

# Dokumentáció

## Előzmények, eredeti háló, felmerült problémák:

A korábbi „Áttétes nyirokcsomó daganatok meghatározása szövettani képek osztályozásával neurális hálókat alkalmazva” témájú projektem megoldására egy Convolutional Neural Network-t építettem a képek osztályozására.

A megoldás során több problémám merült fel, nem sikerült a legjobban a háló felépítése, mert körülbelül csak 75%-os pontossággal osztályozott jól, valamint az adat mennyisége miatt memóriahiányom lépett fel.

Régi modell:

Model: "sequential"		
Layer (type)	Output Shape	Param #
=====		
conv2d (Conv2D)	(None, 46, 46, 16)	448
max_pooling2d (MaxPooling2D)	(None, 23, 23, 16)	0
conv2d_1 (Conv2D)	(None, 21, 21, 32)	4640
max_pooling2d_1 (MaxPooling2D)	(None, 10, 10, 32)	0
conv2d_2 (Conv2D)	(None, 8, 8, 64)	18496
max_pooling2d_2 (MaxPooling2D)	(None, 4, 4, 64)	0
conv2d_3 (Conv2D)	(None, 2, 2, 64)	36928
max_pooling2d_3 (MaxPooling2D)	(None, 1, 1, 64)	0
flatten (Flatten)	(None, 64)	0
dense (Dense)	(None, 512)	33280
dense_1 (Dense)	(None, 1)	513
=====		
Total params: 94,305		
Trainable params: 94,305		
Non-trainable params: 0		

## Feladatok:

Az előzetes egyeztetések során, meg kellett próbálnom csökkenteni a futási időt, új modellt implementálni, batch normalizációt alkalmazni, valamint egy adatbetöltő függvényt alkalmazni.

## Megvalósítások:

A már meglévő CNN-be építettem be batch normalizációs rétegeket, hogy megnézzük ez javít-e a predikció pontosságán.

Layer (type)	Output Shape	Param #
conv2d_25 (Conv2D)	(None, 94, 94, 16)	448
batch_normalization_5 (Batch Normalization)	(None, 94, 94, 16)	64
max_pooling2d_25 (MaxPooling2D)	(None, 47, 47, 16)	0
conv2d_26 (Conv2D)	(None, 45, 45, 32)	4640
batch_normalization_6 (Batch Normalization)	(None, 45, 45, 32)	128
max_pooling2d_26 (MaxPooling2D)	(None, 22, 22, 32)	0
conv2d_27 (Conv2D)	(None, 20, 20, 32)	9248
batch_normalization_7 (Batch Normalization)	(None, 20, 20, 32)	128
max_pooling2d_27 (MaxPooling2D)	(None, 10, 10, 32)	0
conv2d_28 (Conv2D)	(None, 8, 8, 64)	18496
batch_normalization_8 (Batch Normalization)	(None, 8, 8, 64)	256
max_pooling2d_28 (MaxPooling2D)	(None, 4, 4, 64)	0
conv2d_29 (Conv2D)	(None, 2, 2, 64)	36928
batch_normalization_9 (Batch Normalization)	(None, 2, 2, 64)	256
max_pooling2d_29 (MaxPooling2D)	(None, 1, 1, 64)	0
flatten_5 (Flatten)	(None, 64)	0
dense_11 (Dense)	(None, 512)	33280
dense_12 (Dense)	(None, 1)	513
Total params: 104,385		
Trainable params: 103,969		
Non-trainable params: 416		

A modellek tanulás során az epoch-ok pontossága a normalizációs rétegekkel javult viszont maga a végeredményen nem sikerült javulást elérnem.

Normalizációs rétegek nélkül:

```
9/9 [=====] - 1s 165ms/step - loss: 0.6748 - accuracy: 0.6000  
0.6000000238418579
```

Normalizációs rétegekkel:

```
9/9 [=====] - 2s 176ms/step - loss: 1.5332 - accuracy: 0.5200  
0.5199999809265137
```

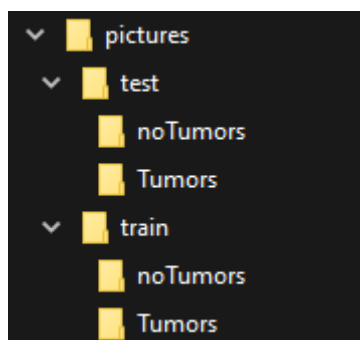
Normalizációs rétegek nélkül:

Epoch 1/20	
26/26 [=====]	- 15s 559ms/step - loss: 0.6832 - accuracy: 0.5818
Epoch 2/20	
26/26 [=====]	- 13s 505ms/step - loss: 0.6763 - accuracy: 0.5939
Epoch 3/20	
26/26 [=====]	- 13s 513ms/step - loss: 0.6765 - accuracy: 0.5939
Epoch 4/20	
26/26 [=====]	- 13s 504ms/step - loss: 0.6762 - accuracy: 0.5939
Epoch 5/20	
26/26 [=====]	- 13s 515ms/step - loss: 0.6759 - accuracy: 0.5939
Epoch 6/20	
26/26 [=====]	- 13s 504ms/step - loss: 0.6763 - accuracy: 0.5939
Epoch 7/20	
26/26 [=====]	- 13s 506ms/step - loss: 0.6754 - accuracy: 0.5939
Epoch 8/20	
26/26 [=====]	- 13s 505ms/step - loss: 0.6761 - accuracy: 0.5939
Epoch 9/20	
26/26 [=====]	- 13s 495ms/step - loss: 0.6758 - accuracy: 0.5939
Epoch 10/20	
26/26 [=====]	- 13s 497ms/step - loss: 0.6765 - accuracy: 0.5939
Epoch 11/20	
26/26 [=====]	- 13s 505ms/step - loss: 0.6756 - accuracy: 0.5939
Epoch 12/20	
26/26 [=====]	- 13s 509ms/step - loss: 0.6756 - accuracy: 0.5939
Epoch 13/20	
26/26 [=====]	- 13s 508ms/step - loss: 0.6756 - accuracy: 0.5939
Epoch 14/20	
26/26 [=====]	- 14s 521ms/step - loss: 0.6759 - accuracy: 0.5939
Epoch 15/20	
26/26 [=====]	- 13s 508ms/step - loss: 0.6755 - accuracy: 0.5939
Epoch 16/20	
26/26 [=====]	- 13s 511ms/step - loss: 0.6751 - accuracy: 0.5939
Epoch 17/20	
26/26 [=====]	- 13s 498ms/step - loss: 0.6749 - accuracy: 0.5939
Epoch 18/20	
26/26 [=====]	- 13s 511ms/step - loss: 0.6751 - accuracy: 0.5939
Epoch 19/20	
26/26 [=====]	- 13s 510ms/step - loss: 0.6744 - accuracy: 0.5939
Epoch 20/20	
26/26 [=====]	- 13s 511ms/step - loss: 0.6735 - accuracy: 0.5939

Normalizációs rétegekkel:

```
Epoch 1/20
26/26 [=====] - 30s 1s/step - loss: 0.7118 - accuracy: 0.5515
Epoch 2/20
26/26 [=====] - 29s 1s/step - loss: 0.6596 - accuracy: 0.6036
Epoch 3/20
26/26 [=====] - 29s 1s/step - loss: 0.6205 - accuracy: 0.6679
Epoch 4/20
26/26 [=====] - 29s 1s/step - loss: 0.5701 - accuracy: 0.7030
Epoch 5/20
26/26 [=====] - 29s 1s/step - loss: 0.4769 - accuracy: 0.7776
Epoch 6/20
26/26 [=====] - 30s 1s/step - loss: 0.3668 - accuracy: 0.8442
Epoch 7/20
26/26 [=====] - 31s 1s/step - loss: 0.3181 - accuracy: 0.8515
Epoch 8/20
26/26 [=====] - 29s 1s/step - loss: 0.2098 - accuracy: 0.9091
Epoch 9/20
26/26 [=====] - 27s 1s/step - loss: 0.1487 - accuracy: 0.9418
Epoch 10/20
26/26 [=====] - 27s 1s/step - loss: 0.1376 - accuracy: 0.9485
Epoch 11/20
26/26 [=====] - 27s 1s/step - loss: 0.1601 - accuracy: 0.9382
Epoch 12/20
26/26 [=====] - 27s 1s/step - loss: 0.1334 - accuracy: 0.9424
Epoch 13/20
26/26 [=====] - 27s 1s/step - loss: 0.1046 - accuracy: 0.9600
Epoch 14/20
26/26 [=====] - 28s 1s/step - loss: 0.0955 - accuracy: 0.9618
Epoch 15/20
26/26 [=====] - 27s 1s/step - loss: 0.0490 - accuracy: 0.9818
Epoch 16/20
26/26 [=====] - 27s 1s/step - loss: 0.0443 - accuracy: 0.9848
Epoch 17/20
26/26 [=====] - 27s 1s/step - loss: 0.0459 - accuracy: 0.9842
Epoch 18/20
26/26 [=====] - 27s 1s/step - loss: 0.0694 - accuracy: 0.9721
Epoch 19/20
26/26 [=====] - 27s 1s/step - loss: 0.1241 - accuracy: 0.9479
Epoch 20/20
26/26 [=====] - 28s 1s/step - loss: 0.1427 - accuracy: 0.9406
```

Sikerült megoldanom a képek beolvasását, hogy egyszerre csak egy részüket tartsa a memóriában, ám ehhez egy külön kóddal kellett megszereznem az adathalmazomat, az alábbi módon:



Ekkor már tudtam alkalmazni a Python beépített képbetöltő függvényét, az ImageDataGeneratort().

A futási időm sajnos nem sikerült csökkentenem, így a sok tesztelgetés során nem futtattam az egész adathalmazon, hanem csak kisebb részletein, hogy lássam a kód tényleges működését. Megpróbáltam új modellt implementálni, de az új adatbetöltő alkalmazása mellett mindez nem sikerült. Első sorban a ResNet50-el és a VGG16-al próbálkoztam.

**Hivatkozások:**

- <https://www.kaggle.com/omniscientist99/metastatic-cancer-in-lymph-nodes-fastai-resnet50>
- <https://github.com/Precillieo/Elixir-Cancer-Diagnosis-AI-Based-System/blob/main/Elixir/Cancer-Prediction.ipynb>
- <https://machinelearningmastery.com/how-to-load-large-datasets-from-directories-for-deep-learning-with-keras/>
- <https://machinelearningmastery.com/how-to-develop-a-convolutional-neural-network-to-classify-photos-of-dogs-and-cats/>
- <https://keras.io/api/applications/>

2021. május 28.

Kubicza Gréta Andrea