FacialRecognition

January 4, 2021

1 Facial Recognition using CNN

[1]:

```
import numpy as np
import pandas as pd
from skimage import io
from keras.utils import to_categorical
import matplotlib.pyplot as plt
%matplotlib inline
```

```
Using TensorFlow backend.
```

1.1 Data Pre-processing

[2]:

```
dt=np.load('ORL_faces.npz',allow_pickle=True)
trainX=dt["trainX"].reshape(240,112,92)
trainY=dt["trainY"]
testX=dt["testX"].reshape(160,112,92)
testY=dt["testY"]
```

[3]:

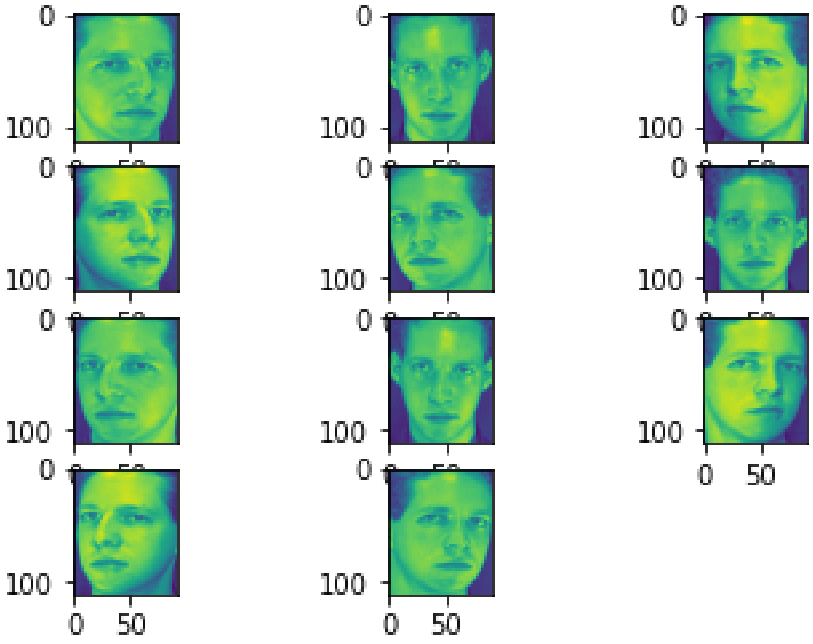
```
trainX=trainX.reshape(-1,112,92,1)
testX=testX.reshape(-1,112,92,1)

#Normalizing the data
trainX=trainX.astype('float32')/225
testX=testX.astype('float32')/255

#encoding the targets to categorical value
ohe_trainY=to_categorical(trainY)
ohe_testY=to_categorical(testY)
```

[100]:

```
for i in range(1,12):
    plt.subplot(4,3,i)
    plt.imshow(trainX[i])
```



[80]:

```
print(trainX.shape,ohe_trainY.shape,testX.shape,ohe_testY.shape)
```

```
(240, 112, 92, 1) (240, 20) (160, 112, 92, 1) (160, 20)
```

1.2 Model Architecting

[4]:

```
import keras
from keras.losses import categorical_crossentropy
from keras.optimizers import Adam
from keras.models import Sequential
from keras.layers import Dense,Dropout,Flatten
from keras.layers import Conv2D,MaxPooling2D
from keras.layers.normalization import BatchNormalization
from keras.layers.advanced_activations import LeakyReLU
```

[5]:

```
model=Sequential()
model.
 ↪add(Conv2D(32,kernel_size=(3,3),activation="linear",input_shape=(112,92,1),padding='same'))
model.add(LeakyReLU(alpha=0.05))
model.add(MaxPooling2D((2,2),padding='same'))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(128,activation='linear'))
model.add(LeakyReLU(alpha=0.05))
model.add(Dropout(0.5))

model.add(Dense(20,activation='softmax'))
model.
 ↪compile(loss='categorical_crossentropy',optimizer=Adam(),metrics=['accuracy'])
model.summary()
model_train=model.
 ↪fit(trainX,ohe_trainY,batch_size=128,epochs=50,verbose=1,validation_data=(testX,ohe_testY))
```

```
Model: "sequential_1"
-----------------------------------------------------------------
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 112, 92, 32)       320
-----------------------------------------------------------------
leaky_re_lu_1 (LeakyReLU)    (None, 112, 92, 32)       0
-----------------------------------------------------------------
max_pooling2d_1 (MaxPooling2 (None, 56, 46, 32)        0
-----------------------------------------------------------------
dropout_1 (Dropout)          (None, 56, 46, 32)        0
-----------------------------------------------------------------
flatten_1 (Flatten)          (None, 82432)             0
-----------------------------------------------------------------
dense_1 (Dense)              (None, 128)               10551424
-----------------------------------------------------------------
leaky_re_lu_2 (LeakyReLU)    (None, 128)               0
-----------------------------------------------------------------
dropout_2 (Dropout)          (None, 128)               0
-----------------------------------------------------------------
dense_2 (Dense)              (None, 20)                2580
=================================================================
Total params: 10,554,324
Trainable params: 10,554,324
Non-trainable params: 0
-----------------------------------------------------------------
Train on 240 samples, validate on 160 samples
Epoch 1/50
240/240 [==============================] - 2s 10ms/step - loss: 7.6039 -
accuracy: 0.0625 - val_loss: 5.4096 - val_accuracy: 0.0500
Epoch 2/50
240/240 [==============================] - 2s 10ms/step - loss: 8.1429 -
accuracy: 0.0542 - val_loss: 3.4554 - val_accuracy: 0.0500
Epoch 3/50
240/240 [==============================] - 2s 10ms/step - loss: 4.5085 -
accuracy: 0.0833 - val_loss: 2.9383 - val_accuracy: 0.1125
Epoch 4/50
```

```
240/240 [==============================] - 2s 10ms/step - loss: 3.2116 -
accuracy: 0.1417 - val_loss: 2.9717 - val_accuracy: 0.1125
Epoch 5/50
240/240 [==============================] - 2s 10ms/step - loss: 3.1104 -
accuracy: 0.1042 - val_loss: 2.8462 - val_accuracy: 0.1937
Epoch 6/50
240/240 [==============================] - 2s 10ms/step - loss: 3.0706 -
accuracy: 0.0958 - val_loss: 2.6564 - val_accuracy: 0.2562
Epoch 7/50
240/240 [==============================] - 2s 10ms/step - loss: 2.8574 -
accuracy: 0.1750 - val_loss: 2.5908 - val_accuracy: 0.2438
Epoch 8/50
240/240 [==============================] - 2s 10ms/step - loss: 2.6832 -
accuracy: 0.1708 - val_loss: 2.4722 - val_accuracy: 0.3438
Epoch 9/50
240/240 [==============================] - 2s 10ms/step - loss: 2.5729 -
accuracy: 0.1750 - val_loss: 2.3625 - val_accuracy: 0.2937
Epoch 10/50
240/240 [==============================] - 2s 10ms/step - loss: 2.6275 -
accuracy: 0.1583 - val_loss: 2.3338 - val_accuracy: 0.3500
Epoch 11/50
240/240 [==============================] - 2s 10ms/step - loss: 2.3134 -
accuracy: 0.2625 - val_loss: 2.2884 - val_accuracy: 0.2375
Epoch 12/50
240/240 [==============================] - 2s 10ms/step - loss: 2.3009 -
accuracy: 0.2583 - val_loss: 2.1267 - val_accuracy: 0.5437
Epoch 13/50
240/240 [==============================] - 2s 10ms/step - loss: 2.1500 -
accuracy: 0.3167 - val_loss: 2.0365 - val_accuracy: 0.5125
Epoch 14/50
240/240 [==============================] - 2s 10ms/step - loss: 2.0901 -
accuracy: 0.3292 - val_loss: 1.8760 - val_accuracy: 0.7063
Epoch 15/50
240/240 [==============================] - 2s 10ms/step - loss: 1.8609 -
accuracy: 0.4083 - val_loss: 1.8308 - val_accuracy: 0.7188
Epoch 16/50
240/240 [==============================] - 2s 10ms/step - loss: 1.7804 -
accuracy: 0.4292 - val_loss: 1.6782 - val_accuracy: 0.7063
Epoch 17/50
240/240 [==============================] - 2s 10ms/step - loss: 1.6483 -
accuracy: 0.4750 - val_loss: 1.5921 - val_accuracy: 0.6313
Epoch 18/50
240/240 [==============================] - 2s 10ms/step - loss: 1.5644 -
accuracy: 0.4958 - val_loss: 1.3564 - val_accuracy: 0.6125
Epoch 19/50
240/240 [==============================] - 2s 10ms/step - loss: 1.2800 -
accuracy: 0.6292 - val_loss: 1.2799 - val_accuracy: 0.8750
Epoch 20/50
```

```
240/240 [==============================] - 2s 10ms/step - loss: 1.2190 -
accuracy: 0.6208 - val_loss: 1.1918 - val_accuracy: 0.9062
Epoch 21/50
240/240 [==============================] - 2s 10ms/step - loss: 1.1799 -
accuracy: 0.6625 - val_loss: 1.0644 - val_accuracy: 0.9187
Epoch 22/50
240/240 [==============================] - 2s 10ms/step - loss: 0.9595 -
accuracy: 0.7417 - val_loss: 0.9392 - val_accuracy: 0.9250
Epoch 23/50
240/240 [==============================] - 2s 10ms/step - loss: 1.0413 -
accuracy: 0.6708 - val_loss: 0.8458 - val_accuracy: 0.9125
Epoch 24/50
240/240 [==============================] - 2s 10ms/step - loss: 0.8513 -
accuracy: 0.7542 - val_loss: 0.8066 - val_accuracy: 0.9187
Epoch 25/50
240/240 [==============================] - 2s 10ms/step - loss: 0.7066 -
accuracy: 0.7792 - val_loss: 0.7795 - val_accuracy: 0.9000
Epoch 26/50
240/240 [==============================] - 2s 10ms/step - loss: 0.6831 -
accuracy: 0.7833 - val_loss: 0.6447 - val_accuracy: 0.9125
Epoch 27/50
240/240 [==============================] - 2s 10ms/step - loss: 0.5509 -
accuracy: 0.8292 - val_loss: 0.5895 - val_accuracy: 0.9250
Epoch 28/50
240/240 [==============================] - 2s 10ms/step - loss: 0.5535 -
accuracy: 0.8458 - val_loss: 0.5096 - val_accuracy: 0.9375
Epoch 29/50
240/240 [==============================] - 2s 10ms/step - loss: 0.4282 -
accuracy: 0.8625 - val_loss: 0.5721 - val_accuracy: 0.9250
Epoch 30/50
240/240 [==============================] - 2s 10ms/step - loss: 0.4993 -
accuracy: 0.8750 - val_loss: 0.5776 - val_accuracy: 0.9500
Epoch 31/50
240/240 [==============================] - 2s 10ms/step - loss: 0.3430 -
accuracy: 0.9250 - val_loss: 0.5679 - val_accuracy: 0.9312
Epoch 32/50
240/240 [==============================] - 2s 10ms/step - loss: 0.4079 -
accuracy: 0.8917 - val_loss: 0.5503 - val_accuracy: 0.9187
Epoch 33/50
240/240 [==============================] - 2s 10ms/step - loss: 0.3459 -
accuracy: 0.8875 - val_loss: 0.4246 - val_accuracy: 0.9125
Epoch 34/50
240/240 [==============================] - 2s 10ms/step - loss: 0.2627 -
accuracy: 0.9167 - val_loss: 0.4148 - val_accuracy: 0.9250
Epoch 35/50
240/240 [==============================] - 2s 10ms/step - loss: 0.2964 -
accuracy: 0.9333 - val_loss: 0.4261 - val_accuracy: 0.9062
Epoch 36/50
```

```
240/240 [==============================] - 2s 10ms/step - loss: 0.2420 -
accuracy: 0.9333 - val_loss: 0.3981 - val_accuracy: 0.9187
Epoch 37/50
240/240 [==============================] - 2s 10ms/step - loss: 0.2546 -
accuracy: 0.9250 - val_loss: 0.3636 - val_accuracy: 0.9625
Epoch 38/50
240/240 [==============================] - 2s 10ms/step - loss: 0.1555 -
accuracy: 0.9792 - val_loss: 0.3488 - val_accuracy: 0.9625
Epoch 39/50
240/240 [==============================] - 2s 10ms/step - loss: 0.1837 -
accuracy: 0.9667 - val_loss: 0.3377 - val_accuracy: 0.9438
Epoch 40/50
240/240 [==============================] - 2s 10ms/step - loss: 0.1760 -
accuracy: 0.9625 - val_loss: 0.3344 - val_accuracy: 0.9500
Epoch 41/50
240/240 [==============================] - 2s 10ms/step - loss: 0.1369 -
accuracy: 0.9625 - val_loss: 0.3438 - val_accuracy: 0.9375
Epoch 42/50
240/240 [==============================] - 2s 10ms/step - loss: 0.1678 -
accuracy: 0.9708 - val_loss: 0.3335 - val_accuracy: 0.9500
Epoch 43/50
240/240 [==============================] - 2s 10ms/step - loss: 0.1388 -
accuracy: 0.9667 - val_loss: 0.3441 - val_accuracy: 0.9375
Epoch 44/50
240/240 [==============================] - 2s 10ms/step - loss: 0.1328 -
accuracy: 0.9875 - val_loss: 0.3293 - val_accuracy: 0.9625
Epoch 45/50
240/240 [==============================] - 2s 10ms/step - loss: 0.1030 -
accuracy: 0.9792 - val_loss: 0.3175 - val_accuracy: 0.9563
Epoch 46/50
240/240 [==============================] - 2s 10ms/step - loss: 0.1355 -
accuracy: 0.9583 - val_loss: 0.3218 - val_accuracy: 0.9375
Epoch 47/50
240/240 [==============================] - 2s 10ms/step - loss: 0.1046 -
accuracy: 0.9875 - val_loss: 0.3243 - val_accuracy: 0.9312
Epoch 48/50
240/240 [==============================] - 2s 10ms/step - loss: 0.1051 -
accuracy: 0.9708 - val_loss: 0.3309 - val_accuracy: 0.9500
Epoch 49/50
240/240 [==============================] - 2s 10ms/step - loss: 0.0886 -
accuracy: 0.9917 - val_loss: 0.3356 - val_accuracy: 0.9312
Epoch 50/50
240/240 [==============================] - 2s 10ms/step - loss: 0.1211 -
accuracy: 0.9667 - val_loss: 0.3089 - val_accuracy: 0.9500
```

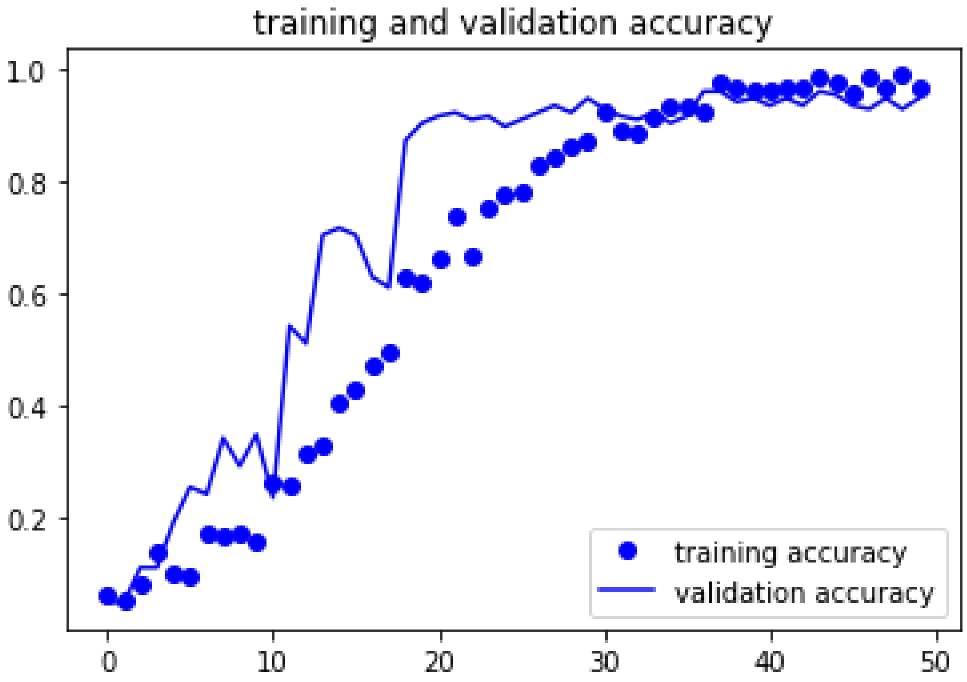
[8]:

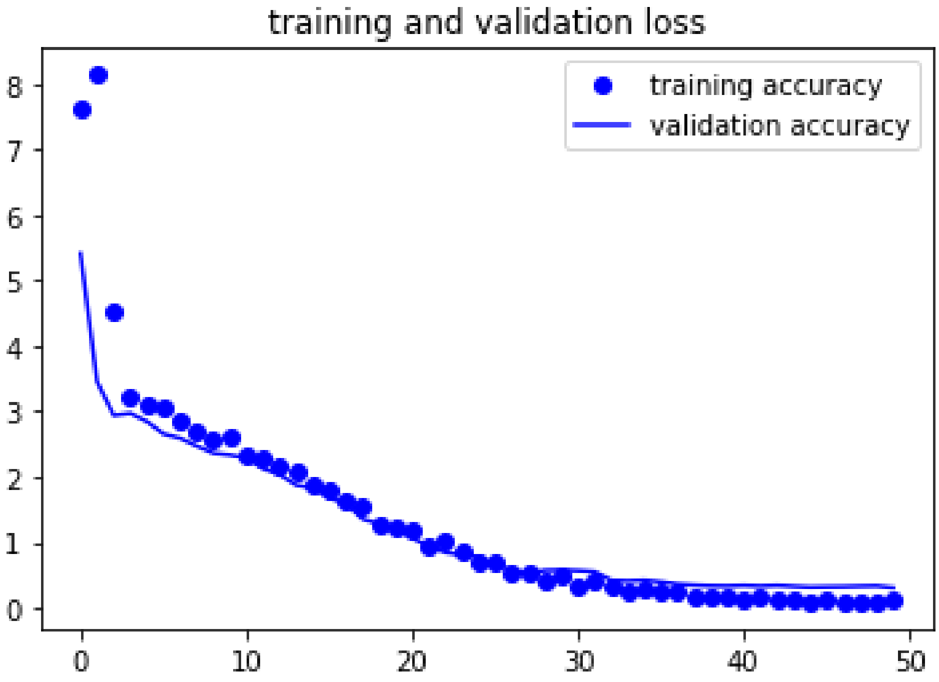
```
# Model Evaluation
test_eval=model.evaluate(testX,ohe_testY,verbose=1)
```

```
print("test loss : ",test_eval[0])
print("test accuracy : ",test_eval[1])
accuracy=model_train.history['accuracy']
val_accuracy=model_train.history['val_accuracy']
loss=model_train.history['loss']
val_loss=model_train.history['val_loss']
epochs=range(len(accuracy))
plt.plot(epochs,accuracy,'bo',label='training accuracy')
plt.plot(epochs,val_accuracy,'b',label='validation accuracy')
plt.title('training and validation accuracy')
plt.legend()
plt.figure()
plt.plot(epochs,loss,'bo',label='training accuracy')
plt.plot(epochs,val_loss,'b',label='validation accuracy')
plt.title('training and validation loss')
plt.legend()
plt.figure()
```

```
160/160 [==============================] - 0s 2ms/step
test loss :  0.3089377969503403
test accuracy :  0.949999988079071
```

```
[8]: <Figure size 432x288 with 0 Axes>
```




training and validation loss
training accuracy
validation accuracy
8
7
6
5
4
3
2
1
0
0
10
20
30
40
50

```
<Figure size 432x288 with 0 Axes>
```