Classify if the GAN-face is wearing glasses or not, using CNN

SADIQA JAFARI
AD20216006

Table of Contents

Introduction

1. The convolutional neural network (CNN) , like other neural networks, is composed of weighted and biased neural layers with the ability to learn.

2. The difference is in the input.

3. The building blocks of CNNs are filters or kernels.

4. Our purpose is to use CNN to determine if a person is wearing glasses or not.

5. We are using glasses or no glasses dataset which is from a Kaggle project from the course T81-855: Applications of Deep Learning at Washington University.

6. A Generative Adversarial Neural Network or (GAN) created all of the people that are in this dataset.

Related Work

Title	Dataset used	Outcomes
▪ Comparison of Face Classification with Single and Multi-model base on CNN.	WIDER FACE, AFW, MAFA	Detect the face mask-wearing , glasses-wearing and gender.
▪ Assessing the impact of color normalization in convolutional neural network-based nuclei segmentation frameworks.	TCGA-Kumar, TNBC, SMH	We evaluated the impact of color normalization on the complex task of segmenting nuclei for computational pathology application .
▪ An efficient cnn model for covid-19 disease detection based on x-ray image classification	Chest x-ray image	This study has been conducted to demonstrate the effective and accurate diagnosis of covid-19 using cnn which was trained on chest x-ray image dataset.

Dataset

- This dataset includes 5000 images of size 1024 x 1024 x 3 which 2788 of them are the people who wearing glasses and 2212 of them are the people with no glasses.
- Images are sorted into folders according to classes : Images in this dataset are divided into two categories which are “ glasses ” and “no glasses”.
- This dataset includes different genders, skin colors and types of hairs.

- Glasses come in many different forms and we need to make sure we can identify all of them.

Implementation

1. libraries: For implementing this work we need this libraries:

- **Glob** : read dataset .
- **Open-cv**: for image processing.
- **Numpy** : for calculation .
- **Sklearn** : for preprocessing .
- **Tensorflow**: main framework.
- **Matplotlib** : for visualization.

```
import glob
import numpy.core.multiarray
import cv2
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from tensorflow.keras.utils import to_categorical
from tensorflow.keras import models
from tensorflow.keras.layers import Dense, Conv2D, Flatten,  MaxPool2D
import matplotlib.pyplot as plt
```

2. Reading dataset:

- Reading images as features
- Resize images (180 x 192 x 3)

```
12 data = []
13
14 labels = []
15
16 i =0
17 for item in glob.glob("glass/*/*"):
18     i+=1
19
20     img = cv2.imread(item, 0)
21
22     img = cv2.resize(img, (180, 192))
23
24     data.append(img)
25     label = item.split("\\")[1]
26     labels.append(label)
27     if i%100 == 0:
28         print("[INFO]... {}".format(i))
29
```

3. Preprocessing:

- Data split configuration : 80% for training and 20% for testing.

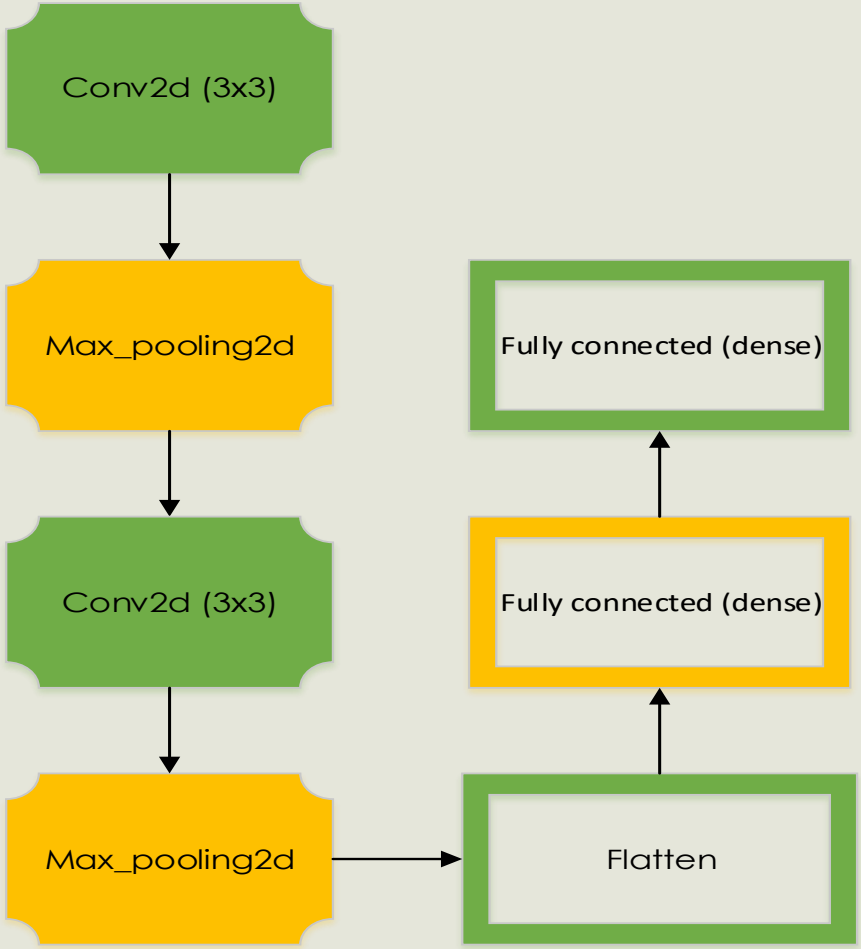
```
33 le = LabelEncoder()
34 labels = le.fit_transform(labels)
35 labels = to_categorical(labels)
36
37
38
39 data = np.array(data)/255.0
40 xtrain, xtest, ytrain, ytest = train_test_split(data,
41                                                 labels,
42                                                 test_size = 0.2)
43 xtrain = xtrain.reshape(4000, 180, 192, 1)
44 xtest = xtest.reshape(1000, 180, 192, 1)
45
```

Model

- Model summary:

Layer(type)	Filters	Kernel size	Activation function	Output shape	Param
Conv2d (Conv2D)	32	(3x3)	Relu	(None, 178, 190, 32)	320
Max_pooling2d (MaxPooling2D)	-	-	-	(None, 89, 95, 32)	0
Conv2d_1 (Conv2D)	64	(3x3)	Relu	(None, 87, 93, 64)	18496
Max_pooling2d (MaxPooling2D)	-	-	-	(None, 43, 46, 64)	0
Conv2d_1 (Conv2D)	128	(3x3)	Relu	(None, 41, 44, 128)	73856
Max_pooling2d (MaxPooling2D)	-	-	-	(None, 20, 22, 128)	0
Flatten (Flatten)	-	-	-	(None,56320)	0
Dense(Dense)	64	-	Relu	(None,64)	3604544
Dense-1(Dense)	2	-	Softmax	(None,2)	130

- layer architecture:



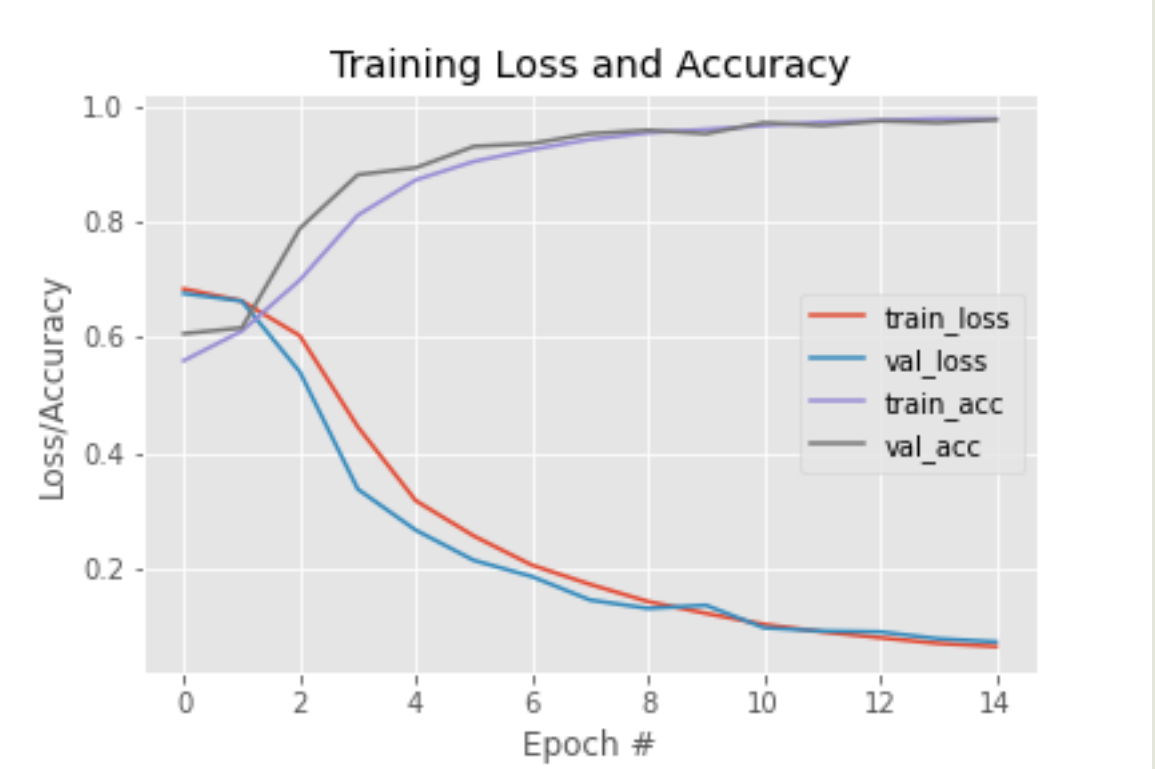
- Training model:

Implementation of our model using sequential , stochastic gradient descent and binary cross entropy then run our model using 15 epochs with batch size 16.

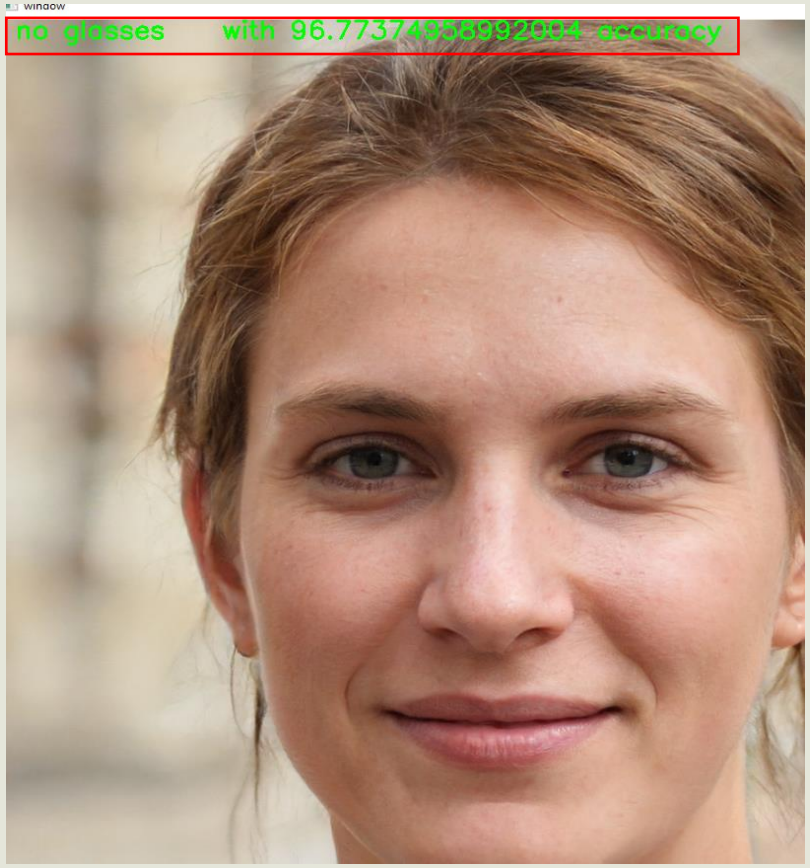
```
47
48 model = models.Sequential([
49                             Conv2D( filters= 32, kernel_size = (3, 3), activation = "relu", input_shape =(180, 192, 1)),
50                             MaxPool2D(),
51                             Conv2D(filters = 64, kernel_size = (3, 3), activation = "relu"),
52                             MaxPool2D(),
53                             Conv2D(filters = 128, kernel_size = (3, 3), activation= "relu"),
54                             MaxPool2D(),
55                             Flatten(),
56                             Dense(units = 64, activation = "relu"),
57                             Dense(units = 2, activation = "softmax")
58                             ])
59
60 model.compile(optimizer = "sgd",
61               loss = "binary_crossentropy",
62               metrics = ["accuracy"])
63
64 H = model.fit(xtrain, ytrain, batch_size = 16, epochs = 15,
65               validation_data = (xtest, ytest))
66
```

Result

- Training loss and accuracy.



Example:



Conclusion:

- We have done an image classification using CNN with 96% accuracy.
- Which means we can classify if a face is wearing glasses or not with 96% accuracy.

Reference

https://www.kaggle.com/jeffheaton/glasses-or-no-glasses

https://ieeexplore.ieee.org/abstract/document/9376825

https://www.hindawi.com/journals/complexity/2021/6621607/

https://www.frontiersin.org/articles/10.3389/fbioe.2019.00300/full

Thank You