Raport Proiect: Clasificarea Tumorilor Cerebrale din Imagini MRI

Cerinta 1: Clasa Personalizata PyTorch pentru Incarcarea Datelor

Am implementat clasa `BrainTumorDataset`, derivata din `torch.utils.data.Dataset`, care incarca datele in mod lazy. Aceasta clasa:

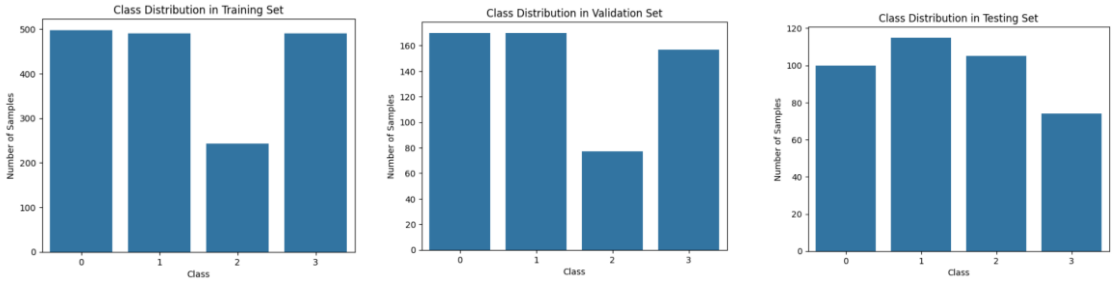
- Primeste lista cu pathurile imaginilor, labelurile, transformarile care trebuie aplicate si tipul setului.
- Are metodele `__len__` si `__getitem__` pentru a returna numarul de imagini si o pereche (imagine, eticheta) la cerere.
- Permite aplicarea de transformari asupra imaginilor inainte de returnare.

Cerinta 2: Impartirea Setului de Date in Antrenare si Validare

Am implementat functia `get_splited_train_data()` care imparte, default, setul de date de antrenare in 80% pentru antrenare si 20% pentru validare, asigurand o distributie echilibrata a imaginilor pe clase.

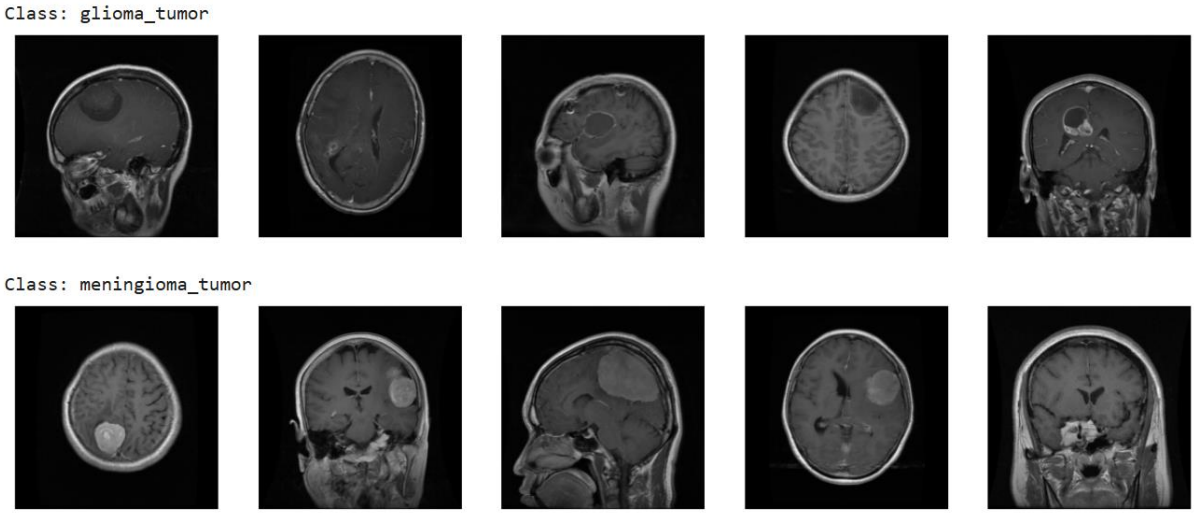
Cerinta 3: Vizualizarea Distributiei Claselor

Am creat o functie pentru vizualizarea distributiei claselor in seturile de antrenare, validare si testare. Graficele arata ca exista clase sub-reprezentate, ceea ce poate influenta performanta modelului. Pentru a rezolva aceasta problema am putea face augmentari pe clasa sub-reprezentata prin rotatii, translatii, inversari pentru a creste numarul de exemple. Am mai putea face o sub-esantionare pe celelalte clase dar datasetul fiind mai restrans nu este un lucru chiar potrivit. Am putea folosii alte modele pentru a genera date pentru acele clase sub-reprezentate.



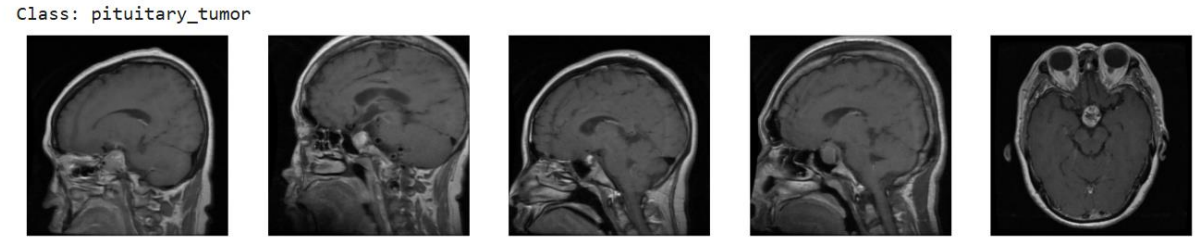
Cerinta 4: Analiza Vizuala a Imaginilor

Am extras si afisat 5 imagini din fiecare categorie, evidentind variabilitatea interna a fiecarei clase si posibilele similitudini intre acestea. Am observat ca :
- tumora glioma este caracterizata printr o pata neagra localizata in mai multe locuri ale creierului.



- tumora meningioma este asemanatoare cu cea glioma, fiind caracterizata tot printr o pata care poate fi localizata in mai multe locuri, dar in acest caz este alba.

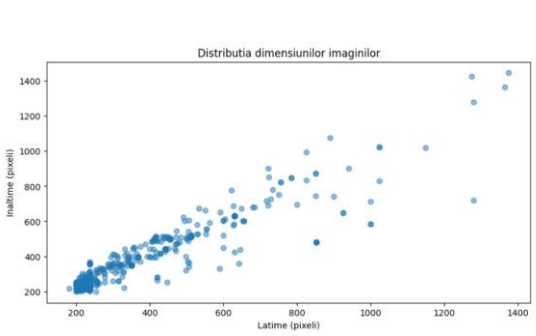
- tumora pituitara se poate distinge printr o pata neagra mica, dar care are o forma specifica si este localizata in centrul creierului

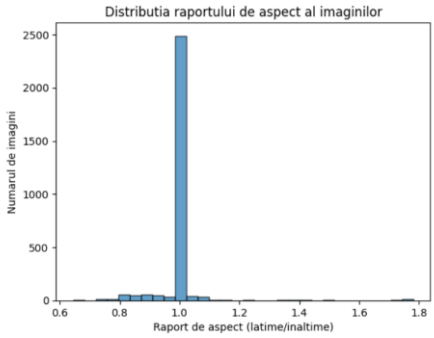


Cerinta 5: Verificarea Consistentei Setului de Date

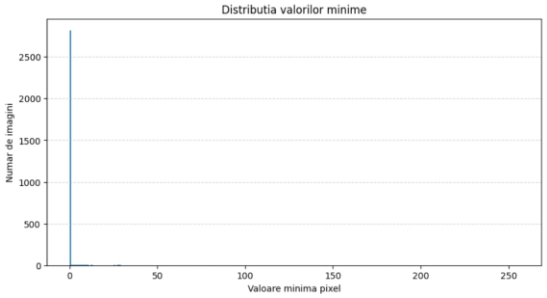
Am realizat verificari pentru:

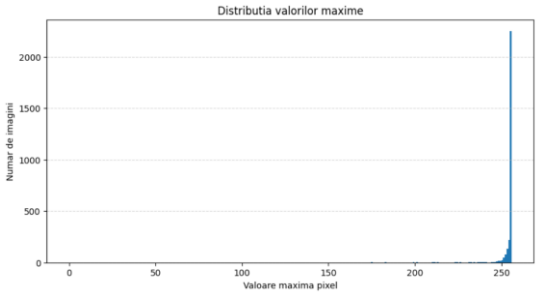
- **Numarul de canale**: Am confirmat ca toate imaginile sunt consistente din acest punct de vedere.
- **Dimensiuni**: Am creat o functie care a aratat ca imaginile au dimensiuni si raportul de aspect care variaza. Acestea vor fi redimensionate la 224x224 cand sunt date ca parametru modelul, deoarece acesta functioneaza doar pe aceasta dimensiune.





- **Normalizarea valorilor pixelilor**: Am verificat valorile pixelilor si am observat ca deja acestea se afla intre 0 si 255.



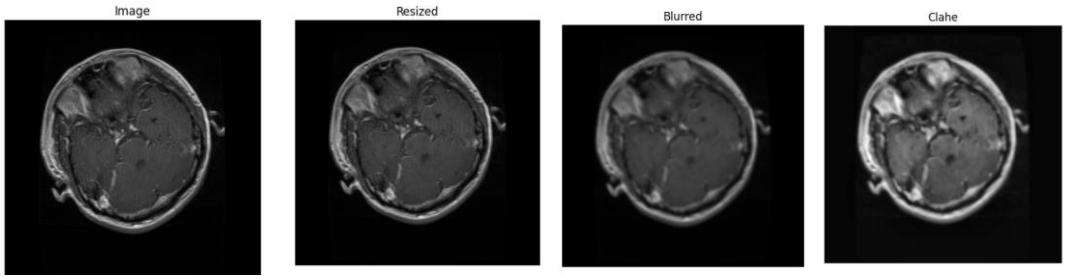


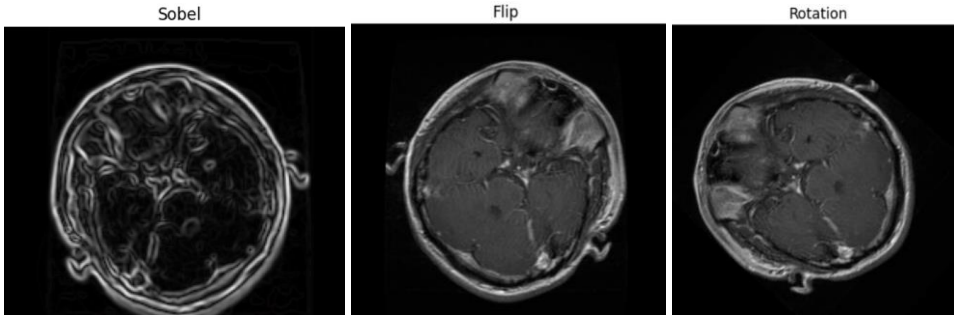
Cerinta 6: Preprocesarea si Normalizarea Imaginilor

Am aplicat si comparat urmatoarele tehnici de preprocesare:

- **Redimensionare (Resize)** – Modelul invata mai rapid si mai eficient cand toate imaginile sunt de aceeasi dimensiune, facilitand procesul de antrenare si optimizarea retelei neuronale.
- **Filtrul Gaussian (Blur) -** Modelul se concentreaza mai mult pe caracteristicile globale ale imaginii, cum ar fi forma si structura generala, in loc sa se piarda in detalii de zgomot irelevante.
- **CLAHE pentru ajustarea contrastului -** Aceasta ajustare imbunatateste vizibilitatea contururilor si a detaliilor fine, ajutand modelul sa identifice mai usor caracteristicile distinctive ale tumorii.
- **Flip si rotatie pentru augmentarea datelor -** Augmentarea datelor reduce riscul de overfitting si ajuta modelul sa fie mai robust la variatiile de orientare ale imaginilor. Astfel, modelul este capabil sa generalizeze mai bine pe imagini care nu au fost vazute anterior.
- **Filtrul Sobel pentru detectarea marginilor**

Am observat impactul fiecarei transformari asupra imaginilor originale si am decis sa nu adaug filtrul Sobel in lista cu transformari pentru model.





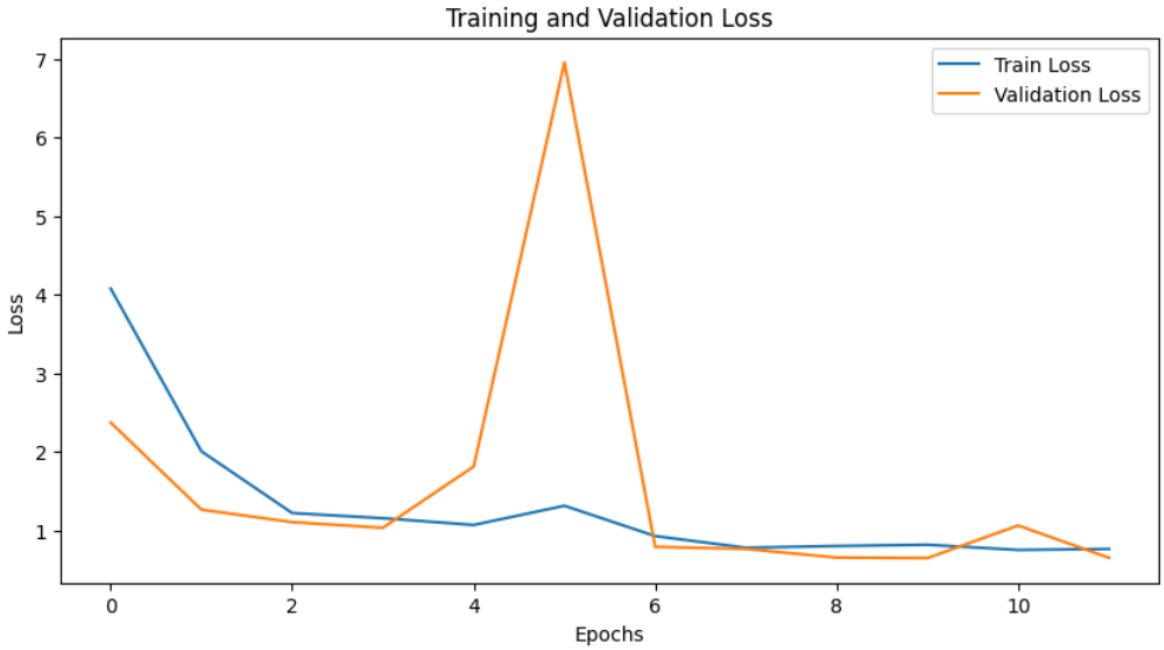
Cerinta 7: Pipeline-ul de Antrenare a Modelului

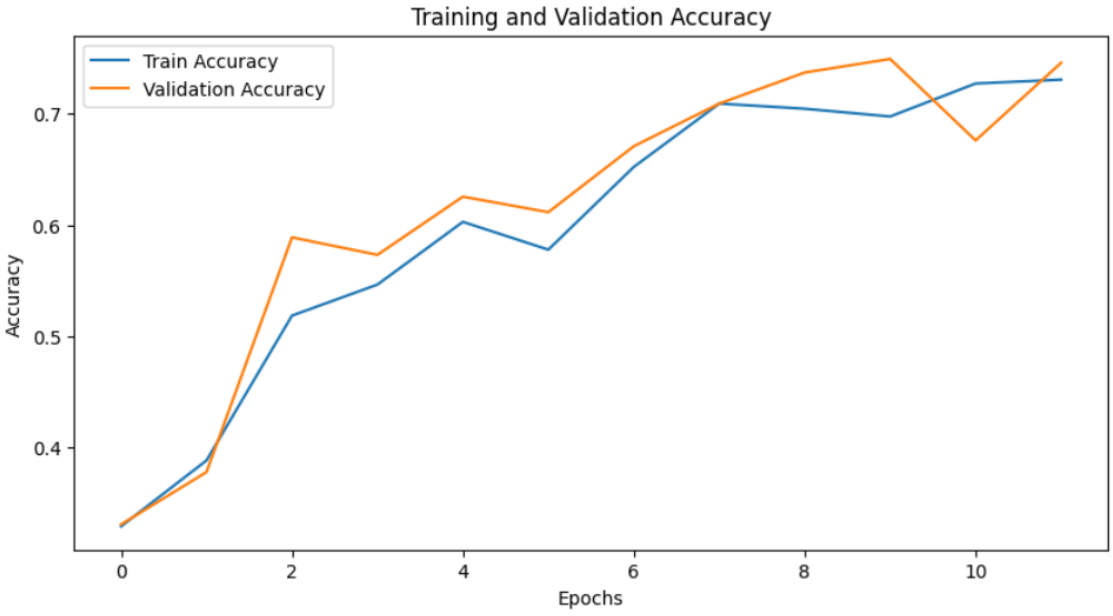
Am implementat un pipeline complet pentru antrenarea modelului `ResNet-50`:

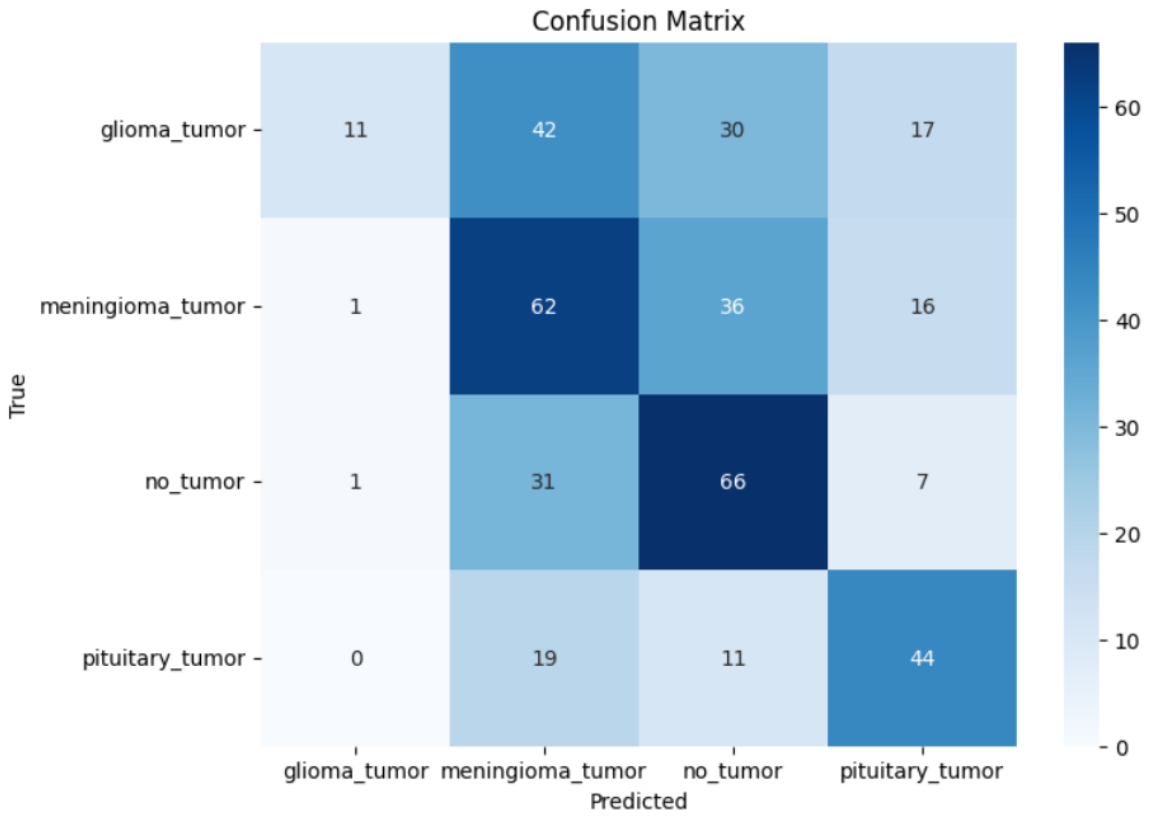
- **Incarcarea datelor**: Am divizat datele in date de antrenare, validare si testare si am folosit `DataLoader` pentru a citi datele.
- **Definirea modelului**: Am folosit `ResNet-50` cu stratul final adaptat pentru 4 clase.
- **Functii de pierdere si optimizator**: Am utilizat `CrossEntropyLoss` ca functie de loss si `SGD ca optimizator`.
- **Antrenarea modelului**: Am efectuat 12 epoci si am monitorizat pierderea si acuratetea pentru seturile de antrenare si validare.
- **Evaluarea modelului**: Am calculat metricele de performanta pe setul de test (acuratete, precizie, recall, F1-score) si am plotat matricea de confuzie.

Rezultate

- **Acuratetea pe setul de test**: 30.13%
- **F1-score**: 0.4076
- **Precision**: 0.5261
- **Recall**: 0.4390


Training and Validation Loss
Train Loss
Validation Loss
Loss
Epochs
0
2
4
6
8
10
1
2
3
4
5
6
7


Training and Validation Accuracy
Train Accuracy
Validation Accuracy
Accuracy
Epochs
0
2
4
6
8
10
0.4
0.5
0.6
0.7


Confusion Matrix
glioma_tumor
meningioma_tumor
no_tumor
pituitary_tumor
True
11
42
30
17
1
62
36
16
1
31
66
7
0
19
11
44
glioma_tumor
meningioma_tumor
no_tumor
pituitary_tumor
Predicted
60
50
40
30
20
10
0