Tema Învățare Automată

Partea 2

Dată publicare: 03/12/2024	**Dată limită rezolvare: 10/01/2025**

1. Descriere generală

Clasificarea imaginilor este problema care a împins cel mai mult dezvoltările inițiale din domeniul rețelelor neurale convoluționale (prin competiția pe setul de date ImageNet 1k).

Dacă prima parte a temei a cerut “ingineria de mână” a unor atribute ce descriu imaginea și folosirea lor ca intrări pentru algoritmi clasici de Învățare Automată, partea a doua cere proiectarea unor arhitecturi de rețele neurale care să efectueze automat atât extragerea de atribute, cât și utilizarea lor în clasificare.

2. Descrierea Seturilor de Date

Seturile de date rămân cele din prima etapă. Reamintim caracteristicile lor:

- Fashion-MNIST - un set de date cu 70000 de imagini grayscale de tip thumbnail (32 x 32) reprezentând 10 tipuri de elemente de vestimentație (e.g. bluze, cizme, pantaloni)
- Fruits-360 - un set de date cu ~55000 de imagini RGB cu 80 de tipuri diferite de fructe singulare

Seturile de date au deja împărțire în date de antrenare (train) și de testare (test).
Obiectivul în fiecare set de date este cel de clasificare a imaginii în categoria corectă.

3. Cerințe

3.1. MLP pe atributele extrase în etapa 1 [2p]

Definiți o arhitectură de tip **Multi-Layered Perceptron**, pe care o antrenați peste vectori de atribute obținuți în urma extragerii de atribute efectuată în prima etapă.

Numărul de atribute extrase depinde de setul de date pe care se aplică (FashionMNIST sau Fruits 360). Pentru scopul uniformității între evaluarile voastre, aplicați procedurile de feature selection (e.g. SelectPercentile, VarianceThreshold, folosirea metodei de Feature Importance Evaluation using Random Forest) astfel încât să ajungeți la:

- Maxim 64 de atribute pentru FashionMNIST
- Maxim 128 de atribute pentru Fruits 360

Definiți și documentați:
- Arhitectura rețelei:
    - Numărul de straturi și numărul de neuroni de pe fiecare strat
    - Funcțiile de activare utilizate
- Caracteristicile procesului de antrenare:
    - Numărul de epoci de antrenare și metodele de regularizare alese (dacă sunt folosite)
    - Funcția de eroare utilizată
    - Dimensiunea batch-ului
    - Optimizatorul folosit

Afișați **pe un același grafic**:
- Curbele de **eroare (loss)** pentru setul de antrenare și cel de test
- Curbele de acuratețe pentru setul de antrenare și cel de test

Comentați rezultatul obținut de algoritmul MLP în comparație cu rezultatele algoritmilor utilizați în etapa anterioară.
Notă! Aplicați procesul de normalizare a vectorilor de intrare înainte de utilizarea lor în antrenarea MLP-ului.

3.2. Arhitectura de tip MLP direct peste imagini [2p]

Definiți o arhitectură de tip **Multi-Layered Perceptron**, pe care o antrenați peste vectori obținuți din **liniarizarea** imaginii.

Definiți și documentați:
- Arhitectura rețelei:
    - Numărul de straturi și numărul de neuroni de pe fiecare strat
    - Funcțiile de activare utilizate
- Caracteristicile procesului de antrenare:
    - Numărul de epoci de antrenare și metodele de regularizare alese (dacă sunt folosite)
    - Funcția de eroare utilizată
    - Dimensiunea batch-ului
    - Optimizatorul folosit

Afișați **pe un același grafic**:
- Curbele de **eroare (loss)** pentru setul de antrenare și cel de test
- Curbele de acuratețe pentru setul de antrenare și cel de test

Notă: normalizați intrările înainte de propagarea prin MLP.

3.3. Arhitectura de tip convoluțional [3p]

Definiți o arhitectură de rețea convoluțională care să fie similară în procesul de construcție cu modelul **DeepConvNet** din laboratorul de introducere în rețele convoluționale.

Rețeaua proiectată trebuie să cuprindă următoarele tipuri de straturi:

- Convoluțional
- Global Pooling
- Fully Connected (Liniar) - în stratul(urile) de final

Folosirea augmentărilor:

În cadrul procesului de antrenare se impune un punct de variabilitate dat de folosirea sau nu a augmentărilor de date.
Pentru arhitectura de tip convoluțional se vor considera două antrenări separate: una **fără augmentări de date**, una **cu augmentări de date.**

În cadrul augmentărilor se vor folosi următoarele proceduri:

- Random Horizontal Flip
- Rotation
- Random Crop and Rescaling

Definiți și documentați:

- Arhitectura rețelei:
- Caracteristicile procesului de antrenare:
    - Numărul de epoci de antrenare și metodele de regularizare alese
    - Funcția de eroare utilizată
    - Dimensiunea batch-ului
    - Optimizatorul folosit

Afișați **pe un același grafic,** odată pentru :

- Curbele de **eroare (loss)** pentru setul de antrenare și cel de test **pentru modelul fără augmentări** și **modelul cu augmentări** (4 curbe în total)
- Curbele de **acuratețe** pentru setul de antrenare și cel de test **pentru modelul fără augmentări** și **modelul cu augmentări** (4 curbe în total)

Note importante:

- Folosiți BatchNormalization în arhitectura voastră
- Normalizați imaginile aducându-le în intervalul [0,1]
- Considerați utilizarea straturilor de tip Dropout în straturile liniare finale, dacă procesul de antrenare vă indică un overfit al modelului
- Asigurați-vă că antrenați rețeaua pentru un număr relevant de epoci (minim 20)

3.4. Utilizarea unei proceduri de finetuning peste arhitectura ResNet-18 [3p]

Folosiți o rețea pre-antrenată pe seturi de date existente (e.g. CIFAR-10) care pot fi adaptate prin procedura de finetuning la sarcinile de clasificare cerute.
Pentru procedura generală de finetuning al unei arhitecturi date (e.g. ResNet-18) urmăriți acest tutorial de la PyTorch.

Note importante:

- Imaginile din seturile de date Fashion-MNIST sunt la rezolutie de 32 x 32, iar cele din Fruits-360 la 100 x 100. Ambele sunt sub dimensiunea standard de 224 x 224 folosită în pre-training pe ImageNet-1k. Ca atare se va folosi un model de ResNet-18 pre-antrenat pe setul de date Cifar10 la rezolutia de 32 x 32. Acest lucru cere ca imaginile din Fruits-360 sa fie rescalate la dimensiunea de 32 x 32 pentru a putea fi propagate prin modelul pre-antrenat.
- Imaginile din Fashion-MNIST sunt în tonuri de gri (un singur canal de intrare), iar cele din CIFAR-10 sunt color. Pentru a putea folosi finetuning este necesară duplicarea canalului de gri din Fashion-MNIST pentru a obține volume similare ca formă cu cele folosite în pre-antrenare.
- Finetuning-ul se desfășoară cu rate de învățare mai mici decât antrenarea de la 0. Folosiți optimizator de tip SGD cu momentum, și un calendar în trepte de reducere a ratei de învățare pentru a avea control asupra acesteia.

Realizați același tip de grafice de ale procesului de antrenare precum în sarcinile anterioare.

4. Raport de evaluare comparativ și predarea temei

Realizați un raport comparativ între rezultatele obținute de fiecare abordare de model neural, și comparați aceste rezultate cu cele obținute de cele mai bune modele realizate în etapa 1.
Faceți comparația în termenii de **average precision, recall** și **F1 score** obținute peste clasele din fiecare set de date, trecând valorile aferente fiecărui algoritm pe câte un rând dintr-un tabel.

Tema va fi încărcată pe Moodle însoțită de un raport sub formă de fișier PDF, care include:

- **Graficele cu curbele de antrenare și test pentru loss și acuratețe. Este obligatorie** prezența în text a **unei interpretări / analize** a diagramelor rezultate.
- **Analiza comparativă** între cele patru abordări (3.1 - 3.4) în termeni de performanță cantitativă și viteză de antrenare. **Este obligatorie** prezența în text a **unei interpretări / analize** a rezultatelor obținute (e.g. ce anume favorizează o abordare față de cealaltă).

Rezultatele temei vor fi prezentate în cadrul laboratoarelor de Învățare Automată, **exclusiv pe baza rapoartelor încărcate.**