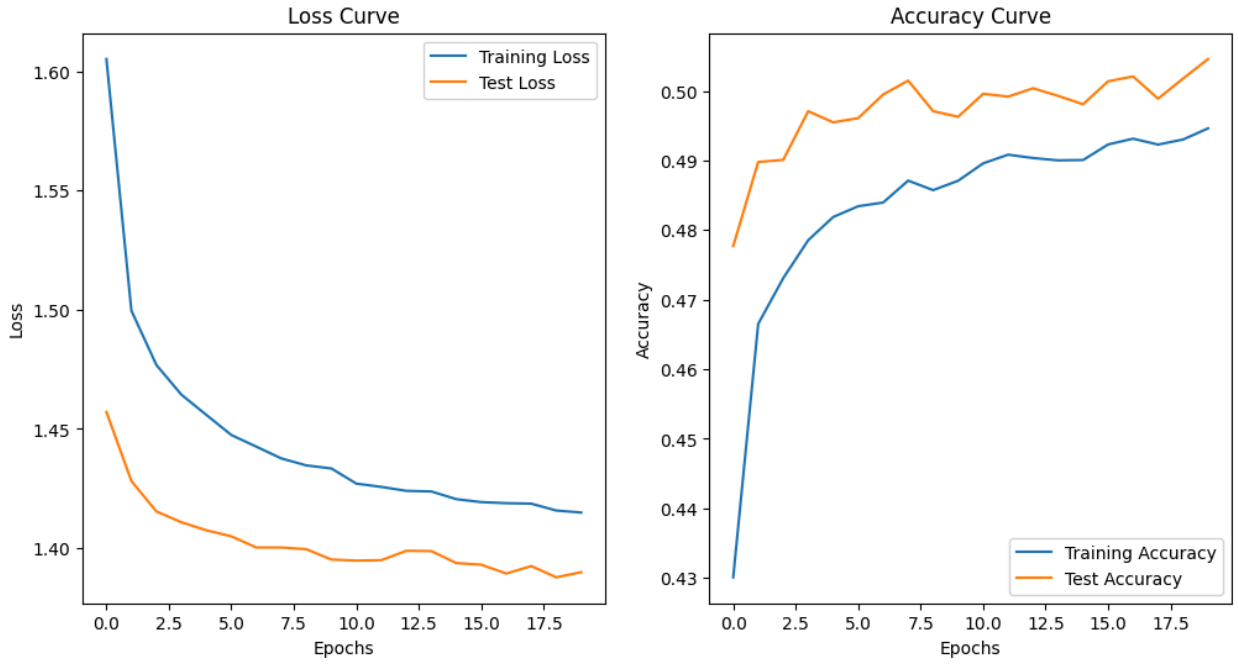
Raport Etapa 2

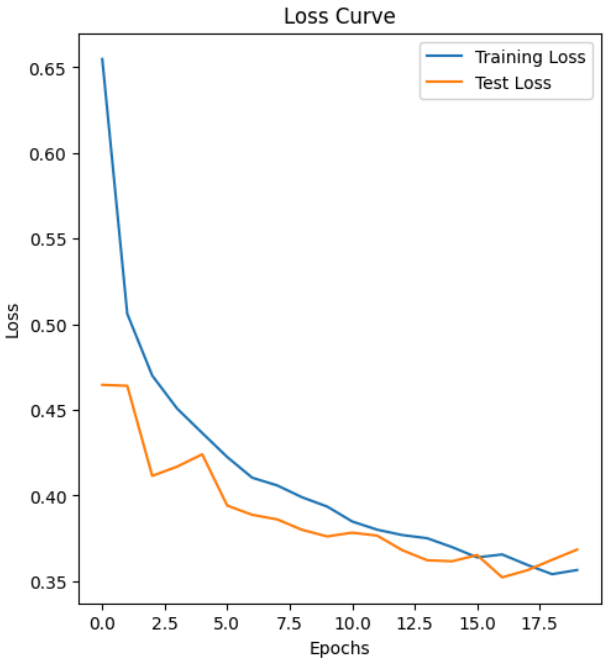
FASHION

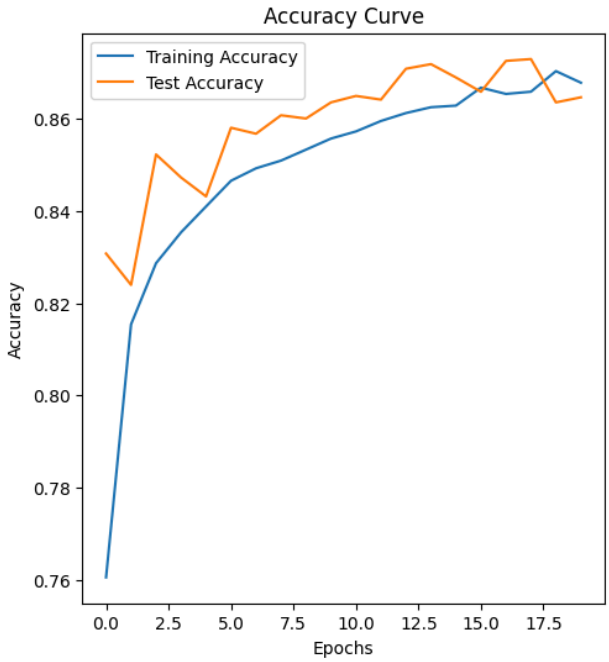
Algoritm	Precision	Recall	F1-Score	Accuracy	Training time (s)
MLP Feature	0.4895	0.5046	0.4883	0.5046	80.89
MLP peste imagini	0.8683	0.8658	0.8656	0.8658	125.51
DeepConvNet Augmentation	0.8513	0.8505	0.8461	0.8505	599.07
DeepConvNet No Augmentation	0.9113	0.9110	0.9089	0.9110	193.69
ResNet-18	0.7181	0.7192	0.7157	0.7192	1228.05

Graficele MLP pe atribute

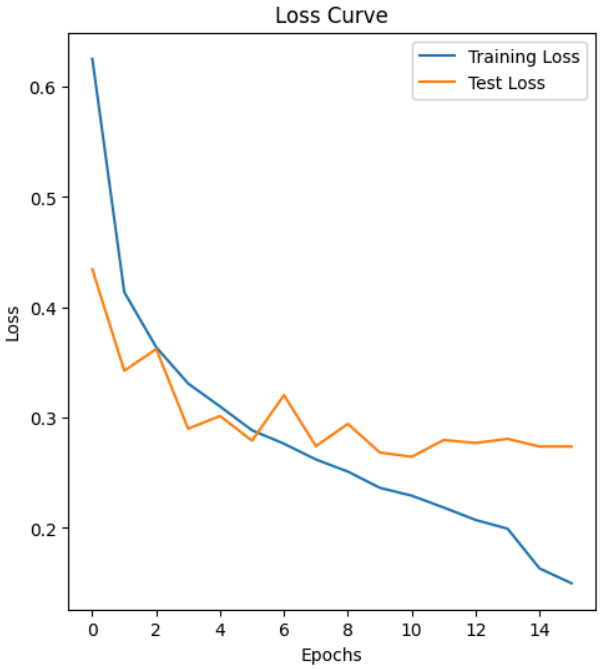


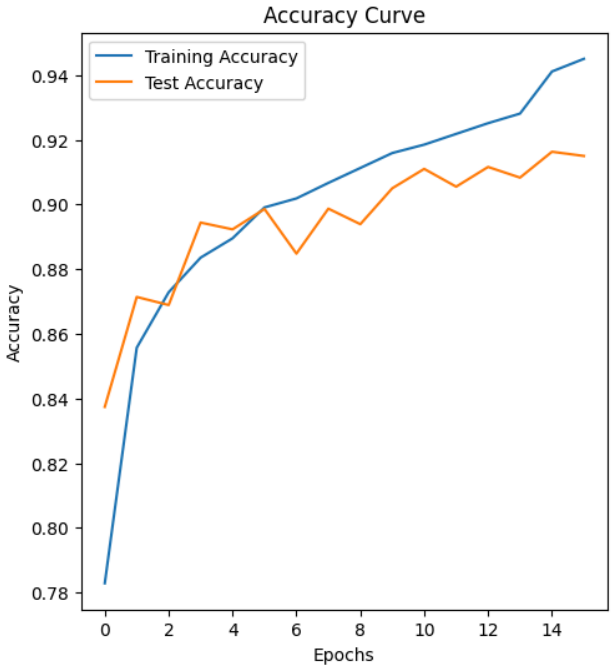
Graficele MLP direct peste imagini



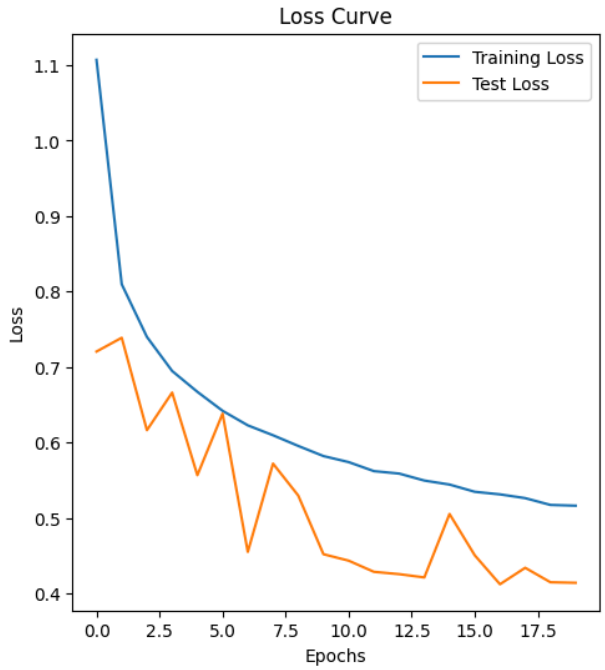


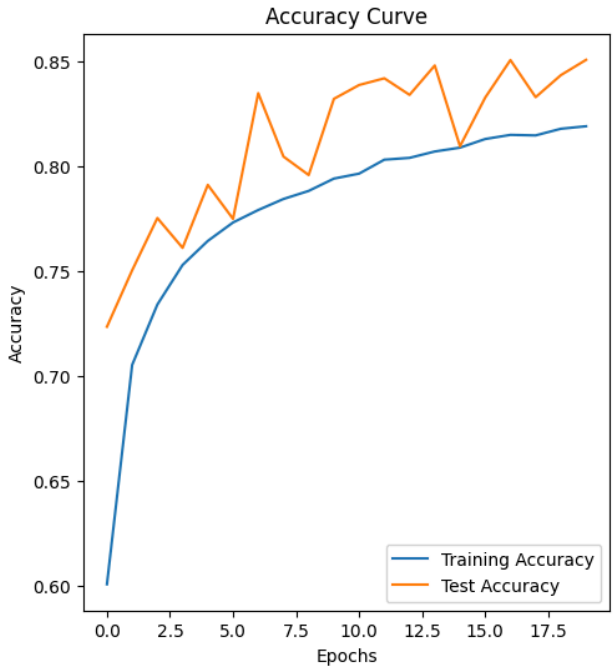
Graficele CNN- No augmentation



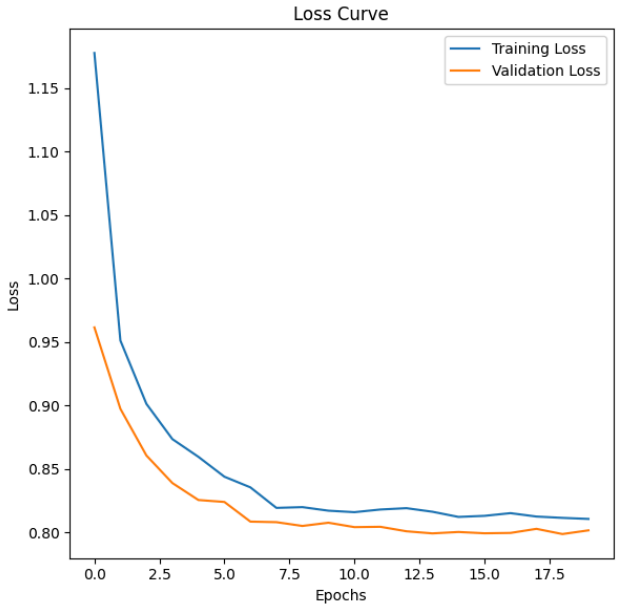


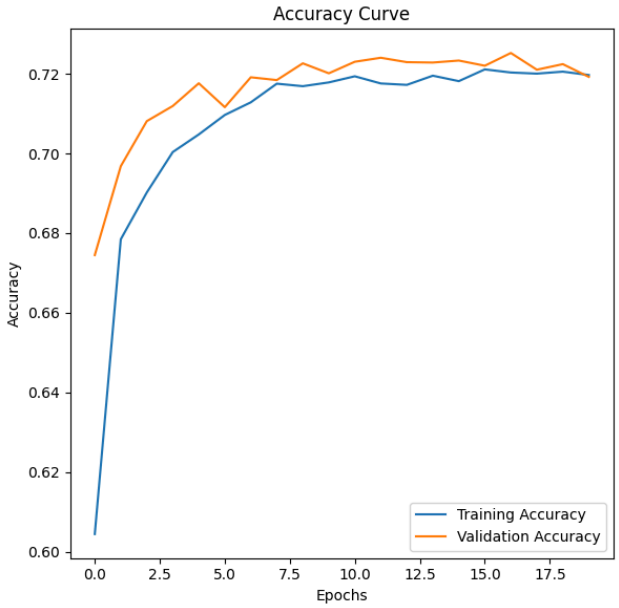
Graficele CNN - Augmentation





Graficele Resnet

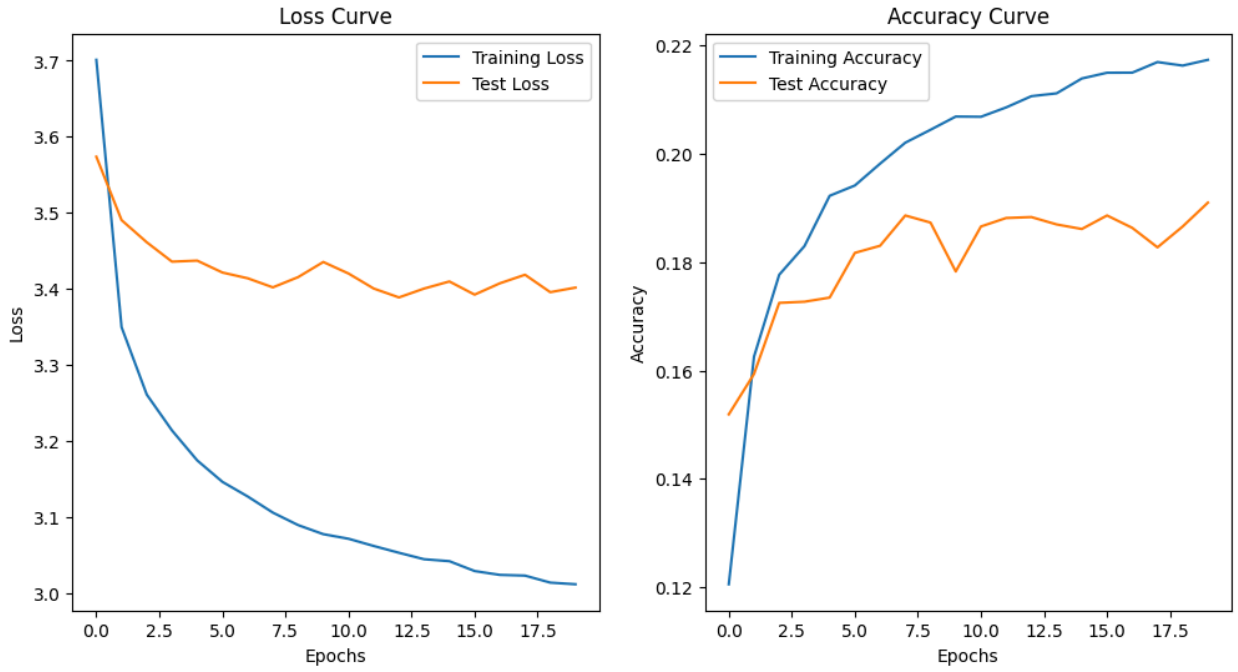




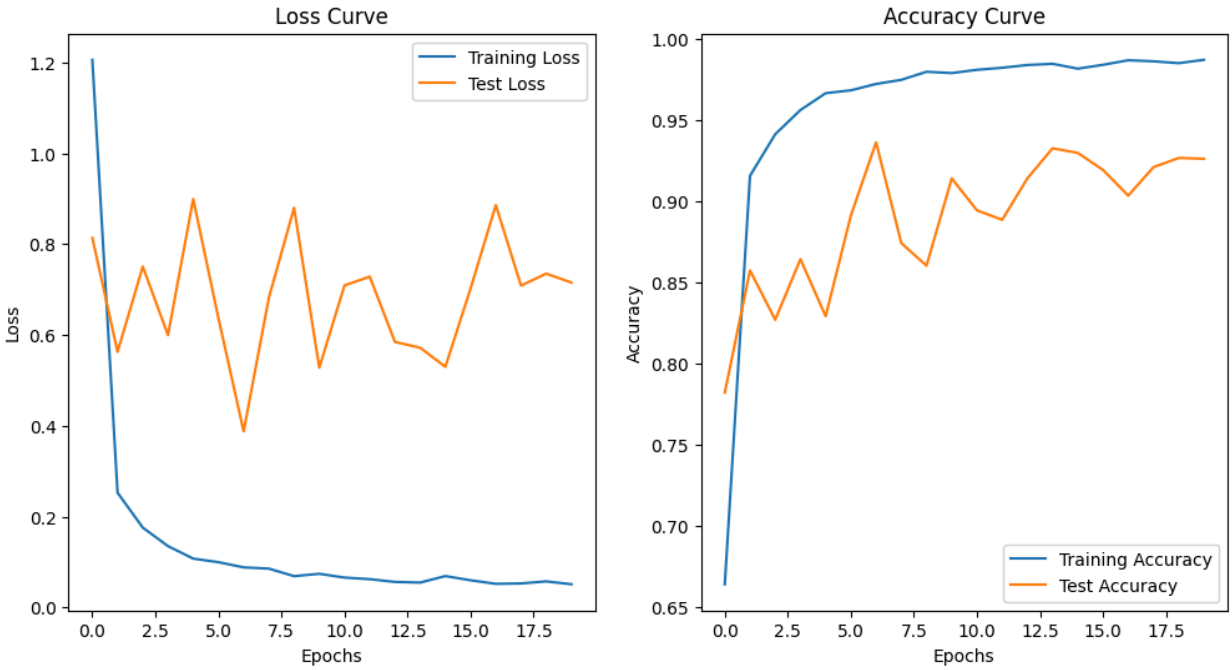
FRUITS

Algoritm	Precision	Recall	F1-Score	Accuracy	Training time
MLP Feature	0.1860	0.1910	0.1680	0.1910	125.80
MLP peste imagini	0.9372	0.9258	0.9236	0.9258	220.95
DeepConvNet Augmentation	0.9232	0.8992	0.8942	0.8992	1096.21
DeepConvNet No Augmentation	0.9807	0.9770	0.9768	0.9770	296.92
ResNet-18	0.8188	0.8135	0.8077	0.8135	1149.90

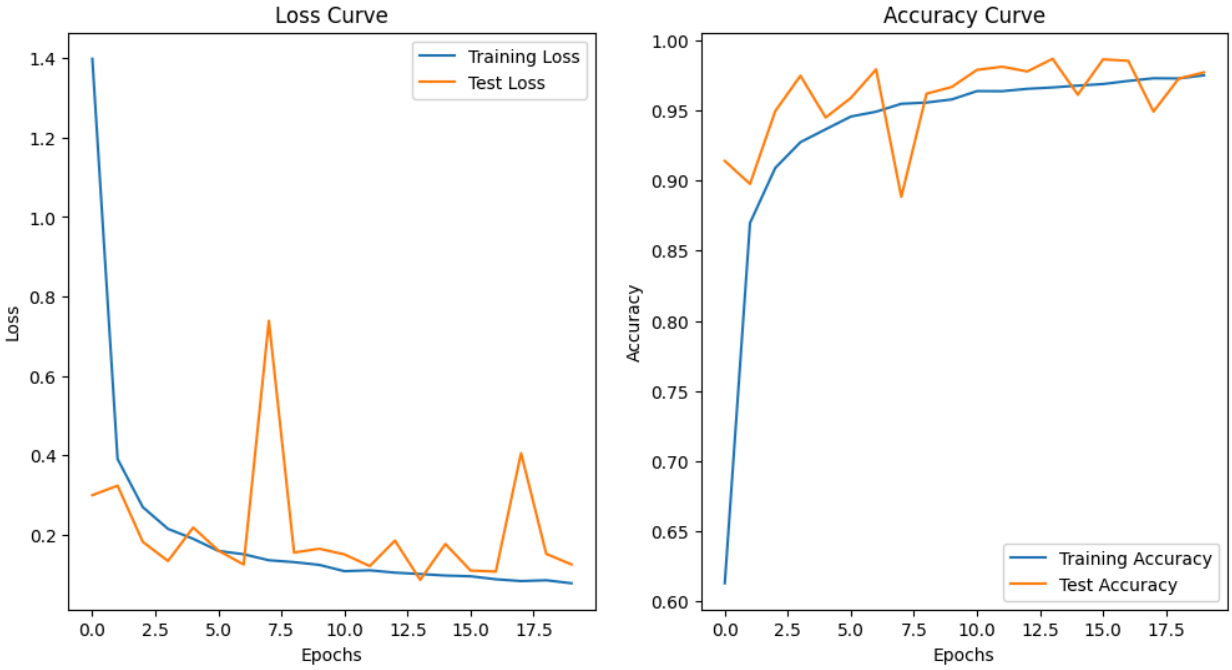
Graficele MLP pe atribute



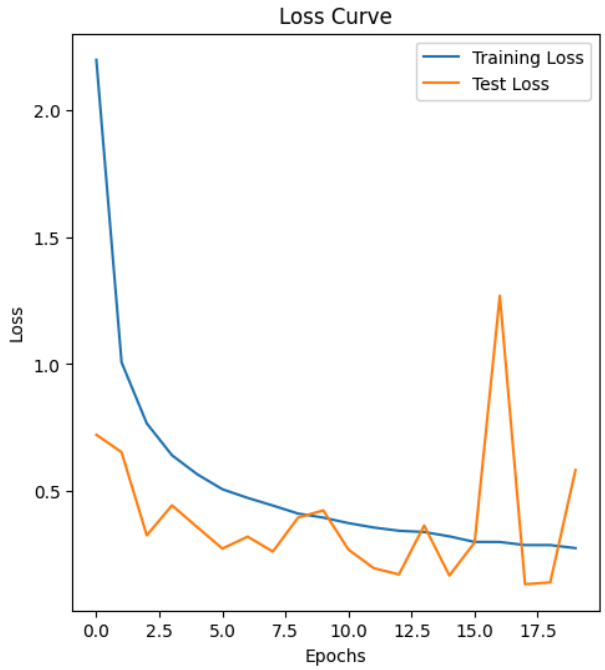
Graficele MLP direct peste imagini

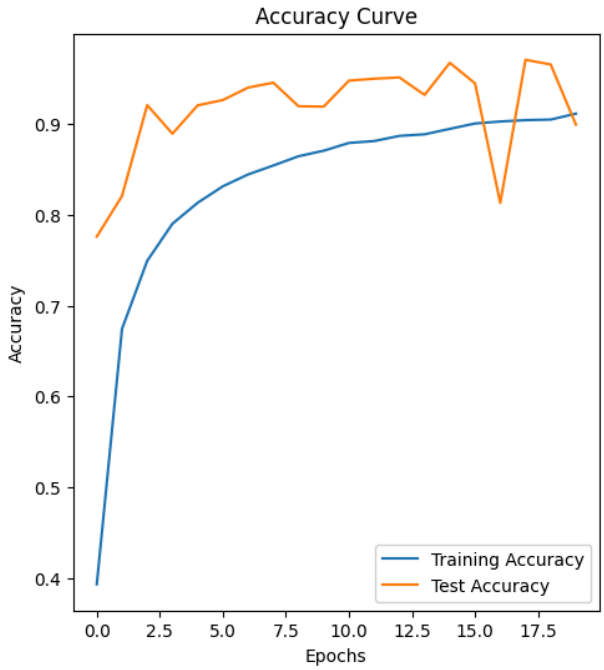


Graficele CNN- No augmentation

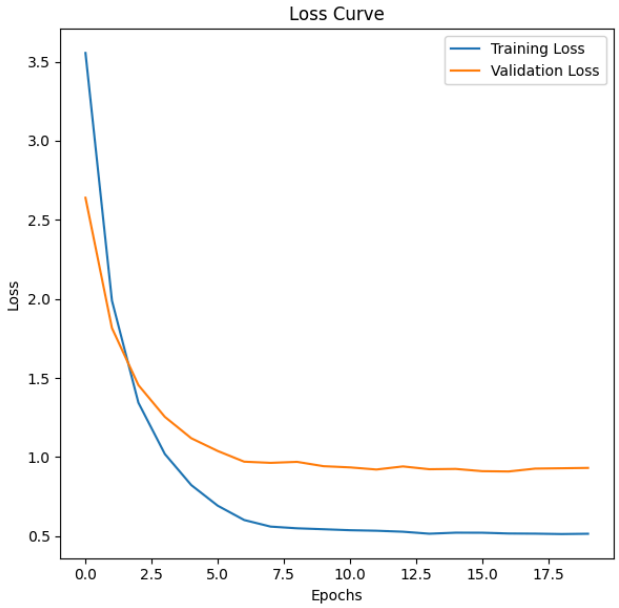


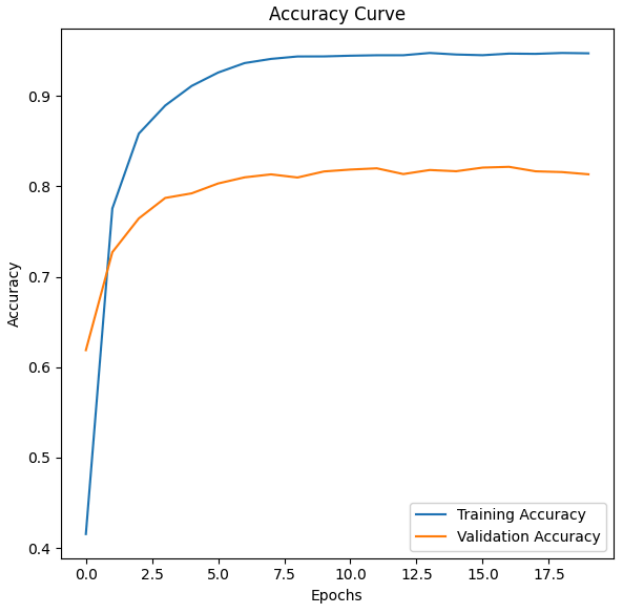
Graficele CNN - Augmentation





Graficele Resnet





Arhitecturile folosite

1. MLP pe atributele extrase in etapa 1 &
2. Arhitectura de tip MLP direct peste imagini

Arhitectura retelei

Straturi si neuroni

Stratul de intrare: 128 neuroni cu functie de activare **ReLU**.

!!!! In cazul MLP direct peste imagini pentru **fruits**, stratul de intrare va avea 512 neuroni, urmand ca urmatorele straturi sa aiba 256, 128 si tot asa.

Straturi ascunse:

Primul strat ascuns: 128 neuroni cu functie de activare **ReLU**.

Al doilea strat ascuns: 64 neuroni cu functie de activare **ReLU**.

Stratul de iesire: neuroni cate clase sunt in total cu activare **Softmax**.

Functiile de activare:

ReLU pentru straturile ascunse.

Softmax pentru stratul de iesire (pentru clasificare multi-clasa).

Caracteristicile Procesului de Antrenare:

Epoci: 20

Funcția de eroare: **Cross-entropy** pentru clasificare multi-clasa.

Dimensiunea batch-ului: 32 de exemple per batch

Optimizator: **Adam**

3. Arhitectura de tip convoluțional

Arhitectura rețelei:

- compusă din 3 straturi convoluționale, urmate de un strat de **Global Average Pooling** și două straturi dense:

1. Conv1:

- 32 de filtre (kernels) cu dimensiunea de 3x3.
- Funcția de activare **ReLU**.
- Normalizare a lotului (**BatchNormalization**).
- Pooling maxim (**MaxPooling2D**) cu fereastra de 2x2.

2. Conv2:

- 64 de filtre cu dimensiunea de 3x3.
- Funcția de activare **ReLU**.
- Normalizare a lotului.
- Pooling maxim cu fereastra de 2x2.

3. Conv3:

- 128 de filtre cu dimensiunea de 3x3.
- Funcția de activare **ReLU**.
- Normalizare a lotului.
- Pooling maxim cu fereastra de 2x2.

4. Global Average Pooling: Acesta este folosit pentru a reduce dimensiunile caracteristicilor extrase, oferind o reprezentare compactă a imaginii.

5. Fully Connected Layers:

- Stratul Dense cu 128 de neuroni și activare ReLU.
- Stratul Dropout cu 70% (pentru a preveni overfitting-ul).
- Stratul final Dense cu numărul de clase, folosind activarea Softmax.

Caracteristicile procesului de antrenare:

1. Numărul de epoci de antrenare și metodele de regularizare alese:

- **Număr de epoci**: 20 epoci.
- **Metode de regularizare**:
    - **Dropout**: Un dropout de 70% în straturile dense pentru a preveni overfitting-ul.
    - **Early Stopping**: Oprirea antrenamentului atunci când pierderea de validare nu se îmbunătățește după 5 epoci consecutive.
    - **ReduceLROnPlateau**: Reducerea ratei de învățare cu un factor de 0.5 dacă pierderea de validare nu se îmbunătățește după 3 epoci.

2. Funcția de eroare utilizată:

- **Funcția de eroare**: **sparse_categorical_crossentropy**, care este utilizată pentru problemele de clasificare multiclasă unde etichetele sunt întregi (fără codificare One-Hot).

3. Dimensiunea batch-ului:

- Dimensiunea batch-ului folosită în procesul de antrenament este **32**, ceea ce înseamnă că rețeaua procesează 32 de imagini la fiecare pas de antrenament.

4. Optimizatorul folosit:

- Se folosește Adam cu un learning rate de 0.001 pentru optimizarea greutăților rețelei**.**
- **Adam**: Optimizatorul **Adam** este utilizat datorită proprietății sale de a adapta rata de învățare pe baza gradientului și a momentului, ceea ce face acest optimizator mai eficient decât optimizatorii tradiționali, cum ar fi SGD. Adam este adesea folosit datorită rapidității și a stabilității sale în procesele de antrenare.

Analiza datelor

FASHION

1. Abordarea MLP pe Feature-uri

Acest model, deși prezintă o precizie și recall rezonabile, performează slab în ceea ce privește acuratețea și F1-Score. Acuratețea scăzută (~50%) sugerează că, deși rețeaua identifică unele trăsături relevante, nu reușește să diferențieze corect toate clasele. Timpul de antrenare este relativ scurt, ceea ce indică un compromis între viteza de antrenare și acuratețea obținută.

2. Abordarea MLP Direct pe Imagini

Această abordare performează mult mai bine decât prima, având o precizie, recall și F1-Score mult mai mari. Acest model se generalizează mai bine, iar precizia și recall-ul sunt apropiate, ceea ce sugerează că modelul nu favorizează falsurile pozitive sau negative. Deși timpul de antrenare este mai mare, acest model aduce o îmbunătățire considerabilă a performanței.

3. DeepConvNet cu Augmentare

Acest model prezintă o performanță excelentă în termeni de precizie, recall și F1-Score. Totuși, timpul de antrenare este mult mai lung comparativ cu modelele anterioare. Augmentarea datelor ajută la îmbunătățirea generalizării modelului, dar costul acestei îmbunătățiri este un timp semnificativ mai mare de antrenare.

4. DeepConvNet Fără Augmentare

Acest model fără augmentare obține cele mai bune performanțe în termeni de precizie, recall, F1-Score și acuratețe. Timpul de antrenare este mai gestionabil comparativ cu modelul cu augmentare. Lipsa augmentării nu a afectat semnificativ generalizarea modelului, făcând această abordare extrem de eficientă.

5. ResNet-18

Deși ResNet-18 este un model puternic și complex, performanța sa nu depășește modelul DeepConvNet fără augmentare. Precizia, recall-ul și F1-Score-ul sunt mai mici, iar timpul de antrenare este semnificativ mai lung. Acest lucru sugerează că, pentru acest set de date, modelele mai simple, precum DeepConvNet, pot fi mai eficiente din punct de vedere al performanței și al timpului de antrenare.

Grafice

Curbele de Pierdere și Acuratețe în Antrenare:

Modelele **MLP** (bazate pe feature-uri și direct pe imagini) au o scădere rapidă a pierderii în primele epoci, dar se blochează la o acuratețe mai scăzută comparativ cu modelele **DeepConvNet** și **ResNet**.

Modelele **DeepConvNet** și **ResNet** prezintă curbe mai line de scădere a pierderii și o îmbunătățire mai constantă a acurateței. Augmentarea ajută la obținerea unor performanțe bune, dar cu un cost semnificativ al timpului de antrenare.

Performanța și Viteza de Antrenare:

Modelele **DeepConvNet fără augmentare** și **ResNet** au performanțe excelente, dar modelul **DeepConvNet fără augmentare** se dovedește a fi mai eficient din punct de vedere al timpului de antrenare.

DeepConvNet cu augmentare obține cele mai bune rezultate în ceea ce privește generalizarea, dar prețul plătit este un timp de antrenare mult mai mare.

MLP-urile, în ciuda unui timp de antrenare mai scurt, nu reușesc să obțină performanțele necesare pentru o clasificare precisă a imaginii.

Concluzii

DeepConvNet fără augmentare a fost abordarea optimă, oferind un echilibru excelent între performanță și eficiență. A obținut cele mai bune rezultate în ceea ce privește precizia, recall-ul, F1-Score-ul și acuratețea, având un timp de antrenare moderat.

ResNet-18, deși un model puternic, nu a depășit DeepConvNet, având o performanță similară, dar un timp de antrenare mult mai mare.

Modelele **MLP** au performat slab în comparație cu modelele **DeepConvNet** și **ResNet**, iar **DeepConvNet cu augmentare**, deși a arătat rezultate bune, a avut un cost semnificativ în ceea ce privește timpul de antrenare.

FRUITS

1. Abordarea MLP pe Feature-uri

- **Performanță**: Modelul MLP pe feature-uri prezintă o performanță relativ slabă în ceea ce privește **precizia**, **recall-ul** și **F1-Score-ul** (toate scorurile sunt sub 0.2). Acest lucru sugerează că, deși rețeaua reușește să identifice unele trăsături ale imaginilor, nu reușește să diferențieze eficient clasele. În plus, **acuratețea** scăzută (aproximativ 0.19) subliniază faptul că acest model nu reușește să obțină rezultate corecte pe majoritatea imaginilor din test.
- **Timp de antrenare**: Timpul de antrenare este de **125.80 secunde**, ceea ce îl face relativ rapid, dar acest compromis se face în fața unei performanțe scăzute.

Interpretare grafică:

- **Curba de pierdere** (loss curve) scade rapid la început, însă apoi se stabilizează la o valoare mai mare, ceea ce sugerează o învățare limitată.
- **Curba de acuratețe** (accuracy curve) arată o îmbunătățire foarte lentă și stagnează la valori foarte mici (~0.18), confirmând performanța slabă a modelului.

2. Abordarea MLP Direct pe Imagini

- **Performanță**: Această abordare performează mult mai bine decât MLP pe feature-uri, având un **F1-Score** și o **precizie** de aproximativ **0.92**, ceea ce indică o îmbunătățire semnificativă. Aceasta se generalizează mult mai bine la setul de date și oferă o **recall** și **precizie** mult mai apropiate, ceea ce sugerează că modelul nu favorizează falsurile pozitive sau negative.
- **Timp de antrenare**: Timpul de antrenare este **220.95 secunde**, mai mare decât în cazul MLP pe feature-uri, dar aduce o îmbunătățire semnificativă a performanței.

Interpretare grafică:

- Curbele de pierdere și acuratețe arată o scădere constantă a pierderii, iar acuratețea crește rapid și se stabilizează la valori mari, aproape de 0.90. Acest lucru sugerează o învățare eficientă, cu o performanță îmbunătățită pe parcursul antrenamentului.

3. DeepConvNet cu Augmentare

- **Performanță**: Modelul **DeepConvNet cu augmentare** prezintă performanțe excelente cu **precizie** și **recall** de **0.92** și un **F1-Score** de **0.89**. Aceste valori indică o generalizare bună a modelului, dar augmentarea adăugată îmbunătățește performanța la costul unui **timp de antrenare mult mai lung**.
- **Timp de antrenare**: Antrenarea durează **1096.21 secunde**, un timp considerabil mai mare decât în cazul altor modele. Acesta este prețul plătit pentru îmbunătățirea performanței prin augmentare.

Interpretare grafică:

- **Curbele de pierdere și acuratețe** arată o scădere rapidă a pierderii și o creștere a acurateței, dar cu unele oscilații pe parcurs, indicând faptul că augmentarea ajută la îmbunătățirea generalizării, dar cu unele instabilități.

4. DeepConvNet Fără Augmentare

- **Performanță**: **DeepConvNet fără augmentare** este cel mai performant model din punct de vedere al **preciziei** (0.98), **recall-ului** (0.98), **F1-Score-ului** (0.97) și **acurateței** (0.98). Aceste scoruri indică o performanță extrem de bună și o generalizare excelentă, fără a fi nevoie de augmentare.
- **Timp de antrenare**: Timpul de antrenare este de **296.92 secunde**, mult mai eficient comparativ cu modelul cu augmentare.

Interpretare grafică:

- **Curbele de pierdere și acuratețe** sunt extrem de stabile, iar modelul scade pierderea rapid și își îmbunătățește constant acuratețea, confirmând performanța ridicată a acestui model.

5. ResNet-18

- **Performanță**: Deși **ResNet-18** este un model puternic, performanțele sale nu sunt la fel de bune ca cele ale modelului **DeepConvNet fără augmentare**. **Precizia** (0.82) și **F1-Score-ul** (0.81) sunt mai mici decât ale DeepConvNet fără augmentare, iar **timpul de antrenare** (1149.90 secunde) este mult mai mare.
- **Timp de antrenare**: ResNet-18 este mult mai complex, ceea ce explică timpul mai lung de antrenare comparativ cu modelele DeepConvNet.

Interpretare grafică:

- **Curbele de pierdere și acuratețe** sunt mult mai line decât în cazul DeepConvNet, indicând o convergență stabilă, dar cu performanțe mai modeste comparativ cu modelele anterioare.

Concluzii

1. **DeepConvNet fără augmentare** este abordarea optimă pentru acest set de date, oferind performanțe excelente în termeni de **precizie**, **recall**, **F1-Score** și **acuratețe**, cu un **timp de antrenare relativ scurt** comparativ cu modelele cu augmentare.
2. **ResNet-18**, deși puternic, nu aduce un avantaj semnificativ față de **DeepConvNet fără augmentare** și suferă de un **timp de antrenare mult mai mare**.
3. **DeepConvNet cu augmentare** oferă cele mai bune rezultate în generalizare, dar costul este un **timp de antrenare semnificativ mai mare**.
4. Modelele **MLP** (atât pe feature-uri, cât și direct pe imagini) au performanțe mult mai slabe, cu o **acuratețe scăzută** și **timp de antrenare scurt**, dar nu reușesc să obțină performanțele necesare pentru clasificarea corectă a imaginilor.