Universitatea Națională de Știință și Tehnologie POLITEHNICA
București
Facultatea de Electronică, Telecomunicații și Tehnologia Informației

Recunoașterea bolilor specifice viței de vie prin analiza de imagini

Proiect de diplomă

prezentat ca cerință parțială pentru obținerea titlului de
Inginer în domeniul *Electronică și Telecomunicații*
programul de studii de licență *Tehnologii și Sisteme de Telecomunicații*

Conducător științific — *Conf. dr. ing. Cristina OPREA*
Absolvent — *Ruxandra-Teodora ZAMFIR*

2024

Universitatea "Politehnica" din Bucureşti **Anexa 1**
Facultatea de Electronică, Telecomunicaţii şi Tehnologia Informaţiei
Program de studiu **TST**

TEMA PROIECTULUI DE DIPLOMĂ
a studentului **ZAMFIR N.L. Ruxandra-Teodora, 443C**

1. Titlul temei: Recunoașterea bolilor specifice viței de vie prin analiza de imagini

2. Descrierea temei și a contribuției personale a studentului (în afara părții de documentare):
Se va dezvolta o aplicație în Python folosind librăria OpenCV care va evalua starea unei culturi de viță de vie prin analiza de imagini / secvențe video. Se vor urmări boli specifice culturilor viticole în vederea unui management eficient al acestor culturi folosind principii ale agriculturii de precizie. Aplicația va analiza starea plantelor, a frunzelor sau a fructelor. În acest scop se vor realiza segmentări automate, evaluarea formelor segmentate, eventual o analiză a culorilor. Aplicația va folosi tehnici de deep learning pentru o clasificare eficientă. Se vor folosi pentru aceasta baze de date cu imagini de test disponibile online.

3. Discipline necesare pt. proiect:
FIPCV; POO; MLF

4. Data înregistrării temei: 2023-12-17 17:14:02

Conducător(i) lucrare,
Conf. dr. ing. Cristina OPREA

Student,
ZAMFIR N.L. Ruxandra-Teodora

Director departament,
Conf. dr. ing. Șerban OBREJA

Decan,
Prof. dr. ing. Mihnea UDREA

Cod Validare: **6a736ec107**

Anexa 5

Declarație de onestitate academică

Prin prezenta declar că lucrarea cu titlul "*Recunoașterea bolilor specifice viței de vie prin analiza de imagini*", prezentată în cadrul Facultății de Electronică, Telecomunicații și Tehnologia Informației a Universității Naționale de Știință și Tehnologie POLITEHNICA București ca cerință parțială pentru obținerea titlului de *Inginer* în domeniul *Electronică și Telecomunicații*, programul de studii *Tehnologii și Sisteme de Telecomunicații* este scrisă de mine și nu a mai fost prezentată niciodată la o facultate sau instituție de învățământ superior din țară sau străinătate.

Declar că toate sursele utilizate, inclusiv cele de pe Internet, sunt indicate în lucrare, ca referințe bibliografice. Fragmentele de text din alte surse, reproduse exact, chiar și în traducere proprie din altă limbă, sunt scrise între ghilimele și fac referință la sursă. Reformularea în cuvinte proprii a textelor scrise de către alți autori face referință la sursă. Înțeleg că plagiatul constituie infracțiune și se sancționează conform legilor în vigoare.

Declar că toate rezultatele simulărilor, experimentelor și măsurătorilor pe care le prezint ca fiind făcute de mine, precum și metodele prin care au fost obținute, sunt reale și provin din respectivele simulări, experimente și măsurători. Înțeleg că falsificarea datelor și rezultatelor constituie fraudă și se sancționează conform regulamentelor în vigoare.

București, *17.06.2024*

Absolvent *Ruxandra-Teodora ZAMFIR*

Zamfir

(semnătură în original)

Cuprins

Listă de figuri

Listă de tabele

Listă de acronime

CNN = Convolutional Neural Network (Rețea convoluțională neuronală ro.)

FD = Flavescence dorée

FN = False negative

FP = False positive

HSV = Hue, Saturation, Value (Nuanță, Saturație, Valoare ro.)

Lab = Lightness, a*, b* (Luminanță, a*, b* ro.)

RGB = Red, Green, Blue (Roșu, Verde, Albastru ro.)

SVM = Support Vector Machine (Mașină de Vectori Suport ro.)

TN = True negative

TP = True positive

UAV = Unmanned Aerial Vehicle (Vehicul Aerian ro.)

VGG = Visual Geometry Group

Introducere

Încă din cele mai vechi timpuri, agricultura a fost o ramură importantă a dezvoltării omenirii, însă managementul calității producției este mai greu atunci când ne gândim la utilizarea instrumentelor clasice, cât și din punct de vedere al costului mare al forței de muncă, fiind necesară monitorizarea cu mare atenție a vegetației pentru a avea în final un produs calitativ, care poate atinge standardele pentru a putea fi acceptat pe piață.

Introducerea tehnologiei în această industrie a rezolvat multe dintre aceste probleme. Diferite metode tehnologice au dus la optimizarea productivității, creșterea calității, cât și la prevenirea distrugerii timpurii a plantelor datorită numeroaselor afecțiuni.

Viticultura este știința care se ocupă cu studiul producției viței de vie. Viticultorului îi revin atribuțiile de a cultiva și recolta strugurii, cât și de a urmări în mod constant evoluția plantelor. Tot acest proces poate fi costisitor dacă este practicat în mod tradițional. Viticultorii realizează o analiză vizuală cu ochiul liber, deși diagnosticele sunt corecte, de cele mai multe ori necesită mai mult timp, precum și forță de muncă, nefiind potrivită la scală largă pentru câmpuri de întinderi mari, de aceea mulți viticultori au fost nevoiți să utilizeze tehnologii avansate datorită cererii mari pe piață a strugurilor în industria alimentară și în producția vinurilor, vinificația fiind influențată de calitatea fructelor. În acest sens, fermierii au început să practice agricultura de precizie. Acest concept se referă la metode de cultivare bazate pe tehnologii noi care au ca scop dezvoltarea durabilă a producției, optimizarea managementului producției, precum și protecția mediului împotriva activităților intense de agricultură.

Viticultura de precizie este o particularizare a acestui concept, concentrată pe optimizarea cultivării strugurilor, pe randamentul culturii și îmbunătățirii sănătății plantației masive de viță de vie prin identificarea din timp a bolilor. Câteva dintre aceste tehnici includ instrumente precum senzori, camere, drone și sateliți. Metodele folosite sunt analiza și procesarea secvențelor video cu drone sau a imaginilor din satelit care oferă o vedere de ansamblu a condiției culturii. Unele dintre cele mai întâlnite sunt sistemele mici de cameră UAV pentru imagini aeriene din regiunea viticolă, sau roboții agricoli care se deplasează prin vița de vie și colectează date în timp real. [1] Asemănător roboților care explorează și descoperă în fiecare zi terenul pe care îl parcurg pe planeta Marte, proces ce se realizează prin intermediul sistemelor cu inteligență artificială cu monitorizare autonomă în timp real.

Motivația

Motivația alegerii temei constă în dorința de a aduce o contribuție semnificativă modernizării sectorului agricol prin punerea în practică a tehnologiilor avansate de analiză a imaginilor și de învățare adâncă și automată, pentru identificarea și gestionarea plantelor afectate. Abordările tehnologice ale acestui proiect contribuie la protejarea mediului și promovează agriculturi mai sustenabile. Într-o lume în care resursele devin din ce în ce mai limitate, folosirea substanțelor chimice pentru tratamente poate avea un impact negativ asupra mediului și a calității produselor. Aplicația vine ca un instrument modern, care să ofere viticultorilor soluții personalizate și precise pentru a preveni și trata afecțiunile plantelor.

Obiectivele generale ale proiectului

Proiectul presupune dezvoltarea unei aplicații de clasificare în limbajul Python, cu scopul de a evalua starea frunzelor de viță de vie prin analiza de imagini. În acest sens se realizează antrenarea unor modele pe o bază de date cu frunze afectate de diferite boli.

În final, obiectivul principal este de a se obține o clasificare precisă prin tehnici de învățare profundă și învățare automată, furnizând o soluție utilă pentru viticultori în vederea optimizării managementului culturii viței de vie.

Metodologia folosită

Metodologia include următoarele etape:

- Colectarea datelor: alegerea imaginilor relevante care vor alcătui baza de date pentru dezvoltarea aplicației.
- Tehnici avansate pentru clasificare: dezvoltarea algoritmilor de învățare automată, arhitecturi de tip CNN pentru a clasifica starea viței de vie.
- Evaluarea rezultatelor: utilizarea metricilor potrivite pentru determinarea performanțelor.

Contribuția personală

Contribuția în cadrul proiectului a constat în alegerea, implementarea tehnicilor de analiză de imagini și dezvoltarea modelelor de învățare automată potrivite pentru clasificarea stării plantelor prin utilizarea librăriei Keras, în limbajul Python.

Organizarea lucrării

Primul capitol reprezintă capitolul de State-of-The-Art, unde am realizat cercetarea câtorva studii și exemplificarea bazelor de date care se concentrează pe recunoașterea bolilor specifice viței de vie. În a doua parte am sintetizat diferite tehnici oferite de librăria OpenCV și câteva tehnici de învățare automată a câtorva publicații relevante domeniului.

În al doilea capitol se prezentă principiile de bază ale rețelelor neuronale convoluționale care se află la baza dezvoltării aplicației de Deep Learning.

Capitolul III începe prin descrierea metodelor de preprocesare a datelor în vederea antrenării modelului. Se prezintă, de asemenea, arhitectura unui model de referință ales pentru verificarea performanței modelului de bază.

Capitolul IV reprezintă evaluarea eficienței aplicației și interpretarea rezultatelor în urma predicțiilor pe setul de date ales prin utilizarea unor metrici de performanță.

I. Tehnici și algoritmi în viticultura de precizie. Analiza de imagini

Prezentele tehnologii viticole încă nu și-au atins potențialul maxim, fiind încă multe provocări existente condiționate de factori de mediu (sol, climă), sau factori de irigare și utilizarea de produse agrochimice (erbicide, pesticide). Există multiple articole științifice listate la [2] privind studiul bolilor plantelor, însă ne vom concentra asupra unor studii relevante temei alese.

Un aspect important al analizei de imagini este faptul că are o mare varietate de domenii și aplicații, însă rolul esențial este acela de a ne extrage informații care ne oferă posibilitatea de a înțelege conținutul vizual, fiindcă de multe ori oamenii nu pot interpreta în mod direct imaginile.

Domeniul agriculturii de precizie utilizează analiza de imagini pentru gestionarea calității produselor prin recunoașterea semnelor precoce ale bolilor, asemeni domeniului medical, unde identificarea simptomelor timpurii ale afecțiunilor din imaginile medicale pot optimiza procesul de vindecare. Detectarea eficientă a bolii poate reduce simptomele, îmbunătățind calitatea vieții persoanelor afectate.

Analiza imaginilor este folosită pentru extragerea anumitor caracteristici sau pattern-uri care conțin detalii subtile sau dificile de observat cu ochiul liber. Caracteristicile pot fi transformate mai departe în date relevante.

Bolile viței de vie pot fi cauzate de fungi sau bacterii. Cele mai comune boli fungice care afectează vița de vie sunt exemplificate la [2] care atacă fructele și frunzele, reducând randamentul și calitatea ar fi: Făinarea viței de vie(Powdery Mildew), Mana viței de vie (Downey Mildew), Putregaiul negru (Black Rot) și Apoplexia viței de vie (Black Measles).

Figura 1.1 Comparație vizuală a bolilor cauzate de fungi din vița de vie [3]

În partea din stânga a figurii 1.1 am ilustrat fructele afectate de boli, iar în partea dreaptă frunzele bolnave. Prin intermediul figurii 1.1 am evidențiat câteva simptome întâlnite: variație a culorii și texturii, apariția unor pete de anumite forme sau semne de deteriorare.

1.1 Baze de date ale bolilor viței de vie utilizate în studii anterioare

În era digitală actuală, utilizarea bazelor de date a devenit un element fundamental în domeniul cercetării, furnizând cercetătorilor acces la cantități impresionante de date și informații. În acest context, vom analiza tipurile de baze de date utilizate în studiile anterioare din domeniul viticulturii de precizie, precum și avantajele și limitările acestora.

1.1.1 Proiectul PlantVillage

„Proiectul PlantVillage a început să colecteze zeci de mii de imagini cu plante de cultură sănătoase și bolnave (Hughes și Salathé, 2015) și le-a pus la dispoziție în mod deschis și gratuit. Aici, raportăm despre clasificarea a 26 de boli la 14 specii de culturi folosind 54.306 imagini” [4].

Baza de date a fost elaborată într-un laborator, iar frunzele au fost extrase din fiecare plantă și ulterior au fost obținute imagini de 256 × 256 pixeli cu frunzele pe un fundal gri. Întrucât lucrarea se concentrează pe culturile de viță de vie, studiul focalizându-se asupra identificării a 3 afecțiuni specifice: Black Measles, Black Rot și Leaf Blight. Am efectuat o comparație între aceste afecțiuni și starea sănătoasă a frunzelor, reprezentată de Healthy Leaf.

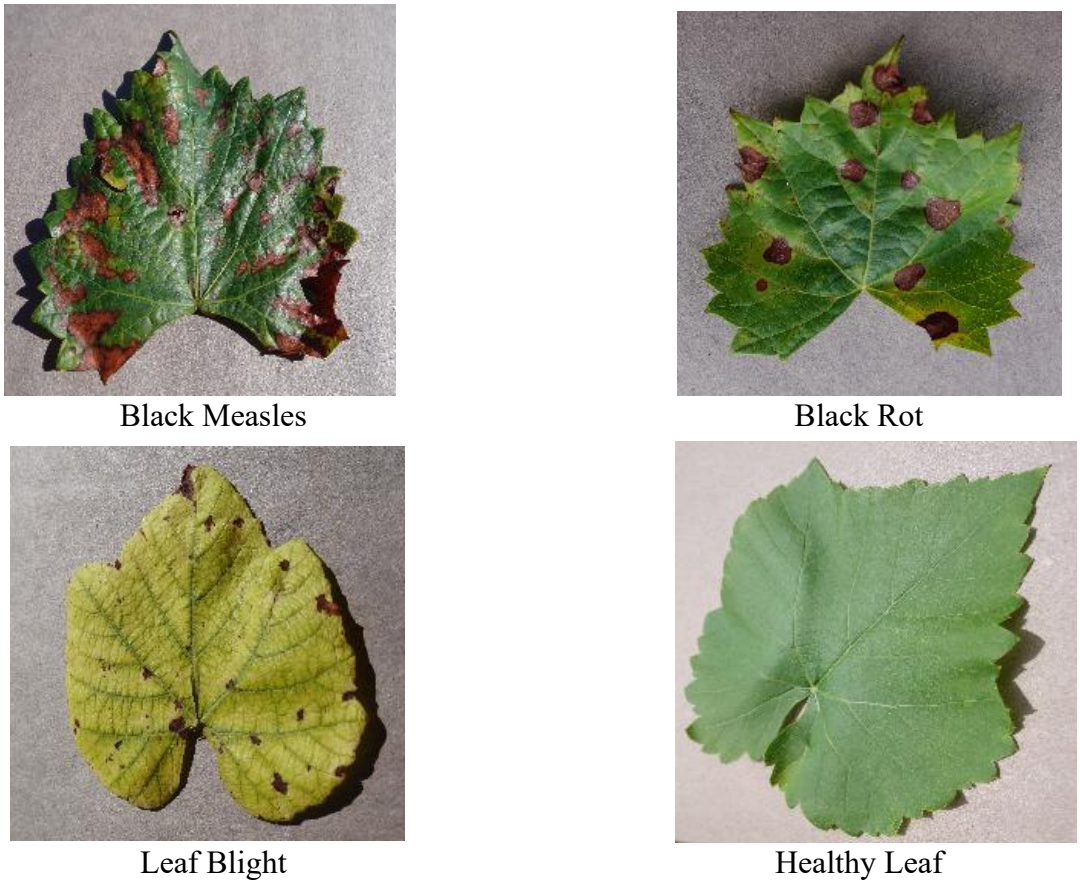


Figura 1.2 Reprezentarea stărilor de bază ale frunzelor de viță de vie din baza de date PlantVillage

Figura 1.2 înfățișează diferențele dintre frunzele sănătoase de viță de vie Healthy Leaf cu frunzele afectate de Black Measles, care prezintă pete alungite și negre, Black Rot, caracterizat prin pete mari, rotunde și negre, și Leaf Blight, care cauzează îngălbenirea și ofilirea frunzelor.

PlantVillage este una dintre cele mai cunoscute baze de date utilizate pentru diverse probleme de clasificare sau detecție a bolilor frunzei de viță de vie. Cele mai multe cercetări [4] au avut rezultate excelente de peste 95% acuratețe pentru încadrarea într-una dintre boli. Cu toate acestea condițiile de realizare ale acestei baze de date sunt speciale, fiind de laborator, captate dintr-un anumit unghi și într-o lumină naturală sau artificială, ceea ce nu conferă întocmai o perspectivă realistă. Însă avantajele sunt multiple fiind o resursă gratuită de informații educaționale despre bolile plantelor gata etichetate pentru cercetare.

1.1.2 Proiectul Bordeaux

Un grup de cercetători de la Universitatea din Bordeaux împreună cu o echipă de experți au stabilit o achiziție de imagini care asigură și o sursă fiabilă de etichetare a datelor: „a fost achiziționat un set de date compus din 1483 de imagini RGB care arată viță de vie care suferă de diferite boli și stres. În special, se concentrează pe boala viței de vie numită Flavescence dorée (FD) și pe factorii și bolile sale de confuzie. Imaginile a 5 soiuri de struguri (Cabernet Sauvignon, Cabernet Franc, Merlot, Ugni Blanc și Sauvignon Blanc) au fost achiziționate pe parcursul a 2 ani (2020 și 2021) în câmp prin detecție proximală” [5].

Baza de date a fost realizată direct în câmpul culturii de viță de vie prin detecție proximală, printre rândurile acesteia, cu ajutorul unei camere RGB la o distanță de 1 sau 2 metri pentru o imagine cât mai cuprinzătoare. Pentru asigurarea unei lumini constante care să nu fie influențată de mediu s-a folosit un bliț industrial. Pentru a oferi o bază de date cât mai exactă în etichetare, experții au realizat o serie de adnotări ale fotografiilor capturate. [5]

În primul rând, există o adnotare la nivel de imagine, pentru problemele de clasificare, întreaga imagine este încadrată într-o anumită stare. Baza de date se concentrează în jurul bolilor Flavescence dorée, Black Measles și alte boli sau deficiențe, care conferă simptome similare viței.

Figura 1.3 Imaginea culturii viței de vie de tip Cabernet Sauvignon afectată de Black Measles

În cazul problemelor de detecție, adnotările au fost realizate de cercetători prin intermediul unor casete de încadrare care înconjoară zone de interes din imagini, zone afectate de boli sau lovituri, și tulpinile care prezintă simptome prin intermediul unor linii.

Figura 1.4 Cadre de delimitare care evidențiază simptomele bolilor [6]

În figura 1.4 adnotările mov realizate de experți reprezintă diferite deficiențe sau alte boli asemănătoare. Etichetele roșii reprezintă frunzele afectate de FD, iar etichetele verzi arată ciorchinii simptomatici. Liniile galbene indică tulpinile afectate. [6]

Baza de date expertizată care conține imagini on-site ale viței de vie reprezintă un efort semnificativ pentru a furniza o resursă utilă în studiul și gestionarea bolilor viței de vie. Întrucât capturarea detaliată a culturii viței de vie poate oferi aspecte care contribuie la dezvoltarea unor strategii sustenabile în viticultură, minimizând utilizarea substanțelor chimice, protejând mediul înconjurător. Dezavantajul ar fi faptul că există limitări în reprezentarea bolilor și a diferențierii dintre acestea.

1.2 Computer Vision pentru analiza stării plantelor

În trecut, viticultorii foloseau metodele clasice de identificare a problemelor din vița de vie, precum analiza vizuală cu ochiul liber. Fiind prea costisitor din punct de vedere al timpului și nefiind întotdeauna eficienți, au apelat la diverse modalități care implicau achiziționarea unor echipamente nou dezvoltate pentru analizarea diferiților factori care pot afecta viile.

Tehnologiile de monitorizare au scopul de a analiza întreaga viță de vie de la distanță cu ajutorul sateliților, UAV sau a aeronavelor, prin capturarea unor imagini care ilustrează starea culturii în ansamblu. Imaginile conferă o reprezentare vizuală a stării vegetației, detectează prezența sau absența bolilor și permit estimarea cantității recoltei. De asemenea, există și monitorizarea proximală care presupune montarea în câmp a unor senzori de umiditate, de temperatură, pentru măsurarea calității solului. [7]

În prezent, tehnologia avansează rapid, iar astfel de soluții pot deveni mai fezabile în viitor. Cu toate acestea, implementarea lor necesită expertiză tehnică și resurse adecvate. Este important să

fie luate în considerare aspecte precum precizia, actualizarea bazei de date și adaptabilitatea algoritmilor la diferite condiții de mediu pentru a asigura o performanță optimă.

Un studiu din 2019, propune un sistem de clasificare automat al frunzelor prin metode de Machine Learning. Se utilizează imagini din baza de date PlantVillage pentru clasificarea între cele 4 clase disponibile pentru culturile de viță de vie, mai exact, Black Measles, Black Rot, Leaf Blight și Healthy Leaf. Metoda propune segmentarea zonei de interes, adică a frunzei, prin îndepărtarea fundalului gri și mai departe identificarea simptomelor bolilor.

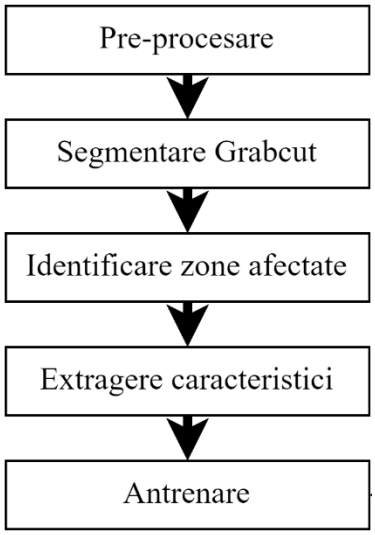


Figura 1.5 Arhitectura sistemului propus [8]

Imaginile au fost preluate din baza de date PlantVillage, cât și de pe Internet. Întrucât acestea provin din surse diverse, pot avea diferite dimensiuni sau probleme legate de zgomotul din poze din cauza factorilor de iluminare slabă. De aceea, în pasul de preprocesare se filtrează zgomotul prin aplicarea unor filtre de tip Gaussian pentru reducerea componentelor de frecvențe înalte, cât și o redimensionare standard a imaginilor la o lățime și înălțime. [8]

Segmentarea Grabcut este un algoritm de Computer Vision prin care se realizează o segmentare a imaginilor pe baza unor informații despre culoare și textură. Este o tehnică des folosită pentru distingerea dintre un obiect și fundal [9].

Figura 1.6 Rezultatul tehnicii de Grabcut pe o imagine din baza de date [8]

Identificarea zonelor afectate de boli presupune un set de pași mai complex prin utilizarea unor operații morfologice de dilatare și eroziune.

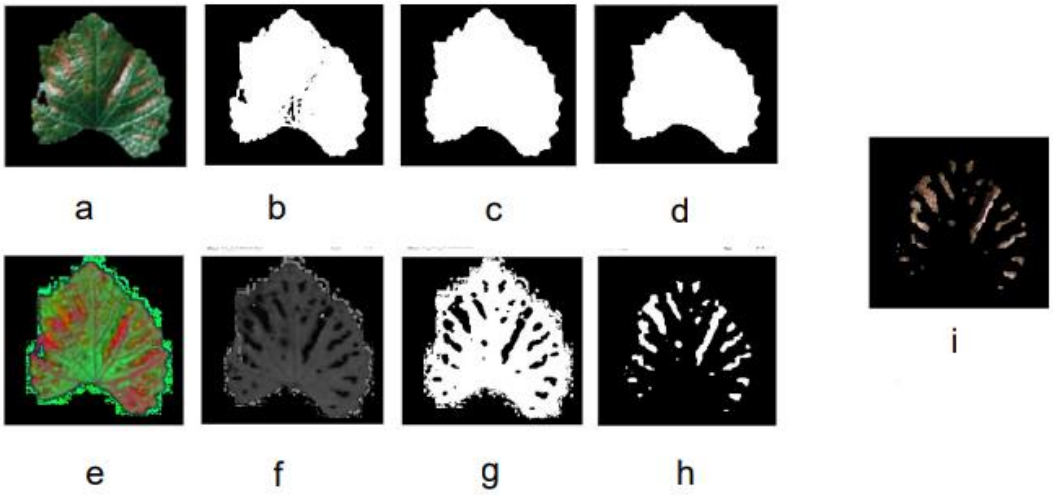


Figura 1.7 Pașii pentru identificarea zonelor afectate de boli[8]

În figura 1.7a este o imagine cu frunza în urma algoritmului de Grabcut. Mai departe această imagine este transformată în nuanțe de gri pentru a se aplica un prag global de alegere a unor pixeli și rezultă masca binară (1.7b). Se poate observa faptul că zona din interiorul frunzei conține zgomot. Acest zgomot este îndepărtat prin operații morfologice numite dilatare (1.7c) și eroziune (1.7d). În contextul morfologiei imaginilor binare, dilatarea și eroziunea sunt două operații fundamentale care modifică forma obiectelor din imagine. Dilatarea este un proces de mărire sau extindere a anumitor regiuni ale imaginii, iar eroziunea este procesul de reducere a anumitor regiuni ale imaginii sau de a elimina detalii fine. [9] Astfel se obține o mască binară a frunzei pentru care s-a înlăturat fundalul.

Imaginea 1.7e reprezintă imaginea 1.7a transformată în spațiul de culoare HSV. Această transformare a spațiului de culoare poate pune în lumină mai multe detalii față de spațiul RGB. În acest caz s-a observat faptul că pentru HSV, canalul H(1.7f) pune în evidență zonele afectate. Se realizează o transformare în imagine binară prin alegerea unui prag de segmentare (1.7g), iar prin înmulțirea cu imaginea binară inversată(1.7d), v-a rezulta o imagine care conține numai zonele de interes (1.7h) care reprezintă practic o mască prin care putem extrage numai părțile afectate (1.7i).

Pentru a realiza diferențierea între cele 4 tipuri de simptome ale bolilor se vor extrage caracteristici din imaginile rezultate în urma procesului, de exemplu 1.7i. Aceste caracteristici pot fi calculate pe baza matricei de co-ocurență la nivel de gri. În esență, matricea conține informații despre modul în care nivelurile de gri sunt dispuse și corelate în imagine, ceea ce poate fi folosit pentru analiza texturii și pentru extragerea de caracteristici din imagini. În funcție de setările specifice și criteriile de calcul ale matricei, valorile din matrice pot varia și pot furniza informații detaliate despre textura imaginii. Cu aceste valori, se calculează ulterior diverse proprietăți și statistici care descriu textura imaginii, cum ar fi contrastul, omogenitatea sau alte măsurători relevante. [10]

Clasificarea s-a realizat prin intermediul algoritmilor de Machine Learning: SVM cu o acuratețe de 91%, Random Forest cu o acuratețe de 74,79% și AdaBoost cu 83%.

Un studiu din 2020, bazat pe imagini cu frunze de viță de vie din baza de date PlantVillage, propune o clasificare binară între frunză sănătoasă și frunză bolnavă. Principalul scop este identificarea simptomelor cauzate de Black Measles.

Metoda presupune o serie de etape. Inițial se realizează segmentarea frunzelor prin îndepărtarea fundalului pentru obținerea unei regiuni de interes care evidențiază zonele afectate de boală. Ulterior se extrag caracteristici de culoare și se clasifică prin intermediul algoritmului SVM.

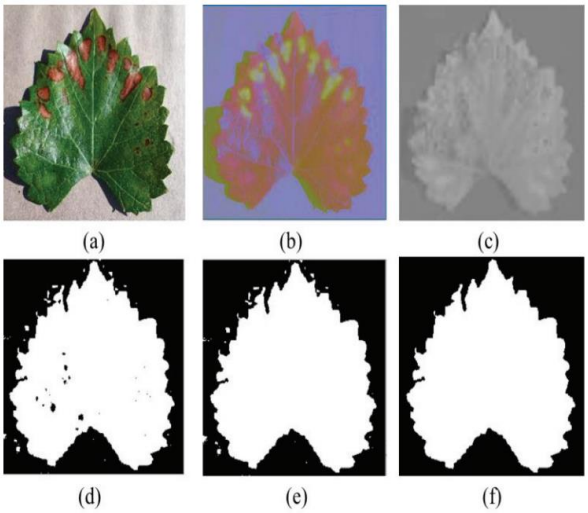


Figura 1.8 Conversie la imagine binară a) Imaginea originală b) Imagine în spațiul Lab c) Canalul b al imaginii d) Imagine binară după aplicare prag global e) Imagine după umplere goluri f) Masca binară după aplicarea conturului maxim [10]

Pentru început imaginea originală (1.8a) este transformată din RGB în Lab (1.8b), urmând ca pe canalul b (1.8c) al imaginii să se realizeze îndepărtarea fundalului. S-a observat faptul că imaginea în formatul de pe canalul b prezintă un contrast evident între frunză și fundal, fundalul conținând informație zgomotoasă, cum ar fi umbra frunzei și pixeli colorați.

S-a aplicat un prag global asupra imaginii, prin care s-a obținut o mască a frunzei segmentate din formatul în nuanțe de gri (1.8d). Imaginea rezultată conține și pixeli care aparțin zonei afectate, interiorul frunzei. Pentru îndepărtarea zgomotului s-a realizat o umplere de goluri(1.8e) și căutarea celui mai mare contur (1.8f) care va conține numai regiunea de interes afectată de boală.

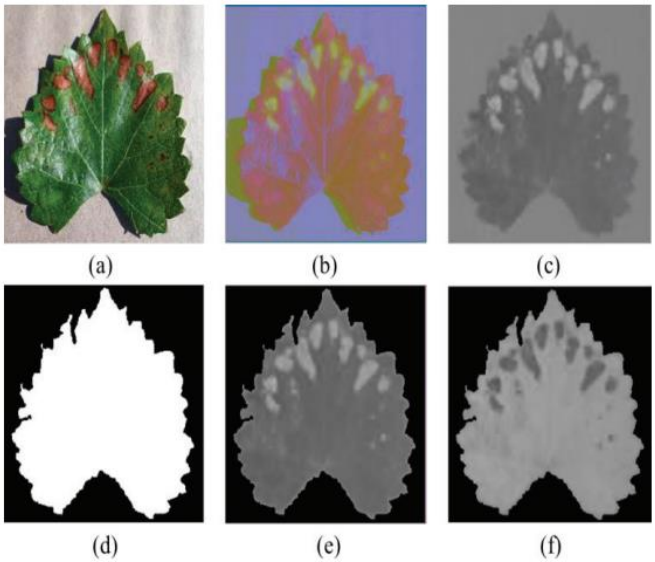


Figura 1.9 Segmentarea și evidențierea zonelor afectate ale frunzei a) Imaginea originală b) Imagine în spațiul Lab c) Canalul a al imaginii d) Masca binară e) Imagine rezultată în urma înmulțirii măștii cu canalul a f) Zonele simptomatice afectate de Black Measles[10]

În urma pasului anterior s-a obținut o mască binară (1.9a) care va pune în evidență petele de pe frunze. S-a observat faptul că pe canalul a (1.9c) al imaginii Lab se oferă informația maximă despre simptomele cauzate de BlackMeasles.

Se va realiza multiplicarea dintre masca binară obținută anterior și canalul a al imaginii(1.9e), care mai departe înmulțită cu (1.8c) vor pune în evidență regiunile de interes afectate de boală(1.9f). Mai departe pentru extragerea caracteristicilor relevante se vor calcula câteva măsurători care descriu distribuția datelor din setul de date. Se calculează media, varianța, deviația standard, skewness și kurtosis pe regiunea de interes. Pentru clasificare s-a implementat un SVM care are performanțe de 97,22% acuratețe. [10]

Un studiu mai recent din 2021, a realizat un sistem de recunoaștere și clasificare a bolilor de pe frunzele de rodie prin tehnici de Machine Learning și procesare de imagini. [11] Procesul de segmentare al zonelor afectate este realizat cu ajutorul algoritmului de K-means.

În primul rând imaginile trec printr-o etapă de preprocesare, în care se redimensionează imaginile din baza de date, se îmbunătățește contrastul pentru a face culorile mai vibrante, mai clare și mai departe se transformă în nuanțe de gri. Se alege un prag de segmentare al zonelor de interes afectate de boală.

Baza de date este colectată din mai multe surse, iar unele imagini conțin mai multă informație nefolositoare, cum ar fi fundalul. În cazul de față fundalul nu este unul gri, simplu, ca în cazul PlantVillage, ci este un fundal care conține verdeață, alte frunze, lucru care împiedică o segmentare ideală. De aceea, algoritmul de K-means este folosit pentru a segmenta culorile cele mai apropiate. K-means este un algoritm de învățare nesupervizată de Machine Learning care împarte setul de date în K grupuri, denumite clustere, în funcție de asemănările lor. Algoritmul are rolul de a identifica k centroizi, iar fiecărui centroid îi sunt atribuite câteva puncte de date, astfel încât aceștia să rămână cât mai compacți posibil și să nu se suprapună.[12]

După realizarea segmentării se evidențiază petele cauzate de fiecare boală în parte. Din aceste imagini sunt extrase, ca și în studiile anterioare, caracteristici care vor fi criterii de diferențiere ale bolilor. Rezultatele clasificării sunt de aproximativ 98% acuratețe.[12]

Algoritmul utilizat este SVM multi-clasă, un algoritm de învățare supravegheată în care fiecare clasă este separată de celelalte.[11] SVM sau mașinile cu vectori suport reprezintă o metodă de învățare prin care se realizează clasificarea datelor prin definirea unui hiperplan care maximizează o margine de separare între instanțele a două clase diferite. [13]

1.3 Deep Learning în agricultura de precizie

Agricultura de precizie a evoluat semnificativ în ultimii ani, transformând modul în care se gestionează producția. În acest context, tehnologiile de învățare profundă, Deep Learning, au devenit un instrument indispensabil, oferind soluții inovatoare pentru provocările complexe cu care se confruntă sectorul agricol, cum ar fi resurse limitate și creșterea cererii alimentare globale.

În subcapitolul anterior am prezentat câteva studii referitoare la cele mai recente tehnici de Computer Vision, însă și aceste tehnici sunt limitate fiind nevoie de alegerea manuală a

caracteristicilor relevante care trebuie extrase. De aceea evoluția în studiul aplicațiilor Deep Learning a luat amploare datorită capacității modelelor de a învăța reprezentări complexe.

Cea mai cunoscută arhitectură pentru problemele de clasificare în domeniul agriculturii de precizie sunt arhitecturile CNN. Un studiu din 2020 a realizat un sistem de clasificare folosind 4 modele de Deep Learning, toate având la bază arhitecturi de tip CNN. Baza de date utilizată este formată din 5 clase. Patru clase aparținând bazei de date PlantVillage: Black Measles. Black Rot, Leaf Blight, Healthy Leaf, iar clasa a 5-a a fost compusă din imagini disponibile online afectate de Phylloxera.

Primul pas este pasul de preprocesare. S-au realizat augumentări asupra bazei de date pentru extinderea acesteia prin intermediul unui generator de imagini care realizează operații de rotație, shift, shear, zoom și flip orizontal pe imagini. [14]

Cea mai simplă arhitectură prezentată este Vanilla CNN, sau o arhitectură CNN construită de la zero din figura de mai jos. Deși o rețea neuronală convoluțională CNN, are rezultate foarte bune de peste 90% acuratețe, există metode prin care acestea se pot îmbunătăți. Există arhitecturi care sunt la bază rețele neuronale convoluționale, însă sunt antrenate pentru a învăța anumite caracteristici, ceea ce se numește transfer de învățare.

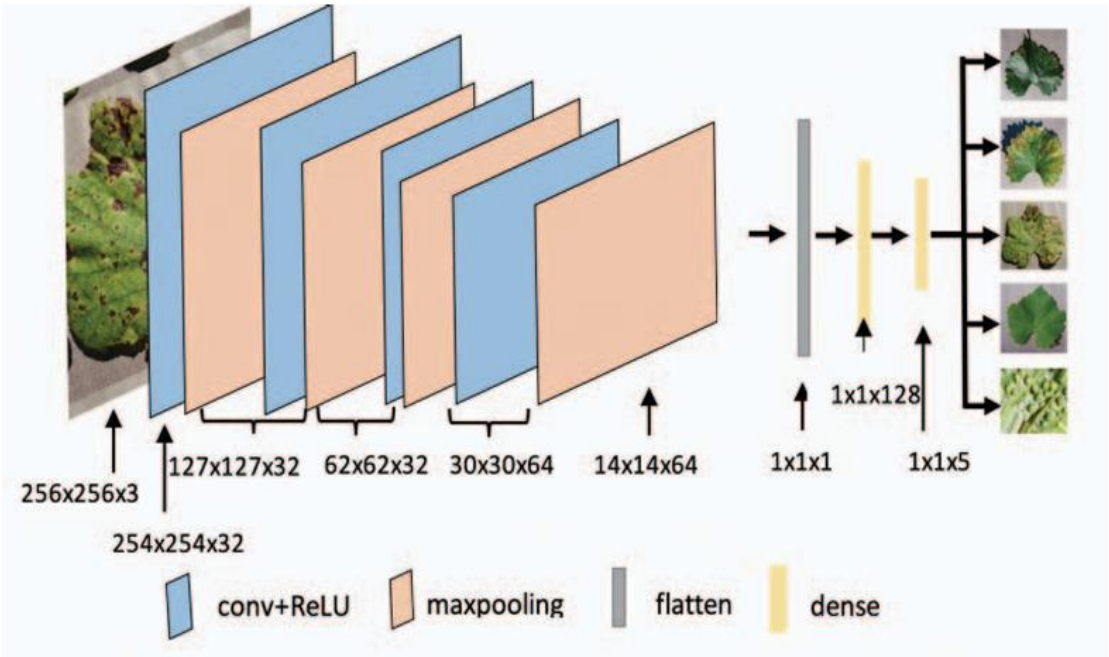


Figura 1.10 Arhitectură CNN [14]

Transferul de învățare este o metodă în domeniul învățării automate și al rețelelor neuronale în care cunoștințele acumulate în urma antrenamentului unei rețele pe un set de date specific sunt transferate și utilizate pentru a îmbunătăți performanța unei alte rețele neuronale destinate unei sarcini diferite sau unui alt set de date.

Procesul implică preluarea unui model pre-antrenat și ajustarea acestuia pentru a se potrivi cu cerințele noii sarcini, cu posibilitatea de rafinare, sau fine-tunning, pentru a adapta modelul la specificul setului de date. [15]

Câteva rețele studiate sunt:

1. VGG16 îmbunătățit prin transfer de învățare
2. MobileNet îmbunătățit cu transfer de învățare
3. AlexNet îmbunătățit prin transfer de învățare

Dintre toate modelele, VGG16 a obținut cea mai bună performanță, mai exact de 99% acuratețe, iar matricea de confuzie pe datele de testare arată că doar 4 imagini cu boala BlackRot au fost prezise greșit și una din clasa Black Measles a fost încadrată greșit. Celelalte 2 modele au performanțe ridicate, rețeaua MobileNet îmbunătățită prezintă o acuratețe de 97%, iar modelul AlexNet îmbunătățit a avut tot 97% acuratețe. [14]

Un alt studiu din 2020 a realizat o clasificare a bolilor specifice frunzelor de roșii folosind 4 modele pre-antrenate de Deep Learning. Cercetătorii au studiat inițial pe un set de date care conținea imagini cu 4 afecțiuni ale frunzelor de roșii fotografiate într-un mediu de laborator controlat, însă aceștia au vrut să implementeze clasificarea și pe imagini ale frunzelor de roșii în mediul lor natural, într-un spațiu necontrolat. [16]

Crearea unui set de date a reprezentat o provocare, deoarece imaginile au fost fotografiate în lumina naturală folosind un telefon. Au reușit să colecteze imagini cu 6 afecțiuni ale frunzelor de roșii preluate din diferite câmpuri agricole care au fost incluse în categorii de către experți. [16]

Pentru a studia performanța modelelor pe cele 2 seturi de date au utilizat următoarele metrici: acuratețea, precizia, recall și F1 – score. [16] Nu au utilizat ca metrică matrice de confuzie deoarece pentru cazul cercetării numărul de imagini pentru fiecare clasă diferă, existând un dezechilibru. De aceea au utilizat metricile menționate anterior pentru a observa o perspectivă globală a performanței modelului.

Modelele pre-antrenate utilizate au fost VGG16 și VGG19, ResNet și InceptionV3. Au obținut rezultatele pentru 2 tipuri de antrenare. Primul a fost antrenarea celor 4 modele pentru a învăța să extragă caracteristici specifice bolilor frunzelor de roșii. Clasificarea imaginilor constă în 2 etape: extragerea de caracteristici realizată de straturi convoluționale și clasificarea efectuată prin straturi dens conectate. [16]

Rezultatele pentru această antrenare au indicat faptul că pe setul de date din laborator performanțele sunt cu mult mai ridicate decât pe setul din mediul natural. Al doilea set de rezultate a fost obținut în urma utilizării transferului de învățare ca în studiul anterior.

Model	Set de date laborator(%)	Set de date câmp(%)
VGG-16	98.50	84.10
VGG-19	98.30	86.30
ResNet	99.40	91.30
Inception V3	99.60	93.70

Tabel 1.1 Rezultatele modelelor pe cele 2 seturi de date [16]

Din tabelul 1.1 putem concluziona faptul că pentru un set de date controlat vom obține rezultate mai ridicate decât pe un set de date colectat pe teren în condiții de ambient natural. Însă rezultatele obținute pe cel de al doilea set sunt destul de bune.

Un studiu din 2019 propune rezolvarea problemei de clasificare a tuturor celor 39 de clase din setul de date PlantVillage prin intermediul unei rețele neuronală convoluțională formată din 9 straturi. Setul de date a fost împărțit in 3 seturi: antrenare, validare și testare. [17]

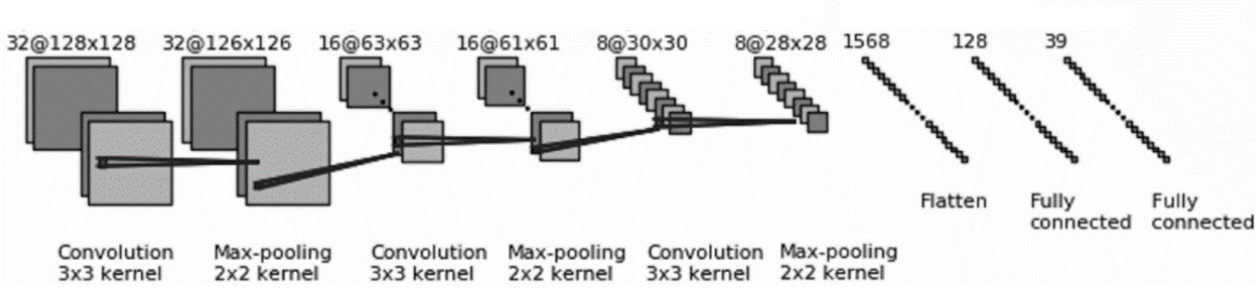


Figura 1.11 Arhitectură CNN propusă cu 9 straturi [17]

Figura 1.11 este o reprezentare vizuală a celor 9 straturi. Principiul din spate este că pe măsura ce imaginea de intrare este procesată de straturile de convoluție se vor extrage caracteristici din ce în ce mai detaliate cu fiecare strat pentru a ajunge la o diferențiere între clasele între care se dorește clasificarea.

De asemenea, studiul se concentrează și pe importanța straturilor de Dropout, pentru reducerea supra-învățării, comparând diverse valori de Dropout. Cele mai bune rezultate au fost pentru un procent de 20%.

Cercetătorii au realizat o analiză comparativă a rețelei CNN propusă cu alte modele cunoscute pentru transferul de învățare: AlexNet, VGG16, InceptionV3 și ResNet. Rezultatele cele mai bune au fost obținute de VGG16 de 93% acuratețe și InceptionV3 de 94% acuratețe. Modelul propus de studiu a fost de 96% acuratețe. [17]

II. Rețele neuronale convoluționale

Metoda folosită pentru problemele de clasificare, recunoaștere și localizare a obiectelor este arhitectura CNN, dezvoltată în 1998 de către Yann LeCun și alții. Inventatorii s-au inspirat biologic dintr-un neuron și au observat pe baza unui studiu mai vechi concentrat pe cortexul vizual animal că există o legătură între o zonă mică bine definită a creierului și zone mici ale câmpului vizual. În final s-a descoperit legătura dintre părți diferite ale câmpului vizual și neuroni individuali care procesează informația. [18]

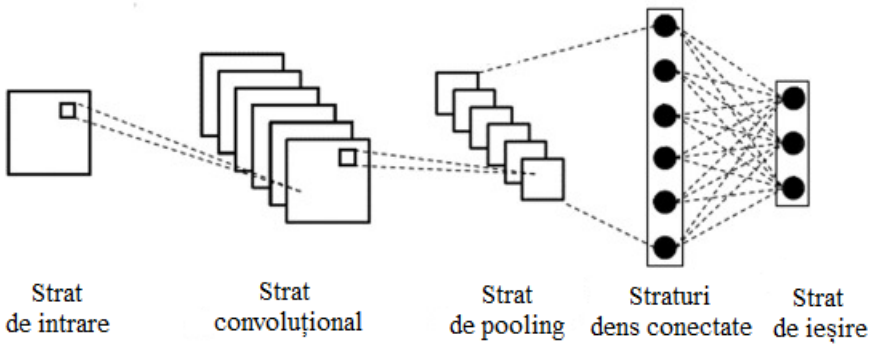


Figura 2.1 Arhitectura generală a unei rețele convoluționale [19]

Arhitectura CNN încearcă să imite acest proces biologic aplicându-l calculatoarelor pentru a le învăța să înțeleagă informația din imagini sau chiar din secvențe video. O rețea neuronală convoluțională este formată la bază din mai multe straturi: convoluție, pooling, de activare, dens conectate.

Intrarea într-o rețea convoluțională reprezintă o imagine având mai multe valori numiți pixeli. Dimensiunea de intrare a primului strat este determinat de imaginea de intrare. În funcție de baza de date, putem avea de exemplu o imagine cu 256×256 pixeli, iar dacă aceasta este color vom avea 256×256×3, 3 fiind numărul de canale RGB.

- Stratul de activare

ReLU este o funcție de activare continuă care introduce non-liniaritate în rețea. În general straturile ReLU urmează după straturile de convoluție, sau straturi dens. [20]

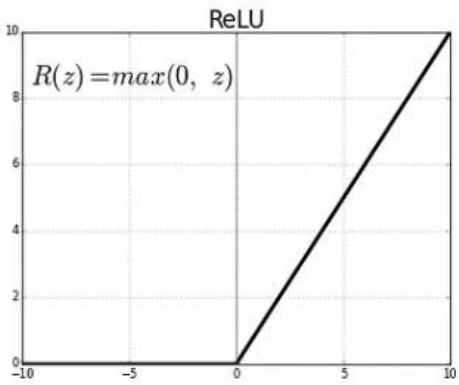


Figura 2.2 Funcția de activare ReLU [21]

Pentru valorile negative funcția va lua valoarea 0, iar pentru o valoare mai mare de 0 va lua valoarea variabilei.

- Stratul de convoluție

În rețelele convoluționale parametrii straturilor sunt structuri unitare tridimensionale, numite filtre. În cadrul operației de convoluție filtrul se suprapune cu imaginea de intrare, se realizează convoluția prin parcurgerea întregii imagini și astfel va rezulta o hartă de trăsături.

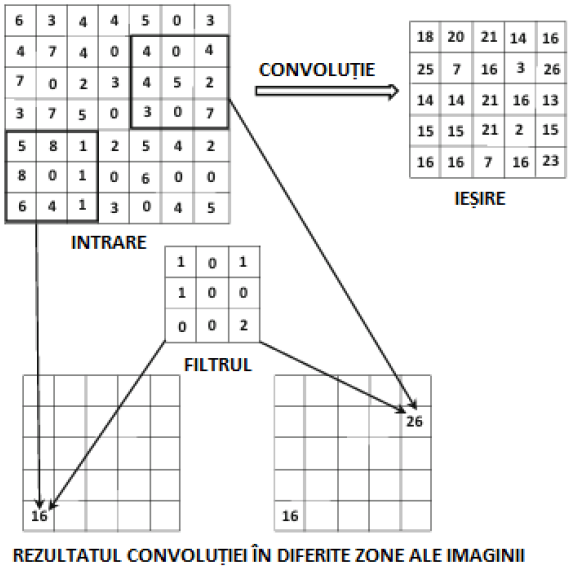


Figura 2.3 Operația de convoluție [20]

În figura 2.2 avem ca exemplu operația de convoluție a unei imagini de intrare pentru care s-a aplicat un filtru de 3×3. Întrucât operația de convoluție ține de regiunile imaginii de intrare putem observa rezultatul aplicat în zone diferite. Valorile s-au obținut prin operațiile următoare:

$$5\times1+8\times1+1\times1+1\times2=16$$

$$4\times1+4\times1+4\times1+7\times2=26$$

Fiecare element din imagine corespunzător ferestrei filtrului se înmulțește cu fiecare element din filtru, apoi sunt adunate. Rezultatul este o variantă a imaginii de intrare, dar transformată, în care avem diverse informații despre caracteristici de culoare, textură sau margini.[20]

- Stratul de pooling

Operația de pooling nu folosește filtre precum convoluția, ci imaginea este parcursă de o regiune pătratică, iar în fiecare zonă se realizează o operație. În general cele mai cunoscute tipuri de pooling sunt Max și Average.

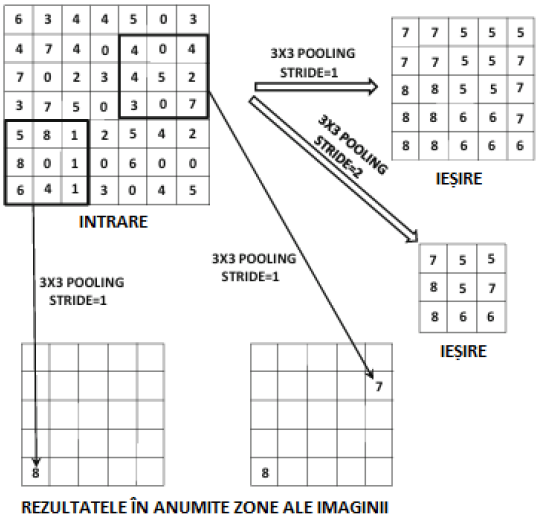


Figura 2.4 Operația de Max Pooling [20]

Figura 2.4 înfățișează operația de Max Pooling cu o regiune de 3×3 peste o imagine de intrare. Din fiecare zonă se extrage valoarea maximă. De exemplu pentru Average Pooling din fiecare zonă se calculează media valorilor. Stride este pasul cu care se parcurge imaginea, cât de mult se deplasează fereastra de Pooling. Putem observa că pentru un pas egal cu 2 dimensiunea imaginii este redusă semnificativ.

- Straturile dens conectate

Sunt straturile complet conectate unde fiecare neuron este conectat cu toți ceilalți neuroni din straturile vecine.

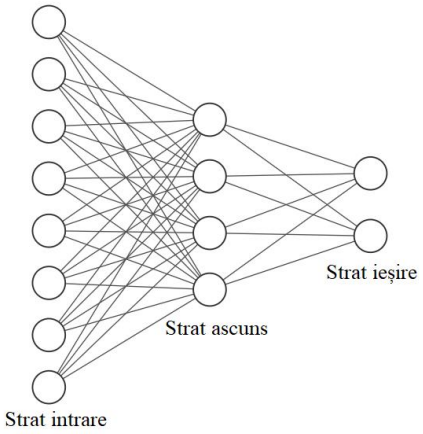


Figura 2.5 Rețea neuronală complet conectată cu 2 straturi [22]

În figura 2.5 avem o rețea complet conectată cu 2 straturi. Stratul ascuns are 4 neuroni, iar stratul de ieșire are 2 neuroni, ceea ce înseamnă că rețeaua rezolvă o problemă de clasificare binară. Stratul de intrare este format din 8 neuroni.

- Stratul de ieșire

Are rolul de a determina rezultatele rețelei pe care o antrenăm și furnizează rezultatele sub forma unor probabilități. Este format din numărul claselor între care se face clasificarea și ne oferă posibilitatea de a ne evalua rețeaua.

Stratul de ieșire este „complet conectat la fiecare neuron din penultimul strat și are o greutate asociată acestuia. S-ar putea folosi activarea logistică, softmax sau liniară, în funcție de natura aplicației (de exemplu, clasificare sau regresie)”. [21]

- Sigmoid

Pentru o clasificare binară între două clase de exemplu se poate utiliza funcția sigmoid. Rezultatul funcției reprezintă o probabilitate, un număr între 0 și 1. Iar în funcție de aceste valori se poate alege un prag, de exemplu 0.5 și în funcție de acesta se face o distribuire a valorilor de ieșire într-o clasă sau alta.

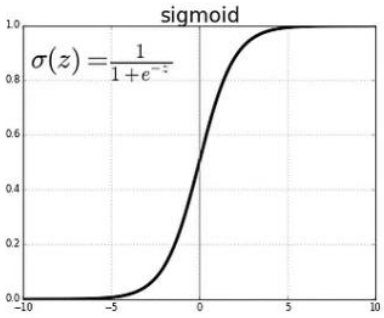


Figura 2.6 Funcția de activare sigmoid [23]

- Softmax

Pentru o clasificare a mai multor clase se poate folosi funcția softmax. Rezultatul funcției matematice este un vector de probabilități de apartenență la o anumită clasă. Suma probabilităților este 1, iar valorile acestora sunt între 0 și 1. Este utilizată pe ultimul strat al rețelei.

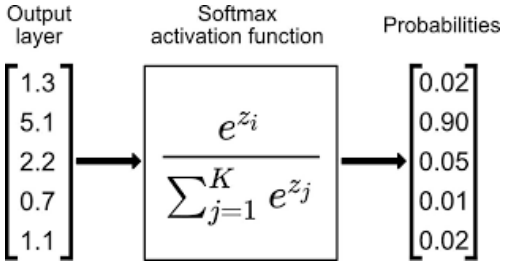


Figura 2.7 Funcția de activare softmax [24]

În figura 2.7 se poate observa faptul că fiecare probabilitate este calculată cu o funcție matematică a valorilor de pe stratul de ieșire. Probabilitatea cea mai mare va fi desemnată ca fiind predicția modelului respectiv, adică pentru figura 2.7 deoarece a doua valoare din vector prezintă cea mai mare probabilitate aceea va fi aleasă ca fiind predicția finală în clasa a doua.

De asemenea pot fi adăugate și alte straturi care au anumite funcții. Câteodată în antrenarea modelelor putem întâlni supra-învățarea, iar pentru a preveni acest fenomen există câteva metode de regularizare pentru a reduce complexitatea modelului ceea ce poate ajuta la ajustarea modelului astfel încât acesta să poată generaliza mai bine pe datele nemaivăzute de acesta, cum ar fi adăugarea straturilor de Dropout.

- Stratul de Dropout

O rețea neuronală standard poate avea toți neuronii legați între ei. Însă o astfel de arhitectură poate fi prea complexă și ineficientă. Fiecare legătură dintre neuroni are o pondere, iar această pondere este actualizată în timpul antrenării și reprezintă cât de importantă este conexiunea respectivă a celor doi neuroni.

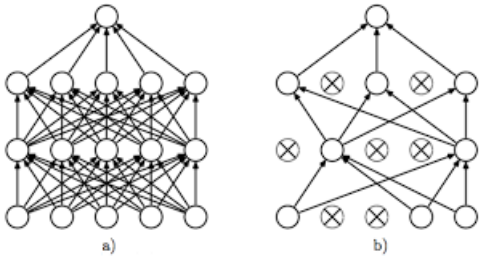


Figura 2.8 Aplicarea Dropout asupra unei rețele neuronale [25]
a) Rețea standard b) După aplicare Dropout

Se poate percepe din figura 2.8b faptul că prin aplicarea operației de Dropout asupra modelului inițial 2.8a, complexitatea modelului a scăzut. Dropout înseamnă dezactivarea aleatorie a unor neuroni de pe un strat într-un procentaj specificat, anulând contribuția lor la ieșire, iar modelul se va baza pe restul căilor din rețea. De exemplu, pentru un procent de 50% vor fi anulați aleatoriu jumătate din neuroni.

III. Modele propuse. Dezvoltarea și antrenarea modelelor

3.1 Preprocesarea datelor. Selecția și prelucrarea bazei de date

Un pas important în crearea unei aplicații care să fie un clasificator precis pentru recunoașterea bolilor plantelor este alegerea unui set de date potrivit. Până acum am observat că în capitolul de State-of The-Art, capitolul 1, multe studii se concentrează pe rezolvarea problemelor de clasificare. De aceea vom încerca să realizăm o recunoaștere a pozelor extrase din baza de date PlantVillage concentrată pe imagini ale bolilor viței de vie.

Primul pas este de a pregăti datele pentru antrenare. Antrenarea se realizează pe un set de date și testarea pe alt set, astfel încât baza de date formată din 1600 de imagini, fiecare clasă conținând câte 400 de imagini. Am împărțit manual datele în antrenare, validare și testare. Am ales un procent de 70% pentru antrenare, apoi cei 30% vor fi împărțiți în mod egal pentru testare și validare.

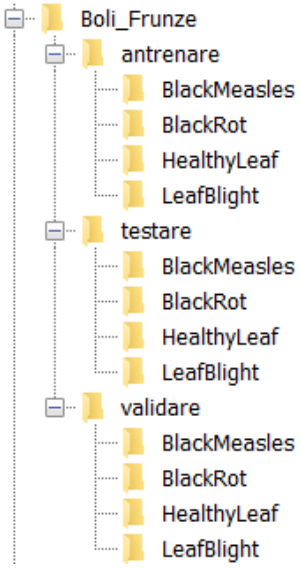


Figura 3.1 Împărțirea datelor în seturile de antrenare, validare și testare

Împărțirea datelor în fișierele de antrenare, validare și testare reprezintă o etapă de pregătire a datelor pentru antrenarea modelelor de învățare automată. Este important să împărțim datele în categoriile reale pentru care se face clasificarea.

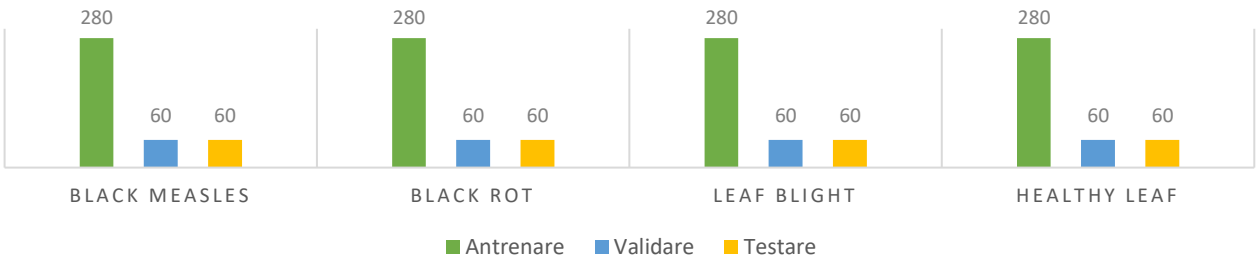


Figura 3.2 Distribuția datelor pentru fiecare clasă a datelor de antrenare, validare și testare

Se poate observa în figura 3.3 distribuția datelor în 70% antrenare, 15% validare și 15% testare pentru fiecare clasă în parte. În total sunt 1120 de imagini de antrenare, 240 de imagini pentru validare și 240 de imagini pentru testare.

```
Found 1120 images belonging to 4 classes.
Found 240 images belonging to 4 classes.
Found 240 images belonging to 4 classes.
Dimensiunea lotului de date pentru antrenare, validare și testare:  (8, 256, 256, 3)
Dimensiunea lotului de etichete pentru antrenare, validare și testare:  (8, 4)
```

Figura 3.3 Mesajele generatorului de date privind distribuirea imaginilor și numărul claselor

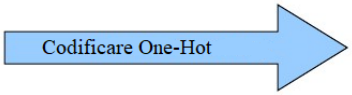
În figura 3.3 se poate confirma faptul că în urma utilizării unui generator de date împărțirea datelor s-a realizat în mod corespunzător procentajelor exprimate. Validarea acestui lucru se realizează prin intermediul funcției din Keras „flow_from_directory” care reprezintă o metodă a clasei ImageDataGenerator utilizată pentru încărcarea imaginilor direct în loturi din directoarele organizate în antrenare, validare și testare pentru fiecare clasă în parte. Clasa ImageDataGenerator din Keras este utilizată pentru generarea unor imagini noi prin transformări ale datelor inițiale de scalare, rotire, shift și multe altele. [26]

Imaginile din baza de date prezentată în capitolul 1.1.1 au dimensiunea de 256×256 pixeli. În antrenarea rețelei dimensionarea setului de date este necesară pentru a se potrivi antrenării unei rețele CNN. De asemenea, o practică comună este normalizarea valorilor de intrare pentru facilitarea procesului de antrenare a modelului de rețea neuronală. Normalizarea acestor valori de intrare se poate realiza mai ușor prin utilizarea clasei ImageDataGenerator. [Anexa 1] Normalizarea ajută la creșterea stabilității procesului de antrenare și poate contribui la o convergență mai eficientă a modelului.

Fiind o problemă de clasificare de multi-categorie fiecărei clase îi este asociată o cifră, un număr întreg, de exemplu, BlackMeasles reprezintă clasa 0, BlackRot – clasa 1, HealthyLeaf – clasa 2, iar LeafBlight – clasa 3. Deși codificarea etichetelor reprezintă o modalitate intuitivă de a atribui etichete datelor, ea prezintă dezavantajul că valorile numerice pot fi interpretate incorect de către algoritmi ca având o ierarhie sau o ordine specifică. Mai exact, utilizarea acestei codificări și permiterea modelului să presupună existența unei ordini naturale între categorii pot conduce la performanțe scăzute.

De aceea, One-Hot encoding este un procedeu de transformare a etichetelor claselor(un număr întreg) într-un vector binar distinct. În Keras prin intermediul funcției „flow_from_directory” care are parametrul „class_mode = categorical” se realizează automat transformarea. [Anexa 1]

id	culoare
1	roșu
2	albastru
3	verde
4	albastru



id	culoare_roșu	culoare_albastru	culoare_verde
1	1	0	0
2	0	1	0
3	0	0	1
4	0	1	0

Figura 3.4 Codificarea One-Hot [27]

Putem observa că pentru fiecare element cu un identificator avem o culoare. Aceste culori vor fi codificate în vectori binari. Asemănător vom face codificarea etichetelor (BlackMeasles, BlackRot, HealthyLeaf, LeafBlight) pentru clasificarea bolilor de viță de vie.

3.2 Modele de clasificare pentru baza de date PlantVillage

Modelul VGG16 este un model cunoscut de rețea neuronală convoluțională propus de Visual Geometry Group (VGG) la Universitatea Oxford. [28] În cadrul proiectului de cercetare ImageNet, modelul VGG16 a fost dezvoltat de la zero și antrenat pe o largă varietate de categorii de imagini, mai exact 1000 de clase. [29]

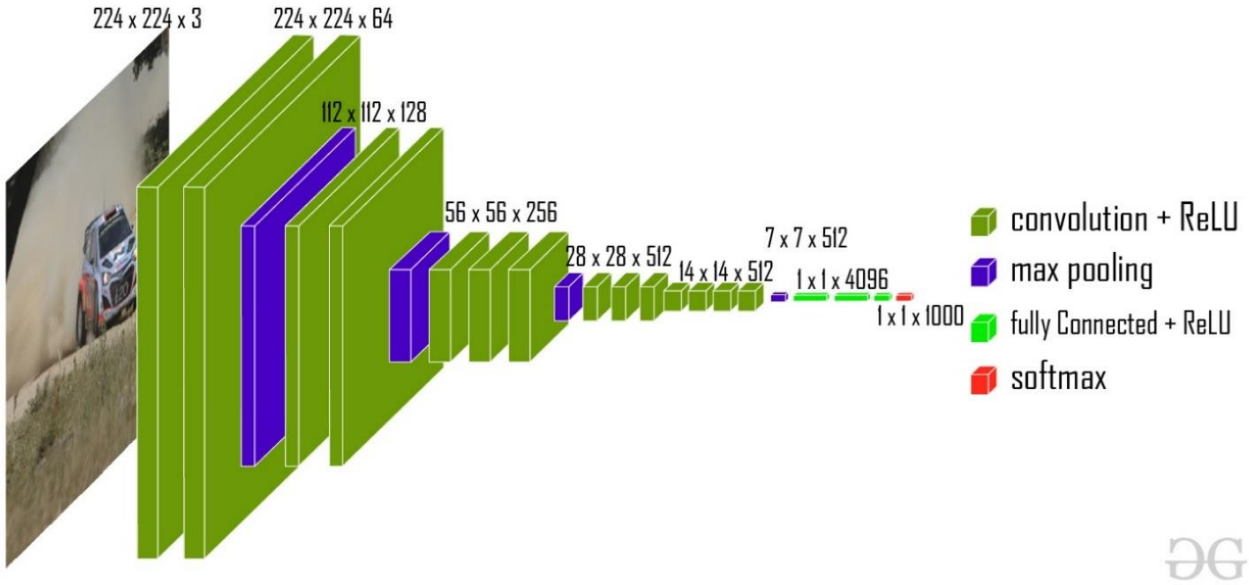


Figura 3.5 Rețea convoluțională VGG16 [28]

Modelul original a fost antrenat pe imagini de dimensiune 224 × 224 × 3 pixeli. Rețeaua este formată din 13 straturi de convoluție și 3 straturi dens conectate, de unde și denumirea de VGG-16. Vom explicita straturile în blocuri, VGG16 are 5 blocuri principale fiecare având un anumit număr de straturi de convoluție.

Primul bloc este format din 2 straturi de convoluție cu 64 de filtre de 3 × 3 dimensiune și padding 1 și al doilea este format din 2 straturi de convoluție, însă numărul de filtre s-a dublat la 128, fiecare având straturile de convoluție și unul de Max Pooling de dimensiune 2 × 2 și pas 2. Blocurile trei, patru și cinci sunt formate fiecare din 3 straturi de convoluție și unul de Max Pooling tot de dimensiune 2 × 2 și pas 2. Blocul trei are numărul de filtre egal cu 256, iar ultimele blocuri 512 de filtre, toate având dimensiunea de 3 × 3. Fiecare strat de convoluție are ca funcție de activare, funcția ReLU.

Ultimele 3 straturi sunt complet conectate. După blocul 5 s-a aplicat un strat de Flatten. Intrarea în straturile dens conectate este de 25088, rezultat în urma transformării hărților de caracteristici de dimensiune 7 × 7 × 512 într-un vector. Primele 2 straturi dens sunt formate din 4096 de neuroni, iar ultimul strat este format din 1000 de neuroni având ca funcție de activare, funcția Softmax, pentru clasificarea celor 1000 de categorii din baza de date ImageNet.

Întrucât imaginile din baza de date PlantVillage aleasă au dimensiunea 256 × 256 × 3 pixeli vom utiliza arhitectura VGG16 în continuare, însă datele de intrare vor fi modificare. De asemenea fiind un model pre-antrenat vom prelua ponderile deja antrenate și vom schimba doar structura straturilor dens pentru a se potrivi cu problema noastră de clasificare.

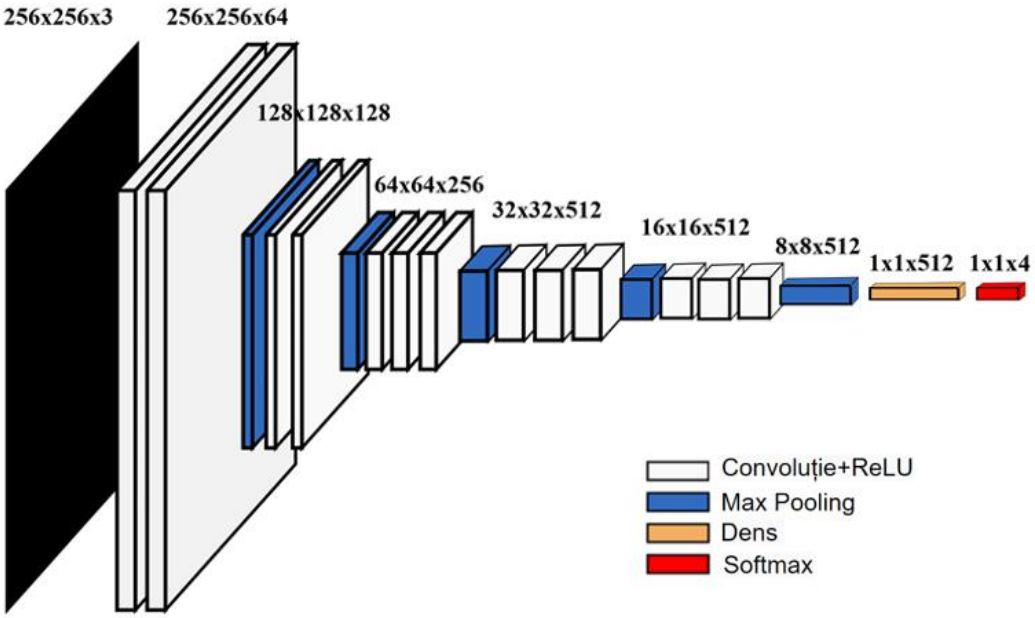


Figura 3.6 Rețea VGG16 pre-antrenată [29]

Rețeaua din figura 3.6 este o rețea clasică VGG16 modificată astfel încât să se potrivească cu datele noastre. Se observă faptul că intrarea are dimensiunea imaginilor din baza de date, iar straturile complet conectate sunt acum doar 2, primul având 512 neuroni și ultimul 4 care reprezintă numărul de clase pe care vrem să îl recunoaștem, având funcția de activare softmax. [Anexa 2]

În figura 3.6 sunt reprezentate și dimensiunile straturilor în urma operațiilor de convoluție și Max Pooling, de aceea pentru calculul dimensiunilor vom folosi următoarele formule:

$$\text{Dimensiunea de ieșire Conv2D} = Floor(\frac{(W-F+2P)}{S} + 1)$$

$$\text{Dimensiunea de ieșire MaxPooling2D} = Floor(\frac{(W-F)}{S} + 1) \text{ [30],}$$

Unde:

- Floor (Funcția Floor): Aceasta este o funcție matematică care, pentru un număr real *x*, returnează cel mai mare număr întreg care este mai mic sau egal cu *x*.
- W(Width): Lățimea imaginii de intrare.
- F(Filter): Dimensiunea filtrelor (kernel-ului) utilizate în stratul Conv2D sau dimensiunea ferestrei de MaxPooling în cazul stratului MaxPooling2D.
- P(Padding): Padding-ul aplicat imaginii de intrare. Padding-ul este adăugat în jurul imaginii de intrare pentru a menține dimensiunea imaginii constantă după aplicarea convoluțiilor.
- S(Stride): Pasul cu care filtrul sau fereastra de MaxPooling se mișcă peste imaginea de intrare. Acesta specifică cât de mult se deplasează filtrul sau fereastra de MaxPooling într-o singură mișcare.

Vom exemplifica rezultatul în urma aplicării unui strat de Conv2D și unul de MaxPooling2D:

$$\text{Dimensiunea de ieșire Conv2D} = Floor\left(\frac{(256-3+2\times1)}{1} + 1\right) = Floor(256) = 256$$

$$\text{Dimensiunea de ieșire MaxPooling2D} = Floor\left(\frac{(256-2)}{2} + 1\right) = Floor(128) = 128$$

În primul rând am reprezentat un model de referință cunoscut. În continuare vom prezenta un model clasic de rețea convoluțională, cât mai simplă formată din câteva straturi consecutive de Convoluție și Max Pooling, care va reprezenta modelul nostru de bază.

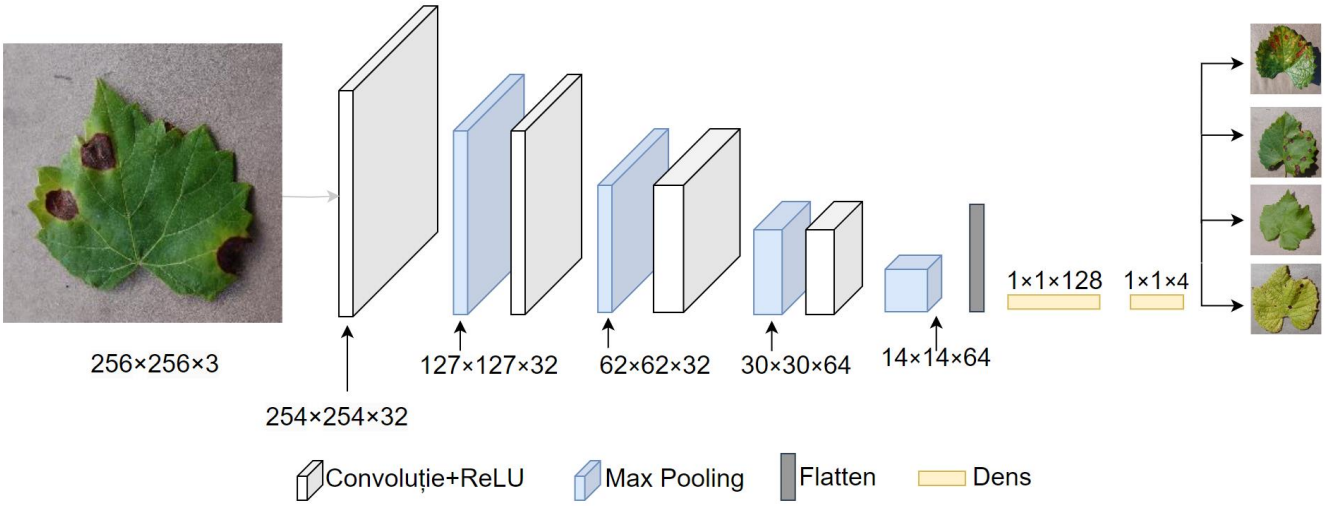


Figura 3.7 Rețea convoluțională

Acest model reprezintă o rețea neuronală convoluțională (CNN) concepută pentru clasificarea imaginilor în patru clase. Arhitectura sa este compusă dintr-o serie de straturi Conv2D și MaxPooling2D, care sunt utilizate pentru a extrage caracteristici semnificative din imagini.

Modelul începe cu două straturi Conv2D, fiecare cu 32 de filtre de dimensiune (3, 3), urmate de straturi de MaxPooling2D cu filtre de dimensiune 2 × 2 și pasul 2. Apoi, urmează încă două straturi Conv2D, fiecare cu 64 de filtre de dimensiune 3 × 3, pentru a captura caracteristici mai complexe. După fiecare strat Conv2D, este aplicat un strat de MaxPooling2D pentru reducerea dimensiunilor.

Modelul se încheie cu un strat Flatten, care transformă caracteristicile spațiale într-un vector unidimensional, și două straturi Dens pentru clasificarea finală. Ultimul strat Dens utilizează funcția de activare softmax pentru a produce probabilități pentru fiecare clasă la ieșire. În timpul antrenamentului, modelul este optimizat folosind algoritmul Adam și funcția de pierdere Categorical Crossentropy. [Anexa 2]

Vom exemplifica rezultatul în urma aplicării primelor straturi de Conv2D și MaxPooling2D:

Dimensiunea de ieșire Conv2D = $Floor\left(\frac{(256-3+2\times0)}{1}+1\right) = Floor(254) = 254$

Dimensiunea de ieșire MaxPooling2D = $Floor\left(\frac{(254-2)}{2}+1\right) = Floor(127) = 127$

IV. Evaluarea aplicației. Rezultate experimentale

În cadrul acestui capitol vom lua în considerare câteva metrici de măsurare a performanțelor modelelor dezvoltate, în vederea unei analize detaliate, cât și a unei comparații între modelul de referință ales și modelul propus.

- Matricea de confuzie

O matrice de confuzie este un tabel în care sunt reprezentate datele de testare pe care modelul nostru le distribuie în anumite clase. Întrucât în învățarea supervizată datele de testare au o etichetă adevărată, iar modelul va face predicții pe datele de intrare, vom analiza diferențele dintre valorile reale și cele prezise, iar din matricea de confuzie vom extrage informații care arată cât de bine clasifică modelul. [31]

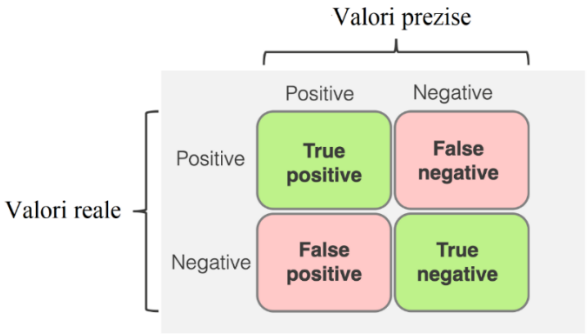


Figura 4.1 Matrice de confuzie [32]

Pentru a explica figura 4.1 vom lua un exemplu de 2 clase. Vom considera pozitiv faptul că o frunză e sănătoasă și negativ faptul că frunza este bolnavă.

- True positive (TP) reprezintă atunci când frunza este sănătoasă și predicția indică faptul că este sănătoasă.
- False negative (FN) reprezintă atunci când frunza este sănătoasă și predicția indică faptul că nu este sănătoasă.
- True negative (TN) reprezintă atunci când frunza nu este sănătoasă și predicția indică faptul că nu este sănătoasă.
- False positive (FP) reprezintă atunci când frunza nu este sănătoasă și predicția indică faptul că este sănătoasă.

Aceste elemente din matricea de confuzie ne indică diferențele dintre adevăr și predicția pe datele de testare, mai departe prin intermediul acestor instanțe putem avea o înțelegere mai profundă a rezultatelor.

În continuare vom explica metricile bazate pe datele rezultate din matricea de confuzie, iar predicțiile corecte vor fi TP și TN, iar erorile de predicție reprezintă FP și FN.

- Acuratețe

$$Acuratețe = \frac{TP + TN}{TP + FN + TN + FP}$$ [33]

Acuratețea reprezintă totalitatea predicțiilor corecte împărțite la toate predicțiile, oferind o privire de ansamblu asupra performanței modelului, indicând procentul total de elemente corect clasificate. [33]

- Recall

$$Recall = \frac{TP}{TP + FN}$$ [33]

Recall reprezintă totalitatea predicțiilor pozitive corecte împărțite la suma dintre toate predicțiile pozitive corecte (TP) și cele negative incorecte (FN). Recall arată cât de bine detectează modelul toate cazurile pozitive. Un recall apropiat de 1 indică faptul că detectează eficient majoritatea cazurilor pozitive. [33]

- Precizie

$$Precizie = \frac{TP}{TP + FP}$$ [33]

Precizia reprezintă totalitatea predicțiilor pozitive corecte împărțite la suma dintre toate predicțiile pozitive făcute de model. O precizie cât mai apropiată de 1 indică faptul că detectează eficient majoritatea cazurilor pozitive corecte și nu există prea multe cazuri negative clasificate incorect ca fiind pozitive. [33]

- F1 – score

$$F1 = \frac{2 \times Precizie \times Recall}{Precizie + Recall}$$ [32]

F1 – score reprezintă o medie armonică dintre precizie și recall. Această metrică urmărește echilibrul dintre cele 2 metrici, mai exact se urmărește reducerea predicțiilor pozitive incorecte (FP), cât și a predicțiilor negative incorecte (FN). [32]

În evaluarea performanței unui model de clasificare a bolilor specifice viței de vie este important să utilizăm metrici potrivite. Acuratețea ne oferă o privire de ansamblu asupra corectitudinii clasificării, recall și precizia sunt metrici care se concentrează asupra predicțiilor incorecte pentru fiecare clasă în parte, iar F1 – score ne arată echilibrul dintre precizie și recall în vederea predicțiilor FP și FN. [Anexa 5]

Întrucât problema noastră de clasificare se concentrează pe recunoașterea a 4 situații în care se poate afla vița de vie, vom utiliza o matrice de confuzie multi-clasă pentru a vedea diferențele dintre performanțele modelului nostru și celor ale modelului de referință.

În momentul în care analizăm o matrice de confuzie multi-clasă vom privi individual fiecare clasă. Matricea de confuzie ne arată diferența dintre distribuția adevărată a datelor și predicțiile modelului. O matrice ideală de confuzie va conține toate valorile nenule pe diagonala principală, indicând faptul că toate imaginile au fost clasificate corect.

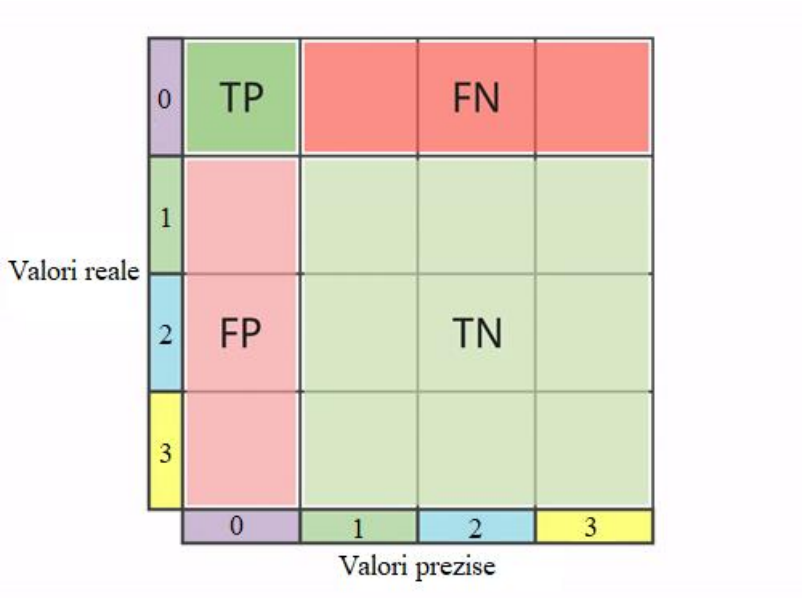


Figura 4.2 Matrice de confuzie multi-clasă [34]

Predicțiile corecte pentru fiecare clasa vor fi pe diagonala principală. Predicțiile incorecte vor fi observate în afara diagonalei principale. Valorile TP și TN cresc în momentul în care o valoare prezisă coincide cu valoarea reală. FN și FP reprezintă erori ale predicției și ele cresc în momentul în care modelul face o predicție greșită, iar valoarea prezisă nu coincide cu valoarea reală.

- VGG16

Pentru început vom testa modelul de referință ales adaptat la problema noastră de clasificare, explicat în capitolul anterior, pentru a putea analiza rezultatele metricilor pe setul nostru de date.

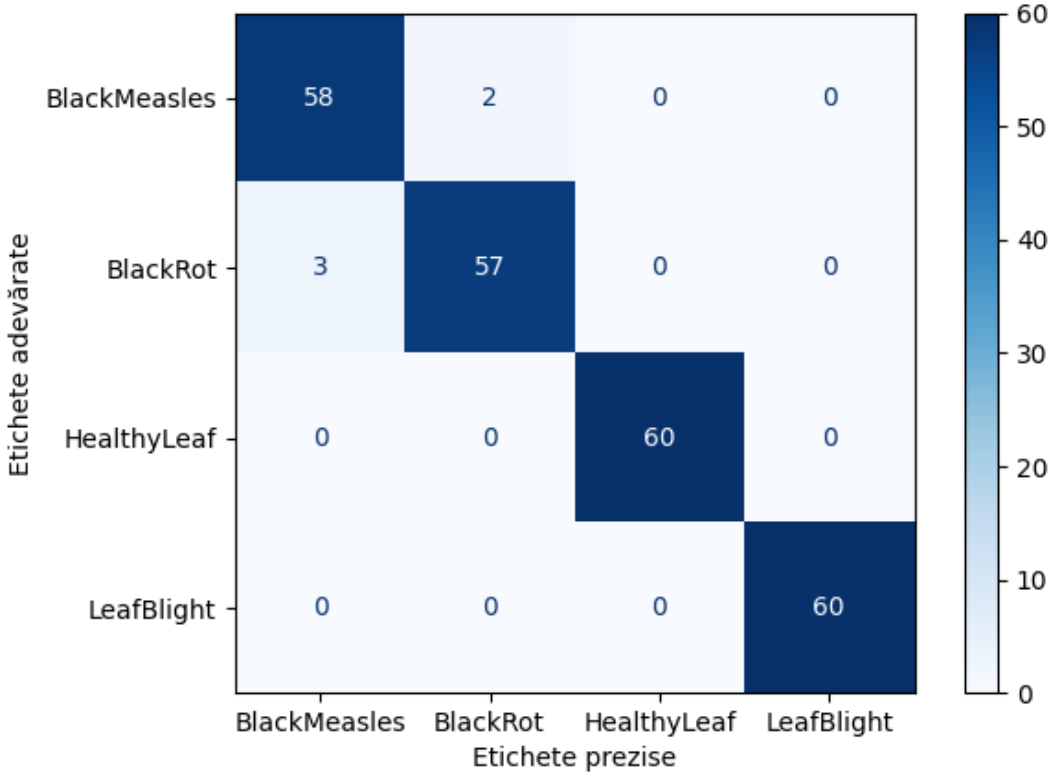


Figura 4.3 Matricea de confuzie pentru rețeaua pre-antrenată VGG16

În figura 4.3 este evident faptul că pentru clasele HealthyLeaf și LeafBlight nu avem erori de predicție, valorile prezise coincid cu cele reale, însă pentru clasa BlackMeasles există 2 erori în care imaginile au fost încadrate greșit într-o altă clasa BlackRot (FN), iar alte 3 imagini care nu aparțineau clasei BlackMeasles au fost prezise greșit (FP) în această clasă. Iar clasa BlackRot prezintă cele mai multe erori FN și 2 imagini care au fost încadrate în categoria BlackRot deși nu erau de acest tip.

Clasa	Precizie [%]	Recall [%]	F1-score [%]	Număr exemple per clasă
0 – BlackMeasles	95	97	96	60
1 – BlackRot	97	95	96	60
2 – HealthyLeaf	100	100	100	60
3 – LeafBlight	100	100	100	60
Acuratețe: 97%				

Tabel 4.1 Raportul de clasificare pentru rețeaua pre-antrenată VGG16

Erorile din matricea de confuzie pentru clasele 0 și 1 din figura 4.3 se pot observa în cadrul metricilor de precizie și recall din tabelul 4.1. Putem concluziona că pentru clase 2 și 3, metrica F1 – score este 1 ceea ce indica faptul că modelul clasifică fiecare imagine corect fără erori.

În cadrul antrenării modelelor trebuie să ne asigurăm de corectitudinea rezultatelor. Fiind un model de învățare automată ne așteptăm ca acesta să se îmbunătățească pe parcurs ce învață, deci ne așteptăm ca acuratețea să crească și ca pierderea să scadă. Cazul ideal este în momentul în care modelul reușește să prezică fiecare valoare corect, ceea ce înseamnă o acuratețe de 100% și o pierdere minima de 0, totuși acest lucru este puțin probabil pentru că până și oamenii fac greșeli în clasificare, un exemplu fiind studiul pe baza de date MNIST. [35]

De asemenea, trebuie să validăm faptul că modelul nostru se va descurca să recunoască și imagini nemaivăzute de acesta, de aceea am și împărțit setul nostru de date în seturi de antrenare și validare. Validarea este un proces prin care ne asigurăm că modelul nostru nu este supra-adaptat și poate fi îmbunătățit în urma antrenării, prin actualizarea hiperparametrilor. Verificarea graficelor de antrenare și validare ne ajută să observăm dacă modelul reușește să generalizeze și să facă predicții pe date noi. În cazul în care acest lucru nu se întâmplă vom observa fenomenul numit overfitting sau supra-învățare. [36]

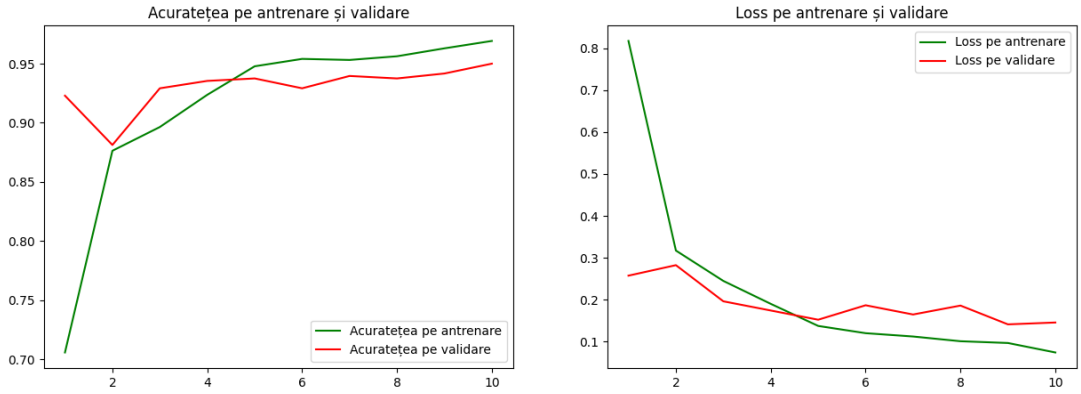


Figura 4.4 Graficele antrenării pentru rețeaua pre-antrenată VGG16 [Anexa 4]

Rețeaua VGG16 este o rețea neuronală profundă antrenată pe un set mare de date. Antrenarea rețelei VGG16 s-a realizat pe 10 epoci prin preluarea ponderilor pre-antrenate.

- CNN

Pentru modelul de bază vom testa aceleași metrici și vom compara rezultatele obținute cu cele ale modelului de referință.

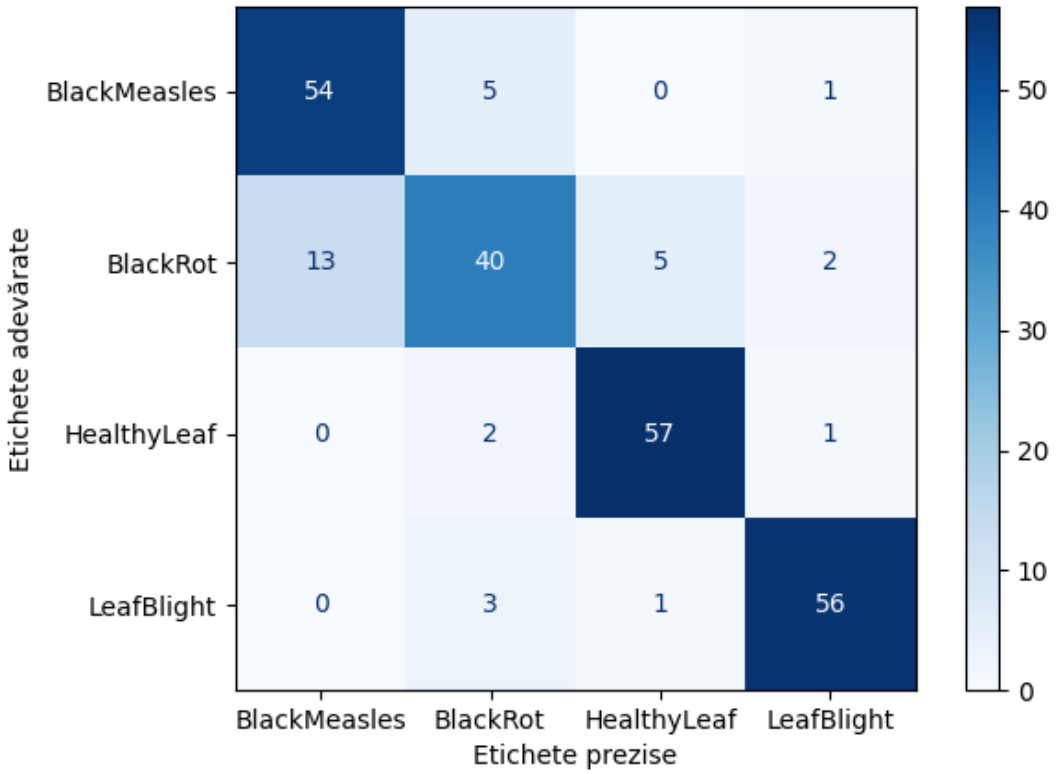


Figura 4.5 Matricea de confuzie pentru rețeaua convoluțională

În primul rând putem identifica faptul că matricea de confuzie prezintă mai multe erori decât matricea din figura 4.3. Putem observa faptul că pentru clasa BlackRot apar cele mai multe erori, însemnând că modelul nostru nu recunoaște foarte bine această clasă, fiind confundată cu o clasa cu caracteristici asemănătoare BlackMeasles. Clasa cu cele mai puține erori este clasa HealthyLeaf având valoarea TP cea mai mare.

Clasa	Precizie [%]	Recall [%]	F1-score [%]	Număr exemple per clasă
0 – BlackMeasles	81	90	85	60
1 – BlackRot	80	67	73	60
2 – HealthyLeaf	90	95	93	60
3 – LeafBlight	93	93	93	60
Acuratețe: 86.25%				

Tabel 4.2 Raportul de clasificare pentru rețeaua convoluțională

Este notabil că în tabelul 4.2 metricile de pe linia clasei BlackRot au valori ceva mai scăzute, însă sunt de un nivel bun. Valoarea cea mai mică recall ne indică faptul că multe dintre imagini care aparțin clasei BlackRot sunt confundate cu clasa BlackMeasles, probabil datorita similitudinii dintre frunzele afectate de boală care de multe ori sunt greu de distins și de către ochiul uman.

Cu toate acestea modelul prezintă per total o acuratețe bună și valori ale metricilor destul de ridicate, însă mai trebuie verificat dacă modelul nostru s-a antrenat în mod corect.

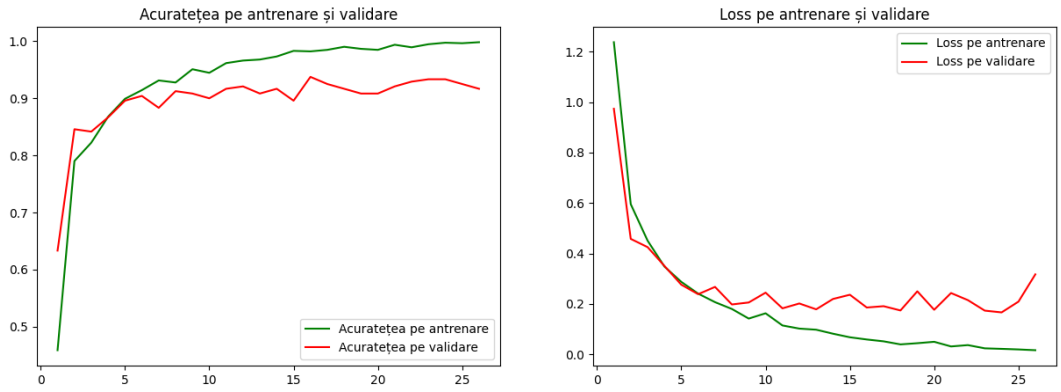


Figura 4.6 Graficele antrenării pentru rețeaua convoluțională [Anexa 4]

Se poate vedea faptul că în urma antrenării graficele pentru acuratețea pe validare și antrenare se îndepărtează, ceea ce poate reprezintă un semn al supra-învățării. Rezultatele matricei de confuzie sunt bune, însă acestea pot fi îmbunătățite.

Inițial am configurat ca antrenarea să ruleze pe 50 de epoci, însă s-a oprit la epoca 25 pentru că am utilizat din biblioteca Keras o funcție de callback numită Early Stopping. Este o tehnică de regularizare prin care se previne supra-învățarea în timpul antrenării și se urmărește evoluția pe setul de validare la fiecare epocă. Dacă performanța nu se îmbunătățește după un anumit număr de epoci se va opri automat antrenarea. [37]

- CNN îmbunătățit

În urma antrenării rețelei CNN am observat o tendință de supra-învățare. Vom încerca o serie de metode de regularizare ale modelului astfel încât să obținem rezultate mai bune. În continuare vom utiliza funcția de Early Stopping, deoarece reprezintă o metodă utilă prin care economisim timpul de antrenare și, de asemenea, ne asigurăm că vom păstra cea mai bună performanță a modelului, prin parametrul „restore_best_weights” care va memora ponderile din epoca cu cea mai bună valoare a metricii de evaluare, adică acuratețea. [Anexa 3]

Vom folosi o altă tehnică întâlnită de regularizare pentru prevenirea supra-învățării, numită Dropout, explicată în capitolul II. Această tehnică de Dropout face modelul capabil să generalizeze mai bine datele noi, deoarece prin anularea aleatorie a neuronilor rețeaua trebuie să găsească noi căi și să se bazeze pe neuroni diferiți. [38]

Prin comparare cu rețeaua de referință, VGG16, am ales să mai adăugăm straturi suplimentare de convoluție cu un număr mai mare de filtre pentru a face modelul să capteze mai multe detalii despre fiecare clasă în parte. Alături de convoluție am adăugat și Max Pooling pentru reducerea dimensiunii datelor de ieșire din stratul de convoluție.

Pentru a observa și vizual modificările aduse arhitecturii prezentate în figura 3.7 am reprezentat grafic rețeaua în figura de mai jos prin intermediul unui software online de creare al diagramelor, numit draw.io. [39]

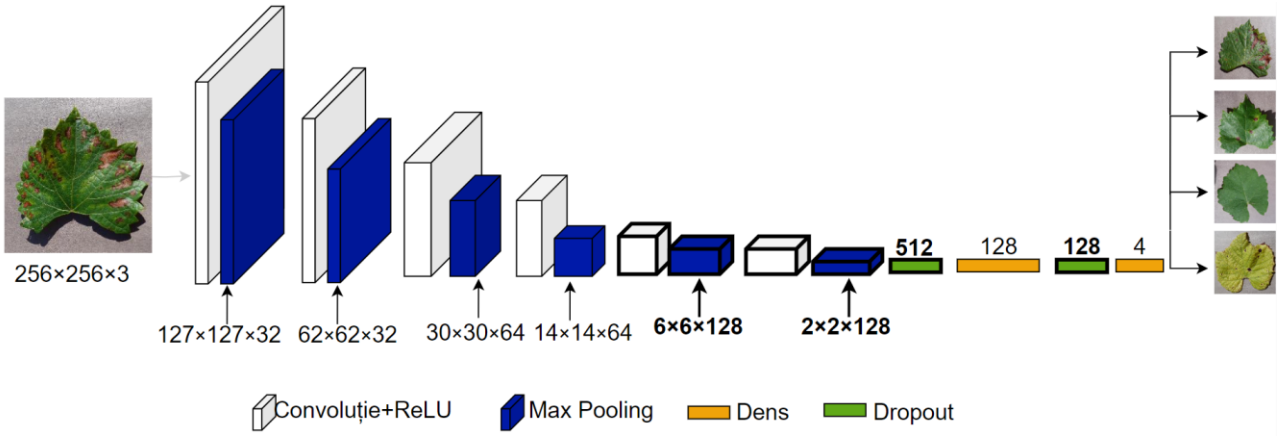


Figura 4.7 Rețea convoluțională îmbunătățită

În figura 4.7 putem observa rețeaua CNN îmbunătățită. Sub fiecare bloc de convoluție și Max Pooling am reprezentat dimensiunea fiecărui filtru în urma operației de Max Pooling pentru o reprezentare mai simplificată a modelului, iar dimensiunile în urma stratului de convoluție au fost exemplificate în capitolul III la reprezentarea aplicării operațiilor de convoluție și Max Pooling. Straturile de Dropout au un procent de anulare de 20%, de asemenea, se poate vedea faptul că am plasat straturile Dropout după stratul de Flatten și după stratul dens cu 128 de neuroni pentru a îmbunătăți performanța modelului. [Anexa 4]

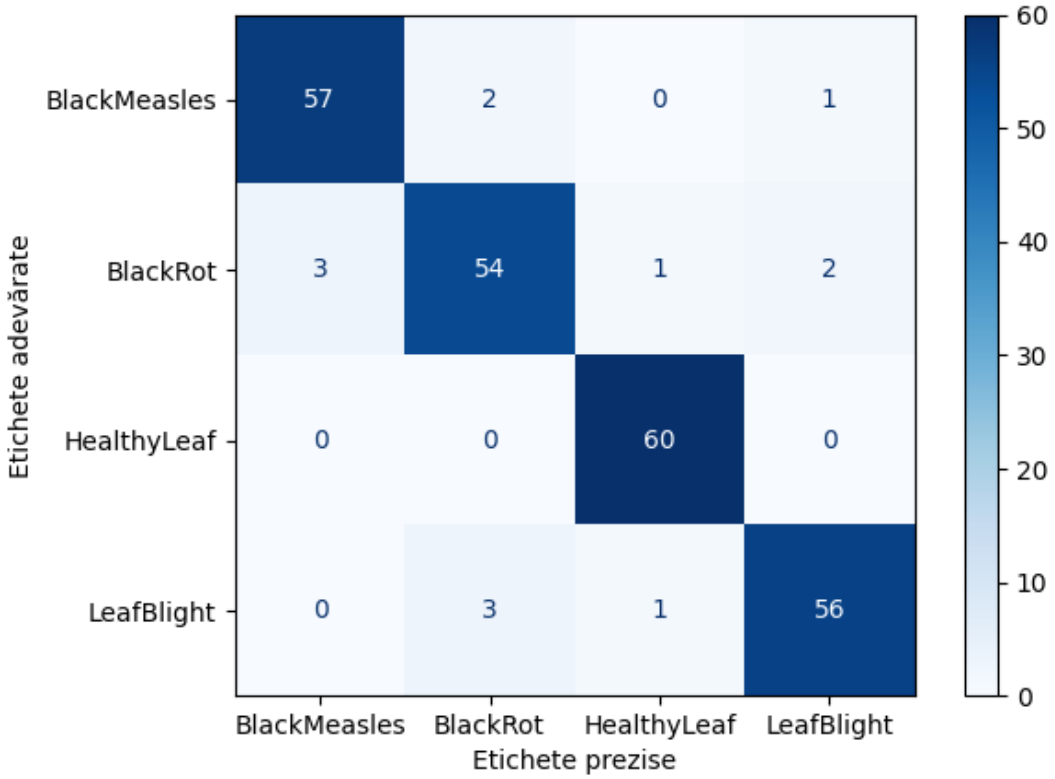


Figura 4.8 Matricea de confuzie pentru rețeaua convoluțională îmbunătățită

Față de matricea de confuzie din figura 4.5, putem constata o distribuire mai uniformă a datelor TP pentru fiecare clasă în parte în figura 4.8. Clasa BlackRot nu mai prezintă așa multe erori de predicție, iar pentru clasa HealthyLeaf modelul a atins maximul prin predicția corectă a fiecărei imagini. De asemenea, pentru clasele BlackMeasles și LeafBlight a scăzut numărul erorilor.

Clasa	Precizie [%]	Recall [%]	F1-score [%]	Număr exemple per clasă
0 – BlackMeasles	95	95	95	60
1 – BlackRot	92	90	91	60
2 – HealthyLeaf	97	100	98	60
3 – LeafBlight	95	93	94	60
Acuratețe: 94.58%				

Tabel 4.3 Raportul de clasificare pentru rețeaua convoluțională îmbunătățită

Prin analogie cu tabelul 4.1, putem compara faptul că valorile metricilor pentru clasa BlackRot sunt mult mai ridicate. Per total tabelul 4.3 prezintă o performanță ridicată. Comparativ cu modelul de referință, modelul de bază are o recunoaștere mai bună a preciziei pentru clasa BlackMeasles și un recall mai bun pentru clasa BlackRot, însă performanțele pentru clasa HealthyLeaf și LeafBlight sunt mai scăzute decât performanțele modelului de referință.

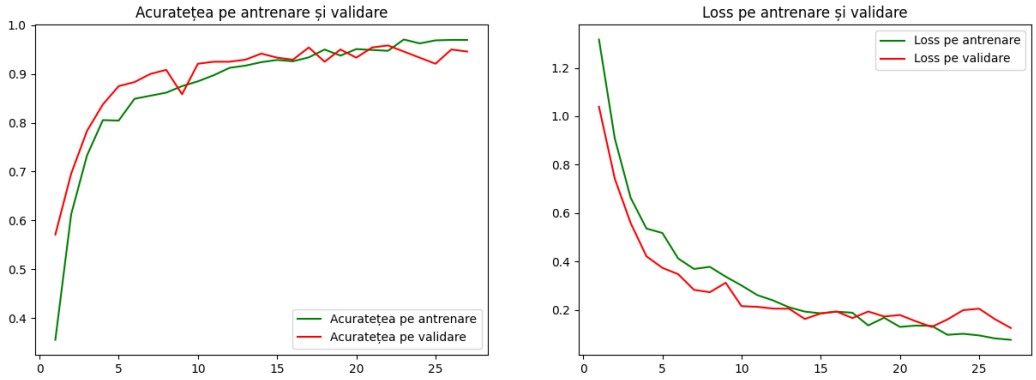


Figura 4.9 Graficele antrenării pentru rețeaua convoluțională îmbunătățită [Anexa 4]

În urma modificărilor aduse rețelei CNN de bază se remarcă faptul că rezultatele sunt mai bune, iar pe graficul de antrenare din figura 4.9 nu există semne de supra-învățare, curbele de antrenare și validare au aceeași evoluție, acestea sunt apropiate și se suprapun, sugerând că modelul nostru se comportă asemănător pe datele de antrenare și pe cele de validare.

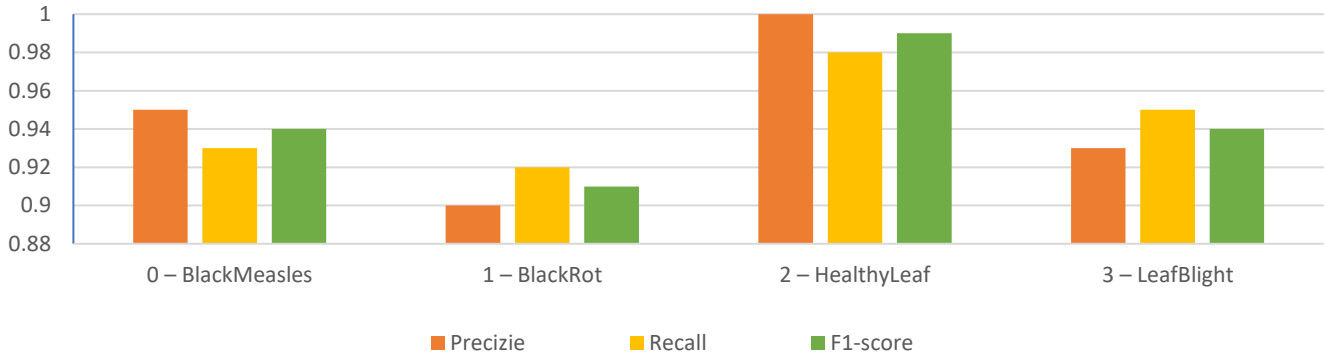


Figura 4.10 Raportul de clasificare pentru rețeaua convoluțională îmbunătățită

Valorile metricilor de evaluare din figura 4.10 au performanțe ridicate. În general modelul are capacitatea de a clasifica corect majoritatea cazurilor, pentru clasa HealthyLeaf fiind cel mai performant, iar pentru BlackRot are valori puțin mai mici, însă tot cu un nivel bun.

Pentru început pentru evaluarea modelului am creat o bază de testare formată din imagini din baza de date PlantVillage conținând din 240 de imagini.

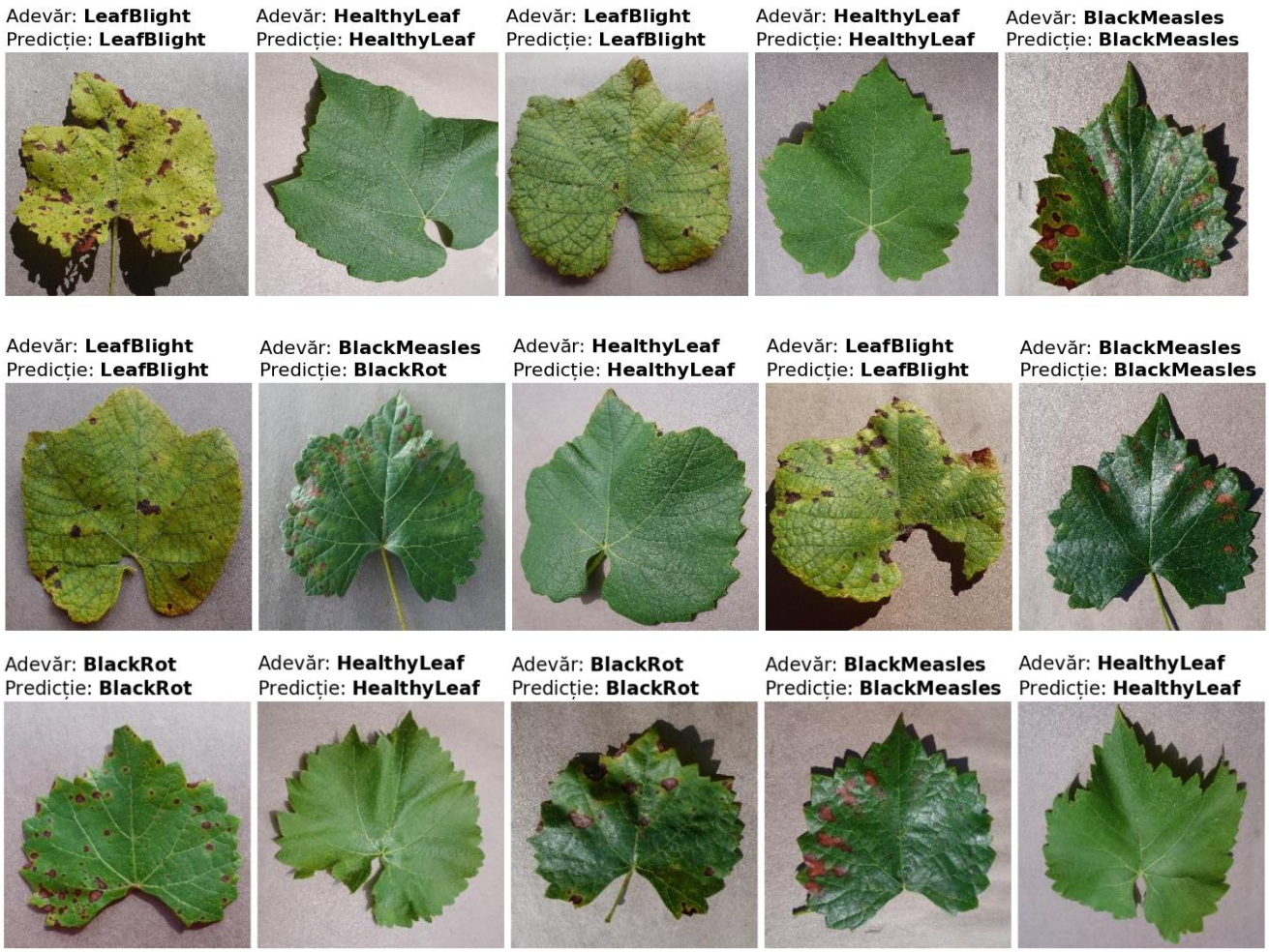


Figura 4.11 Predicții pe setul de testare PlantVillage

În figura 4.11 avem afișate câteva dintre predicțiile modelului nostru CNN îmbunătățit. Pe rândul 2, coloana 2 vom evidenția un tip de eroare. Dacă privim din punct de vedere al clasei BlackMeasles această eroare este de tip FN, unde eticheta adevărată este BlackMeasles, iar predicția este o altă clasă BlackRot. Dacă privim din punct de vedere al clasei BlackRot reprezintă un tip de eroare FP, fiindcă pentru clasa BlackRot a apărut o predicție în plus care nu este adevărată.

Pentru a evalua capacitatea modelului CNN antrenat pe baza de date PlantVillage de a generaliza în condiții reale, vom testa în continuare modelul pe un set de imagini colectate din surse online. Aceste imagini arată frunze fotografiate în mediul lor natural, cu diferite condiții de iluminare, zgomot de fundal și unghiuri de fotografiere variate.

Acest test este important pentru a vedea dacă modelul nostru poate să recunoască corect bolile plantelor și în afara laboratorului, unde frunzele pot fi afectate de mai mulți factori în același timp. Evaluarea pe aceste imagini ne ajută să determinăm cât de bine generalizează modelul și ce îmbunătățiri ar putea fi necesare pentru utilizarea lui în practică, pe teren.

În continuare vom realiza o analiză comparativă pe setul de date extins dintre modelul de referință și modelul de bază optimizat prin intermediul metricilor de evaluare a performanței aplicației prezentate la începutul acestui capitol.

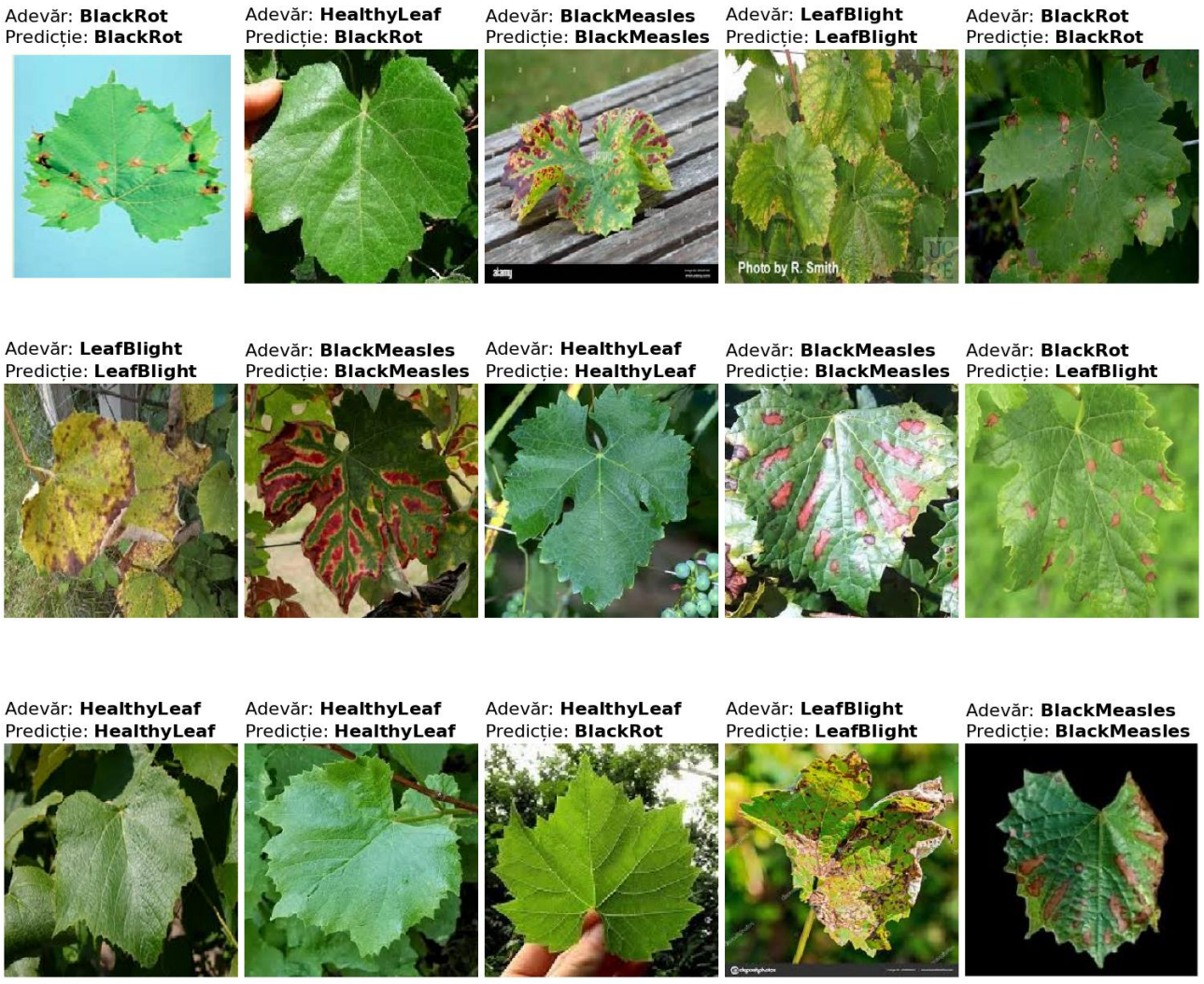


Figura 4.12 Predicții ale modelului de bază pe imagini de test disponibile online

În figura 4.12 sunt afișate câteva tipuri de imagini de test alese pentru verificarea fiabilității modelului. Este vizibil faptul că imaginile din figura 4.11 conțin multe frunze aflate și în mediul lor natural înconjurate de diverse obiecte, ceea ce reprezintă o provocare pentru modelul nostru antrenat pe baza de date PlantVillage care este formată din frunze singure pe o masă gri într-o anumită lumină. Această provocare ne va indica dacă rețeaua noastră este pregătită pentru aplicarea pe cazuri reale.

Baza de testare cu imagini disponibile online este formată din 48 de imagini, distribuite în mod egal pentru cele 4 categorii, rezultând 12 exemple pentru fiecare clasă. În continuare vom analiza și metricile de performanță. Ne așteptăm ca rezultatele să fie mai scăzute întrucât noul set conține variante diferite față de ceea ce a fost învățat. Cu toate acestea ne așteptăm ca modelul nostru să fie capabil să învețe cât mai multe caracteristici specifice fiecărei boli și să nu fie influențat de fundalul gri.

Imaginile provenite din surse online au diferite dimensiuni, de aceea este necesar ca acestea să treacă printr-un proces de redimensionare pentru ca tipul datelor să se potrivească cu tipul de date acceptate de model. Redimensionarea s-a realizat prin intermediul unui parametru din cadrul funcției „flow_from_directory”, numit „target_size” care a primit valoarea de 256 × 256 pixeli. În rest preprocesarea s-a realizat în același mod prezentat la începutul capitolului III.

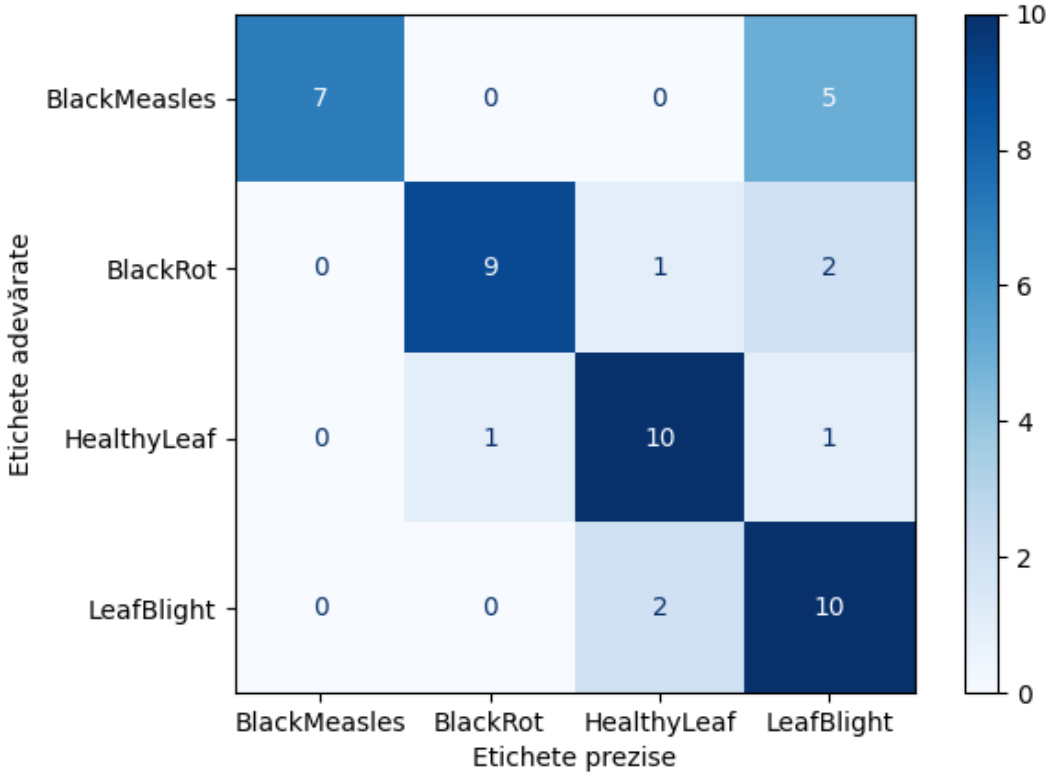


Figura 4.13 Matricea de confuzie pentru VGG16 pe setul de date extins

Matricea de confuzie din figura 4.13 prezintă mult mai multe erori decât ce am observat până acum, dar se explică ținând cont de setul de date ales. Pare că modelul confundă mai multe clase între ele. Comparativ cu rezultatele pe setul de testare PlantVillage, din figura 4.3, clasele cel mai bine recunoscute sunt din nou HealthyLeaf și LeafBlight.

Clasa	Precizie [%]	Recall [%]	F1-score [%]	Număr exemple per clasă
0 – BlackMeasles	100	58	74	12
1 – BlackRot	90	75	82	12
2 – HealthyLeaf	77	83	80	12
3 – LeafBlight	56	83	67	12
Acuratețe VGG16: 75%				

Tabel 4.4 Raportul de clasificare pentru VGG16 pe setul de imagini disponibile online

Cu siguranță primul lucru pe care l-am analizat a fost acuratețea care este mult mai scăzută decât pe setul de date PlantVillage din tabelul 4.1, lucru care poate fi explicat de faptul că imaginile de antrenare sunt similare cu cele de testare aparținând din același set PlantVillage au o oarecare similitudine fiind creat într-un mediu de laborator în aceleași condiții.

Per total valorile metricilor au valori medii, indicând că modelul nostru se descurcă suficient de bine cu datele noi. Dar valorile metricilor ne indică pentru ce clasă modelul nostru are anumite deficiențe: multe dintre imaginile BlackMeasles au fost încadrate într-o altă clasa, prezentând cele mai puține valori de TP, multe predicții fiind departe de adevăr.

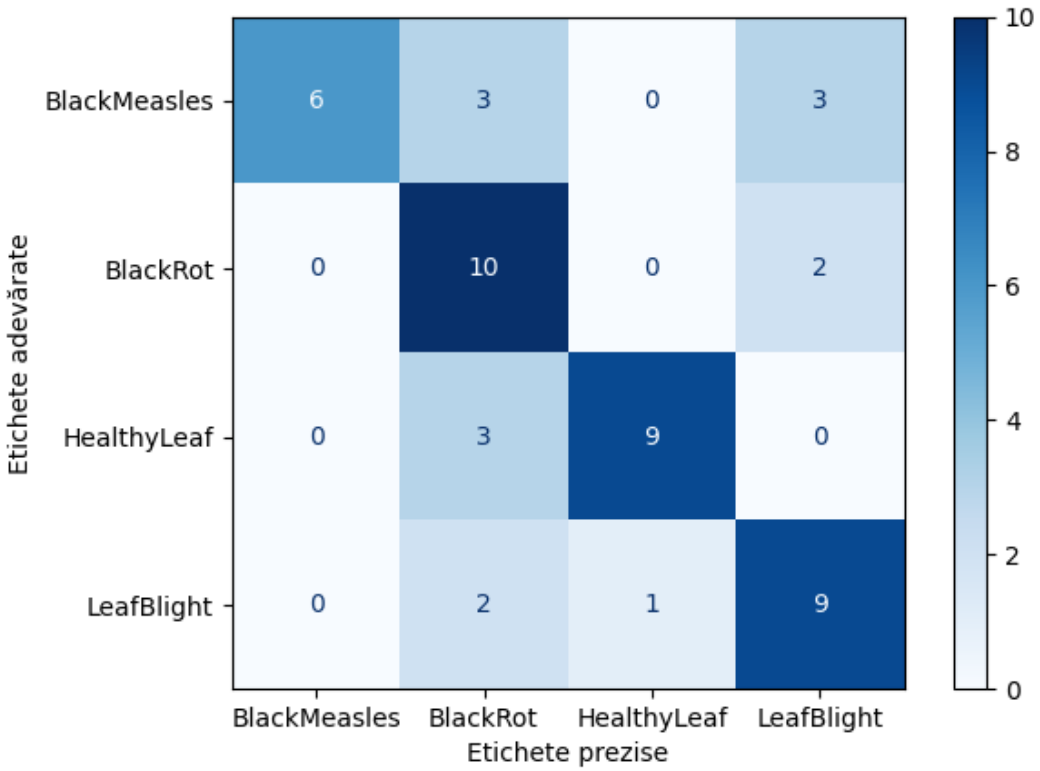


Figura 4.14 Matricea de confuzie pentru CNN pe setul de date extins

În figura 4.14 matricea de confuzie prezintă mai multe erori decât matricea de confuzie din figura 4.13, dar în același timp putem observa că totuși modelul de bază CNN este capabil să recunoască majoritatea cazurilor TP. Cu siguranță modelul pare să confunde cel mai mult clasa BlackMeasles cu clasele BlackRot și LeafBlight ceea ce va duce la un recall scăzut.

Clasa	Precizie [%]	Recall [%]	F1-score [%]	Număr exemple per clasă
0 – BlackMeasles	100	50	67	12
1 – BlackRot	59	83	69	12
2 – HealthyLeaf	91	83	87	12
3 – LeafBlight	64	75	69	12
Acuratețe CNN: 72.92%				

Tabel 4.5 Raportul de clasificare pentru CNN pe setul de imagini disponibile online

Din câte putem remarca, modelul de bază prezintă o performanță apropiată de modelul de referință ceea ce indică faptul că acesta este mai bine optimizat. Din matricea de confuzie am observat că jumătate din valorile clasei BlackMeasles sunt confundate cu alte clase ceea ce va duce la un recall de 50%, ceea ce înseamnă că modelul nostru întâmpină dificultăți în identificarea clasei BlackMeasles.

Din rezultatele obținute anterior la evaluarea aplicațiilor pe setul de testate PlantVillage am observat tendința modelului VGG16 de a recunoaște mai bine clasele HealthyLeaf și LeafBlight, iar a modelului de bază CNN recunoaște mai bine clasa BlackRot. Însă precizăm faptul că există multiple surse de provenire a imaginilor online, cât și varietăți ale culturilor de viță de vie, iar simptomele bolilor pot varia în funcție de acești factori. [40]

PlantVillage este un început bun pentru problemele de clasificare a bolilor însă este un set sub-reprezentat când vine vorba de diversitatea bolilor. În general am observat că tendința imaginilor disponibile online este de a ilustra frunzele afectate în ultimele stadii, iar PlantVillage este concentrată mai mult pe detecția timpurie a simptomelor. De aceea există o insuficiență a reprezentării frunzelor într-un mediu natural. [41]

Concluzii și posibile îmbunătățiri viitoare

Scopul acestei lucrări este de a dezvolta o aplicație folosind limbajul Python pentru a evalua starea unei culturi de viță de vie prin analiza de imagini. În urma studiului tehnicilor deja existente și utilizate în agricultura de precizie și al bazelor de date disponibile online am considerat că baza de date PlantVillage este un set reprezentativ des folosit în rezolvarea problemelor de clasificare.

În urma cercetării metodelor de învățare automată am observat că arhitecturile CNN prezintă o eficiență mai mare întrucât metodele de Computer Vision prezintă o parte pe preprocesare laborioasă ce necesită mai mult timp. Așadar ca urmare a dezvoltării, antrenării și testării modelului de bază de tip învățare profundă și pe un set de date care conține imagini în diverse ipostaze și nu numai pe un fundal gri am observat faptul că modelul nostru are potențialul de a deveni un instrument în managementul culturilor de viță de vie.

Contribuția mea a constat în următoarele aspecte:

- Selecția unei baze de date adecvată temei proiectului;
- Organizarea și prelucrarea bazei de date prin împărțirea în antrenare, validare și testare;
- Includerea unui model de referință performant VGG16 pentru o analiză comparativă cu modelul de bază;
- Simularea unui model de bază și dezvoltarea unei variante mai eficiente în clasificare a acestuia;
- Interpretarea rezultatelor în vederea evaluării modelului de bază;
- Alegerea unei baze de date de test cu imagini disponibile online și testarea performanței modelelor pe acest set pentru o evaluare suplimentară a capacității modelului de bază la diverse condiții din situațiile reale.

PlantVillage oferă o perspectivă restrânsă atunci când vine vorba de analiza stării unei culturi de viță de vie, întrucât simptomele unei boli se pot regăsi pe lângă frunze, pe tulpini și ciorchinii de struguri. O analiză a întregii plante necesită un set de date mai complex care ar conține poze capturate direct în câmpul de viță de vie, pe locul de producție, pentru o monitorizare mai atentă, fiind importantă și analiza stării strugurilor.

Eventualele îmbunătățiri viitoare se concentrează în jurul setului de date, fiind nevoie de o diversificare a acestuia luând în considerare condițiile de mediu în care se regăsesc plantele. Modelul dezvoltat poate să fie baza unui echipament de recunoaștere a bolilor de viță de vie atașat unui UAV care capturează în timp real imagini de pe câmpul culturilor de struguri, însă pentru acest proiect modelul ar necesita o antrenare mai riguroasă pe mai multe date care conțin diferite unghiuri sau iluminări ale frunzelor. O altă îmbunătățire ar fi extinderea studiului pe alte părți ale plantelor de viță, precum struguri sau tulpini.

Bibliografie

[1] K. P. Seng, L. -M. Ang, L. M. Schmidtke and S. Y. Rogiers, "Computer Vision and Machine Learning for Viticulture Technology," in IEEE Access, vol. 6, pp. 67494-67510, 2018, doi: 10.1109/ACCESS.2018.2875862.

[2],,Identification Guide to the Major Diseases of Grapes" https://agriculture.canada.ca/en/agricultural-production/crop-protection/agricultural-pest-management-resources/identification-guide-major-diseases-grapes . Accesat la data de: 17.12.2023.

[3] Făinarea la vița de vie https://www.horticultorul.ro/insecte-boli-daunatori-fungicide-insecticide-ingrasaminte-pesticide/fainarea-la-vita-de-vie/. Accesat la data de: 18.12.2023.

[4] Using Deep Learning for Image-Based Plant Disease Detection https://www.frontiersin.org/articles/10.3389/fpls.2016.01419/full. Accesat la data de: 18.12.2023.

[5] An expertized grapevine disease image database focused on Flavescence dorée and its confounding diseases https://data.mendeley.com/datasets/3dr9r3w3jn/2. Accesat la data de: 11.02.2024.

[6] Two-stage automatic diagnosis of Flavescence Dorée based on proximal imaging and artificial intelligence: a multi-year and multi-variety experimental study. https://oeno-one.eu/article/view/5460 . Accesat la data de: 11.02.2024.

[7] Matese, Alessandro, and Salvatore Filippo Di Gennaro. "Technology in precision viticulture: A state of the art review." International journal of wine research (2015): 69-81.

[8] Jaisakthi, S. M., P. Mirunalini, and D. Thenmozhi. "Grape leaf disease identification using machine learning techniques." 2019 International conference on computational intelligence in data science (ICCIDS). IEEE, 2019.

[9] Szeliski, Richard. Computer vision: algorithms and applications. Springer Nature, 2022.

[10] Singh, Ujjwal, et al. "Computer vision technique for detection of grape esca (Black Measles) disease from grape leaf samples." *2020 International Conference on Contemporary Computing and Applications (IC3A)*. IEEE, 2020.

[11] Madhavan, Mangena Venu, et al. "Recognition and classification of pomegranate leaves diseases by image processing and machine learning techniques." *Computers, Materials & Continua* 66.3 (2021): 2939-2955.

[12] Kubat, Miroslav. "Introduction to machine learning." Advanced Topics in Artificial Intelligence: International Summer School Prague, Czechoslovakia, July 6–17, 1992 Proceedings. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. 104-138.

[13] SVM Classifier Based Grape Leaf Disease Detection https://sci-hub.se/10.1109/casp.2016.7746160 https://ieeexplore.ieee.org/abstract/document/7746160. Accesat la data de: 09.03.2024.

[14] Huang, Z., Qin, A., Lu, J., Menon, A., & Gao, J. (2020, November). „Grape leaf disease detection and classification using machine learning”. In 2020 international conferences on internet of things (iThings) and IEEE green computing and communications (GreenCom) and IEEE cyber, physical and social computing (CPSCom) and IEEE smart data (SmartData) and IEEE congress on Cybermatics (Cybermatics) (pp. 870-877). IEEE.

[15] Aggarwal, Charu C. "Neural networks and deep learning." Springer 10.978 (2018): 3.

[16] Ahmad, Iftikhar, et al. "Optimizing pretrained convolutional neural networks for tomato leaf disease detection." Complexity 2020.1 (2020): 8812019.

[17] Geetharamani, G., and Arun Pandian. "Identification of plant leaf diseases using a nine-layer deep convolutional neural network." Computers & Electrical Engineering 76 (2019): 323-338.

[18] Santos, Thiago T., et al. "Grape detection, segmentation, and tracking using deep neural networks and three-dimensional association." Computers and Electronics in Agriculture 170 (2020): 105247.

[19] Convolutional Layer https://www.sciencedirect.com/topics/computer-science/convolutional-layer . Accesat la data de: 10.03.2024.

[20] Skansi, Sandro. Introduction to Deep Learning: from logical calculus to artificial intelligence. Springer, 2018.

[21] Activation Functions in Neural Networks https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6 . Accesat la data de: 10.03.2024.

[22] NN-SVG Publication-ready NN-architecture schematics. https://alexlenail.me/NN-SVG/index.html . Accesat la data de: 11.03.2024.

[23] Softmax Activation Function Explained https://towardsdatascience.com/softmax-activation-function-explained-a7e1bc3ad60 . Accesat la data de: 11.03.2024.

[24] Implementing Drop Out Regularization in Neural Networks https://www.tech-quantum.com/implementing-drop-out-regularization-in-neural-networks/. Accesat la data de: 12.03.2024.

[25] Tutorial on using Keras flow_from_directory and generators https://vijayabhaskar96.medium.com/tutorial-image-classification-with-keras-flow-from-directory-and-generators-95f75ebe5720 . Accesat la data de: 17.04.2024.

[26] Building a One Hot Encoding Layer with TensorFlow https://towardsdatascience.com/building-a-one-hot-encoding-layer-with-tensorflow-f907d686bf39 . Accesat la data de: 17.04.2024.

[27] VGG-16 | CNN model https://www.geeksforgeeks.org/vgg-16-cnn-model/ . Accesat la data de: 25.04.2024.

[28] Tammina, Srikanth. "Transfer learning using vgg-16 with deep convolutional neural network for classifying images." International Journal of Scientific and Research Publications (IJSRP) 9.10 (2019): 143-150.

[29] Aulia, Suci, and Dadi Rahmat. "Brain tumor identification based on VGG-16 architecture and

CLAHE method." JOIV: International Journal on Informatics Visualization 6.1 (2022): 96-102.

[30] Calculate Output Size of Convolutional and Pooling layers in CNN. https://androidkt.com/calculate-output-size-convolutional-pooling-layers-cnn/ . Accesat la data de: 18.04.2024.

[31] Confusion Matrix in Machine Learning https://www.geeksforgeeks.org/confusion-matrix-machine-learning/ . Accesat la data de: 11.03.2024.

[32] How to interpret a confusion matrix for a machine learning model https://www.evidentlyai.com/classification-metrics/confusion-matrix . Accesat la data de: 11.03.2024.

[33] Confusion Matrix https://vtantravahi.medium.com/confusion-matrix-96fb002e13d1 . Accesat la data de: 11.03.2024.

[34] Visual Guide to the Confusion Matrix https://towardsdatascience.com/visual-guide-to-the-confusion-matrix-bb63730c8eba . Accesat la data de: 13.05.2024.

[35] Russell, Stuart, and Peter Norvig. "Artificial Intelligence: A Modern Approach, 4th, Global ed." (2022).

[36] Ying, Xue. "An overview of overfitting and its solutions." Journal of physics: Conference series. Vol. 1168. IOP Publishing, 2019.

[37] Prechelt, Lutz. "Early stopping-but when?." Neural Networks: Tricks of the trade. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002. 55-69.

[38] Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting." The journal of machine learning research 15.1 (2014): 1929-1958.

[39] Flowchart Maker and Online Diagram Software. https://app.diagrams.net/ . Accesat la data de: 30.05.2024.

[40] Deshpande, Priya, and Sharada Kore. "Disease Detection for Grapes: A Review." International Conference on Computational Intelligence. Singapore: Springer Nature Singapore, 2022.

[41] Mohimont, Lucas, et al. "Computer vision and deep learning for precision viticulture." Agronomy 12.10 (2022): 2463.

Anexe

Anexa 1. Încărcare date

```
#########################################################################
director_antrenare = r"C:\Users\Ruxi\Desktop\Boli_Frunze\antrenare"
director_validare = r"C:\Users\Ruxi\Desktop\Boli_Frunze\validare"
director_testare = r"C:\Users\Ruxi\Desktop\Boli_Frunze\testare"

date_antrenare = ImageDataGenerator(rescale=1. / 255)
date_validare = ImageDataGenerator(rescale=1. / 255)
date_testare = ImageDataGenerator(rescale=1. / 255)

dimensiune_batch = 8
generator_antrenare = date_antrenare.flow_from_directory(director_antrenare,
                                                         batch_size =
dimensiune_batch, class_mode='categorical')
generator_validare = date_validare.flow_from_directory(director_validare,
                                                         batch_size =
dimensiune_batch, class_mode='categorical')
generator_testare = date_testare.flow_from_directory(director_testare,
                                                         batch_size =
dimensiune_batch, class_mode='categorical')


for batch_date, batch_etichete in generator_antrenare:
    print('Dimensiunea lotului de date pentru antrenare, validare și testare: ',
batch_date.shape)
    print('Dimensiunea lotului de etichete pentru antrenare, validare și
testare: ', batch_etichete.shape)
    break
for batch_date, batch_etichete in generator_validare:
    print('Dimensiunea lotului de date în validare: ', batch_date.shape)
    print('Dimensiunea lotului de etichete în validare: ', batch_etichete.shape)
    break
for batch_date, batch_etichete in generator_testare:
    print('Dimensiunea lotului de date în testare: ', batch_date.shape)
    print('Dimensiunea lotului de etichete în testare: ', batch_etichete.shape)
    break
```

Anexa 2. Definire modele

```
#########################################################################
        Modelele au fost implementate utilizând drept ghid instrucțiunile din
biblioteca Keras: https://keras.io/guides/sequential_model/

def definireModelCNN():

    modelCNN = models.Sequential()
    modelCNN.add(layers.Conv2D(32, input_shape=(256, 256, 3), kernel_size=(3,
3),activation='relu'))
    modelCNN.add(layers.MaxPooling2D(pool_size=(2, 2)))

    modelCNN.add(layers.Conv2D(32, kernel_size=(3, 3), activation='relu'))
    modelCNN.add(layers.MaxPooling2D(pool_size=(2, 2)))

    modelCNN.add(layers.Conv2D(64, kernel_size=(3, 3), activation='relu'))
    modelCNN.add(layers.MaxPooling2D(pool_size=(2, 2)))

    modelCNN.add(layers.Conv2D(64, kernel_size=(3, 3), activation='relu'))
    modelCNN.add(layers.MaxPooling2D(pool_size=(2, 2)))
```

```
    modelCNN.add(layers.Flatten())
    modelCNN.add(layers.Dense(128, activation="relu"))
    modelCNN.add(layers.Dense(4, activation='softmax'))

    modelCNN.compile(loss='categorical_crossentropy',
optimizer=Adam(learning_rate=1e-4), metrics=['accuracy'])
    modelCNN.summary()
    return modelCNN

def definireModelCNNImbunatatit():

    modelCNN = models.Sequential()

    modelCNN.add(layers.Conv2D(32, input_shape=(256, 256, 3), kernel_size=(3,
3),activation='relu'))
    modelCNN.add(layers.MaxPooling2D(pool_size=(2, 2)))

    modelCNN.add(layers.Conv2D(32, kernel_size=(3, 3), activation='relu'))
    modelCNN.add(layers.MaxPooling2D(pool_size=(2, 2)))

    modelCNN.add(layers.Conv2D(64, kernel_size=(3, 3), activation='relu'))
    modelCNN.add(layers.MaxPooling2D(pool_size=(2, 2)))

    modelCNN.add(layers.Conv2D(64, kernel_size=(3, 3), activation='relu'))
    modelCNN.add(layers.MaxPooling2D(pool_size=(2, 2)))

    modelCNN.add(layers.Conv2D(128, kernel_size=(3, 3), activation='relu'))
    modelCNN.add(layers.MaxPooling2D(pool_size=(2, 2)))

    modelCNN.add(layers.Conv2D(128, kernel_size=(3, 3), activation='relu'))
    modelCNN.add(layers.MaxPooling2D(pool_size=(2, 2)))

    modelCNN.add(layers.Flatten())
    modelCNN.add(Dropout(0.2))
    modelCNN.add(layers.Dense(128, activation="relu"))
    modelCNN.add(Dropout(0.2))
    modelCNN.add(layers.Dense(4, activation='softmax'))

    modelCNN.compile(loss='categorical_crossentropy',
optimizer=Adam(learning_rate=1e-4), metrics=['accuracy'])
    modelCNN.summary()
    return modelCNN

#######################################################################
        Modelul VGG16 a fost implementat utilizând drept ghid instrucțiunile din
biblioteca Keras: https://keras.io/api/applications/vgg/#vgg16-function și
https://keras.io/api/applications/#usage-examples-for-image-classification-
models
def definireModelVGGPreantrenat():

    modelPreantrenat = VGG16(weights='imagenet', include_top=False,
input_shape=(256, 256, 3))
    modelPreantrenat.summary()

    for layer in modelPreantrenat.layers:
        layer.trainable = False

    modelVGG = models.Sequential()
    modelVGG.add(modelPreantrenat)

    modelVGG.add(Flatten())
    modelVGG.add(Dense(512, activation="relu"))
    modelVGG.add(Dense(4, activation="softmax"))
```

```
    modelVGG.compile(loss='categorical_crossentropy',
optimizer=Adam(learning_rate=1e-4), metrics=['accuracy'])
    modelVGG.summary()
    return modelVGG
```

Anexa 3. Antrenare

```
#########################################################################
model = definireModelCNN()
# model = definireModelCNNImbunatatit()
# model = definireModelVGGPreantrenat()

epoci = 50
early_stop = EarlyStopping(monitor='val_accuracy', patience=3, verbose=1,
restore_best_weights=True)
istoric = model.fit(generator_antrenare, validation_data=generator_validare,
epochs=epoci, callbacks=[early_stop])
```

Anexa 4. Vizualizare performanțe în antrenare

```
#########################################################################
        Funcția de vizualizare a graficelor de antrenare a fost implementată
utilizând drept ghid laboratorul materiei MLF
def vizualizarePerformanteAntrenare(istoric):

    acc = istoric.history['accuracy']
    val_acc = istoric.history['val_accuracy']

    loss = istoric.history['loss']
    val_loss = istoric.history['val_loss']

    epochs = range(1, len(acc) + 1)
    plt.figure(figsize=(15,5))
    plt.subplot(121)
    plt.title('Acuratețea pe antrenare și validare')
    plt.plot(epochs, acc, 'g', label = 'Acuratețea pe antrenare')
    plt.plot(epochs, val_acc, 'r', label = 'Acuratețea pe validare')
    plt.legend()

    plt.subplot(122)
    plt.title('Loss pe antrenare și validare')
    plt.plot(epochs, loss, 'g', label = 'Loss pe antrenare')
    plt.plot(epochs, val_loss, 'r', label = 'Loss pe validare')
    plt.legend()

    return
vizualizarePerformanteAntrenare(istoric)
```

Anexa 5. Testare - Evaluare performanțe utilizând diferite metrici

```
#########################################################################
        Metricile au fost implementate utilizând biblioteca sklearn :
https://scikit-learn.org/stable/api/sklearn.metrics.html

etichete_adevarate = []
for i in range(len(generator_testare)):

    imagine, eticheta = generator_testare.next()
    etichete_adevarate.append(np.argmax(eticheta, axis=1))

clase_concatenate = np.concatenate(etichete_adevarate)
```

```
clase_adevarate = np.array(clase_concatenate)

predictii = model.predict(generator_testare)
clase_prezise = np.argmax(predictii, axis=1)

matrice_confuzie = confusion_matrix(clase_adevarate, clase_prezise)

print("Matricea de confuzie:")
print(matrice_confuzie)

afișare_matrice = ConfusionMatrixDisplay(
confusion_matrix=matrice_confuzie,
display_labels=["BlackMeasles", "BlackRot", "HealthyLeaf", "LeafBlight"])

afișare_matrice.plot(cmap=plt.cm.Blues)

plt.gcf().set_size_inches(7, 4.8)
plt.xlabel('Etichete prezise')
plt.ylabel('Etichete adevărate')

raport_clasificare = classification_report(clase_adevarate, clase_prezise)

print("Raport de clasificare:")
print(raport_clasificare)

loss, acuratete = model.evaluate(generator_testare, verbose=1)
print("Acuratetea:", "{:.2f}".format(acuratete * 100))
plt.show()
```