



Neumann János Egyetem  
Műszaki és Informatikai  
Kar

# Mesterséges intelligencia alapjai

## Beszámoló a beadandó feladathoz

Bodor Bence

X0TUFQ

Mérnökinformatikus szak

Választott feladat: 9. Autótípus felismerés  
képek alapján

2024.12.04

## Tartalomjegyzék

|                                    |           |
|------------------------------------|-----------|
| <b>1. Feladat leírása</b>          | <b>3</b>  |
| <b>2. Forráskód</b>                | <b>4</b>  |
| <b>3. Nehézségek és megoldások</b> | <b>12</b> |
| 3.1 Adathalmaz problémák           | 12        |
| 3.2 Modell teljesítménye           | 12        |
| <b>4. Tanulságok</b>               | <b>13</b> |
| 4.1 Adatminőség fontossága         | 13        |
| 4.2 Iteratív fejlesztés            | 13        |
| 4.3 Erőforrások hatékonysága       | 13        |
| <b>5. Összegzés</b>                | <b>13</b> |

## **1. Feladat leírása**

A feladat célja egy autó osztályozó modell létrehozása volt, amely képes különböző autótípusok azonosítására. Ehhez konvolúciós neurális hálózatot (CNN) használtam, amely egy ismert megoldás a számítógépes látás problémákra.

## 2. Forráskód

```
image_Car_Calssifier.ipynb ×
AI_beadandó > image_Car_Calssifier.ipynb > ...
+ Code + Markdown | ▶ Run All | Clear All Outputs | Outline ... Python 3.11.9

import numpy as np
import pandas as pd

import os
for dirname, _, filenames in os.walk('.'):
    print(os.path.join(dirname))

[20] Python

...
.\Cars Dataset
.\Cars Dataset\test
.\Cars Dataset\test\Audi
.\Cars Dataset\test\Hyundai Creta
.\Cars Dataset\test\Mahindra Scorpio
.\Cars Dataset\test\Rolls Royce
.\Cars Dataset\test\Swift
.\Cars Dataset\test\Tata Safari
.\Cars Dataset\test\Toyota Innova
.\Cars Dataset\train
.\Cars Dataset\train\Audi
.\Cars Dataset\train\Hyundai Creta
.\Cars Dataset\train\Mahindra Scorpio
.\Cars Dataset\train\Rolls Royce
.\Cars Dataset\train\Swift
.\Cars Dataset\train\Tata Safari
.\Cars Dataset\train\Toyota Innova
```

A tanításhoz és teszteléshez használt dataset könyvtárszerkezete.

```
image_Car_Calssifier.ipynb ×
AI_beadandó > image_Car_Calssifier.ipynb > ...
+ Code + Markdown | ▶ Run All | Clear All Outputs | Outline ... Python 3.11.9

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,Dropout,Convolution2D,MaxPooling2D,Flatten
import tensorflow as tf
import matplotlib.pyplot as plt
from IPython.display import HTML

[1] Python

from tensorflow.keras.preprocessing.image import ImageDataGenerator
IMAGE_SIZE = 128

train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=10,
    horizontal_flip=True
)
train_generator = train_datagen.flow_from_directory(
    'Cars Dataset/train',
    target_size=(IMAGE_SIZE,IMAGE_SIZE),
    class_mode="sparse",
)

[4] Python

... Found 3352 images belonging to 7 classes.
```

Szükséges könyvtárak beimportálása, illetve képek betöltése, normalizálása és méretezése

```
image_Car_Calssifier.ipynb X
AI_beadandó > image_Car_Calssifier.ipynb > ...
+ Code + Markdown | ▶ Run All ≡ Clear All Outputs | ≡ Outline ... Python 3.11.9

count=0
for image_batch, label_batch in train_generator:
    print(image_batch[0])
    break

[[[0.6766087 0.73634356 0.79215693]
 [0.64166296 0.70922893 0.7801047 ]
 [0.60188967 0.69411767 0.7749535 ]
 ...
 [0.33143583 0.52161086 0.6961839 ]
 [0.33208773 0.52204543 0.6972704 ]
 [0.33273965 0.5224801 0.6983569 ]]

 [[0.6774779 0.73677814 0.79215693]
 [0.64427054 0.7114019 0.7811912 ]
 [0.60384536 0.69411767 0.77473617]
 ...
 [0.35306168 0.5373453 0.723665 ]
 [0.3548001 0.5386491 0.7256207 ]
 [0.3565385 0.53995293 0.72757643]]

 [[0.67834705 0.7372128 0.79215693]
 [0.64687806 0.71357495 0.78227764]
 [0.60580105 0.69411767 0.7745189 ]
 ...
 [0.38438603 0.5608386 0.7490558 ]
 [0.38612446 0.5621424 0.749925 ]
 [0.38786286 0.5634462 0.75079423]]]
```

Minta (image\_batch) és címke (label\_batch) kiolvasása a tesztadatokból,  
majd az első képhez tartozó pixel adatok megjelenítése.

```
image_Car_Calssifier.ipynb X
AI_beadandó > image_Car_Calssifier.ipynb > ...
+ Code + Markdown | ▶ Run All ≡ Clear All Outputs | ≡ Outline ... Python 3.11.9

class_names = list(train_generator.class_indices.keys())
class_names

['Audi',
 'Hyundai Creta',
 'Mahindra Scorpio',
 'Rolls Royce',
 'Swift',
 'Tata Safari',
 'Toyota Innova']
```

A tréning osztályok neveinek letárolása a tréningadatok könyvtárstruktúrából.

```
image_Car_Calssifier.ipynb X
AI_beadandó > image_Car_Calssifier.ipynb > ...
+ Code + Markdown | Run All Clear All Outputs | Outline ... Python 3.11.9

test_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=10,
    horizontal_flip=True)

test_generator = test_datagen.flow_from_directory(
    'Cars Dataset/test',
    target_size=(IMAGE_SIZE, IMAGE_SIZE),
    class_mode="sparse"
)

[8] Python
... Found 813 images belonging to 7 classes.
```

'ImageDataGenerator' a tesztadatokra is alkalmazva, hasonlóan a tréningadatokhoz.

```
image_Car_Calssifier.ipynb X
AI_beadandó > image_Car_Calssifier.ipynb > ...
+ Code + Markdown | Run All Clear All Outputs | Outline ... Python 3.11.9

SZ = 128

model = Sequential()
model.add(Convolution2D(32, (3, 3), input_shape=(sz, sz, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Convolution2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(units=96, activation='relu'))
model.add(Dropout(0.40))
model.add(Dense(units=32, activation='relu'))
model.add(Dense(units=7, activation='softmax'))

[10] Python
... C:\Users\Djokko\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p...
super().__init__(activity_regularizer=activity_regularizer, **kwargs)

model.summary()

[11] Python
... Model: "sequential"
```

Sequential modell létrehozása a konvolúciós hálózathoz. A konvolúciós rétegek 32 szűrővel 'relu'-s aktivációval lettek létrehozva. A 'Dense' 32 és 96 neuronos rétegek szintén 'relu' aktivációval, illetve egy 7 kimenetű 'softmax' osztályzó réteg.

...

| Layer (type)                   | Output Shape         | Param #   |
|--------------------------------|----------------------|-----------|
| conv2d (Conv2D)                | (None, 126, 126, 32) | 896       |
| max_pooling2d (MaxPooling2D)   | (None, 63, 63, 32)   | 0         |
| conv2d_1 (Conv2D)              | (None, 61, 61, 32)   | 9,248     |
| max_pooling2d_1 (MaxPooling2D) | (None, 30, 30, 32)   | 0         |
| flatten (Flatten)              | (None, 28800)        | 0         |
| dense (Dense)                  | (None, 96)           | 2,764,896 |
| dropout (Dropout)              | (None, 96)           | 0         |
| dense_1 (Dense)                | (None, 32)           | 3,104     |
| dense_2 (Dense)                | (None, 7)            | 231       |

...

**Total params:** 2,778,375 (10.60 MB)

...

**Trainable params:** 2,778,375 (10.60 MB)

...

**Non-trainable params:** 0 (0.00 B)

A modell rétegeinek és paramétereinek összefoglalása.

image\_CatClassifier.ipynb

AI beadandó > image\_CatClassifier.ipynb > ...

Code | Markdown | Run All | Clear All Outputs | Outline | Python 3.11.9

```
model.compile(optimizer='adam', loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False), metrics=['acc
```

```
history = model.fit(
    train_generator,
    validation_data=test_generator,
    epochs=50
)
```

C:\Users\DJokko\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11\_qbz5n2kfra8p0\LocalCache\local-packages\Py

self.\_warn\_if\_super\_not\_called()

Epoch 1/50  
105/105 — 33s 293ms/step - accuracy: 0.2102 - loss: 1.9771 - val\_accuracy: 0.3284 - val\_loss: 1.790

Epoch 2/50  
105/105 — 34s 324ms/step - accuracy: 0.3228 - loss: 1.7850 - val\_accuracy: 0.3850 - val\_loss: 1.688

Epoch 3/50  
105/105 — 32s 304ms/step - accuracy: 0.3512 - loss: 1.7177 - val\_accuracy: 0.4440 - val\_loss: 1.573

Epoch 4/50  
105/105 — 35s 328ms/step - accuracy: 0.4266 - loss: 1.6008 - val\_accuracy: 0.4994 - val\_loss: 1.419

Epoch 5/50  
105/105 — 33s 318ms/step - accuracy: 0.4638 - loss: 1.4828 - val\_accuracy: 0.5092 - val\_loss: 1.408

Epoch 6/50  
105/105 — 34s 322ms/step - accuracy: 0.4982 - loss: 1.3998 - val\_accuracy: 0.5609 - val\_loss: 1.249

Epoch 7/50  
105/105 — 35s 327ms/step - accuracy: 0.5274 - loss: 1.2976 - val\_accuracy: 0.5781 - val\_loss: 1.208

Epoch 8/50  
105/105 — 34s 326ms/step - accuracy: 0.5690 - loss: 1.1919 - val\_accuracy: 0.5904 - val\_loss: 1.177

Epoch 9/50  
105/105 — 35s 326ms/step - accuracy: 0.5843 - loss: 1.1331 - val\_accuracy: 0.6002 - val\_loss: 1.141

...

Epoch 49/50  
105/105 — 36s 338ms/step - accuracy: 0.8529 - loss: 0.3871 - val\_accuracy: 0.7109 - val\_loss: 0.985

Epoch 50/50  
105/105 — 35s 328ms/step - accuracy: 0.8488 - loss: 0.3845 - val\_accuracy: 0.7319 - val\_loss: 1.062

Output is truncated. View as a [scrollable element](#) or open in a [text editor](#). Adjust cell output [settings](#)...

Modell 'fordítása' az 'adam' optimalizáló és a veszteségfüggvény implementálása, model tanítása 50 'epoch'-on keresztül(a felett már csökken a tanulás pontossága).

```
image_Car_Calssifier.ipynb
AI_beadandó > image_Car_Calssifier.ipynb > ...
+ Code + Markdown | ▶ Run All | Clear All Outputs | Outline ... Python 3.11.9

scores = model.evaluate(test_generator)
scores

[14] Python

... 26/26 ————— 5s 206ms/step - accuracy: 0.7171 - loss: 1.1324

... [1.0995163917541504, 0.7183271646499634]
```

A tesztadatok alapján a modell kiértékelése, és az elért pontszám megjelenítése.

```
image_Car_Calssifier.ipynb
AI_beadandó > image_Car_Calssifier.ipynb > ...
+ Code + Markdown | ▶ Run All | Clear All Outputs | Outline ... Python 3.11.9

history.history.keys()
type(history.history['loss'])
len(history.history['loss'])
history.history['loss'][:5]

[15] Python

... [1.9073936939239502,
1.7830756902694702,
1.66820228099823,
1.5639196634292603,
1.4661492109298706]
```

```
image_Car_Calssifier.ipynb
AI_beadandó > image_Car_Calssifier.ipynb > ...
+ Code + Markdown | ▶ Run All | Clear All Outputs | Outline ... Python 3.11.9

acc = history.history['accuracy']
val_acc = history.history['val_accuracy']

loss = history.history['loss']
val_loss = history.history['val_loss']

[16] Python

import matplotlib.pyplot as plt
EPOCHS = 50

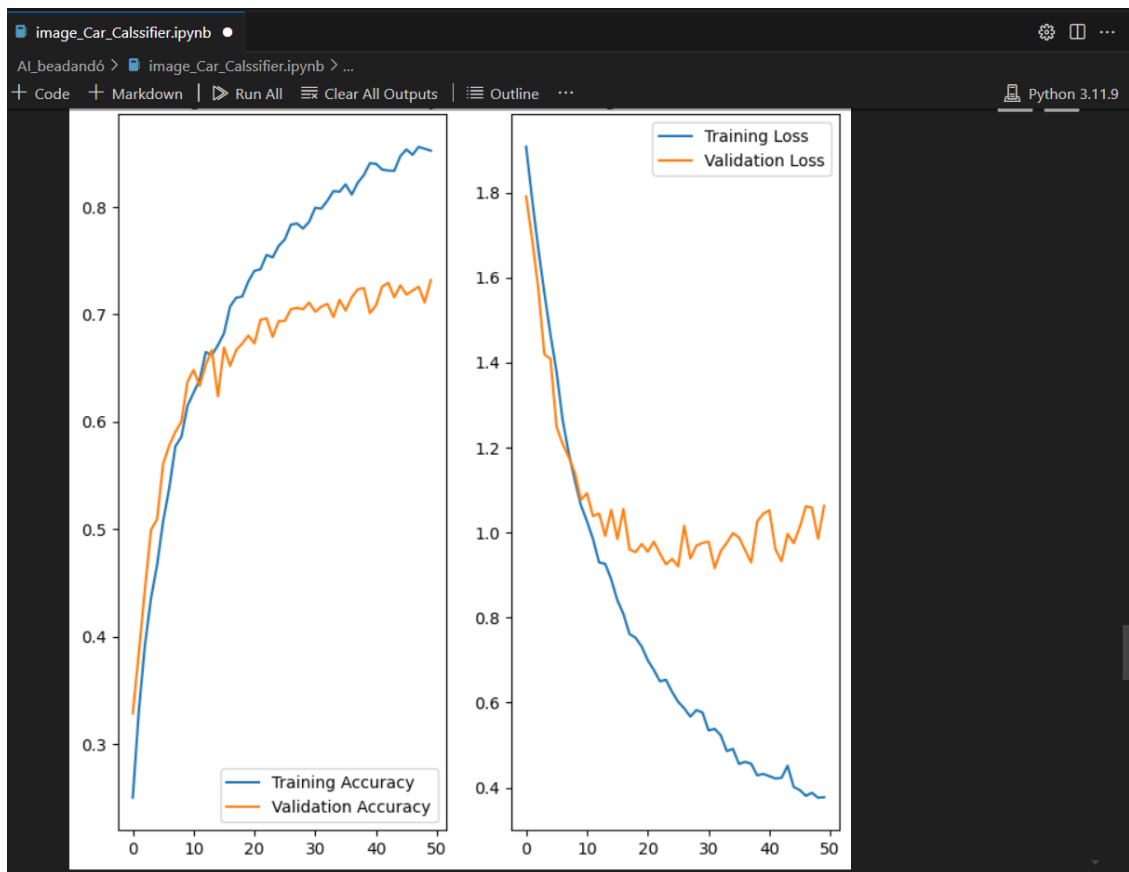
plt.figure(figsize=(8, 8))
plt.subplot(1, 2, 1)
plt.plot(range(EPOCHS), acc, label='Training Accuracy')
plt.plot(range(EPOCHS), val_acc, label='Validation Accuracy')
plt.legend(loc='lower right')
plt.title('Training and Validation Accuracy')

plt.subplot(1, 2, 2)
plt.plot(range(EPOCHS), loss, label='Training Loss')
plt.plot(range(EPOCHS), val_loss, label='Validation Loss')
plt.legend(loc='upper right')
plt.title('Training and Validation Loss')
plt.show()

[17] Python
```

Az adatok elmentése: pontosság , veszteség





Tanulási görbék ábrázolása(tréning- és validációs pontosság és veszteség)

```
image_Car_Calssifier.ipynb
AL_beadandó > image_Car_Calssifier.ipynb > ...
+ Code + Markdown | ▶ Run All | Clear All Outputs | Outline ... Python 3.11.9

def predict(model, img):
    img_array = tf.keras.preprocessing.image.img_to_array(images[i])
    img_array = tf.expand_dims(img_array, 0)

    predictions = model.predict(img_array)

    predicted_class = class_names[np.argmax(predictions[0])]
    confidence = round(100 * (np.max(predictions[0])), 2)
    return predicted_class, confidence

[18] Python

plt.figure(figsize=(15, 15))
for images, labels in test_generator:
    for i in range(8):
        ax = plt.subplot(3, 3, i + 1)
        plt.imshow(images[i])

        predicted_class, confidence = predict(model, images[i])
        actual_class = class_names[int(labels[i])]

        plt.title(f"Actual: {actual_class},\n Predicted: {predicted_class}.\n Confidence: {confidence}%")

        plt.axis("off")
    break

[24] Python
```

```
... 1/1 ██████████ 0s 55ms/step
     1/1 ██████████ 0s 63ms/step
     1/1 ██████████ 0s 53ms/step
     1/1 ██████████ 0s 63ms/step
     1/1 ██████████ 0s 77ms/step
     1/1 ██████████ 0s 70ms/step
     1/1 ██████████ 0s 63ms/step
     1/1 ██████████ 0s 55ms/step
```

Egy kép feldolgozása és megjósolja az osztályát és annak valószínűségét, majd visszatérési értéként visszaadja a megjósolt osztály nevét és a magabiztossági értékét. Megjelenít 8 teszt képet és az osztályt, megadja megjósolt osztályokat és a magabiztossági értékeket.

image\_Car\_Calssifier.ipynb

AI\_beadandó > image\_Car\_Calssifier.ipynb > ...

Code

Markdown


Run All

Clear All Outputs


Outline

Python 3.11.9

Actual: Tata Safari,  
Predicted: Toyota Innova.  
Confidence: 36.25%




Actual: Toyota Innova,  
Predicted: Toyota Innova.  
Confidence: 99.91000366210938%




Actual: Mahindra Scorpio,  
Predicted: Mahindra Scorpio.  
Confidence: 99.83000183105469%

Actual: Rolls Royce,  
Predicted: Audi.  
Confidence: 84.41999816894531%




Actual: Toyota Innova,  
Predicted: Toyota Innova.  
Confidence: 100.0%




Actual: Audi,  
Predicted: Toyota Innova.  
Confidence: 72.5%

Actual: Mahindra Scorpio,  
Predicted: Audi.  
Confidence: 93.1500015258789%



Actual: Toyota Innova,  
Predicted: Toyota Innova.  
Confidence: 98.91999816894531%



Spaces: 4Cell 10 of 17Go Live

### **3. Nehézségek és megoldások**

3.1 Adathalmaz problémák: A képek különböző méretűek és minőségűek voltak, ami nehezítette a modell számára az egységes tanulást.

Megoldás: Az 'ImageDataGenerator' segítségével egységesítettem a képméreteket, és bővítéssel növeltem az adatok sokféleségét (pl. forgatás, tükrözés).

3.2 Modell teljesítménye: Az első modellek alacsony pontosságot értek el.

Megoldás: Több iterációban optimalizáltam a hálózat struktúráját (pl. dropout rétegeket adtam hozzá a túlilleszkedés csökkentésére). Különböző aktivációs függvényeket és optimalizáló algoritmusokat próbáltam ki.

## **4. Tanulságok**

4.1 Adatminőség fontossága: A projekt rámutatott, hogy az adatok előkészítése kulcsfontosságú. A megfelelő méretre hozott és bővített adatok jelentősen javították a modell teljesítményét.

4.2 Iteratív fejlesztés: A hálózat felépítése és hiperparaméterezése iteratív folyamat. Minden módosítás után szükség volt teljesítménytesztelésre.

4.3 Erőforrások hatékonysága: Nagy modellek esetén a megfelelő hardverhasználat (pl. GPU) alapvető fontosságú.

## **5. Összegzés**

A projekt során egy működőképes autóosztályozó modellt fejlesztettem. A felmerült problémákat a kísérleti megközelítések és technikai eszközök segítségével sikerült megoldani.