Documentatie proiect - Inteligenta Artificiala

Bondoc Ana – Maria

Grupa 241

1. Scopul proiectului

Scopul acestui proiect este antrenarea unui model pentru clasificarea unor imagini în două clase de scanări ale creierului: una care conţine anomalii(1) şi una normală(0).

2. Metode de lucru

În rezolvarea proiectului am testat urmatoarele modele, prezentate în continuare: Naïve-Bayes, retele neuronale convolutionale.

3. Modelul Naive-Bayes

Primul model incercat a fost naive bayes.

Am folosit modelul Multinomial Naive Bayes (MNB) din libraria sklearn. Modelul presupune că toate caracteristicile sunt independente şi urmează o anumită distribuţie de probabilitate, numită distribuţie multinomială. Învaţă să estimeze probabilitatea caracteristicilor pentru fiecare clasă şi probabilitatea pentru fiecare clasă în sine.

Am citit din fisierul data toate imaginile intr-un vector pe care, ulterior, l-am convertit intr-un numpy array. Am impartit datele in date de antrenare(primele 15000), date de validare(urmatoarele 2000) si date de testare(ultimele 5149). Apoi, am citit label-urile corespunzatoare atat pentru datele de antrenare cat si pentru cele de validare.

Folosesc linspace pentru a obtine capetele intervalelor cu care putem categorisii pixelii. Impart pixeli in 3 intervale:

[0. 74.66666667 149.33333333 224.]

Am incercat sa categorisesc pixelii in intervalul [0, 256], dar acuratetea a scazut.

categoria 0 - toti pixelii au valori intre 0 si 74.66666667

categoria 1 - toti pixelii au valori intre 74.66666667 si 149.33333333

categoria 2 - toti pixelii au valori intre 149.33333333 si 224

Functia *valori_bin* transforma matricea array care contine pixeli, intr-o matrice de aceleasi dimensiuni in care pixelii sunt inlocuiti cu categoriile corespunzatoare. Aceasta functie foloseste conceptul din laborator.

Accuracy: 0.722

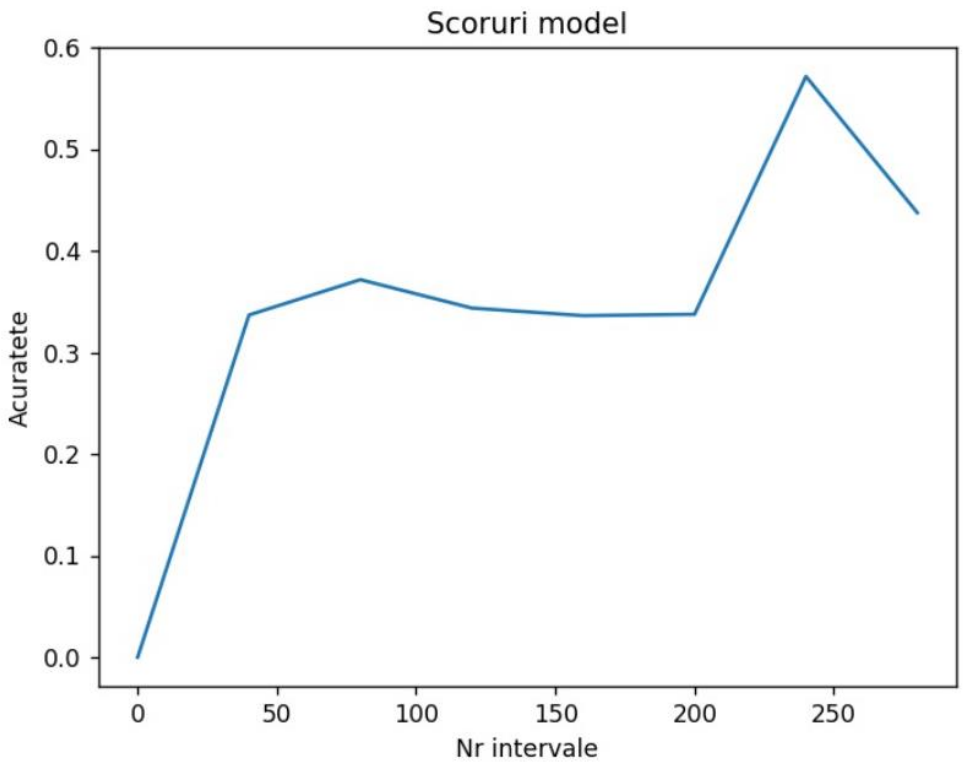
f1_score: 0.40598290598290604

Matrice de confuzie pe datele de validare:

[[1254. 470.]

[86. 190.]]

Grafic:



4. Retea neuronala convolutionala

Pentru construirea celui de-al doilea model am folosit Convolutional Neural Network(CNN).

```
model = Sequential([

    Conv2D(16, (3,3), 1, activation = 'relu', input_shape = (224, 224, 3)),
    BatchNormalization(),
    Conv2D(16, (3,3), 1, activation = 'relu'),
    BatchNormalization(),
    MaxPooling2D((2, 2)),
    Lambda(lambda x : x / 255.0),
    Conv2D(32, (3,3), 1, activation = 'relu'),
    BatchNormalization(),
    Conv2D(32, (3,3), 1, activation = 'relu'),
    BatchNormalization(),
```

```
    MaxPooling2D((2, 2)),
    Conv2D(64, (3,3), 1, activation = 'relu'),
    BatchNormalization(),
    Conv2D(64, (3,3), 1, activation = 'relu'),
    BatchNormalization(),
    MaxPooling2D((2, 2)),
    Conv2D(128, (3,3), 1, activation = 'relu'),
    BatchNormalization(),
    Conv2D(128, (3,3), 1, activation = 'relu'),
    BatchNormalization(),
    MaxPooling2D((2, 2)),
    Flatten(),
    Dense(256, activation = 'relu'),
    Dense(256, activation = 'relu'),
    Dense(256, activation = 'relu'),
    Dense(1, activation = 'sigmoid')
])
```

Acest model m-a ajutat sa obtin acuratetea maxima. Am testat numeroase valori pentru numarul de filtre si numarul de straturi care au dus la modelul de mai sus. De asemenea, straturile de BatchNormalization au crescut acuratetea semnificativ deoarece modelul a putut fi antrenat mai eficient. Am adaugat trei straturi Dense cu 256 de neuroni, deoarece conecteaza fiecare neuron din stratul curent la toți neuronii din stratul precedent.

Am folosit kernel de 3 x 3 care reprezinta o matrice de 3 x 3 ce gliseaza cu cate o pozitie in matricea de pixeli a unei imagini.

Primul strat pe care l-am adaugat este un strat convolutional cu 16 filtre, cu marimea filtrului de 3 × 3, glisarea de 1 si cu functia de activare relu. Fiind primul strat din retea, am specificat si input shape-ul de (224 × 224 × 3).

Al doilea strat este un strat de tip BatchNormalization. Are ca scop aducerea valorilor de ieșire ale stratului într-un interval standard, astfel încât rețeaua să poată fi antrenată mai eficient.

Al treilea strat pe care l-am adaugat este un strat de tipul MaxPooling2D, care reduce dimensiunea inputului.

Stratul Lambda imparte valoarea pixelilor la 255.0 oferind valori cuprinse intre 0 si 1. Am incercat sa fac acest lucru de la citirea imaginilor, dar a condus la memory exceeded.

Urmatoarele straturi convolationale au numarul de filtre crescut respectiv 32, 64, 128.

La final, am adaugat un strat de tip Flatten, pentru a aplatiza output-ul intr-un vector de dimensiune 1.

Am pus 3 straturi Dense cu cate 256 de neuroni si functia de activare relu. Al patrulea Dense este un strat complet conectat, dintr-un singur neuron cu functia de activare sigmoid.

La compilarea modelului am folosit functia binary_crossentropy, optimizerul adam folosit pentru a ajusta ponderile modelului în timpul antrenării si ca metrica acuratetea.

Am antrenat modelul in 20 de epoci, pe datele de antrenare, cu batch_size de 64.

Test loss: 0.4829079508781433

Test accuracy: 0.9004999995231628

Summary:

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 222, 222, 16)      448

batch_normalization (BatchN  (None, 222, 222, 16)     64
ormalization)

conv2d_1 (Conv2D)            (None, 220, 220, 16)      2320

batch_normalization_1 (Batc  (None, 220, 220, 16)     64
hNormalization)

max_pooling2d (MaxPooling2D  (None, 110, 110, 16)     0
)

lambda (Lambda)              (None, 110, 110, 16)      0

conv2d_2 (Conv2D)            (None, 108, 108, 32)      4640

batch_normalization_2 (Batc  (None, 108, 108, 32)     128
hNormalization)

conv2d_3 (Conv2D)            (None, 106, 106, 32)      9248

batch_normalization_3 (Batc  (None, 106, 106, 32)     128
hNormalization)

max_pooling2d_1 (MaxPooling  (None, 53, 53, 32)       0
2D)

conv2d_4 (Conv2D)            (None, 51, 51, 64)        18496

batch_normalization_4 (Batc  (None, 51, 51, 64)       256
hNormalization)

conv2d_5 (Conv2D)            (None, 49, 49, 64)        36928

batch_normalization_5 (Batc  (None, 49, 49, 64)       256
hNormalization)

max_pooling2d_2 (MaxPooling  (None, 24, 24, 64)       0
2D)

conv2d_6 (Conv2D)            (None, 22, 22, 128)       73856

batch_normalization_6 (Batc  (None, 22, 22, 128)      512
hNormalization)

conv2d_7 (Conv2D)            (None, 20, 20, 128)       147584

batch_normalization_7 (Batc  (None, 20, 20, 128)      512
hNormalization)

max_pooling2d_3 (MaxPooling  (None, 10, 10, 128)      0
2D)

flatten (Flatten)            (None, 12800)             0

dense (Dense)                (None, 256)               3277056

dense_1 (Dense)              (None, 256)               65792

dense_2 (Dense)              (None, 256)               65792

dense_3 (Dense)              (None, 1)                 257

=================================================================
Total params: 3,704,337
Trainable params: 3,703,377
Non-trainable params: 960
_________________________________________________________________
```

f1_score: 0.5446224256292906

Matrice de confuzie:

[[1682. 42.]
 [157. 119.]]

Grafice: