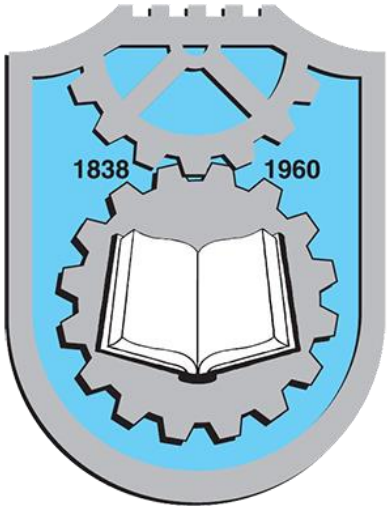
Univerzitet u Kragujevcu
Fakultet inženjerskih nauka



Predmet:
Osnovi mašinskog i dubokog učenja

Tema: CNN – Nošenje maske
(FaceMask Recognition)

Student:
Anđela Manojlović 628/2018

Profesor: Vladimir M. Milovanović

Kragujevac, april 2022. godine

Sadržaj

Opis zadatka

Zadatak ovog projekta je da, na osnovu ulaza koji predstavlja snimak kamere u datom trenutku, dobijemo izlaz koji će predstavljati informaciju da li je u kadru osoba sa maskom,ili bez maske. Kamera ne mora biti isključivo kamera onog uređaja sa kog se program pokreće, može se iskoristiti i npr. kamera telefona, ukoliko na njemu (i na računaru sa kog pokrećemo program) imamo instaliranu aplikaciju DroidCam.

Korišćene tehnologije

Kod je pisan u programskom jeziku Python. Razlog tome je što podržava biblioteke sa puno funkcija korisnih pri treniranju konvolucijskih neuronskih mreža. Korišćene biblioteke su :

Tensorflow – biblioteka otvorenog koda za Python numerička izračunavanja, biblioteka koja olakšava process sticanja podataka,obučavanja modela,predviđanja I unapređivanja budućih rezultata.

Keras – softverska biblioteka otvorenog koda koja pruža Python interfejs za veštačke neuronske mreže. Keras deluje kao interfejs za biblioteku Tensorflow.

OpenCV – biblioteka programskih funkcija,uglavnom u realnom vremenu,prvobitno razvijena u Intelovom istraživačkom centru. Sadrži 2D I 3D alatke, sistem za prepoznavanje lica (korišćeno u ovom projektu) , prepoznavanje pokreta, identifikaciju objekata , praćenje kretanja, i još puno toga korisnog za veštačke neuronske mreže, i mašinsko učenje uopšte.

Osim navedenih biblioteka korišćen je i :

Numpy – proširenje na Python programskom jeziku, dodajući podršku za velike višedimenzionalne nizove i matrice, zajedno sa velikim bibliotekama matematičkih funkcija na visokom nivou koje operišu nad ovim nizovima.

Analiza koda

Za realizaciju ovog projekta, najpre je potrebno da se slike obrade na odgovarajući način. Baza podataka sa slikama je preuzeta sa sajta Humans in the Loop, sa linka https://humansintheloop.org/resources/datasets/mask-dataset-download/?submissionGuid=0576b7ae-d010-4c99-a224-e0caeea28187 , ali je zbog lakšeg treniranja skraćena, i u konačnoj verziji ima nešto manje od 1.400 slika.

Slike se obrađuju u fajlu face_mask_pretprocessing.py na sledeći način:
-Svaka slika treba da bude istih dimenzija i to 100x100 piksela

-slike se prevode u nijanse crne I bele boje, I to je (kao i prethodna stavka obrade) suštinski omogućeno sledećim funkcijama:

`gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)`

`cv2.resize(gray,(img_size,img_size))`

pri čemu je `img_size=100`

Obrađene slike se smeštaju u niz data[], a u niz target[] se smešta njihova oznaka (“mask”/”no mask”).

Svrha poslednjih 8 linija koda je prevođenje skupa podataka u oblik odgovarajuć za dalju obradu pomoću biblioteke Keras.

```
import numpy as np

data=np.array(data)/255.0
data=np.reshape(data,(data.shape[0],img_size,img_size,1))
target=np.array(target)


from keras.utils import np_utils

new_target=np_utils.to_categorical(target)


np.save('data',data)
np.save('target', new_target)
```

(Slika 1) Deo koda face_mask_pretprocessing.py

Treniranje modela vrši se u fajlu training.py.

Najpre učitavamo slike I njihove oznake, tj. skupove data[] I target[].

Na sledećoj slici je prikazan deo koda koji realizuje neuronsku mrežu sa dva konvolucijska sloja i dva fully-connected sloja(jedan fullz-connected i soft-max)

-Prvi sloj je konvolucijski sloj sa 200 3x3 filtera, relu aktivacionom funkcijom koju sledi max-pooling.

-Drugi sloj je takođe konvolucijski sloj sa 100 3x3 filtera,relu aktivacionom funkcijom koju sledi max-pooling.

-Nakon konvolucijskih slojeva sledi fully-connected sloj sa 50 neurona u kojima je aktivaciona funkcija takođe reLu

-Izlazni sloj je softmax sloj sa dva izlaza („mask“/“no mask“)

```
model = Sequential()

model.add(Conv2D(200,(3,3),input_shape=data.shape[1:]))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
#kernel_regularizer = keras.regularizers.l1()

model.add(Conv2D(100,(3,3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
#kernel_regularizer = keras.regularizers.l2(0.01)

model.add(Flatten())
model.add(Dropout(0.5))

model.add(Dense(50,activation='relu'))

model.add(Dense(2,activation='softmax'))
#kernel_regularizer = keras.regularizers.l1_l2(0.1)
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics = ['accuracy'])
```

(Slika 2) Realizacija neuronske mreže u training.py

Optimizator učenja je Adam. Metoda model.compile konfiguriše kako će se model učiti.

Za regularizaciju parametara je korišćena Dropout metoda sa 50% odbacivanja po prolasku kroz trening skup. Moguće je dodatno optimizovati mrežu dodavanjem l1, l2 regularizacije.

U ovom fajlu se takođe I deli dataset na test I training skupove, gde test skup sadrži 10% ukupnog broja slika,a training skup ostalih 90%, dev skup (skup za

podešavanje hiperparametara) nije bio potreban, jer su hiperparametri izvučeni iz arhitekture mreže pogodne za ovaj zadatak.

U poslednjoj liniji koda se koristi funkcija model.fit koja uči neuronsku mrežu na trening I test skupu sa 15 epoha :

```
history=model.fit(train_data,train_target,batch_size=1,epochs=15 , callbacks=[checkpoint] , validation_split=0.2)
```

U fajlu detecting.py se naš model koristi na praktičnom primeru.

```
face_clsfr = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
```

Ova linija koda učitava klasifikator pomoću kojeg OpenCV nalazi lica na snimku web kamere.

U ovom fajlu se sa live snimka web kamere iseca frame na kome se traže lica. Za svako pronađeno lice se radi ista obrada kao u face_mask_pretprocessing delu (prebacivanje slike u crno-belu, i promena dimenzija), i zatim se tako obrađena slika funkcijom model.predict daje modelu na procenu da li je na slici lik sa maskom ili ne, u zavisnosti od čega se menja okvir na snimku.

```
while(True):
    ret,img = source.read()
    gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    faces = face_clsfr.detectMultiScale(gray,1.3,5)

    for x,y,w,h in faces:

        face_img = gray[y:y+w, x:x+w]
        resized=cv2.resize(face_img,(100,100))
        normalized = resized/255.0
        reshaped = np.reshape(normalized,(1,100,100,1))
        result = model.predict(reshaped)

        label = np.argmax(result,axis = 1)[0]

        cv2.rectangle(img,(x,y),(x+w, y+h), color_dict[label],2)
        cv2.rectangle(img,(x,y-40), (x+w,y) , color_dict[label], -1)
        cv2.putText(img, labels_dict[label], (x, y-10), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (255,255,255),2)
```

(Slika 3) detekcija lica I prikaz prozora

U fajlu tacnost.py je deo koda koji testira tačnost našeg modela I ona iznosi približno 95.6%. Tačnost se prikazuje I za svaku epohu u toku treniranja, a fajl tacnost.py je tu da ubrza proveru tačnosti.

tacnost.py - C:\Users\Andjela\Desktop\face_mask_detection\tacnost.py (3.6.8)

File Edit Format Run Options Window Help

```
from sklearn.model_selection import train_test_split
import numpy as np
from keras.models import load_model
import cv2
import numpy as np
import time
model = load_model('model-004.model')
data=np.load('data.npy')
target=np.load('target.npy')
train_data,test_data,train_target,test_target=train_test_split(data,target,test_size=0.1)



print(model.evaluate(test_data,test_target))
```

(Slika 4) Kod koji procenjuje tačnost procene modela

```
====== RESTART: C:\Users\Andjela\Desktop\face_mask_detection\tacnost.py ======
1/5 [=====>.....................] - ETA: 1s - loss: 0.1215 - accuracy: 0.9375
2/5 [===========>..................] - ETA: 0s - loss: 0.0732 - accuracy: 0.9688
3/5 [=================>............] - ETA: 0s - loss: 0.0787 -
accuracy: 0.9688
4/5 [======================>...
...] - ETA: 0s - loss: 0.0943 - accuracy: 0.9609
5/5 [==============================] - ETA: 0s - loss: 0.1095 - accuracy: 0.9565
5/5 [==============================] - 1s 264ms/step - loss: 0.1095 - accuracy: 0.9565
[0.1095460057258606, 0.95652174949646]
>>>
```

(Slika 5) Procena tačnosti modela

Literatura

1. Wikipedia https://www.wikipedia.org/
2. geeksForGeeks https://www.geeksforgeeks.org/
3. stackOverFlow https://stackoverflow.com/
4. Coursera https://www.coursera.org/