Fejlesztői dokumentáció

Alphabet_recogn.py

Feladat

Kézzel írt karakter felismerése homogén háttér előtt. Létező modell alapján.

Függőségek: OpenCV, Keras, NumPy, sys, getopt

Környezet

Python 3.9.7 futtatására alkalmas operációs rendszer. Fejlesztői környezet VS Code.

Forráskód

A paraméterben kapott elérési utat megjegyeztetjük az inputfile változóba. Ezek után betöltjük az előre elkészített modellünket, létrehozzuk hozzá a szótárat, mivel amikor megjósoljuk a képen található betű/szám osztályát, azt ez alapján tudjuk behatárolni, hogy mit jelenítsen meg. Ezek után OpenCV segítségével megnyitjuk az inputfile-ba mentett elérési útról a képet, ezt egy imgcopy nevű változóban előfeldolgozzuk a modellünk számára, majd ezen a képen végrehajtjuk a jóslást. Az eredetileg beolvasott képünket átméretezzük és OpenCV segítségével ráírjuk a megjósolt eredményünket, majd megjelenítjük.

```
import cv2
                                                      A szükséges könyvtárak
import numpy as np
from keras.models import load model
                                                             importálása
import sys, getopt
argv=sys.argv[1:]
inputfile="
try:
  opts, args = getopt.getopt(argv,"hi:",["infile="])
except getopt.GetoptError:
                                                                       Command
  print('Alphabet_recogn.py -i <inputfile>')
                                                                 Prompt/Powershell-ről
  sys.exit(2)
                                                                    indított program
for opt, arg in opts:
                                                               argumentum ellenőrzése
  if opt == '-h':
     print('Alphabet_recogn.py -i <inputfile>')
     sys.exit()
  elif opt in ("-i", "--infile"):
     inputfile = arg
```

```
model = load_model('modelname.h5')
model.summary()
                                                                                  Modell
alph dict = \{0:'0',1:'1',2:'2',3:'3',4:'4',5:'5',6:'6',7:'7',8:'8',9:'9',
                                                                               betöltése, és
        10:'a',11:'b',12:'c',13:'d',14:'e',15:'f',16:'g',17:'h',18:'i',19:'j',
        20:'k',21:'l',22:'m',23:'n',24:'o',25:'p',26:'q',27:'r',28:'s',29:'t',
                                                                                 a hozzá
        30:'u',31:'v',32:'w',33:'x',34:'y',35:'z',36:'A',37:'B',38:'C',39:'D',
                                                                              tartozó szótár
        40:'E',41:'F',42:'G',43:'H',44:'l',45:'J',46:'K',47:'L',48:'M',49:'N',
                                                                               létrehozása
        50:'O',51:'P',52:'Q',53:'R',54:'S',55:'T',
        56:'U',57:'V',58:'W',59:'X',60:'Y',61:'Z'}
img= cv2.imread(inputfile)
ima copy =ima.copy()
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
img = cv2.resize(img, (400,400))
                                                                            Kép betöltése
img copy = cv2.GaussianBlur(img copy, (3,3), 0)
                                                                           argumentumból,
img gray = cv2.cvtColor(img copy, cv2.COLOR BGR2GRAY)
                                                                          előfeldolgozás és
_, img_thresh = cv2.threshold(img_gray, 100, 255,
                                                                                 tipp
                      cv2.THRESH_BINARY_INV)
                                                                           meghatározása
img_final = cv2.resize(img_thresh, (24,24))
img final =np.reshape(img final, (1,24,24,1))
img_pred = alph_dict[np.argmax(model.predict(img_final))]
tipp=np.argmax(model.predict(img_final))
print(tipp)
print('Number of arguments: {}'.format(len(sys.argv)))
print('Argument(s) passed: {}'.format(str(sys.argv)))
cv2.putText(img, "Tipp: " + img_pred, (20,370),
                                                                        A megjósolt tippünk
      cv2.FONT HERSHEY TRIPLEX, 1.3, color = (255.0.0))
                                                                           ráhelyezése a
cv2.imshow('Handwritten character recognition', img)
                                                                           képre, ablak
cv2.waitKey(0)
                                                                             bezárása
cv2.destroyAllWindows()
                                                                          gombnyomásra
```

Fejlesztési lehetőségek

Az argumentumoknál több fájl fogadása.

Több betűről készült kép fogadása, azok helyzetének detektálása és ezután a felismerésük.

Modell megadásának lehetősége argumentumban.

seq_mod.py

Feladat

Modell létrehozása kézzel írt betűk és számok felismerésére

Függőségek: TensorFlow, Keras

Forráskód

Beolvassuk az előre maghatározott helyről az adatokat, melyeket két részre bontunk, egyikkel betanítjuk a modellt a másikkal, pedig validáljuk a pontosságát. Ezután létrehozzuk a modellt. Ehhez a Keras Sequential modellt használtam. Ez egy egyszerű modell típus egy bemenettel és egy kimenettel, rétegek egymásutániságával. Ezek után összeállítjuk a modellt és betanítjuk a meglevő adatainkkal, majd elmentjük azt későbbi felhasználásra, emellett kiírjuk a képernyőre a modell pontosságát a veszteségek mértékét, valamint a modell összegzését.

```
import keras
                                                        A szükséges könyvtárak
from keras import layers
                                                              importálása
import tensorflow as tf
from tensorflow.keras.optimizers import Adam
IMG_WIDTH=24
                                                              Főbb konstansok
IMG HEIGHT=24
                                                                 kiemelése
DATA DIR=r"path to the dataset directory"
NUM_CLASSES = 62
trainset=tf.keras.utils.image_dataset_from_directory(
  DATA DIR, labels="inferred",
  label_mode="categorical", class_names=None,
  color_mode="grayscale", batch_size=64,
  image size=(IMG WIDTH, IMG HEIGHT),
  shuffle=True, seed=3.
  validation split=0.2, subset="training",
  interpolation="bilinear",
  follow_links=False,
  crop_to_aspect_ratio=False
                                                                  Adatok beolvasása és
                                                                  felbontása a rendezett
valset=tf.keras.utils.image dataset from directory(
                                                                   könyvtárstruktúrából
  DATA_DIR, labels="inferred", label_mode="categorical",
  class_names=None, color_mode="grayscale",
  batch_size=64,
  image_size=(IMG_WIDTH, IMG_HEIGHT),
  shuffle=True, seed=3,
  validation_split=0.2, subset="validation",
  interpolation="bilinear",
  follow links=False,
  crop_to_aspect_ratio=False
```

```
model = keras.Sequential(
     keras.Input(shape=(IMG_HEIGHT,IMG_WIDTH,1)).
     layers.Conv2D(32, kernel_size=(3, 3), activation="relu"),
     layers.MaxPooling2D(pool_size=(2, 2)),
                                                                            Modell és a
     layers.Conv2D(64, kernel_size=(3, 3), activation="relu"),
                                                                              rétegek
     layers.MaxPooling2D(pool size=(2, 2)),
                                                                            létrehozása
     layers.Flatten(),
     layers.Dropout(0.5),
     layers.Dense(NUM_CLASSES, activation="softmax"),
model.compile(optimizer = Adam(learning_rate=0.001),
loss='categorical crossentropy', metrics=['accuracy'])
                                                                              Modell
history = model.fit(trainset, batch_size=64, epochs=3.
                                                                           összeállítása,
                   validation_data = valset)
                                                                           betanítása és
model.summary()
score=model.evaluate(valset,verbose=0)
                                                                              mentése
print(score)
model.save(r'modelname.h5')
print("The validation accuracy is :", history.history['val_accuracy'])
                                                                              Adatok
print("The training accuracy is :", history.history['accuracy'])
print("The validation loss is :", history.history['val_loss'])
                                                                              kiírása a
                                                                            képernyőre
print("The training loss is :", history.history['loss'])
```

Fejlesztési lehetőségek

Modell rétegeinek bővítése. Másik modell bevezetése több input fogadásához. Adatbeolvasás beállításainak megváltoztatása

Szerző

Zsédenyi Bálint Ádám (F5LMZJ) SZE Mérnökinformatikus hallgató