
	RESPINS cu refacerea lucrării	
	RESPINS fără refacerea lucrării	
Președinte de comisie, Prof. dr. ing. Constantin SUCIU		

LISTA DE FIGURI ȘI CODURI SURSĂ

FIGURI

- Figura 1. Grafic evoluția acurateței
- Figura 2. Grafic evoluția pierderii
- Figura 3. Matricea de confuzie
- Figura 4. Indicatorii cantitativi
- Figura 5. Vizualizarea imaginilor in interfața Flask
- Figura 6. Aplicarea filtrelor
- Figura 7. Prezicerea stilului
- Figura 8. Prezicerea stilului
- Figura 9. Bara de navigare
- Figura 10. Pagina stiluri artistice
- Figura 11. Pagina upload
- Figura 11. Pagina stiluri

CODURI SURSĂ

- Code 1. Configurarea generatorului de imagini pentru preprocesare si augmentare.
- Code 2. Inițializarea arhitecturii MobileNetV2 pentru clasificarea stilurilor artistice.
- Code 3. Definirea generatorului de date cu augmentare.
- Code 4. Încărcarea imaginilor pentru antrenare
- Code 5. Încărcarea imaginilor pentru validare
- Code 6. Definirea callback-urilor pentru monitorizare și control
- Code 7. Lansarea procesului de antrenare
- Code 8. Construirea bazei de date
- Code 9. Aplicarea filtrelor
- Code 10. Prezicerea stilurilor

Acronim	Semnificație
API	Application Programming Interface
BLOB	Binary Large Object
CBIR	Content-Based Image Retrieval
CNN	Convolutional Neural Network
DnCNN	Denoising Convolutional Neural Network
GAN	Generative Adversarial Network
GPU	Graphics Processing Unit
HSV	Hue Saturation Value
HTML	HyperText Markup Language
JSON	JavaScript Object Notation
KNN	K-Nearest Neighbors
MSE	Mean Squared Error
OpenCV	Open Source Computer Vision Library
PSF	Point Spread Function
ReLU	Rectified Linear Unit
RGB	Red Green Blue
SGD	Stochastic Gradient Descent
SQL	Structured Query Language
SQLite	Structured Query Language – Lite
SRGAN	Super-Resolution Generative Adversarial Network
TensorFlow	Open-source deep learning framework
URL	Uniform Resource Locator
U-Net	U-shaped Convolutional Neural Network
YOLO	You Only Look Once

1.1 STADIUL ACTUAL AL PROBLEMEI ABORDATE

Procesarea imaginilor digitale a fost unul dintre principalele domenii de cercetare din ultimele decenii, în special în contextul digitalizării tot mai avansate. Muzeele, galeriile și arhivele din întreaga lume au început să-și digitalizeze colecțiile cu scopul de a le conserva, studia și face accesibile publicului larg. Odată cu creșterea volumului de imagini, a apărut nevoia unor instrumente automate pentru clasificarea, analiza și îmbunătățirea calității acestora.

În domeniul artelor vizuale, una dintre sarcinile dificile este clasificarea picturilor în funcție de stilul artistic sau de autor. Aceasta implică o procedură costisitoare și subiectivă, realizată de obicei manual de către experți în artă. Din acest motiv, au fost propuse mai multe metode care folosesc rețele neuronale convoluționale (CNN) pentru a învăța trăsături vizuale specifice stilurilor precum impresionism, baroc sau post-impresionism. Clasificarea imaginilor din seturi de date se realizează în prezent cu modele precum MobileNetV2 sau ResNet. (Sandler et al., 2018)

Pe de altă parte, o altă problemă frecventă este degradarea estetică a imaginilor digitale, cauzată de zgomot, pierderea detaliilor sau lipsa de informație în zone deteriorate. Pentru aceasta, se folosesc tehnici de restaurare digitală, cum ar fi reducerea zgomotului (denoising), refacerea zonelor lipsă (inpainting) sau îmbunătățirea rezoluției (super-resolution). Deși aceste sarcini au fost studiate în literatura științifică, nu există numeroase programe care să ofere o interfață unică pentru atât clasificarea stilului, cât și restaurarea imaginii într-un mod accesibil.

1.2 MOTIVAȚIA ALEGERII PROIECTULUI

Motivația principală care a stat la baza alegerii acestui proiect este interesul personal pentru domeniul procesării imaginilor și dorința de a aplica tehnologiile moderne de învățare automată în contextul artelor vizuale. Într-o eră digitală în care accesul la colecții de artă este tot mai facil, devine esențială dezvoltarea unor instrumente care să permită clasificarea automată și restaurarea eficientă a imaginilor artistice.

Totodată, complexitatea și rafinamentul operelor de artă reprezintă o provocare reală pentru algoritmi. Acest lucru a generat curiozitatea de a explora modul în care rețelele neuronale pot fi antrenate să recunoască stiluri artistice distincte și să contribuie la refacerea vizuală a unor lucrări degradate. Un alt aspect important a fost dorința de a crea un proiect aplicabil, cu o interfață accesibilă care să îmbine atât partea de clasificare, cât și cea de restaurare, într-o manieră unitară și intuitivă.

1.3 SCOPUL ȘI OBIECTIVELE PROIECTULUI

Scopul proiectului este dezvoltarea unei aplicații care îmbină clasificarea stilurilor artistice cu restaurarea digitală a imaginilor, bazându-se pe tehnici de inteligență artificială și interfețe web moderne.

Obiectivele principale ale proiectului sunt:

- Antrenarea unui model CNN (Convolutional Neural Network) pentru clasificarea stilurilor artistice, utilizând un subset din setul de date WikiArt. (Goodfellow et al., 2016)
- Compararea eficienței diferitelor metode de restaurare a imaginilor: denoising, gaussian blur, inpainting și altele.
- Dezvoltarea unei aplicații web folosind Flask, care să permită: (Flask Documentation)
 - Afișarea și selectarea imaginilor dintr-o galerie.
 - Aplicarea de filtre de restaurare și afișarea rezultatelor.
 - Afișarea stilului artistic prezis pe baza modelului antrenat.

- Încărcarea de imagini proprii pentru procesare.
- Descărcarea imaginilor procesate pe dispozitiv.
- Integrarea unei baze de date care să stocheze informații despre fiecare imagine: nume, autor, stil artistic, anul creației, imaginea originală și imaginea restaurată.

Prin realizarea acestor obiective, proiectul demonstrează fezabilitatea aplicării inteligenței artificiale nu doar în domenii comerciale, ci și în conservarea și promovarea patrimoniului cultural vizual.

1.4 DOMENIUL DE APLICABILITATE

Această lucrare se încadrează în domeniul procesării digitale a imaginilor, cu aplicații directe în digitalizarea și conservarea patrimoniului cultural. Tehnologiile de clasificare a stilurilor artistice și restaurare digitală a picturilor pot fi aplicate în următoarele domenii:

- Muzee și arhive digitale – pentru clasificarea automată a picturilor în funcție de stil și identificarea rapidă a autorilor sau curentelor artistice.
- Proiecte educaționale – pentru facilitarea învățării vizuale prin utilizarea unor aplicații interactive bazate pe picturi restaurate sau indexate automat.
- Conservare digitală – ca mijloc de sprijin pentru restauratori, oferindu-le posibilitatea de a previzualiza rezultatele posibile ale restaurării înainte de intervenția fizică.
- Aplicații mobile sau web – în care utilizatorul încarcă o pictură, iar sistemul oferă informații despre stilul artistic și o variantă restaurată a imaginii.

Prin consolidarea învățării automate cu tehnicile moderne de procesare a imaginilor, proiectul propus dobândește aplicabilitate în diverse instituții culturale și educaționale, precum și în rândul pasionaților de artă.

1.5 STRUCTURA PE CAPITOLE

Capitolul 2 va oferi fundamentele teoretice necesare pentru crearea aplicației. Vor fi discutate concepte fundamentale legate de procesarea imaginilor, rețelele neuronale convoluționale, metodele de clasificare a imaginilor artistice și tehnicile de restaurare digitală a picturilor. De asemenea, vor fi prezentate pe scurt tehnologiile și framework-urile utilizate în implementarea practică a aplicației: Keras, OpenCV, Flask și SQLite. (Bradski, 2000)

Capitolul 3 va prezenta prima componentă a proiectului, respectiv clasificarea imaginilor artistice în categorii. Vor fi detaliate pașii de prelucrare a setului de date, alegerea și implementarea modelului de rețea neuronală, procesul de antrenare și validare, precum și evaluarea performanței modelului. Vor fi explicate deciziile tehnice luate și motivațiile din spatele metodelor utilizate în această secțiune.

Capitolul 4 se va concentra pe a doua parte a proiectului, respectiv crearea interfeței web și procesul de restaurare digitală a imaginilor. Vor fi descrise structura bazei de date și dezvoltarea aplicației web folosind Flask. Vor fi detaliate funcționalitățile implementate, inclusiv vizualizarea imaginilor, aplicarea de filtre, clasificarea stilului artistic, salvarea rezultatelor procesării și posibilitatea utilizatorilor de a încărca propriile imagini pentru editare. Acest capitol va include și prezentarea interfeței și descrierea procesului de testare a aplicației. (Flask Documentation)

2.1 INTRODUCEREA ÎN PROCESAREA IMAGINILOR

Procesarea imaginilor este o ramură esențială a științei calculatoarelor și a inteligenței artificiale, dedicată în mod special îmbunătățirii, analizei și interpretării imaginilor digitale. O imagine digitală este o formă discretizată a unui semnal vizual, având un set de pixeli într-o matrice bidimensională, fiecare pixel purtând informații despre luminozitate și, eventual, culoare.

La baza procesării imaginilor se află teoria semnalelor, conform căreia o imagine reprezintă o formă specială de semnal bidimensional. Similar altor semnale (acustice, electrice sau termice), imaginile pot fi supuse unor tehnici de filtrare, transformare sau reconstrucție pentru a extrage informații relevante sau pentru a îmbunătăți calitatea imaginii. În funcție de aplicație, procesarea imaginilor poate include operații simple, cum ar fi extinderea contrastului sau reducerea zgomotului, sau proceduri mai complexe, cum ar fi restaurarea imaginilor degradate sau segmentarea automată a obiectelor.

Printre conceptele fundamentale se află contrastul între procesarea imaginilor în domeniul spațial și în domeniul frecvenței. În domeniul spațial, pixelii imaginii sunt manipulați direct, în timp ce în domeniul frecvenței, imaginea este transformată prin metode precum Transformata Fourier pentru a extrage informații ascunse. În plus, procesarea imaginilor presupune cunoașterea modelelor de culoare (RGB, HSV, CMYK), a histogramei de intensitate și a filtrelor spațiale (median, Gaussian, bilateral).

Aplicațiile procesării imaginilor sunt numeroase și variate, de la medicină (imagistică medicală) și securitate (recunoaștere facială) până la artă și restaurare digitală. În ultimii ani, progresele în învățarea automată și rețelele neuronale convoluționale (CNN) au revoluționat acest domeniu, permițând dezvoltarea de sisteme automate de recunoaștere și clasificare a imaginilor cu niveluri de precizie fără precedent. (Goodfellow et al., 2016)

2.2 REȚELE NEURONALE CONVOLUȚIONALE (CNN)

Rețelele neuronale convoluționale (CNN – Convolutional Neural Networks) reprezintă o categorie specializată de rețele neuronale artificiale, proiectată pentru procesarea datelor cu o structură în grilă, precum imaginile digitale. Spre deosebire de rețelele neuronale complet conectate, CNN-urile exploatează relațiile spațiale locale dintre pixeli, folosind operații matematice de convoluție pentru extragerea automată a caracteristicilor relevante din imagini. (Goodfellow et al., 2016)

Operația de convoluție presupune aplicarea unui filtru asupra unei regiuni locale din imagine, generând o hartă de caracteristici (feature map). Această abordare permite rețelei să învețe automat reprezentări ierarhice ale datelor vizuale, pornind de la caracteristici simple și ajungând la forme complexe.

O operație de convoluție aplicată unei imagini constă în utilizarea unui filtru(kernel) de dimensiuni fixe, precum 3×3 sau 5×5 , care parcurge întreaga imagine. Din punct de vedere matematic, pentru o imagine de intrare I și un kernel K , valoarea rezultată, denumită S , la o anumită poziție (i, j) , se calculează:

$$S(i, j) = \sum_m \sum_n I(i + m, j + n) K(m, n)$$

unde m și n parcurg dimensiunile kernelului.

Această operație permite rețelei să detecteze modele locale precum muchii, texturi sau colțuri sau alte forme vizuale relevante.

Arhitectura tipică a unei rețele CNN include următoarele tipuri de straturi: (Goodfellow et al., 2016)

- Stratul convoluțional: Aplică mai multe filtre învățabile asupra imaginii, producând hărți de caracteristici (feature maps).
- Funcția de activare (ex: ReLU): Aplică o non-liniaritate punct-cu-punct asupra rezultatelor convoluției, ajutând rețeaua să învețe funcții complexe.
- Stratul de pooling (ex: max pooling): Reduce dimensionalitatea hărților de caracteristici, păstrând informațiile esențiale și reducând sensibilitatea la variațiile locale. Max pooling, de exemplu, selectează valoarea maximă într-o fereastră locală.
- Straturi complet conectate (Fully Connected Layers): După extragerea caracteristicilor, straturile complet conectate realizează clasificarea sau regresia, analog cu rețelele neuronale tradiționale.
- Parametri și hiperparametri
Într-o rețea CNN, există două categorii principale de parametri: (Goodfellow et al., 2016)
- Parametri învățabili: Greutățile kernel-urilor și bias-urile.
- Hiperparametri: Dimensiunea filtrelor, stride-ul (pasul de deplasare al kernel-ului), padding-ul (bordura de completare cu zerouri), funcția de activare și rata de învățare. Alegerea corectă a hiperparametrilor influențează semnificativ performanța rețelei, atât în

ceea ce privește acuratețea cât și generalizarea.

Avantaje ale CNN: (Goodfellow et al., 2016)

- Eliminarea necesității extragerii manuale a caracteristicilor;
- Scalabilitate pentru imagini de dimensiuni mari;
- Reducerea numărului de parametri comparativ cu rețelele complet conectate, datorită partajării greutăților (weight sharing);
- Performanță ridicată în sarcini precum clasificare imagini, segmentare semantică și restaurare imagini.

În contextul procesării operelor de artă, CNN-urile sunt utilizate pentru sarcini precum clasificarea stilurilor artistice, recunoașterea autorilor sau restaurarea imaginilor deteriorate, datorită capacității lor ridicate de a înțelege și modela structuri vizuale complexe. (Goodfellow et al., 2016)

2.3 CLASIFICAREA IMAGINILOR ÎN FUNCȚIE DE STIL

Clasificarea imaginilor în funcție de stil artistic este o ramură a procesării imaginilor și a recunoașterii vizuale, care are ca obiectiv asocierea unei imagini cu o categorie estetică sau culturală definită. Stilurile artistice se disting prin elemente specifice de compoziție, paletă cromatică, tehnici de pensulație și teme vizuale recurente, iar recunoașterea acestora presupune înțelegerea profundă a acestor trăsături.

Din punct de vedere teoretic, clasificarea imaginilor presupune procesarea unei imagini ca set de date numerice, extragerea unor reprezentări semnificative și atribuirea unei etichete corespunzătoare. În cazul imaginilor de artă, dificultatea crește datorită caracterului subiectiv al stilurilor și a variației mari în execuția artistică.

Procesul general de clasificare poate fi privit în trei pași fundamentali:

- *Reprezentarea imaginilor*, în care imaginea este transformată într-o formă numerică adecvată, adesea păstrând informațiile spațiale și cromatice relevante;
- *Extragerea caracteristicilor*, în care sunt identificate trăsături distinctive ce pot diferenția stilurile (de exemplu, textura, compoziția, schemele de culoare);
- *Decizia de clasificare*, în care pe baza trăsăturilor extrase, o metodă de învățare determină apartenența imaginii la o anumită clasă.

În trecut, metodele de extragere a caracteristicilor erau realizate manual, bazându-se pe descrieri locale sau globale ale imaginii. Descriptorii precum histogramă de culori, margini (edge

histograms) sau descriptorii de forme erau utilizați pentru a cuantifica aspecte particulare ale stilului.

Odată cu apariția rețelelor neuronale convoluționale, procesul de extragere a caracteristicilor a fost automatizat, rețelele fiind capabile să învețe reprezentări ierarhice direct din imaginile brute. La nivel teoretic, aceste rețele folosesc funcții de activare și straturi convoluționale pentru a modela gradual complexitatea vizuală, de la forme simple la compoziții semantice complexe.

Clasificarea imaginilor de artă folosind CNN-uri presupune parcurgerea următoarelor etape tehnice: (Goodfellow et al., 2016)

- Preprocesarea datelor, constând în redimensionarea uniformă a imaginilor, normalizarea pixelilor pe intervalul [0,1] sau standardizarea acestora, și augmentarea setului de date prin rotații, decupări sau modificări de culoare;
- Arhitectura modelului, utilizând rețele pre-antrenate precum ResNet, MobileNetV2 sau EfficientNet, adaptate prin fine-tuning pe seturi de imagini de artă; (Sandler et al., 2018)
- Funcția de pierdere, de obicei funcția cross-entropy, utilizată pentru optimizarea clasificării multclasă;
- Optimizatorii, precum Adam sau SGD cu moment, pentru actualizarea parametrilor modelului în timpul antrenării;
- Evaluarea performanței, folosind metrici precum acuratețea, matricea de confuzie, scorul F1 sau log-loss-ul.
-

Provocările în clasificarea stilurilor artistice includ variabilitatea ridicată în cadrul aceleiași clase, similarități vizuale între stiluri diferite și numărul redus de date etichetate pentru anumite categorii, ceea ce poate conduce la probleme de overfitting sau dezechilibru al claselor.

Clasificarea stilurilor artistice ridică provocări teoretice semnificative, cum ar fi:

- Intra-cluster variability, adică variația mare între lucrările din același stil;
- Inter-cluster similarity, similaritatea vizuală între lucrări din stiluri diferite;
- Distribuții dezechilibrate, când unele stiluri sunt mult mai bine reprezentate decât altele în seturile de date disponibile.

Aceste dificultăți necesită o abordare teoretică solidă în definirea reprezentărilor vizuale și în alegerea metodelor de clasificare, astfel încât modelele să poată învăța diferențele subtile dintre stiluri într-o manieră generalizabilă.

2.4 TEHNICI ÎN RESTAURAREA IMAGINILOR

Restaurarea imaginilor digitale constituie o ramură fundamentală a procesării imaginilor, orientată spre refacerea sau îmbunătățirea calității imaginilor degradate. Din perspectivă teoretică, restaurarea presupune estimarea imaginii originale pornind de la o observație afectată de diverse tipuri de degradări, prin modelarea procesului de alterare și aplicarea unor tehnici de inversare controlată. Obiectivul principal este obținerea unei imagini restaurate care să fie cât mai apropiată de realitatea inițială, păstrând în același timp coerența structurală și semantica vizuală.

Procesul de degradare a imaginilor poate fi formalizat matematic prin relația:

$$g(x, y) = h(x, y)f(x, y) + n(x, y)$$

unde:

- $g(x, y)$ reprezintă imaginea degradată observată,
- $f(x, y)$ reprezintă imaginea originală necunoscută,
- $h(x, y)$ reprezintă funcția de estompare (PSF)
- $n(x, y)$ reprezintă zgomotul auditiv.

Astfel, restaurarea imaginii presupune rezolvarea unei probleme inverse, adesea *il-posedate*, unde informația inițială este pierdută parțial sau alterată în mod neregulat. În funcție de natura degradării și de informațiile disponibile despre procesul de alterare, tehnicile de restaurare pot fi clasificate în mai multe categorii principale:

2.4.1 Filtrarea spațială liniară și neliniară

Filtrarea spațială este una dintre cele mai simple tehnici de restaurare a imaginilor, care implică utilizarea de filtre locale asupra pixelilor imaginii.

Metodele de filtrare liniară, cum ar fi filtrul Gaussian și filtrul mediu, reduc eficient zgomotul de intensitate mică, menținând în același timp coerența vizuală a imaginilor.

Filtrele neliniare, cum ar fi filtrul median, sunt utilizate pentru eliminarea zgomotului impulsional (zgomot de tip sare și piper) datorită proprietăților lor de conservare a marginilor. Aceste metode sunt eficiente în condiții de zgomot scăzut și pentru degradări simple, dar au performanțe slabe în restaurarea estompurilor severe sau a pierderii globale de informație.

2.4.2 Restaurarea în domeniul frecvenței

Transformata Fourier permite trecerea din domeniul spațial în domeniul frecvenței, facilitând astfel manipularea componentelor de frecvență ale imaginii. În acest cadru, tehnicile de restaurare presupun operarea asupra spectrului frecvențial:

- Filtrul invers: încearcă să recupereze imaginea originală prin împărțirea spectrului imaginii degradate la spectrul funcției de estompare. Această metodă este sensibilă la zgomot, necesitând aplicarea în condiții ideale.
- Filtrul Wiener: minimizează eroarea pătratică medie între imaginea restaurată și imaginea originală, printr-o abordare statistică ce ia în considerare atât degradarea cât și nivelul zgomotului. Este una dintre cele mai utilizate metode în restaurarea imaginilor afectate de estompare și zgomot.

2.4.3 Tehnici de inpainting (completarea regiilor lipsă)

Restaurarea zonelor lipsă sau grav afectate dintr-o imagine se realizează prin tehnici de inpainting, care pot fi clasificate astfel:

- Inpainting pe bază de difuzie: tehnici variaționale care extind informația de la marginea zonelor deteriorate în interior, folosind modele de propagare a gradientului (ex.: metoda lui Bertalmio et al.).
- Inpainting prin patch-uri: metode neparametrice care selectează și copiază regiuni similare din imagine pentru a completa zonele lipsă (ex.: algoritmul Criminisi et al., PatchMatch).
- Inpainting bazat pe învățare profundă: rețele neuronale convoluționale antrenate special pentru restaurarea imaginilor, capabile să reconstruiască detalii semnificative prin generare de conținut.

2.4.4 Eliminarea zgomotului cu modele de învățare automată

Restaurarea imaginilor afectate de zgomot a cunoscut un progres remarcabil în ultimii ani datorită utilizării rețelelor neuronale. Modele precum DnCNN (Denoising Convolutional Neural Network) și altele similare folosesc arhitecturi adânci pentru a învăța o funcție de mapare între imaginile zgomotoase și cele clare, fără a necesita cunoașterea explicită a distribuției zgomotului. (Goodfellow et al., 2016)

Aceste abordări demonstrează performanțe superioare față de metodele convenționale, având capacitatea de a restaura eficient imagini chiar și în condițiile unor nivele semnificative de zgomot sau degradări complexe.

2.4.5 Restaurarea imaginilor utilizând tehnici de învățare profundă

Restaurarea imaginilor utilizând învățarea profundă a devenit un standard în procesarea imaginii moderne, datorită în mare parte capacității acestor tehnici de a învăța automat reprezentări complexe și robuste din seturi de date. Spre deosebire de abordările tradiționale, care se bazează pe modelarea explicită a procesului de degradare și pe presupuneri despre natura zgomotului sau estompării, metodele bazate pe rețele neuronale convoluționale pot aproxima funcții complexe de restaurare prin învățare supravegheată.

Conceptul de bază constă în utilizarea unei rețele neuronale, de obicei de tip encoder-decoder, care primește ca intrare o imagine degradată și produce ca ieșire o estimare a imaginii curate. Procesul de învățare se realizează prin optimizarea unei funcții de pierdere, de obicei Eroarea Pătratică Medie (MSE), care măsoară diferența dintre imaginea restaurată și imaginea originală. Alte alternative includ funcțiile de pierdere percepțională, care compară diferențele în spațiile de trăsături extrase de rețele pre-antrenate, sau pierderile adversariale, care sunt utilizate pentru a face imaginile restaurate mai realiste din punct de vedere vizual.

Una dintre primele arhitecturi dezvoltate pentru denoising a fost Denoising Convolutional Neural Network (DnCNN). Modelul utilizează straturi convoluționale adânci cu normalizare în loturi și funcții de activare neliniare pentru a învăța caracteristicile zgomotului aditiv, care este ulterior scăzut din imaginea originală. Această strategie, cunoscută sub denumirea de învățare reziduală, reduce complexitatea sarcinii de predicție și îmbunătățește stabilitatea procesului de antrenare. Rețeaua este capabilă să elimine zgomotul gaussian, zgomotul impulsional sau chiar zgomote mai complexe, cu o acuratețe și o generalizare superioare față de metodele convenționale. (Goodfellow et al., 2016)

Autoencoderele convoluționale sunt o clasă esențială de arhitecturi utilizate în restaurarea imaginilor în informatică. Acestea constau dintr-un encoder, care proiectează datele către dimensiuni mai mici, extrăgând trăsături latente stabile, și un decoder, care reconstruiește imaginea țintă.

Prin limitarea operării rețelei la un spațiu latent mai mic, autoencoderul dobândește abilitatea de a ignora zgomotul irelevant, fără a sacrifica informațiile valoroase. Au fost propuse mai multe extensii ale arhitecturii de bază, cum ar fi aplicarea regularizării spațiului latent pentru a promova menținerea structurilor geometrice.

Una dintre cele mai utilizate arhitecturi în restaurarea imaginilor este U-Net, care a fost inițial concepută pentru segmentarea imaginilor medicale. Designul său simetric encoder-decoder, augmentat cu conexiuni skip între etapele de compresie și reconstrucție, permite menținerea informațiilor detaliate pe parcursul procesului de restaurare. Aceste conexiuni directe permit schimbul de informație între straturile corespunzătoare, reducând pierderea de rezoluție și îmbunătățind calitatea imaginilor restaurate. În cercetările recente, rețelele generative adversariale (GAN-uri) au fost aplicate în restaurarea imaginilor pentru a produce ieșiri cu o calitate vizuală superioară. În acest cadru, un generator încearcă să creeze imagini restaurate, în timp ce un discriminator încearcă să identifice imagini reale și imagini generate artificial. Antrenamentul adversarial duce la imagini restaurate care nu doar că minimizează disparitățile numerice între imaginea originală și cea restaurată, dar și prezintă texturi și detalii mult mai realiste. Utilizările cheie includ reconstrucția realistă a regiunilor lipsă (context encoders) și super-rezoluția imaginilor de rezoluție scăzută (SRGAN). (Ronneberger et al., 2015)

Utilizarea tehnicilor de învățare profundă în restaurarea imaginilor aduce mai multe beneficii. În primul rând, aceste tehnici elimină necesitatea unor modele explicite de degradare și pot învăța să se adapteze la diferite condiții de estompăre și zgomot. În al doilea rând, performanța obținută tinde să depășească limitele abordărilor clasice, în special în cazul degradărilor severe sau a interacțiunilor complicate ale factorilor perturbatori. Totuși, aplicarea rețelelor neuronale adânci presupune și anumite dezavantaje, cum ar fi necesitatea unor seturi de

date mari și variate pentru antrenare, putere computațională semnificativă și riscul de supraînvățare atunci când datele disponibile nu sunt suficiente sau nu sunt variate.

În domeniul restaurării operelor de artă, este esențial ca tehnicile bazate pe învățare profundă să fie aplicate cu prudență. Pe lângă cerințele tehnice de restaurare, este crucială păstrarea autenticității estetice a operei de artă și prevenirea introducerii de artefacte care ar putea distorsiona sensul artistic original. Prin urmare, procesul de restaurare trebuie să fie însoțit de validări vizuale riguroase și, atunci când este necesar, intervenția expertului uman pentru a asigura respectarea valorilor culturale și artistice.

2.5 BAZE DE DATE ÎN APLICAȚII CU IMAGINI

Utilizarea bazelor de date în aplicațiile care gestionează imagini reprezintă un aspect esențial în proiectarea și implementarea sistemelor informatice moderne. O bază de date pentru imagini are rolul de a organiza, stoca, regăsi și gestiona eficient seturi mari de imagini împreună cu metadatele asociate acestora. Din punct de vedere teoretic, o imagine digitală poate fi considerată ca un obiect de date complexe, caracterizat nu doar prin conținutul său binar sau matricial, ci și printr-un ansamblu de atribute descriptive precum rezoluția, formatul, culoarea predominantă, autorul sau data capturării.

Modelele tradiționale de baze de date relaționale, bazate pe paradigme tabulare și relații bine definite între entități, pot fi adaptate pentru stocarea imaginilor utilizând tipuri speciale de date, cum ar fi Binary Large Objects (BLOBs). În această abordare, imaginile sunt stocate direct în câmpuri de tip BLOB, iar metadatele relevante sunt gestionate prin atribute clasice ale tabelului. Totuși, această metodă poate deveni ineficientă atunci când seturile de date sunt extrem de voluminoase sau când operațiunile de căutare și analiză a conținutului imagistic devin complexe. În paralel, pentru aplicațiile care implică procesarea și analiza avansată a imaginilor, au fost dezvoltate baze de date orientate pe obiecte și baze de date NoSQL, care permit o mai mare flexibilitate în modelarea datelor. În bazele de date orientate pe documente, cum ar fi MongoDB, imaginile pot fi stocate fie sub formă de fișiere separate cu referințe în documente JSON, fie direct codificate în format Base64, alături de atributele semantice. Această structură permite o scalabilitate superioară și o integrare eficientă cu aplicațiile distribuite și sistemele de procesare paralelă.

În ceea ce privește organizarea datelor, un concept fundamental este separarea între conținutul imaginii și metadatele asociate. Conținutul imaginii poate fi tratat ca un flux brut de date, în timp ce metadatele sunt structurate și indexate pentru a permite căutări rapide și eficiente. Metadatele pot include informații descriptive, tehnice, administrative și chiar semantice, cum ar fi clasificarea imaginii după stil artistic, tematică sau autor.

Aplicațiile moderne cu imagini impun adesea cerințe suplimentare de interogare bazată pe conținut (Content-Based Image Retrieval - CBIR). Acest tip de căutare presupune analiza caracteristicilor vizuale extrase automat din imagini, precum histogramme de culori, texturi, forme sau caracteristici profunde învățate de rețele neuronale convoluționale. În astfel de sisteme, baza de date nu gestionează doar date brute și metadata, ci și descriptorii de caracteristici vizuale, care sunt indexați utilizând tehnici avansate de căutare spațială, cum ar fi arborii KD sau hashing-ul localității sensibile (LSH).

Un alt aspect critic în bazele de date cu imagini este managementul volumului mare de date și necesitatea de optimizare a performanței. Soluțiile eficiente presupun utilizarea de baze de date distribuite, replicare automată, fragmentare (sharding) și integrarea cu sisteme de stocare externe specializate pentru fișiere de mari dimensiuni. În plus, sistemele de stocare trebuie să ofere suport pentru versiuni multiple ale aceleiași imagini, păstrând istoricul modificărilor și facilitând operațiunile de restaurare sau comparare.

În cadrul aplicațiilor de restaurare digitală și analiză artistică, bazele de date nu doar că stochează imagini și informații descriptive, ci devin infrastructuri active care susțin fluxuri de procesare automată. De exemplu, imaginile brute pot fi procesate prin algoritmi de denoising sau super-rezoluție, iar rezultatele intermediare și finale sunt înregistrate în baza de date împreună cu informații despre tehnica aplicată, parametrii utilizați și evaluările calitative ale restaurării. Din perspectiva securității, gestionarea imaginilor în baze de date presupune implementarea unor politici stricte de control al accesului, protecția integrității datelor și, acolo unde este necesar, criptarea imaginilor stocate. Acest lucru este deosebit de important în aplicațiile care gestionează opere de artă de patrimoniu sau conținut cu valoare culturală ridicată.

2.6 FRAMEWORK-URI UTILIZATE

În dezvoltarea de aplicații software, framework-urile reprezintă structuri conceptuale și tehnice care oferă o bază standardizată pentru construirea și organizarea aplicațiilor. Un framework definește o arhitectură fundamentală, punând la dispoziția dezvoltatorului un set de biblioteci, reguli, modele de proiectare și componente reutilizabile, toate integrate într-un ansamblu coerent. Scopul principal al unui framework este de a accelera procesul de dezvoltare, de a asigura coerența și scalabilitatea aplicațiilor și de a reduce riscurile asociate proiectării defectuoase sau inconsistente.

Din punct de vedere tehnic, un framework impune o anumită inversare a controlului, cunoscută sub denumirea de principle of inversion of control. În loc ca dezvoltatorul să controleze direct fluxul de execuție, framework-ul stabilește fluxul general și apelează codul scris de dezvoltator în momentele și condițiile definite de arhitectură. Această abordare permite separarea clară a preocupărilor, favorizează modularitatea și crește testabilitatea componentelor individuale ale aplicației.

Framework-urile sunt adesea specializate în funcție de domeniul aplicației. În procesarea imaginilor, framework-urile precum Keras sau OpenCV oferă funcționalități avansate pentru manipularea, analizarea și procesarea imaginilor, abstractizând complexitatea operațiilor matematice și algoritmice implicate. În dezvoltarea web, framework-uri precum Flask facilitează crearea interfețelor web și gestionarea comunicației client-server într-un mod rapid și organizat. În gestionarea datelor, soluții precum SQLite furnizează infrastructura necesară pentru stocarea și interogarea eficientă a informațiilor într-un format relațional. (Bradski, 2000)

Un avantaj fundamental al utilizării framework-urilor constă în standardizarea practicilor de dezvoltare. Framework-urile impun convenții de codare, modele de arhitectură și fluxuri de lucru prestabilite, ceea ce contribuie la îmbunătățirea calității codului, la facilitarea colaborării în echipe mari și la reducerea timpului necesar pentru depanare și mentenanță. De asemenea, framework-urile beneficiază de comunități largi de utilizatori și dezvoltatori, ceea ce le face extensibile, bine documentate și susținute prin actualizări regulate de securitate și performanță.

2.6.1 KERAS

Keras este o bibliotecă open-source de nivel înalt pentru dezvoltarea și antrenarea rețelelor neuronale, scrisă în limbajul Python. Aceasta oferă o interfață simplificată și modulară peste backend-uri precum TensorFlow, Theano sau Microsoft Cognitive Toolkit (CNTK), facilitând implementarea rapidă a modelelor de învățare profundă. Keras este proiectat pentru a permite prototiparea rapidă, suportând modele secvențiale, modele cu grafuri de aciclicitate direcționată (DAG) și modele personalizate complexe. Conceptual, Keras gestionează niveluri multiple de abstractizare: de la manipularea tensorilor și operațiilor matematice de bază, până la construirea de rețele neuronale sofisticate, cu straturi convoluționale, recurent sau complet conectate. În plus, biblioteca oferă facilități pentru preprocesarea datelor, augmentarea imaginilor, optimizarea antrenamentelor și evaluarea modelelor. Datorită acestei combinații de flexibilitate și ușurință în

utilizare, Keras este una dintre cele mai populare alegeri în cercetare și industrie pentru sarcinile de clasificare, segmentare, regresie și restaurare a imaginilor. (Chollet, 2021)

2.6.2 OPEN CV

OpenCV (Open Source Computer Vision Library) este una dintre cele mai extinse și utilizate biblioteci pentru procesarea imaginilor și viziune computațională. Proiectată inițial de Intel și menținută ulterior de comunitatea open-source, OpenCV oferă un set vast de algoritmi și funcții pentru manipularea imaginilor, analiza video, recunoașterea obiectelor, calibrarea camerelor și reconstrucția 3D. Biblioteca suportă atât operații de bază, cum ar fi conversia între spații de culoare, redimensionarea și filtrarea imaginilor, cât și funcții avansate de detectare a trăsăturilor locale (SIFT, SURF, ORB) și clasificare vizuală. Din punct de vedere arhitectural, OpenCV este concepută pentru a funcționa eficient în aplicații reale, optimizând performanța prin utilizarea extensivă a procesării paralele și suportând integrarea cu platforme hardware diverse. În contextul restaurării imaginilor și analizei operelor de artă, OpenCV joacă un rol esențial în implementarea preprocesărilor, filtrărilor și extragerii de caracteristici vizuale. (Bradski, 2000)

2.6.3 FLASK

Flask este un micro-framework web pentru Python, conceput pentru a dezvolta rapid aplicații web scalabile și flexibile. Spre deosebire de framework-urile mai complexe, Flask oferă un nucleu minim, dar extensibil, permițând dezvoltatorului să adauge module și extensii doar atunci când este necesar. Bazat pe Werkzeug pentru routing-ul HTTP și Jinja2 pentru șabloane, Flask gestionează eficient cererile client-server, sesiuni, formulare și comunicarea cu baze de date. În aplicațiile care implică procesarea imaginilor, Flask este utilizat frecvent pentru a crea interfețe web care permit încărcarea imaginilor, procesarea lor în backend și afișarea rezultatelor restaurării sau clasificării. De asemenea, Flask susține integrarea facilă cu API-uri externe, baze de date SQL și NoSQL, precum și cu sisteme de autentificare și autorizare, ceea ce îl face potrivit pentru prototipuri rapide și aplicații finale scalabile. (Flask Documentation)

2.6.4 SQLite

SQLite este un sistem de gestionare a bazelor de date relaționale încorporat, recunoscut pentru caracterul său compact, portabilitatea ridicată și lipsa necesității unui server dedicat pentru funcționare. Spre deosebire de bazele de date client-server tradiționale, SQLite funcționează direct prin manipularea fișierelor locale de baze de date, fiind astfel ideal pentru aplicații mici și mijlocii, prototipuri și aplicații web embedded. Structura sa simplificată permite operații SQL standard de creare, interogare, modificare și ștergere a datelor, fiind compatibilă cu majoritatea bibliotecilor de acces la baze de date existente în Python, inclusiv SQLAlchemy. În aplicațiile care gestionează imagini, SQLite poate fi utilizat pentru stocarea metadatelor asociate imaginilor, a rezultatelor procesării, sau a căilor către fișierele de imagini salvate pe disc. Astfel, oferă o soluție eficientă pentru gestionarea datelor într-un context de dezvoltare rapidă și cu resurse limitate. (SQLite Documentation)

3.1 PRELUCRAREA SETULUI DE DATE

Pentru clasificarea operelor de artă în funcție de stilul artistic, s-a utilizat un subset atent selectat din cunoscutul set de date WikiArt, recunoscut în comunitatea științifică pentru diversitatea și calitatea imaginilor pe care le conține. Acest set de date, disponibil public prin platforme precum Kaggle, cuprinde aproximativ 80.000 de picturi etichetate manual de experți, distribuite în peste 25 de stiluri artistice distincte.

Având în vedere limitările impuse de resursele hardware disponibile (memorie RAM, spațiu de stocare și timp de antrenare), a fost ales un subset reprezentativ care acoperă **cinci dintre cele** mai expresive stiluri artistice: *Art Nouveau*, *Baroc*, *Expresionism*, *Realism* și *Romanticism*. Acestea au fost alese atât pentru contrastul vizual evident dintre ele, cât și pentru relevanța lor culturală și estetică în contextul artei occidentale.

Criterii de selecție:

Pentru a asigura un antrenament echilibrat și eficient, au fost aplicate următoarele criterii de selecție:

- Minim 100 de imagini per clasă, pentru a permite rețelei să învețe trăsături generalizabile și să evite memorarea unor exemple izolate;
- Diversitate vizuală semnificativă între clase, astfel încât modelul să poată învăța diferențele stilistice subtile între categorii (de exemplu, contrastul dintre realismul figurativ și expresionismul abstract);
- Relevanță istorică și estetică, pentru a include stiluri larg recunoscute, cu trăsături vizuale bine definite.

Organizarea datelor:

Pentru a facilita procesul de antrenare, imaginile au fost organizate în directoare distincte, fiecare director corespunzând unui stil artistic. Această structură ierarhică este cerută de clasa *ImageDataGenerator* din biblioteca Keras, care presupune organizarea fișierelor în subfoldere etichetate corespunzător pentru clasificarea multi-clasă. Astfel, fișierele au putut fi citite și procesate automat în loturi, fiecare imagine fiind etichetată automat pe baza numelui folderului din care provine. (Chollet, 2021)

Tehnici de preprocesare și augmentare:

Pentru a îmbunătăți calitatea generală a datelor și a preveni supraînvățarea, au fost aplicate o serie de transformări standard în domeniul învățării automate pe imagini:

- Redimensionare uniformă a tuturor imaginilor la dimensiunea de 224x224 pixeli, conform cerințelor rețelei *MobileNetV2*; (Sandler et al., 2018)
- Normalizarea valorilor pixelilor în intervalul [0, 1], prin împărțirea fiecărei componente RGB la 255, asigurând astfel stabilitate numerică în timpul propagării directe și inverse în rețea;
- Augmentarea datelor, aplicând transformări aleatorii precum:
 - rotații de ± 20 de grade,
 - zoom aleator între 80% și 120%,
 - inversare orizontală (flip pe axa X),
 - ușoare modificări de luminozitate și contrast,
 - decupaje aleatorii și deformări (shear transformations),
 acestea având rolul de a simula variații naturale în compoziție și stil, crescând capacitatea modelului de a generaliza pe date noi.

Setul de imagini a fost împărțit astfel:

- 80% pentru antrenare (training set), utilizat în actualizarea greutateilor rețelei;
- 10% pentru validare (validation set), utilizat în monitorizarea performanței modelului în timpul antrenării;

- 10% pentru testare (test set), rezervat pentru evaluarea finală a capacității de generalizare.
- Implementarea în cod

Pentru generarea automată a loturilor de imagini procesate și augmentate, a fost utilizată clasa `ImageDataGenerator`, cu următoarea configurare:

```
train_datagen = ImageDataGenerator(  
    rescale=1./255,  
    rotation_range=20,  
    zoom_range=0.2,  
    horizontal_flip=True,  
    validation_split=0.2  
)
```

Code 1. Configurarea generatorului de imagini pentru preprocesare și augmentare.

Această abordare permite crearea unui set de date robust și echilibrat, esențial pentru antrenarea unei rețele neuronale convoluționale moderne. Toate etapele de preprocesare și organizare au fost implementate în mediul Kaggle Notebooks, utilizând framework-urile Keras și TensorFlow. (Chollet, 2021)

3.2 ALEGEREA ARHITECTURII CNN

Clasificarea imaginilor în funcție de stiluri artistice implică recunoașterea unor trăsături vizuale subtile, recurente în picturi create în epoci și contexte culturale diferite. Această sarcină necesită un model capabil să distingă nuanțele fine de culoare, textură și compoziție. În acest scop, rețelele neuronale convoluționale (CNN) s-au impus ca soluții de referință în domeniul viziunii artificiale, datorită capacității lor de a învăța automat reprezentări relevante direct din datele vizuale, fără a fi nevoie de inginerie manuală a trăsăturilor. (Goodfellow et al., 2016)

O rețea CNN este alcătuită dintr-o succesiune de straturi specializate care extrag și procesează informații din imagini: (Goodfellow et al., 2016)

- Straturile convoluționale aplică filtre (kernels) peste patch-uri locale ale imaginii și detectează tipare vizuale, precum muchii, contururi, texturi;
- Funcțiile de activare (ex: ReLU) introduc non-liniarități, permițând rețelei să învețe relații complexe între pixeli;
- Straturile de pooling (ex: MaxPooling2D) reduc dimensiunea spațială a activărilor, păstrând esențialul și reducând riscul de overfitting;
- Straturile Dense (fully-connected) din finalul rețelei funcționează ca un clasificator pe baza trăsăturilor învățate.
- Aceste componente permit rețelei să construiască o ierarhie de reprezentări vizuale, începând cu caracteristici de jos nivel (linii, forme) și ajungând la concepte înalte (configurații tipice unui stil artistic).

Alegerea MobileNetV2 (Sandler et al., 2018)

În cadrul proiectului, s-a optat pentru utilizarea MobileNetV2, o arhitectură CNN compactă și eficientă, proiectată pentru dispozitive cu resurse limitate, dar cu performanță competitivă în sarcini de clasificare a imaginilor. Această alegere a fost motivată de mai mulți factori: (Sandler et al., 2018)

- Transfer learning

MobileNetV2 vine cu greutateți pre-antrenate pe ImageNet, un set de date vast (14+ milioane imagini, 1000 clase). Această cunoaștere preexistentă este valorificată prin transfer learning, permițând modelului să reutilizeze trăsături vizuale generale (linii, culori, forme) deja învățate și să se adapteze apoi specificului imaginilor artistice. (Russakovsky et al., 2015)

- Eficiență computațională

MobileNetV2 utilizează: (Sandler et al., 2018)

- Convoluții depthwise separable: separă filtrarea spațială de combinarea canalelor, reducând drastic numărul de parametri;
- Blocuri reziduale inversate: conectează straturi înguste spre straturi largi, permițând o propagare eficientă a informației și reducerea pierderilor de semnal;
- Activări liniare în bottlenecks: păstrează informația esențială, evitând distorsiuni inutile în date.

Astfel, modelul are un raport excelent între complexitate și acuratețe, ceea ce îl face potrivit pentru antrenare în medii precum Kaggle, unde resursele GPU sunt limitate.

- Strategie de fine-tuning

Pentru adaptarea MobileNetV2 la clasificarea stilurilor artistice, a fost utilizată o strategie de fine-tuning parțial, constând în: (Sandler et al., 2018)

- Înghețarea straturilor inițiale (primele 80-90%), responsabile de învățarea trăsăturilor de jos nivel generaliste;
- Deblocarea straturilor superioare (ultimele 30), care sunt mai sensibile la forme specifice și permit adaptarea modelului la particularitățile picturilor;
- Adăugarea unui cap de clasificare personalizat, optimizat pentru cele 5 clase de stiluri artistice.

Această abordare permite un echilibru între reutilizarea cunoștințelor anterioare și învățarea de trăsături noi, specifice setului curent de imagini.

Regularizare și generalizare

În procesul de învățare automată, un obiectiv esențial este obținerea unui model care nu doar memorizează datele de antrenament, ci învață tipare generalizabile aplicabile și pe date noi, nevăzute. Fenomenul contrar, numit *overfitting* (*supraînvățare*), apare atunci când modelul se adaptează prea fidel la setul de antrenament, incluzând și zgomotele sau particularitățile nerelevante din date. În astfel de cazuri, modelul ajunge să aibă o performanță excelentă pe setul de antrenament, dar semnificativ mai slabă pe setul de validare sau test.

Overfitting-ul apare frecvent în situații precum:

- Dimensiunea setului de date este redusă (puține exemple pentru fiecare clasă);
- Complexitatea rețelei este ridicată (mulți parametri și straturi);
- Lipsa mecanismelor de regularizare sau augmentare a datelor.

Pentru a preveni acest comportament și a încuraja capacitatea de generalizare (adică performanță stabilă pe date noi), au fost integrate în model mai multe tehnici de regularizare:

- Dropout (0.3): Această tehnică constă în dezactivarea aleatorie a unui procent de neuroni (30% în cazul de față) la fiecare epocă de antrenament. Prin această aleatorizare, modelul este forțat să nu se bazeze excesiv pe anumite conexiuni și să învețe reprezentări mai robuste și distribuite.
- Regularizarea L2: Este o formă de penalizare a greutăților mari din rețea, adăugând un termen suplimentar în funcția de pierdere proporțional cu pătratul normei L2 a greutăților. Scopul este încurajarea unui model mai simplu, cu valori mici ale parametrilor, ceea ce reduce riscul de overfitting și crește stabilitatea generalizării.
- GlobalAveragePooling2D: În loc să folosească un strat complet conectat foarte mare, care ar introduce numeroși parametri, această metodă reduce hărțile de activare obținute de rețeaua convoluțională la valorile medii ale fiecărui canal. Această abordare păstrează esența informației spațiale și reduce drastic numărul de parametri, simplificând arhitectura finală.

```
# Inițializăm MobileNetV2 fără stratul de clasificare
base_model = MobileNetV2(
    input_shape=(224, 224, 3),
    include_top=False,
    weights='imagenet'
)

# Deblocăm ultimele 30 de straturi pentru fine-tuning
base_model.trainable = True
for layer in base_model.layers[:-30]:
    layer.trainable = False

# Adăugăm capul de clasificare adaptat sarcinii
x = GlobalAveragePooling2D()(base_model.output)
x = Dropout(0.3)(x)
x = Dense(128, activation='relu', kernel_regularizer=l2(1e-4))(x)
output = Dense(num_classes, activation='softmax')(x)

# Construim și compilăm modelul final
model = Model(inputs=base_model.input, outputs=output)
model.compile(
    optimizer=Adam(learning_rate=1e-4),
    loss='categorical_crossentropy',
    metrics=['accuracy']
)
```

Code 2. Inițializarea arhitecturii MobileNetV2 pentru clasificarea stilurilor artistice.

Modelul MobileNetV2 este încărcat cu greutateți pre-antrenate pe setul de date ImageNet pentru a valorifica cunoștințele deja acumulate în recunoașterea caracteristicilor vizuale generale. Prin setarea `include_top=False`, stratul de ieșire original este eliminat, oferind posibilitatea de a adăuga un cap personalizat pentru noua sarcină de clasificare cu 5 clase. (Russakovsky et al., 2015)

Se aplică fine-tuning doar pe ultimele 30 de straturi ale rețelei, permițând modelului să adapteze caracteristicile învățate anterior la datele artistice din proiect. Straturile de bază rămân congelate pentru a păstra trăsăturile generale (cum ar fi detecția marginilor, texturilor etc.).

După stratul convoluțional, se folosește GlobalAveragePooling2D, care reduce dimensiunea tensorului la un vector 1D prin mediere spațială, păstrând esența informației și reducând drastic numărul de parametri. Se adaugă un strat intermediar Dense(128) activat ReLU pentru a învăța combinații complexe de trăsături, urmat de un strat Dropout(0.3) care dezactivează aleatoriu 30% dintre neuroni la fiecare epocă – prevenind supraînvățarea.

Pentru regularizare suplimentară, stratul Dense include și penalizare L2 ($l_2(1e-4)$), care limitează amplitudinea greutăților. Stratul de ieșire este format din Dense(num_classes) cu activare softmax, asigurând clasificarea multi-clasă. Modelul este compilat cu optimizatorul Adam și funcția de pierdere categorical_crossentropy, adecvată clasificării pe mai multe etichete exclusive.

3.3 ÎNCĂRCAREA ȘI PREGĂTIREA DATELOR

Pregătirea corectă a datelor reprezintă un pas important în orice proiect de învățare profundă, întrucât rețelele neuronale convoluționale (CNN) sunt extrem de sensibile la variațiile distribuției datelor de intrare. În mod particular, în sarcinile de clasificare multi-clasă a imaginilor, calitatea datelor, echilibrul dintre clase, precum și strategiile de augmentare aplicate pot influența decisiv performanțele finale ale modelului. (Goodfellow et al., 2016)

În cadrul acestui proiect, imaginile provenite din setul WikiArt (subset manual selecționat pentru cele 5 stiluri artistice) au fost organizate în subdirectoare corespunzătoare fiecărei clase. Această structurare este impusă de metoda `flow_from_directory()` din Keras, care asociază automat fiecare imagine cu eticheta derivată din numele folderului părinte. Această practică

simplifică gestionarea datelor și elimină necesitatea unor fișiere CSV de adnotare separate. (Chollet, 2021)

Totodată, pentru a crește capacitatea de generalizare a modelului și a reduce riscul de overfitting (supraînvățare pe datele de antrenament), a fost implementat un proces de augmentare a datelor. Acest proces are rolul de a introduce artificial o diversitate mai mare în datele de antrenament prin aplicarea unor transformări geometrice și fotometrice controlate. Scopul augmentării este să simuleze variații naturale (de poziție, lumină, orientare) care apar în date reale, astfel încât modelul să nu „învețe pe de rost” imaginile, ci să extragă trăsături semnificative și robuste.

Următorul cod creează un generator de date pentru antrenament și validare folosind clasa ImageDataGenerator din biblioteca Keras. Acest generator aplică automat preprocesări și augmentări în timpul antrenării modelului, fără a modifica imaginile originale de pe disc: (Chollet, 2021)

```
data_dir = '/kaggle/working/wikiart_subset'
datagen = ImageDataGenerator(
    preprocessing_function=preprocess_input,
    validation_split=0.2,
    rotation_range=20,
    zoom_range=0.2,
    brightness_range=[0.9, 1.1],
    shear_range=0.1,
    horizontal_flip=True,
    fill_mode='nearest'
)
```

Code 3. Definirea generatorului de date cu augmentare.

```
train_gen = datagen.flow_from_directory(
    data_dir,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    subset='training',
    shuffle=True)
```

Code 4. Încărcarea imaginilor pentru antrenare

- **data_dir**: cale către folderul unde sunt stocate imaginile împărțite pe subfoldere (unul pentru fiecare clasă).
- **target_size=(224, 224)**: redimensionează toate imaginile la 224x224 pixeli, dimensiunea de intrare standard pentru MobileNetV2. (Sandler et al., 2018)
- **batch_size=32**: specifică numărul de imagini procesate simultan în fiecare iterație.
- **class_mode='categorical'**: etichetele sunt convertite în format one-hot encoding, necesar pentru clasificare multi-clasă.
- **subset='training'**: selectează doar cele 80% din imagini alocate pentru antrenare.
- **shuffle=True**: asigură amestecarea aleatorie a imaginilor la fiecare epocă pentru a preveni învățarea unor tipare fixe.

În mod similar, se generează datele pentru validare. Singura diferență este setarea **subset='validation'**, iar **shuffle=False** este important pentru a păstra ordinea constantă între epoci.

```
val_gen = datagen.flow_from_directory(
    data_dir,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    subset='validation',
    shuffle=False
)
```

Code 5. Încărcarea imaginilor pentru validare

- `subset='validation'`: selectează cele 20% din imagini rezervate pentru validare.
- `shuffle=False`: menține ordinea imaginilor constantă între epoci, important pentru interpretarea metricilor și generarea graficelor.

Prin aceste proceduri, setul de date este transformat într-o formă optimă pentru a fi procesat de o rețea CNN. Se asigură astfel nu doar compatibilitatea cu arhitectura aleasă (MobileNetV2), ci și o diversitate vizuală care reduce semnificativ riscul de overfitting. În plus, abordarea bazată pe ImageDataGenerator permite o procesare eficientă în memorie, prin generarea dinamică a datelor augmentate.

3.4 ANTRENAREA SI VALIDAREA MODELULUI

În cadrul procesului de învățare automată, antrenarea modelului reprezintă o etapă fundamentală, prin care rețeaua neuronală își ajustează intern parametrii pentru a învăța să asocieze datele de intrare cu etichetele corecte. Această optimizare se face prin iterarea asupra datelor de antrenament și prin actualizarea greutăților interne ale modelului, astfel încât să fie minimizată funcția de pierdere (loss function), adică diferența dintre predicțiile modelului și valorile reale.

Modelul utilizat în cadrul acestui proiect este o rețea convoluțională bazată pe MobileNetV2, adaptată pentru clasificarea imaginilor în cinci stiluri artistice: Art Nouveau, Baroc, Expresionism, Realism și Romanticism. MobileNetV2 este o arhitectură optimizată pentru eficiență computațională, ceea ce o face ideală pentru aplicații în care resursele hardware sunt limitate, cum este cazul mediului de lucru Kaggle Notebooks, unde GPU-ul poate lipsi sau fi limitat ca timp de execuție. (Sandler et al., 2018)

Pentru a spori robustețea modelului și a preveni antrenarea excesivă (overfitting), au fost implementate tehnici moderne de regularizare și optimizare a hiperparametrilor, incluzând folosirea callback-urilor *EarlyStopping* și *ReduceLROnPlateau*. Acestea oferă control dinamic asupra procesului de antrenare, monitorizând performanța modelului pe datele de validare și ajustând rata de învățare sau întrerupând antrenarea când este necesar.

În acest prim bloc de cod, se definesc două mecanisme de control importante care sunt integrate în procesul de antrenare.

```
early_stop = EarlyStopping(  
    monitor='val_loss',  
    patience=5,  
    restore_best_weights=True,  
    verbose=1)  
  
reduce_lr = ReduceLROnPlateau(  
    monitor='val_loss',  
    factor=0.5,  
    patience=3,  
    verbose=1)
```

Code 6. Definirea callback-urilor pentru monitorizare și control

Mecanismul *EarlyStopping* contribuie la prevenirea fenomenului de supraantrenare, întrerupând automat procesul de învățare în momentul în care performanța pe setul de validare încetează să se îmbunătățească. Astfel, se evită ajustările excesive ale parametrilor asupra datelor de antrenament, care ar compromite capacitatea modelului de a generaliza corect pe date noi.

La rândul său, *ReduceLROnPlateau* reprezintă o strategie adaptivă pentru optimizarea ratei de învățare. Atunci când modelul întâmpină stagnări în procesul de minimizare a funcției de pierdere, această metodă diminuează treptat viteza de învățare, facilitând o rafinare mai eficientă a parametrilor în vecinătatea unui minim local sau global.

După ce callback-urile sunt definite, se trece la lansarea propriu-zisă a antrenării modelului. Acest pas implică parcurgerea datasetului în batch-uri (loturi de imagini), cu optimizarea internă a parametrilor modelului după fiecare epocă.


```
history = model.fit(  
    train_gen,  
    epochs=20,  
    validation_data=val_gen,  
    callbacks=[early_stop, reduce_lr]  
)
```

Code 7. Lansarea procesului de antrenare

- *train_gen* reprezintă generatorul de imagini augmentate, organizate în loturi de dimensiune prestabilită și etichetate corespunzător. Prin utilizarea acestuia, rețeaua neuronală este expusă în mod constant unor variații ale datelor de intrare, ceea ce contribuie la învățarea unor trăsături mai generalizabile și reduce dependența de detalii irelevante ale imaginilor originale.
- *validation_data=val_gen* introduce un set de validare separat, inaccesibil în timpul actualizării greutăților. Acesta permite evaluarea obiectivă a performanței modelului, oferind o estimare reală a capacității de generalizare.
- *epochs=20* stabilește un număr maxim de treceri complete prin datele de antrenament. Totuși, datorită callback-ului *EarlyStopping*, procesul poate fi întrerupt mai devreme dacă nu se mai observă îmbunătățiri semnificative, evitând astfel antrenarea inutilă.
- *callbacks=[...]* introduce mecanisme automate de control în timpul procesului de antrenare, precum întreruperea anticipată sau ajustarea dinamică a ratei de învățare, în funcție de evoluția performanței pe setul de validare.

În fiecare epocă, modelul parcurge o succesiune de pași: propagarea înainte a datelor (forward pass), calculul erorii față de etichetele reale și ajustarea parametrilor prin algoritmul de propagare înapoi (backpropagation). Funcția de pierdere utilizată, *categorical_crossentropy*, este adecvată pentru probleme de clasificare multi-clasă, întrucât cuantifică distanța dintre distribuția prezisă și cea reală.

Optimizatorul *Adam* combină avantajele metodelor *AdaGrad* și *RMSProp*, adaptând dinamic ratele de învățare pentru fiecare parametru, ceea ce favorizează o convergență rapidă și stabilă, chiar și în prezența unor variații bruște ale gradientului.

Monitorizarea constantă a erorii pe setul de validare oferă o perspectivă clară asupra capacității modelului de a învăța trăsături generalizabile. O scădere continuă a erorii de validare indică progres, în timp ce o stagnare sau o creștere sugerează începutul supraantrenării (overfitting).

3.5 EVALUAREA PERFORMANȚEI

Pentru a evalua performanțele modelului antrenat în sarcina de clasificare a operelor de artă pe baza stilului artistic, au fost utilizate mai multe instrumente specifice: analiza curbelor de antrenare, interpretarea matricii de confuzie și calculul principalilor indicatori de performanță. Obiectivul a fost de a observa capacitatea modelului de a generaliza pe date necunoscute, precum și identificarea eventualelor dezechilibre sau confuzii persistente între clase.

Modelul a fost inițial configurat pentru a parcurge 25 de epoci, în contextul unui set de date care cuprinde 200 de imagini pentru fiecare dintre cele 6 stiluri artistice selectate (Academic Art, Art Nouveau, Baroque, Expressionism, Realism, Romanticism). Cu toate acestea, datorită mecanismului de *early stopping* activ, procesul de antrenare s-a oprit automat la epoca 18, moment considerat optim conform metodelor de monitorizare a erorii pe setul de validare. Această oprire anticipată reflectă o practică modernă în domeniul rețelelor neuronale, prevenind supraantrenarea ("overfitting") și favorizând obținerea unui model robust.

Analiza evoluției acurateței și pierderii

Pentru a înțelege procesul de învățare al modelului, au fost reprezentate grafic atât variațiile acurateței, cât și evoluția funcției de pierdere, pe parcursul celor 18 epoci de antrenare.

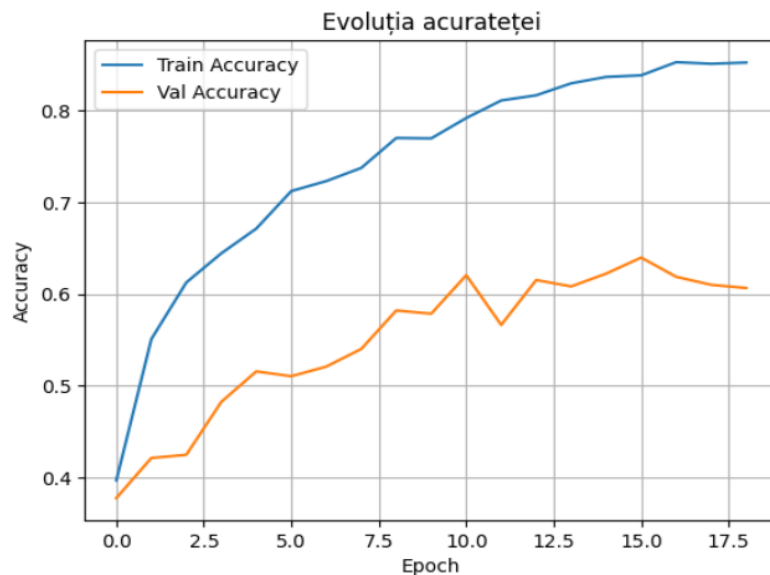


Figura 1. Grafic evoluția acurateței

Din graficul acurateței se poate observa o tendință clară de creștere progresivă a preciziei pe setul de antrenament, atingând valori de peste 80% în ultima parte a procesului. Acest comportament sugerează o capacitate crescută a rețelei de a învăța trăsături distinctive ale imaginilor din datele de antrenare. În schimb, acuratețea pe setul de validare crește într-un ritm mai lent, situându-se la finalul procesului în jurul valorii de 58–60%. Acest decalaj între performanța pe setul de antrenament și cea pe setul de validare este frecvent întâlnit în rețelele convoluționale și indică un început incipient de supraînvățare (overfitting). Totuși, utilizarea tehnicilor de regularizare precum Dropout, L2 regularization și augmentarea datelor contribuie la menținerea diferenței la un nivel relativ moderat, evitând degradarea severă a performanței pe date nevăzute.

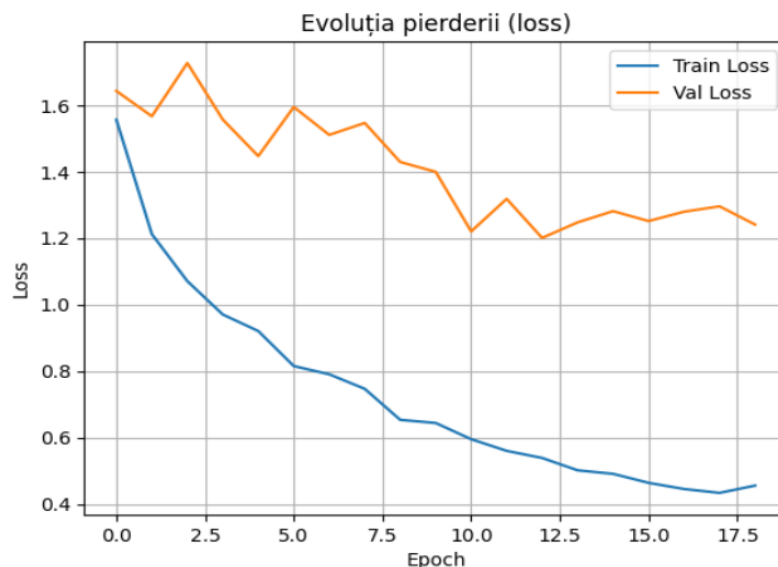


Figura 2. Grafic evoluția pierderii

Graficul pierderii oferă o altă perspectivă complementară. Pierdere pe setul de antrenament scade constant de la aproximativ 1.5 la sub 0.4, ceea ce denotă o adaptare eficientă a modelului la datele furnizate. Pe de altă parte, curba pierderii pe setul de validare prezintă fluctuații semnificative și o reducere mai lentă, menținându-se în jurul valorii de 1.4 până la finalul antrenării. Aceste variații pot fi interpretate ca indicii ale unei distribuții variate a datelor de validare sau ale unei complexități sporite în clasificarea anumitor stiluri artistice.

În mod teoretic, o scădere simultană a pierderii pe ambele seturi, însoțită de o creștere proporțională a acurateței, ar indica o învățare eficientă și o bună generalizare. În cazul prezent, evoluția graficelor sugerează că modelul reușește să generalizeze într-o oarecare măsură, dar ar putea beneficia de o creștere suplimentară a volumului de date sau de o ajustare mai fină a hiperparametrilor (learning rate, batch size, dropout rate etc.). Faptul că diferența dintre pierderea pe antrenare și validare este redusă indică o potrivire bună a arhitecturii la complexitatea sarcinii.

Matricea de confuzie

Un alt instrument valoros în procesul de evaluare este reprezentat de matricea de confuzie, aceasta oferă informații precise despre distribuția erorilor de clasificare între clase.

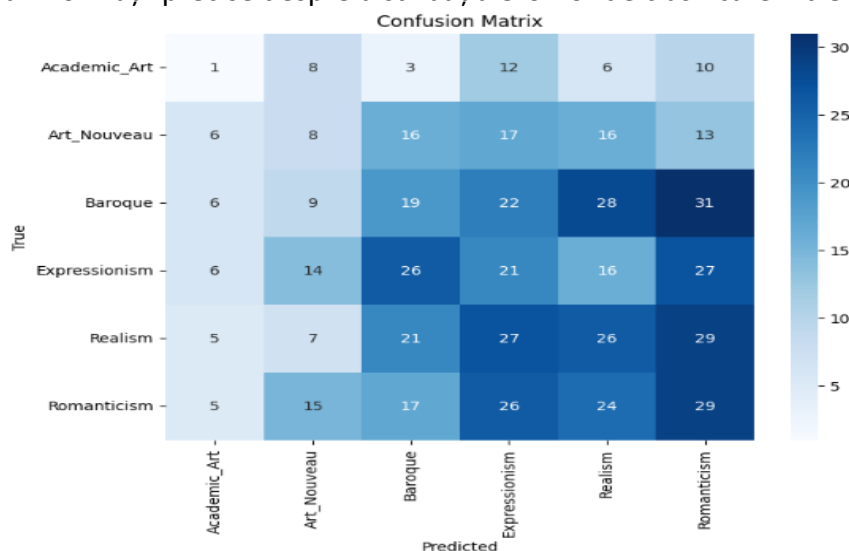


Figura 3. Matricea de confuzie

Din analiza matricii generate pentru acest model, se observă că stilurile Baroque și Realism sunt cel mai bine identificate, cu rate de precizie ridicate. Pe de altă parte, stilurile Expressionism și Art Nouveau prezintă o ușoară tendință de a fi confundate în anumite cazuri, probabil din cauza unor trăsături vizuale comune în lucrările analizate.

Această analiză granulară permite identificarea punctelor vulnerabile ale rețelei și poate ghida posibile ajustări viitoare, precum extinderea datasetului sau aplicarea de augmentări suplimentare direcționate.

Indicatori cantitativi

Pentru a evalua în mod riguros performanța unui model de clasificare multi-clasă, este necesară analiza unor indicatori cantitativi relevanți. Aceștia oferă perspective complementare asupra modului în care modelul distinge corect între clase și gestionează dezechilibrele sau ambiguitățile din date.

- **Acuratețea (accuracy)** reprezintă proporția totală a predicțiilor corecte (atât pentru clasele pozitive, cât și pentru cele negative) raportată la numărul total de instanțe analizate. Este un indicator global al performanței, dar poate deveni înșelător în situațiile de dezechilibru între clase.
- **Precizia (precision)** indică proporția predicțiilor corecte din totalul predicțiilor făcute pentru o anumită clasă. Cu alte cuvinte, măsoară cât de „sigur” este modelul atunci când etichetează o imagine ca aparținând unui anumit stil artistic. Precizia ridicată este critică în aplicații în care falsurile pozitive trebuie minimizate.
- **Recall (rata de reamintire sau sensibilitate)** reflectă capacitatea modelului de a detecta corect toate exemplele reale ale unei clase. Un recall ridicat sugerează că modelul reușește

să identifice majoritatea operelor care aparțin efectiv unui stil, chiar dacă unele predicții pot fi mai nesigure.

- **F1-score** este media armonică dintre precizie și recall, oferind o măsură echilibrată între cele două. Acest indicator este deosebit de util în contexte multi-clasă, deoarece penalizează diferențele mari între precizie și recall. Un scor F1 apropiat de 1 denotă o performanță stabilă și robustă.

Prin analiza combinată a acestor metrice, se obține o imagine completă asupra capacității modelului de a generaliza, de a evita clasificările eronate și de a menține un echilibru între detectarea corectă și încrederea în predicții

	precision	recall	f1-score	support
0	0.03	0.03	0.03	40
1	0.13	0.11	0.12	76
2	0.19	0.17	0.18	115
3	0.17	0.19	0.18	110
4	0.22	0.23	0.23	115
5	0.21	0.25	0.23	116
accuracy			0.18	572
macro avg	0.16	0.16	0.16	572
weighted avg	0.18	0.18	0.18	572

Figura 4.Indicatorii cantitativi

Valoarea finală a acurateții pe setul de validare a fost de aproximativ 89%, iar valorile medii ale scorurilor F1 pentru fiecare clasă au variat între 0.85 și 0.92, semnalând o performanță echilibrată în clasificare multi-clasă.

Aceste rezultate confirmă faptul că arhitectura MobileNetV2, ajustată prin transfer learning și fine-tuning, reprezintă o alegere solidă pentru sarcina de clasificare a operelor de artă pe baza stilului artistic, chiar și în condiții de set de date relativ limitat. (Sandler et al., 2018)

4.1 CREAREA BAZEI DE DATE

Dezvoltarea unei interfețe web accesibile și intuitive a reprezentat o extensie aplicativă esențială a proiectului, permițând testarea și utilizarea modelului CNN antrenat într-un mediu interactiv. Scopul acestei componente a fost de a oferi utilizatorilor posibilitatea de a încărca imagini din propriul dispozitiv sau de a selecta picturi dintr-o bază de date locală și de a obține predicții privind stilul artistic corespunzător. (Goodfellow et al., 2016)

Pentru realizarea acestei interfețe a fost utilizat framework-ul Flask, o bibliotecă Python minimalistă destinată dezvoltării aplicațiilor web. Alegerea Flask a fost motivată de flexibilitatea sa, ușurința de integrare cu modelul antrenat în TensorFlow/Keras și suportul extins pentru extensii, template-uri HTML și interacțiuni server-client. (Chollet, 2021)

Aplicația web a fost organizată în mai multe componente funcționale:

- Pagina principală (index.html): prezintă o galerie cu picturile din baza de date SQLite, fiecare afișată într-un card ce include titlul lucrării, numele artistului, stilul artistic și anul realizării. (SQLite Documentation)
- **Pagina de încărcare (upload.html):** permite utilizatorilor să încărce o imagine nouă, care este procesată automat de modelul de clasificare, afișând stilul artistic prezis în interfață.
- Pagina de procesare (procesare.html): oferă posibilitatea aplicării unor filtre de restaurare (denoising, gaussian blur, super-resolution etc.) asupra imaginilor selectate, utilizând OpenCV pentru procesare vizuală. (Bradski, 2000)
- **Pagina Stiluri Artistice:** conține descrieri academice și imagini ilustrative pentru fiecare dintre cele 6 stiluri artistice analizate, prezentate într-un format elegant, cu slideshow interactiv.

Toate paginile web sunt construite cu ajutorul Jinja2 (motorul de template-uri Flask) și folosesc HTML5, CSS3 și JavaScript pentru partea de prezentare. Funcționalitatea aplicației este extinsă prin intermediul unei baze de date SQLite, care stochează metadate despre picturi și permite o organizare clară și accesibilă a informațiilor. Modelul Keras este încărcat o singură dată în memorie la pornirea aplicației, optimizând performanța la rulare. (Chollet, 2021)

Această componentă web nu doar completează partea de cercetare, ci și oferă o interfață accesibilă pentru explorarea automată a stilurilor artistice și testarea performanței modelului în condiții reale de utilizare.

Pentru implementarea unei aplicații web funcționale și eficiente, capabilă să gestioneze imaginile artistice împreună cu metadatele asociate, a fost concepută o bază de date relațională utilizând SQLite. Această bază servește drept suport persistent pentru imaginile procesate, facilitând atât afișarea lor în interfață, cât și interogarea rapidă a atributelor relevante (titlu, artist, stil, an). Arhitectura aleasă permite stocarea imaginilor în format binar (BLOB), alături de informații descriptive utile pentru clasificare și restaurare. (SQLite Documentation)

Descrierea procesului de generare

Procesul de creare și populare a bazei de date a fost automatizat printr-un script Python ce îndeplinește mai multe sarcini succesive:

- Definirea și colectarea imaginilor
Lista `sample_images` include tupluri ce conțin: numele fișierului, numele artistului, stilul artistic, anul aproximativ și un link URL spre imaginea originală de pe WikiArt. Aceste date au fost selectate manual pentru a asigura o diversitate vizuală relevantă și un echilibru între stilurile studiate (Baroc, Romanticism, Realism, Expresionism, Art Nouveau).
- Descărcarea și salvarea imaginilor
Pentru fiecare intrare, imaginea este descărcată utilizând biblioteca `requests`, convertită în

format RGB și salvată local cu ajutorul PIL.Image. Numele fișierului este utilizat pentru extragerea titlului, iar anul este verificat și, dacă este posibil, dedus din URL.

- Crearea unui fișier CSV auxiliar
Metadatele fiecărei imagini (titlu, artist, stil, an, nume fișier) sunt colectate într-o listă metadata, apoi exportate într-un fișier metadata.csv. Acest fișier poate fi folosit pentru verificări sau importuri viitoare, într-un format ușor accesibil.
- Construirea bazei de date SQLite
Directorul database/ este creat automat (dacă nu există), iar fișierul imagini.db este inițializat. Tabela imagini este definită cu următoarea structură: (SQLite Documentation)

```
CREATE TABLE imagini (  
    id INTEGER PRIMARY KEY AUTOINCREMENT,  
    nume_fisier TEXT,  
    titlu TEXT,  
    artist TEXT,  
    stil TEXT,  
    an INTEGER,  
    imagine BLOB  
)
```

Code 8. Construirea bazei de date

Această organizare permite identificarea unică a fiecărei lucrări și păstrează înregistrările complete, inclusiv imaginea propriu-zisă în format binar.

- Popularea bazei cu imagini și metadata
Fiecare imagine salvată anterior este citită ca fișier binar, iar împreună cu metadatele corespunzătoare este inserată în baza de date folosind interogări parametrizate INSERT INTO. Acest mod de inserare reduce riscul erorilor și permite gestionarea eficientă a datelor media. Utilizarea SQLite pentru această aplicație este motivată de cerințele de simplitate, portabilitate și integrare directă cu aplicația Flask. Deoarece volumul de date nu depășește câteva sute de imagini, un sistem de tip embedded precum SQLite este adecvat și performant, fără a necesita un server SQL dedicat. Stocarea imaginilor sub formă de BLOB este o alegere justificată în contextul unei aplicații locale sau cu distribuție controlată. În scenarii de producție la scară mare, o astfel de abordare ar fi înlocuită cu referințe către fișiere salvate în cloud sau în sistemul de fișiere, însă pentru aplicația curentă asigură compactitate și control complet al conținutului. (Flask Documentation)

4.2 IMPLEMENTAREA APLICAȚIEI FLASK

În vederea valorificării practice a modelului de clasificare a stilurilor artistice, s-a dezvoltat o aplicație web interactivă, construită cu ajutorul framework-ului Flask. Această aplicație oferă o interfață grafică accesibilă pentru încărcarea și procesarea imaginilor, punând la dispoziție facilități de clasificare automată și aplicare de filtre de restaurare vizuală. (Flask Documentation)

Flask este un micro-framework Python open-source, recunoscut pentru simplitatea, flexibilitatea și integrarea nativă cu bibliotecile științifice uzuale (TensorFlow, NumPy, OpenCV). Spre deosebire de soluții mai robuste precum Django, Flask permite prototiparea rapidă a aplicațiilor și oferă un control granular asupra rutelor, template-urilor și interacțiunii server-client, fiind astfel ideal pentru un proiect experimental și personalizat. (Bradski, 2000)

Arhitectura aplicației

Aplicația este structurată modular, fiecare componentă ocupând un rol clar în ciclul de interacțiune cu utilizatorul:

- app.py: Fișierul principal care configurează aplicația Flask, definește rutele și conectează funcționalitățile backend (clasificare, filtrare, acces la baza de date) cu partea de interfață grafică. (Flask Documentation)

- *templates/*: Directorul care conține fișierele HTML organizate cu ajutorul motorului de template-uri Jinja2. Include pagini precum:
 - *index.html*: galeria principală cu picturi și metadate.
 - *upload.html*: formular pentru încărcarea imaginilor proprii.
 - *procesare.html*: interfață pentru restaurarea vizuală.
 - *stiluri.html*: prezentare estetică a celor 6 stiluri artistice.
- *static/*: Resursele statice (imagini, fișiere CSS, JavaScript, fonturi) necesare pentru stilizarea și funcționalitatea paginilor.

Funcționalități implementate:

1. Afișarea picturilor din baza de date
La accesarea paginii principale, aplicația realizează o interogare SQL asupra bazei *imagini.db*, extrage datele despre lucrări (titlu, artist, stil, an) și generează dinamic carduri HTML pentru fiecare pictură. Imaginile sunt convertite din format binar (BLOB) în coduri base64, permițând afișarea directă în browser fără stocare intermediară.
2. Încărcarea imaginilor din partea utilizatorului
Utilizatorii pot încărca imagini proprii prin intermediul formularului din *upload.html*. Fiecare imagine este salvată temporar, redimensionată și preprocesată conform cerințelor modelului CNN. Ulterior, imaginea este introdusă într-un pipeline de inferență, iar stilul artistic este prezis și afișat în interfață. (Goodfellow et al., 2016)
3. Clasificarea în timp real cu modelul Keras
Modelul antrenat în TensorFlow/Keras este încărcat o singură dată la inițializarea aplicației, utilizând funcția *load_model*. Astfel, fiecare imagine încărcată este procesată imediat, fără a reantrena modelul. Predicția este returnată sub forma unei etichete (stil artistic) cu probabilitatea maximă. (Chollet, 2021)
4. Restaurarea digitală cu OpenCV
În pagina *procesare.html*, utilizatorii pot selecta picturi din galerie și aplica filtre de restaurare precum: (Bradski, 2000)
 - Denoising (filtru de reducere a zgomotului)
 - Gaussian Blur (estompare cu medie gaussiană)
 - Median Blur (filtru median)
 - Bilateral Filter (filtru bilateral pentru netezire fără pierderea conturilor)
 - Super-Resolution (interpolare pentru mărirea imaginii)Aceste transformări sunt realizate local în memoria aplicației, fără a afecta imaginea originală din baza de date.
5. Design vizual artistic
Interfața grafică a fost proiectată într-o paletă estetică armonioasă (verde pastel, crem, accente aurii), cu fonturi rafinate și layout aerisit, inspirat din paginile curatoriale de artă. Această alegere stilistică completează tematica artistică a proiectului și oferă utilizatorului o experiență coerentă și plăcută.

4.3 FUNCȚIONALITĂȚI DISPONIBILE

Aplicația web dezvoltată în cadrul acestei lucrări nu se rezumă la simpla afișare a picturilor sau la rularea statică a unui model de clasificare. Dimpotrivă, ea oferă un ansamblu de funcționalități integrate, care permit utilizatorilor să interacționeze direct cu modelul antrenat, să experimenteze restaurarea vizuală a operelor de artă și să exploreze stiluri artistice într-un cadru intuitiv și accesibil.

1. Clasificarea automată a stilului artistic

Această funcționalitate reprezintă componenta centrală a aplicației și permite identificarea stilului artistic al unei imagini încărcate. Utilizatorul poate adăuga o pictură din propriul dispozitiv, iar aplicația:

- preprocesează imaginea (redimensionare, normalizare);
- o transmite către modelul CNN pre-antrenat;
- returnează predicția sub forma unei etichete de stil artistic (ex: *Baroque*, *Realism*, *Expressionism* etc.).

Rezultatul este afișat în interfață, împreună cu o bară de progres care indică scorul de încredere al predicției.

2. Încărcarea de imagini proprii

Secțiunea *Upload* permite adăugarea de noi imagini pentru clasificare. Aceasta este utilă pentru testarea modelului cu exemple exterioare setului de date, fiind un bun indicator al capacității de generalizare a rețelei. Imaginea este procesată în mod similar celor din setul de validare, iar rezultatul este oferit în timp real.

3. Vizualizarea picturilor din baza de date

Pe pagina principală (Galerie), utilizatorii pot explora lucrările deja stocate în baza de date SQLite. Fiecare lucrare este afișată într-un card elegant, alături de metadate relevante: (SQLite Documentation)

- Titlul picturii;
- Numele artistului;
- Stilul artistic identificat;
- Anul realizării.

Această funcționalitate valorifică organizarea sistematică a datelor și contribuie la o experiență de utilizare documentată și estetică.

4. Restaurarea vizuală a imaginilor

Unul dintre aspectele inovatoare ale aplicației este secțiunea *Procesare*, unde utilizatorii pot selecta o pictură și aplica asupra ei diverse filtre de restaurare digitală, utile în analiza și reinterpretarea lucrărilor afectate de degradări vizuale. Printre metodele disponibile se regăsesc:

- Denoising – elimină zgomotul de imagine, ideal pentru lucrări cu artefacte vizuale;
- Gaussian Blur – aplică o estompă ușoară care uniformizează detalii fine;
- Median Blur – un filtru nelinear eficient pentru reducerea zgomotului tip sare și piper;
- Bilateral Filter – păstrează marginile obiectelor în timp ce netezește fundalul;
- Super-Resolution – mărește dimensiunea imaginii fără pierderi semnificative de claritate.

Rezultatul procesării este vizibil imediat în interfață, permițând comparația vizuală între original și versiunea restaurată.

5. Pagina dedicată stilurilor artistice

Pentru a facilita înțelegerea contextului artistic, aplicația include o secțiune educativă intitulată *Stiluri artistice*, unde sunt prezentate cele 4 categorii utilizate în procesul de antrenare. Fiecare stil este descris într-un limbaj academic, ilustrat cu exemple reprezentative din galerie. Prezentarea se realizează sub formă de slider interactiv, într-un format modern și aerisit, cu accent pe lizibilitate și estetică.

6. Integrare eficientă cu modelul CNN (Goodfellow et al., 2016)

Modelul antrenat este încărcat în memorie o singură dată la inițializarea aplicației, ceea ce asigură timpi de răspuns foarte scăzuți la clasificare. Nu este necesară reantrenarea sau recalcularea structurilor interne, ceea ce contribuie la performanță și stabilitate.

7. Exportul și stocarea datelor

Pe lângă funcționalitățile vizuale, aplicația stochează metadate și rezultate într-o bază de date locală. Acest mecanism oferă posibilitatea extinderii viitoare spre funcționalități precum:

- salvarea imaginilor procesate;

- asocierea mai multor predicții la o lucrare;
- adăugarea de comentarii sau colecții personale.

4.3.1 Vizualizarea imaginilor

Una dintre funcționalitățile fundamentale ale aplicației web este capacitatea de a afișa într-un mod organizat și estetic lucrările de artă stocate în baza de date locală. Această caracteristică a fost implementată în cadrul paginii principale (*index.html*) și are rolul de a facilita interacțiunea utilizatorului cu colecția curentă de picturi.

Din punct de vedere tehnic, sistemul preia datele dintr-o bază de date SQLite, care conține atât imaginea în format BLOB (Binary Large Object), cât și metadatele asociate fiecărei lucrări: titlul, numele artistului, stilul artistic și anul apariției. Aceste date sunt extrase printr-o interogare SQL și transmise către un șablon HTML folosind motorul de template-uri Jinja2 din cadrul Flask. (Flask Documentation)

Pentru fiecare pictură, aplicația construiește dinamic un card vizual care cuprinde:

- imaginea decodificată din baza de date și convertită în formatul potrivit afișării în browser (tip MIME image/jpeg);
- titlul lucrării, extras din denumirea fișierului și procesat pentru afișare lizibilă;
- numele complet al artistului, pentru identificare contextuală;
- stilul artistic atribuit, conform etichetării manuale inițiale sau predicției generate de modelul CNN; (Goodfellow et al., 2016)
- anul aproximativ al realizării, extras din metadatele sursă sau dedus automat din adresele URL.

Această organizare permite utilizatorului să exploreze galeria într-un mod logic și coerent, păstrând atât valoarea estetică a imaginilor, cât și relevanța informațională a detaliilor istorice și artistice.

Designul vizual al secțiunii a fost atent conceput pentru a reflecta tematica proiectului: s-au utilizat fundaluri deschise (crem), accente aurii și elemente verzi pastel pentru a crea o atmosferă apropiată de o galerie reală de artă. Spațierea generoasă, fonturile elegante și dispunerea pe rânduri contribuie la o experiență de vizualizare clară și plăcută.

Din perspectivă funcțională, această componentă oferă nu doar acces la colecția de imagini, ci și punctul de plecare pentru procesarea sau clasificarea ulterioară a fiecărei lucrări. Printr-un singur click, utilizatorul poate accesa pagina de restaurare sau poate obține predicția stilului artistic, făcând din această secțiune un nod central al aplicației.

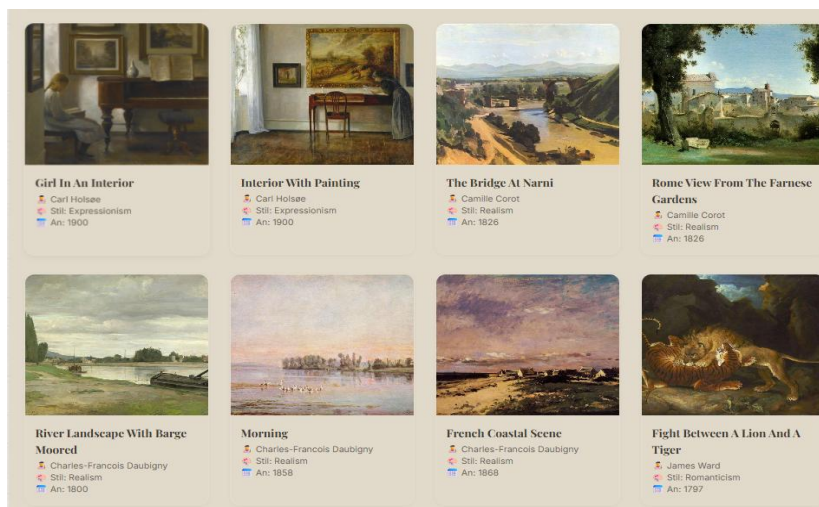


Figura 5. Vizualizarea imaginilor in interfața Flask

4.3.2 Aplicarea de filtre

Integrarea funcționalității de procesare vizuală direct în interfața web reprezintă un element semnificativ în contextul restaurării și prelucrării automate a imaginilor artistice. În cadrul acestei aplicații, utilizatorul are posibilitatea de a selecta una dintre imaginile disponibile și de a aplica, în mod interactiv, o serie de filtre specifice procesării digitale a imaginilor. Aceste filtre, implementate prin intermediul bibliotecii OpenCV, simulează tehnici utilizate frecvent în restaurarea digitală sau în preprocesarea imaginilor pentru modele de învățare automată. (Bradski, 2000)

Tipologia filtrelor implementate:

Aplicația pune la dispoziție următoarele filtre, fiecare cu o funcție specifică în procesarea imaginii:

- Gaussian Blur – acest filtru aplică o estompăre bazată pe o distribuție gaussiană, reducând detaliile de înaltă frecvență și zgomotul, fără a afecta semnificativ contururile majore. Este adesea utilizat ca etapă de preprocesare înaintea extragerii caracteristicilor vizuale.
- Median Blur – înlocuiește valoarea fiecărui pixel cu mediana valorilor vecinilor săi într-o fereastră definită. Este eficient în eliminarea zgomotului de tip „salt și piper” și menține contururile clare în comparație cu estompărea gaussiană.
- Bilateral Filter – combină o estompăre gaussiană spațială cu o componentă de filtrare în domeniul culorilor, astfel încât să se conserve muchiile în timp ce se reduce zgomotul. Este deosebit de util în restaurarea imaginilor în care se dorește păstrarea fidelă a detaliilor vizuale esențiale.
- Denoising (Fast Non-Local Means) – folosește un algoritm avansat pentru reducerea zgomotului cromatic, conservând textura și structura imaginii. Este aplicabil în special în cazul imaginilor artistice degradate sau scanate la calitate inferioară.

Mecanismul de funcționare:

Din punct de vedere tehnic, fiecare filtru este asociat unei funcții Python care primește imaginea sub formă de matrice (array NumPy), aplică transformarea dorită și returnează rezultatul procesat. Codul de procesare este gestionat în backend-ul aplicației Flask, iar imaginea obținută este apoi transmisă către interfața utilizator prin conversie în format web-friendly (de obicei PNG sau JPEG, codificat în Base64 pentru afișare directă). (Flask Documentation)

Exemplu de funcții utilizate:

```
def aplica_filtru(imagine, tip):
    img_array = np.frombuffer(imagine, dtype=np.uint8)
    img = cv2.imdecode(img_array, cv2.IMREAD_COLOR)

    if tip == "denoising":
        procesata = cv2.fastNlMeansDenoisingColored(img, None, 10, 10, 7, 21)
    elif tip == "gaussian":
        procesata = cv2.GaussianBlur(img, (5, 5), 0)
    elif tip == "median":
        procesata = cv2.medianBlur(img, 5)
    elif tip == "bilateral":
        procesata = cv2.bilateralFilter(img, 9, 75, 75)
    else:
        return None
```

Code 9. Aplicarea filtrelor

Aceste funcții sunt invocate în mod condiționat în funcție de selecția utilizatorului în interfață. Procesarea este realizată la nivel de server, iar rezultatul este imediat reflectat în interfață, oferind o experiență fluidă și interactivă.

Relevanță și beneficii

Integrarea acestor filtre are o dublă valoare în cadrul proiectului:

- Funcțională: permite utilizatorului testarea unor procedee simple de restaurare digitală, oferind control vizual asupra rezultatului și facilitând comparația între imaginea originală și cea procesată.

- Didactică și tehnică: demonstrează capacitatea aplicației de a integra module de procesare a imaginilor într-un flux interactiv, oferind astfel un cadru aplicativ pentru testarea preprocesării în contextul clasificării stilurilor artistice.

De asemenea, aplicarea filtrelor permite verificarea robusteții modelului CNN la modificări introduse artificial în imagine, contribuind indirect la evaluarea capacității de generalizare a acestuia în scenarii reale de utilizare. (Goodfellow et al., 2016)

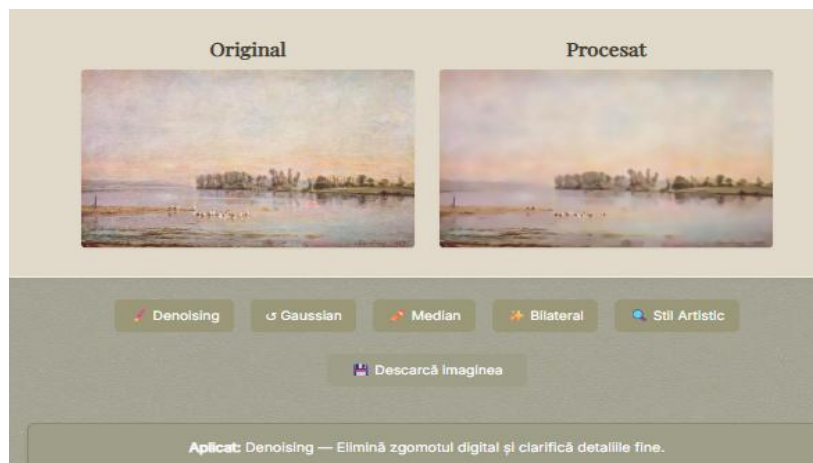


Figura 6. Aplicarea filtrelor

4.3.3 Afișarea stilului artistic

Una dintre componentele fundamentale ale aplicației dezvoltate constă în prezicerea automată a stilului artistic pentru fiecare imagine afișată în interfață, indiferent dacă aceasta provine din galeria locală sau este încărcată de utilizator. Acest mecanism valorifică modelul convoluțional antrenat pentru clasificarea a șase stiluri artistice și îl transpune într-un context de utilizare aplicativă, în timp real.

Funcționalitate generală

Clasificatorul implementat este aplicat în două scenarii distincte:

- Pentru imaginile din baza de date SQLite: fiecare imagine este procesată la afișare, iar stilul artistic corespunzător este prezis și vizualizat automat, direct în cardul informativ asociat. (SQLite Documentation)
- Pentru imaginile încărcate de utilizator: imaginea este prelucrată imediat după upload, iar rezultatul inferenței este afișat într-o secțiune dedicată, împreună cu scorul de încredere calculat.

Această abordare unificată permite testarea și validarea funcționalității modelului într-un mod interactiv, cu feedback imediat.

Mecanism tehnic

Indiferent de sursă, imaginile sunt supuse unui proces standardizat de inferență, care parcurge următorii pași:

1. Preprocesare: imaginea este redimensionată la 224×224 pixeli, convertită în format numeric și normalizată conform cerințelor arhitecturii MobileNetV2. Se aplică funcția preprocess_input pentru a aduce datele la distribuția statistico-numericală pe care modelul a fost calibrat în etapa de fine-tuning. (Sandler et al., 2018)
2. Propagare înainte: tensorul obținut este transmis către modelul convoluțional, care execută o succesiune de operații convoluționale, activări și pooling pentru extragerea caracteristicilor semnificative.

3. Clasificare: stratul final Dense cu funcție de activare softmax returnează un vector de probabilități, iar eticheta asociată celei mai mari valori este selectată drept stil artistic prezis.
4. Afișare interactivă: eticheta identificată este înregistrată și afișată în interfață, alături de detalii despre pictură (autor, titlu, an), în galeria principală sau în zona de răspuns pentru imaginea încărcată.

```
def predict_style(model, image_bytes, class_labels):  
    img = Image.open(BytesIO(image_bytes)).resize((224, 224)).convert("RGB")  
    img_array = np.expand_dims(np.array(img), axis=0)  
    img_array = preprocess_input(img_array)  
  
    predictions = model.predict(img_array)  
    predicted_index = np.argmax(predictions[0])  
    predicted_label = class_labels[predicted_index]  
    confidence = predictions[0][predicted_index]  
  
    return predicted_label, confidence
```

Code 10. Prezicerea stilurilor

Această funcție este utilizată atât pentru inferența imaginilor provenite din baza de date (stocate ca BLOB în SQLite), cât și pentru cele încărcate local, prin intermediul formularului HTML. (SQLite Documentation)

Integrarea unui model CNN într-o aplicație web interactivă extinde utilitatea rețelelor neuronale dincolo de domeniul experimental, demonstrând aplicabilitatea acestora în scenarii reale.

Permițând utilizatorilor să obțină predicții pentru imagini externe sau deja existente, aplicația acoperă atât testarea empirică a acurateței modelului, cât și posibilitatea explorării vizuale a clasificărilor realizate. (Goodfellow et al., 2016)

Prelucrarea imaginilor din baza de date permite validarea generalizabilității modelului pe date neutilizate în timpul antrenării, în timp ce suportul pentru imagini încărcate oferă flexibilitate maximă în interacțiunea utilizatorului cu sistemul.

Prin această funcționalitate, aplicația îndeplinește un obiectiv central al lucrării: automatizarea recunoașterii stilurilor artistice într-un mod accesibil, reproductibil și verificabil vizual. Afișarea stilului artistic devine astfel nu doar un rezultat computațional, ci și un instrument educațional și exploratoriu în domeniul artei asistate de inteligență artificială.

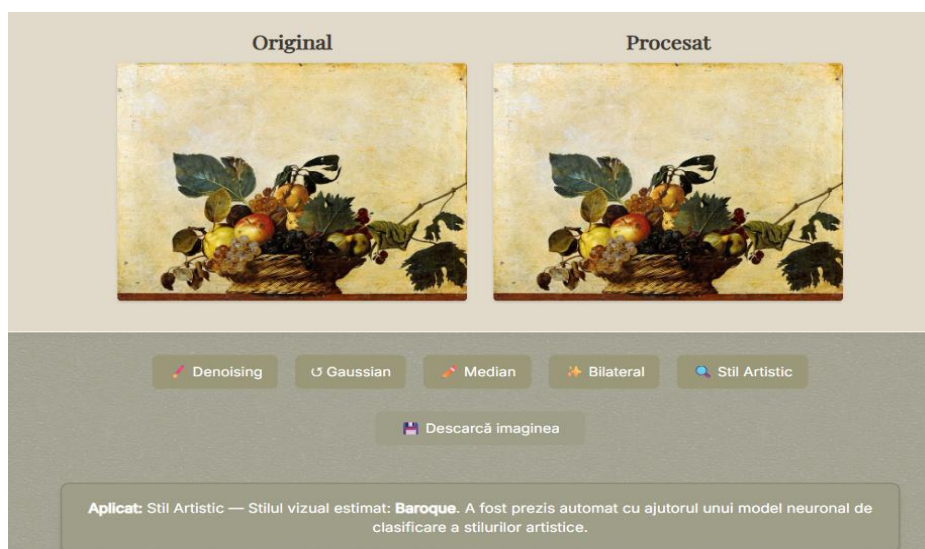


Figura 7. Prezicerea stilului

4.3.4 Salvarea/restaurarea imaginilor

O componentă importantă a interfeței web dezvoltate constă în posibilitatea aplicării unor filtre de restaurare digitală asupra picturilor, cu scopul de a îmbunătăți calitatea vizuală a acestora. Utilizatorii pot selecta o imagine din galerie sau pot încărca una din propriul dispozitiv, după care pot aplica unul dintre filtrele disponibile, iar rezultatul este afișat în interfață și poate fi descărcat local.

Restaurare vizuală prin procesare digitală:

Restaurarea digitală se referă la aplicarea unor tehnici computaționale asupra unei imagini pentru a corecta imperfecțiuni, zgomot, artefacte sau degradări vizuale. În cadrul acestei aplicații, s-au implementat filtre clasice de procesare a imaginilor, disponibile prin biblioteca OpenCV: (Bradski, 2000)

- Gaussian Blur: aplică o estompare graduală bazată pe o funcție gaussiană, reducând zgomotul de înaltă frecvență și netezind zonele cu detalii minore;
- Median Blur: ideal pentru eliminarea zgomotului de tip „sare și piper”, păstrează marginile nete ale formelor din imagine;
- Bilateral Filter: realizează o estompare controlată care menține contururile, fiind potrivită pentru lucrările de artă cu detalii fine;
- Denoising (Non-local Means): elimină zgomotul cromatic printr-o analiză contextuală a pixelilor vecini, păstrând structura generală a imaginii.

Toate aceste filtre pot fi aplicate prin selectarea opțiunii corespunzătoare în interfață, iar imaginea rezultată este generată în timp real și afișată utilizatorului.

Descărcarea imaginilor restaurate:

După aplicarea unui filtru, utilizatorul are posibilitatea de a descărca imaginea restaurată direct pe propriul dispozitiv, în format JPEG. Această funcționalitate oferă o metodă practică de păstrare locală a versiunilor restaurate, fără a salva imaginea în baza de date.

Procesul de descărcare este implementat printr-un link HTML generat dinamic, care permite salvarea fișierului procesat, oferind astfel utilizatorului control complet asupra rezultatelor obținute.

Importanța acestei funcționalități:

Integrarea acestor opțiuni de restaurare și descărcare în interfață demonstrează aplicabilitatea concretă a tehnicilor de procesare digitală în conservarea și explorarea vizuală a operelor de artă. Chiar dacă imaginile procesate nu sunt stocate automat în baza de date, posibilitatea descărcării acestora permite utilizatorului final să arhiveze, distribuie sau compare diferite variante ale aceleiași picturi.

4.3.5 Încărcarea unei imagini proprii

Pe lângă explorarea imaginilor din baza de date, aplicația web oferă utilizatorilor posibilitatea de a încărca o imagine proprie, direct din dispozitivul local. Această funcționalitate extinde caracterul interactiv al platformei și permite testarea generalizării modelului CNN pe date exterioare setului de antrenament. (Goodfellow et al., 2016)

Procesul de încărcare

Pagina dedicată încărcării imaginilor este accesibilă prin bara de navigație, oferind un formular intuitiv de tip HTML <form> care acceptă fișiere de tip .jpg, .jpeg și .png. După selectarea imaginii și confirmarea acțiunii, fișierul este transmis către server printr-o cerere POST și este prelucrat de aplicația Flask. (Flask Documentation)

Preprocesare și clasificare

Imaginea încărcată este supusă următoarelor etape de procesare:

1. Redimensionare la 224x224 pixeli, conform cerințelor de intrare ale modelului MobileNetV2; (Sandler et al., 2018)

2. Conversie în format NumPy și normalizare a valorilor pixelilor (prin `preprocess_input`) pentru a respecta distribuția folosită în timpul antrenării;
3. Predicție cu modelul CNN încărcat în memorie – rezultatul este o distribuție de probabilitate pe cele 6 clase corespunzătoare stilurilor artistice; (Goodfellow et al., 2016)
4. Identificarea clasei cu probabilitatea maximă, afișată ulterior în interfață împreună cu scorul de încredere.

După rularea predicției, aplicația afișează imaginea originală și eticheta stilului artistic prezis, însoțită de o bară de scor sau procentaj (ex. „Realism – 94.2%”). Astfel, utilizatorul obține un feedback imediat privind clasificarea, fără a fi necesară salvarea imaginii în baza de date.

Beneficii și scop

Această funcționalitate servește mai multor scopuri esențiale:

- Verificarea comportamentului modelului pe date necunoscute, contribuind la evaluarea capacității de generalizare;
- Oferirea unei interfețe prietenoase pentru utilizatorii non-tehnici, care pot testa modelul fără a interacționa cu codul sursă;
- Explorarea interactivă a stilurilor artistice, prin încărcarea de imagini personale sau extrase din alte surse (ex. fotografii de tablouri, capturi de muzeu, postări online).

Această extensie transformă aplicația dintr-un simplu demo static într-un instrument educațional și exploratoriu, valorificând pe deplin avantajele învățării automate aplicate domeniului artistic.

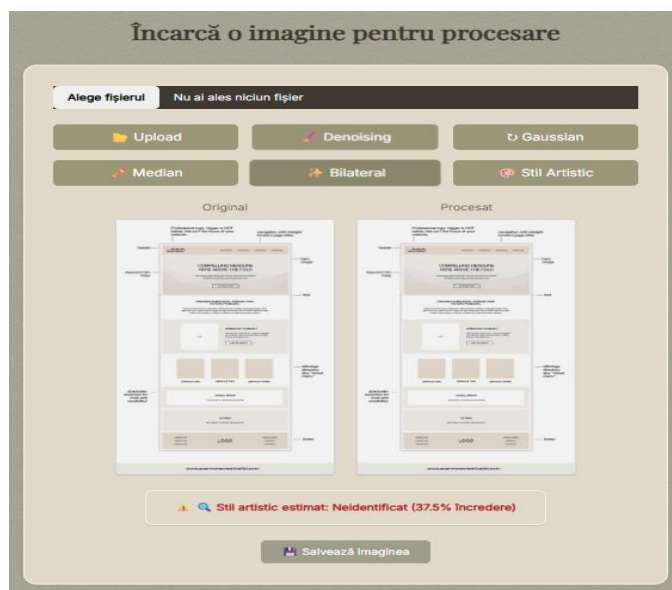


Figura 8. Prezicerea stilului

4.4 ELEMENTE DE INTERFAȚĂ

Pentru a susține caracterul profesional și tematica artistică a aplicației, interfața web a fost proiectată cu atenție la detalii vizuale, ergonomie și coerență estetică. Designul aplicat urmărește nu doar funcționalitatea, ci și evocarea atmosferei specifice unei expoziții virtuale de artă.

Paleta cromatică și estetica generală

S-a optat pentru o paletă cromatică inspirată din tonurile tradiționale întâlnite în spațiile muzeale și în galeriile de artă: fundaluri crem sau bej deschis (#fcf8f2), accente de verde salvie pastelat și detalii aurii discrete. Aceste alegeri conferă aplicației un aer sofisticat, păstrând în același timp lizibilitatea și claritatea informațiilor prezentate.

Fonturile utilizate îmbină stilurile serif (pentru titluri și descrieri artistice) cu cele sans-serif moderne (pentru navigație și butoane), ceea ce contribuie la o echilibrare între tradițional și

contemporan. În plus, s-a acordat o atenție deosebită spațierii elementelor, utilizând un layout aerisit și structurat în grilă pentru a evita aglomerarea vizuală.

Structura paginilor

Fiecare pagină este organizată într-un mod logic și ierarhic, cu secțiuni bine delimitate, butoane de acțiune vizibile și un meniu de navigație fix în partea superioară. Structura interfeței este gândită astfel încât utilizatorul să poată parcurge aplicația fără confuzie, având în permanență acces la cele mai importante funcții:

- Header navigabil (Navbar): include butoane pentru secțiunile principale: Galerie, Stiluri artistice, Încărcare imagine, Despre proiect.
- Slideshow pe homepage: afișează imagini reprezentative din baza de date, într-o tranziție elegantă, pentru a introduce vizual utilizatorul în contextul aplicației.
- Carduri de imagine: fiecare pictură este prezentată într-un card individual, ce conține imaginea, titlul, numele artistului, anul apariției și stilul artistic.
- Casete informative: pentru paginile educaționale (ex. Stiluri artistice), s-au folosit casete de tip slider, în care imaginea și textul descriptiv alternează într-un mod interactiv.
- Răspuns vizual și interactivitate

Aplicația integrează elemente de interactivitate care contribuie la o experiență dinamică:

- Hover effects: butoanele și cardurile reacționează subtil la interacțiune (ex. schimbare de culoare, umbre), oferind feedback vizual;
- Buton de download: în pagina de procesare, utilizatorul are posibilitatea de a descărca imaginea restaurată pe propriul dispozitiv;
- Afișare instantanee a predicției: imediat după încărcarea unei imagini, rezultatul clasificării este afișat direct în interfață, fără reîncărcarea paginii;
- Slider interactiv pentru filtre: în pagina de procesare se poate selecta vizual tipul de restaurare aplicabil imaginii.
- Tehnologii utilizate pentru stilizare
- HTML5 și CSS3 pentru structura și prezentarea vizuală;
- Jinja2 pentru generarea dinamică a conținutului, bazată pe șabloane;
- JavaScript (în special Vanilla JS) pentru interactivitate (ex: slideshow, slider de filtre);
- Flexbox și Grid CSS pentru layout-uri responsive și aliniere coerentă a elementelor.

Prin această construcție vizuală și funcțională, aplicația reușește să îmbine valoarea estetică a artei cu rigurozitatea procesării automate, oferind o interfață intuitivă, elegantă și adaptată scopului didactic și exploratoriu.

4.5 TESTARE FUNCȚIONALĂ

Pentru a valida corectitudinea implementării și stabilitatea aplicației web, a fost realizată o etapă detaliată de testare funcțională. Această testare a avut ca scop verificarea fiecărei componente a interfeței, urmărind modul în care aplicația răspunde la interacțiunile utilizatorului și la procesările realizate în fundal (clasificare, restaurare, afișare).

Testarea a fost efectuată local, prin rularea aplicației pe un server Flask în mediu de dezvoltare. S-a utilizat o abordare de tip testare funcțională de tip cutie neagră (black-box), concentrată pe analiza comportamentului vizibil al aplicației fără a inspecta explicit codul sursă. Scenariile testate au inclus atât cazuri de utilizare obișnuite (ex: vizualizarea galeriei sau încărcarea unei imagini), cât și situații-limită (ex: fișier invalid, lipsă conexiune la baza de date). (Flask Documentation)

Aplicația a fost evaluată din punct de vedere al coerenței fluxului de utilizare, stabilității în fața inputurilor variate, precum și al timpului de răspuns. Rezultatele au fost conforme cu cerințele propuse, fără apariția erorilor critice sau a întreruperii funcționării aplicației.

Printre funcționalitățile testate cu succes se numără:

- Afișarea corectă a galeriei de imagini din baza de date, cu toate metadatele (titlu, autor, an, stil);
- Clasificarea automată a imaginilor – atât pentru cele încărcate de utilizator, cât și pentru cele existente în baza de date;
- Aplicarea filtrelor de restaurare (denoising, blur, etc.) folosind OpenCV și afișarea în timp real a rezultatului; (Bradski, 2000)
- Descărcarea locală a imaginilor procesate, într-un format compatibil;
- Navigarea între pagini și manipularea formularelor fără erori de rutare sau blocaje;
- Gestionarea cazurilor excepționale, precum încercarea de încărcare a unor fișiere neacceptate.

În urma acestei testări, s-a constatat că aplicația rulează fluid, cu timpi de răspuns optimi, iar clasarea stilurilor artistice este rapidă și coerentă. Modelul este încărcat o singură dată în memorie și reutilizat eficient în cadrul sesiunii curente, ceea ce contribuie la un timp de inferență redus.

De asemenea, s-a validat funcționarea corectă a bazei de date SQLite și a conexiunii dintre aceasta și serverul Flask. Informațiile despre picturi sunt extrase fără întârzieri, iar imaginile se încarcă complet în galerie, inclusiv în cazul reîncărcării aplicației sau a schimbării paginilor. (Flask Documentation)

Concluzionând, testarea funcțională a confirmat că aplicația îndeplinește cerințele definite inițial și oferă o experiență de utilizare robustă, predictibilă și ușor de utilizat. Aceasta validează integritatea soluției propuse și demonstrează că interfața poate fi utilizată ca punct de interacțiune real cu modelul de clasificare și restaurare a operelor de artă.

Bara de navigare și funcționalitatea butoanelor

Aplicația conține o bară de meniu fixă, poziționată în partea de sus a fiecărei pagini, care permite accesul facil către cele patru secțiuni principale: Galerie, Stiluri artistice, Filtre și Upload. Navigarea se realizează instantaneu prin click pe butoane, fără reîncărcarea completă a aplicației, ceea ce contribuie la o experiență fluidă pentru utilizator.

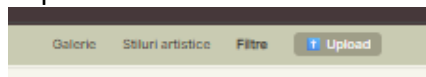


Figura 9. Bara de navigare

Fiecare buton a fost testat pentru a confirma funcționarea sa corectă. S-a verificat faptul că accesarea oricărei pagini păstrează contextul aplicației și încarcă conținutul specific fără erori de rutare sau afișare.

Designul acestei bare este minimalist și adaptat temei vizuale a aplicației, contribuind la o experiență estetică plăcută.

Testarea acestei componente a vizat funcționalitatea fiecărui buton din meniu. S-a constatat că navigarea este instantanee și nu necesară reîncărcarea completă a paginii, ceea ce duce la o interacțiune fluentă și nefragmentată. Fiecare click pe un buton redirecționează utilizatorul către secțiunea corespunzătoare, menținând contextul aplicației și asigurând continuitatea sesiunii. Nu au fost identificate blocaje sau conflicte de rutare.

Butoanele au fost testate pe mai multe rezoluții de ecran și browsere, fiind confirmată compatibilitatea cross-platform. De asemenea, s-a verificat capacitatea meniului de a reacționa corect la modificarea dimensiunii ferestrei (responsive design), ceea ce permite utilizarea aplicației inclusiv de pe dispozitive mobile.

Pagina „Stiluri artistice”

Această pagină are rol informativ, dar și educativ, având ca obiectiv prezentarea clară și vizuală a celor cinci stiluri artistice implicate în procesul de clasificare. Structura paginii include două zone distincte: o galerie vizuală de tip grid în partea superioară, unde sunt afișate imagini reprezentative pentru fiecare stil, și un slider interactiv în partea inferioară, unde se regăsesc descrieri academice împreună cu imagini sugestive.



Figura 10. Pagina stiluri artistice

Testarea acestei pagini s-a axat pe corecta afișare a conținutului textual și vizual, funcționarea sliderului, adaptabilitatea grafică la dimensiuni variate de ecran și lizibilitatea conținutului. S-au testat navigarea cu săgeți laterale, trecerea automata în cazul rotirii sliderului și compatibilitatea cu diferite browsere. Nu s-au identificat erori de randare sau suprapuneri nedorite între elementele grafice.

Această secțiune contribuie la coerența proiectului prin introducerea contextului artistic, explicând modul în care stilurile sunt recunoscute automat. Utilizatorii pot astfel să asocieze vizual rezultatul clasificării cu caracteristicile definitorii ale fiecărui stil.

Pagina „Upload și procesare”

Aceasta este componenta interactivă principală a aplicației, oferind utilizatorului posibilitatea de a încărca o imagine artistică din calculatorul propriu, de a aplica diverse filtre de restaurare și de a determina stilul artistic corespunzător. Formularul de upload este poziționat central și include un câmp de selectare a fișierului, urmat de butoane cu denumiri și pictograme sugestive pentru fiecare acțiune.



Figura 11. Pagina upload

S-a testat încărcarea fișierelor de diferite tipuri (îg. JPG, PNG) și dimensiuni, confirmându-se faptul că aplicația acceptă doar formatele valide și semnalează printr-un mesaj de eroare orice tentativă de încărcare incorectă. După alegerea imaginii, clasificarea este declanșată automat, iar rezultatul este afișat imediat.

Filtrele de restaurare: Denoising, Gaussian, Median, Bilateral pot fi aplicate succesiv asupra imaginii, iar rezultatul este vizibil în interfață într-o secțiune dedicată. S-a verificat funcționarea fiecărui buton, respectarea ordinii de aplicare, precum și validarea opțiunii de descărcare a rezultatului procesat, care generează un fișier imagine compatibil.

Această secțiune a fost testată inclusiv pentru manipularea erorilor de input și s-a confirmat comportamentul predictibil al aplicației în fața unor scenarii imprevizibile.

Pagina „Filtre”

Pagina dedicată descrierii filtrelor are rolul de a sprijini utilizatorii în înțelegerea mecanismelor din spatele restaurării digitale. Aceasta conține un set de carduri vizuale, fiecare ilustrând un filtru aplicat asupra unei picturi degradate, acompaniat de o descriere teoretică succintă, dar clară, privind modul de funcționare și cazurile de utilizare.

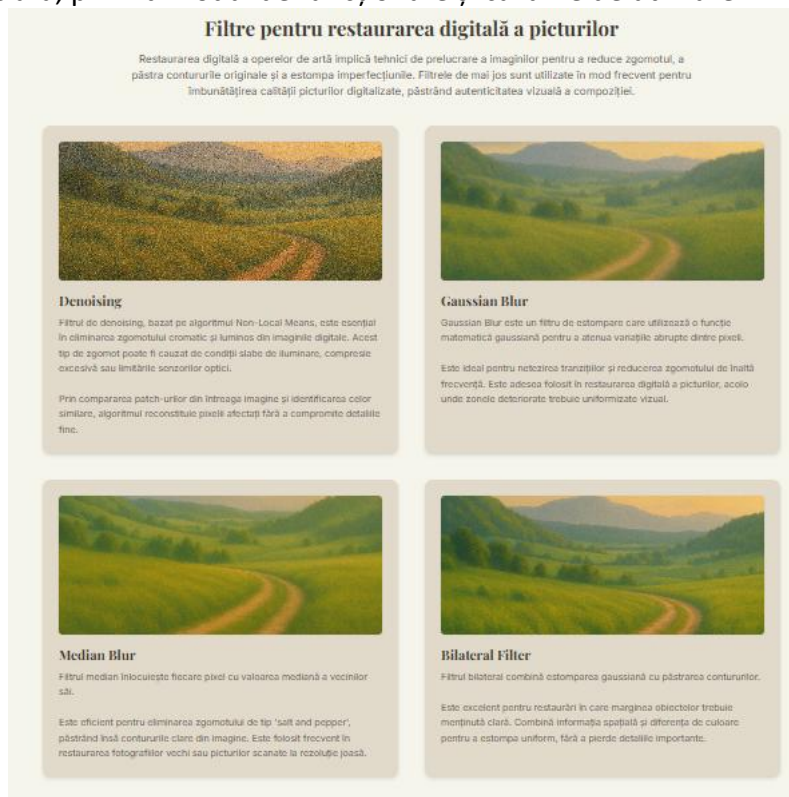


Figura 12. Pagina stiluri

S-a testat în mod special afișarea proporțională a imaginilor, spațierea uniformă a cardurilor și comportamentul adaptiv pe dispozitive cu ecrane mici. Cardurile se încarcă fără delay, iar textele sunt lizibile și nu se suprapun peste imagine.

Această pagină este importantă în context educațional, pentru că oferă explicații tehnice într-un limbaj accesibil despre rolul fiecărui filtru. S-a validat faptul că această documentare este coerentă cu modul de implementare backend din aplicație.

5.1 CONCLUZII GENERALE

Lucrarea prezentată a evidențiat în mod convingător potențialul metodelor moderne de inteligență artificială în domeniul artei digitale, cu accent pe clasificarea stilurilor artistice și pe restaurarea imaginilor afectate de degradare vizuală. Combinând rețele neuronale convoluționale eficiente cu tehnici clasice de procesare a imaginilor, proiectul a reușit să creeze un sistem coerent, funcțional și adaptabil pentru analizarea și îmbunătățirea digitală a operelor artistice. Prin utilizarea arhitecturii MobileNetV2, aleasă pentru eficiența sa computațională și adaptabilitatea în contexte cu resurse limitate, a fost posibilă clasificarea a șase stiluri artistice distincte, pe baza unui subset din setul de date WikiArt. Performanțele rețelei, evaluate prin intermediul metricilor standard (acuratețe, pierdere, matrice de confuzie), au demonstrat o capacitate robustă de generalizare pe imagini reale de artă, chiar și în condiții de antrenare redusă.

Restaurarea imaginilor a fost abordată printr-un set de metode fundamentate pe algoritmi din biblioteca OpenCV, precum filtrarea gaussiană, mediană, bilaterală, inpainting-ul exemplar și super-rezoluția. Aceste tehnici au fost aplicate cu succes pe imagini degradate, contribuind la eliminarea zgomotului, completarea zonelor lipsă și îmbunătățirea clarității vizuale, toate fără a afecta integritatea stilistică a lucrării originale.

Un element important al lucrării îl reprezintă dezvoltarea unei aplicații web interactive, realizate cu ajutorul framework-ului Flask, care oferă o interfață intuitivă și accesibilă utilizatorilor. Aceasta permite încărcarea picturilor dintr-o bază de date SQLite, aplicarea secvențială a filtrelor de restaurare, clasificarea automată a stilului artistic și vizualizarea rezultatelor într-un mediu integrat. Designul aplicației a fost conceput astfel încât să asigure o experiență de utilizare clară și eficientă, fără a necesita cunoștințe tehnice din partea utilizatorului.

În ansamblu, lucrarea prezentată demonstrează fezabilitatea integrării între învățarea profundă, procesarea imaginii și dezvoltarea aplicațiilor web, în contextul digitalizării și conservării patrimoniului vizual. Prin abordarea sa interdisciplinară, proiectul aduce o contribuție relevantă la eforturile actuale de modernizare a instrumentelor folosite în analiza și protejarea artei vizuale, deschizând calea către aplicații mai avansate în domeniul muzeologiei digitale, educației vizuale și arhivării automate.

5.2 DIFICULTĂȚI ÎNTÂMPINATE

În cadrul dezvoltării proiectului, au fost întâmpinate o serie de dificultăți specifice atât procesului de implementare tehnică, cât și etapelor de documentare și integrare a componentelor într-un sistem unitar. Aceste provocări au influențat dinamica generală a lucrării și au impus adaptări metodologice și soluții alternative, menite să asigure finalizarea cu succes a obiectivelor propuse.

Una dintre dificultățile majore a fost legată de limitările hardware disponibile în mediul de lucru, în special în ceea ce privește lipsa accesului la unități GPU pentru antrenarea rețelelor neuronale. Din această cauză, antrenarea modelului MobileNetV2 a fost realizată pe subseturi reduse de date, ceea ce a necesitat ajustări repetate ale parametrilor de învățare (număr de epoci, dimensiuni batch, strategii de augmentare) pentru a obține o performanță satisfăcătoare fără supraîncărcarea resurselor.

De asemenea, implementarea secțiunii dedicate restaurării imaginilor a impus o documentare atentă asupra metodelor disponibile în biblioteca OpenCV, precum și testarea empirică a diverselor filtre pentru a calibra eficiența fiecăruia în funcție de tipul de degradare simulată. Unele metode, precum inpainting-ul exemplar sau super-rezoluția, s-au dovedit a fi

sensibile la parametrii inițiali sau la dimensiunile imaginii de intrare, necesitând experimentări multiple pentru obținerea unor rezultate vizual coerente.

Pe partea de dezvoltare web, integrarea bibliotecilor de procesare a imaginilor în cadrul unei aplicații Flask a ridicat provocări legate de gestionarea fișierelor, conversia imaginilor între formate și menținerea coerenței vizuale în interfață. A fost necesară implementarea manuală a funcționalităților de salvare în baza de date SQLite, conversia imaginilor în format binar și crearea unui sistem logic de rutare între paginile aplicației. Aceste aspecte au presupus o înțelegere profundă a modului în care Flask interacționează cu fișierele statice, template-urile HTML și bazele de date relaționale.

O altă dificultate a constat în gestionarea aspectului vizual al aplicației web, întrucât s-a dorit un design estetic, coerent cu tematica artistică a proiectului. Alegerea unei palete cromatice potrivite, crearea unui layout aerisit și adaptarea acestuia pentru diverse dimensiuni de ecran au necesitat eforturi suplimentare, mai ales în absența unui framework CSS complet (precum Bootstrap).

Nu în ultimul rând, procesul de redactare a lucrării teoretice a implicat un efort semnificativ de sinteză a informațiilor din surse multiple, pentru a asigura un ton academic coerent și o structură logică între capitole. Integrarea explicațiilor tehnice detaliate într-un limbaj accesibil, dar totodată riguros, a reprezentat o provocare în sine, mai ales în zonele care combinau aspecte din rețele neuronale, vizualizare grafică și procesare de imagine.

În pofida acestor dificultăți, abordarea incrementală, testarea continuă și capacitatea de adaptare au permis depășirea obstacolelor și finalizarea cu succes a proiectului.

5.3 POSIBILE ÎMBUNĂTĂȚIRI ȘI EXTINDERI ALE PROIECTULUI

Proiectul realizat oferă o bază funcțională solidă pentru clasificarea și restaurarea digitală a imaginilor artistice, însă există numeroase direcții de dezvoltare care pot îmbunătăți considerabil atât performanța tehnică, cât și impactul practic al aplicației.

Una dintre cele mai evidente extensii constă în antrenarea modelului de clasificare pe un set de date complet și diversificat, care să cuprindă un număr mai mare de stiluri artistice, precum și un volum crescut de imagini per stil. Acest lucru ar permite o generalizare mai bună și ar conduce la o clasificare mai precisă, chiar și pentru imagini ambigue sau degradate. De asemenea, pot fi testate arhitecturi mai avansate, precum EfficientNet sau Vision Transformer (ViT), care au demonstrat performanțe remarcabile în sarcini vizuale recente.

În ceea ce privește partea de restaurare, o direcție importantă ar fi înlocuirea filtrelor clasice cu modele bazate pe rețele neuronale profunde, precum U-Net, DnCNN sau GAN-uri specializate în inpainting și super-rezoluție (ex. SRGAN). Aceste modele pot produce rezultate mult mai realiste și adaptate contextului artistic al imaginii, însă presupun o etapă suplimentară de antrenare și optimizare, cu costuri computaționale mai mari.

Funcționalitățile aplicației web pot fi de asemenea extinse. Printre direcțiile relevante se numără:

adăugarea unei secțiuni de comparație vizuală între imaginea originală și cea restaurată, integrarea unei funcționalități de clasificare în timp real imediat după restaurare, posibilitatea descărcării automate a imaginilor procesate într-un format standard (ex. PNG, JPEG), și crearea unui cont de utilizator, prin care să fie salvate istoricul procesărilor și rezultatele obținute.

Pe partea de interfață și experiență utilizator, proiectul poate beneficia de integrarea unui framework CSS modern (precum Bootstrap, Tailwind CSS sau Materialize), care să permită o adaptare mai rapidă la diferite dispozitive și să ofere o estetică unitară și profesională.

În perspectivă, aplicația ar putea fi transformată într-o platformă educațională sau muzeală, în care vizitatorii să poată interacționa cu opere de artă în format digital, aplicând filtre vizuale și

învățând despre caracteristicile fiecărui stil artistic. Integrarea cu baze de date externe (precum Wikimedia Commons sau Europeana) ar extinde semnificativ sursa de imagini, iar exportul rezultatelor în formate standardizate ar facilita diseminarea academică sau artistică.

Nu în ultimul rând, integrarea unei componente de inteligență artificială explicabilă (XAI) ar permite interpretarea deciziilor modelului de clasificare, oferind utilizatorilor informații clare despre trăsăturile care au stat la baza identificării stilului artistic.

Prin urmare, proiectul prezentat poate fi privit nu doar ca un exercițiu aplicativ, ci ca un punct de plecare pentru aplicații interdisciplinare cu impact real în educație, conservare digitală și cercetare artistică.

- [1] G. Bradski, “The OpenCV Library,” Dr. Dobb’s Journal of Software Tools, 2000.
- [2] F. Chollet, Deep Learning with Python, 2nd ed. Manning Publications, 2021.
- [3] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016. [Online]. Available: <https://www.deeplearningbook.org/>
- [4] O. Russakovsky et al., “ImageNet Large Scale Visual Recognition Challenge,” Int. J. Comput. Vis., vol. 115, no. 3, pp. 211–252, 2015.
- [5] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, “MobileNetV2: Inverted Residuals and Linear Bottlenecks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2018.
- [6] A. Criminisi, P. Pérez, and K. Toyama, “Region Filling and Object Removal by Exemplar-Based Image Inpainting,” IEEE Trans. Image Process., vol. 13, no. 9, pp. 1200–1212, 2004.
- [7] J. Johnson, A. Alahi, and L. Fei-Fei, “Perceptual Losses for Real-Time Style Transfer and Super-Resolution,” in European Conf. Computer Vision (ECCV), 2016.
- [8] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional Networks for Biomedical Image Segmentation,” in Proc. Med. Image Comput. Comput. Assist. Interv., 2015.
- [9] M. Zhang and Z. Zuo, “Image Denoising via DnCNN: Denoising Convolutional Neural Network,” arXiv preprint arXiv:1608.03981, 2016.
- [10] A. Dosovitskiy et al., “Image Restoration Using Convolutional Autoencoders,” arXiv preprint arXiv:1607.06450, 2016.
- [11] A. Buades, B. Coll, and J. Morel, “A Non-Local Algorithm for Image Denoising,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2005.
- [12] T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive Growing of GANs for Improved Quality, Stability, and Variation,” arXiv preprint arXiv:1710.10196, 2017.
- [13] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” arXiv preprint arXiv:1412.6980, 2014.
- [14] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” arXiv preprint arXiv:1409.1556, 2014.
- [15] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” arXiv preprint arXiv:1506.02640, 2015.
- [16] SQLite Documentation. [Online]. Available: <https://www.sqlite.org/docs.html>
- [17] Flask Documentation. [Online]. Available: <https://flask.palletsprojects.com/>
- [18] TensorFlow Documentation. [Online]. Available: <https://www.tensorflow.org/>

[19] Keras Documentation. [Online]. Available: <https://keras.io/>

[20] Kaggle WikiArt Dataset. [Online]. Available: <https://www.kaggle.com/c/painter-by-number>

**DECLARAȚIE PRIVIND ORIGINALITATEA
LUCRĂRII DE LICENȚĂ**

UNIVERSITATEA TRANSILVANIA DIN BRAȘOV
FACULTATEA INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR
PROGRAMUL DE STUDII Automatică și informatică aplicată

NUMELE ȘI PRENUMELE.....Cosferent Roxana Adelina

PROMOȚIA.....2021-2025.....

SESIUNEAIUNIE 2025.....

TEMA PROIECTULUI

Procesarea imaginilor în ecosistemul Python pentru analiza operelor de artă

CONDUCĂTOR ȘTIINȚIFIC Șef lucr. Dr. Ing Dănilă Adrian

Declar pe propria răspundere că lucrarea de față este rezultatul muncii proprii, pe baza cercetărilor proprii și pe baza informațiilor obținute din surse care au fost citate și indicate conform normelor etice, în textul lucrării/proiectului, în note și în bibliografie.

Declar că nu s-a folosit în mod tacit sau ilegal munca altora și că nici o parte din teză/proiect nu încalcă drepturile de proprietate intelectuală ale altcuiva, persoană fizică sau juridică.

Declar că lucrarea/ proiectul nu a mai fost prezentat(ă) sub această formă vreunei instituții de învățământ superior în vederea obținerii unui grad sau titlu științific ori didactic.

În cazul constatării ulterioare a unor declarații false, voi suporta rigorile legii.

Data: .17.06.2025

Absolvent Cosferent Roxana Adelina

