Eye detection

November 20, 2023

[35]:

```python
import numpy as np
import os
import matplotlib.pylab as plt
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.preprocessing import image
from keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential, load_model
from tensorflow.keras.layers import Dense, Conv2D, MaxPooling2D, Flatten
from keras.callbacks import ModelCheckpoint

from sklearn.metrics import f1_score
from sklearn.metrics import confusion_matrix, classification_report
import seaborn as sns
```

[23]:

```python
train_path="data_eyes_closure/train"
test_path="data_eyes_closure/test"

train_datagen = ImageDataGenerator (rescale= 1./255, shear_range= 0.2,␣
 ↪zoom_range= 0.2, horizontal_flip=True)
test_datagen = ImageDataGenerator (rescale= 1./255)

x_train=train_datagen.flow_from_directory(train_path, target_size =␣
 ↪(256,256),batch_size = 128, color_mode = 'grayscale',class_mode =␣
 ↪'categorical')
x_test=train_datagen.flow_from_directory(test_path,target_size =␣
 ↪(256,256),batch_size = 128, color_mode = 'grayscale',class_mode =␣
 ↪'categorical')

x_train.class_indices
```

```
Found 1234 images belonging to 2 classes.
Found 218 images belonging to 2 classes.
```

[23]:

```
{'Closed': 0, 'Open': 1}
```

[24]:

```python
classes=2
model = Sequential()
```

```python
model.add(Conv2D(32, (3,3), padding = 'same', input_shape = (256,256,1),␣
 ↪activation = 'relu'))
model.add(MaxPooling2D(pool_size = (2,2)))

model.add(Conv2D(64, (3,3), padding = 'same', activation = 'relu'))
model.add(MaxPooling2D(pool_size = (2,2)))

model.add(Conv2D(128,(3,3), padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Flatten())

model.add(Dense(64, activation = 'relu'))

model.add(Dense(classes, activation = 'softmax'))
print(model.summary())
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_3 (Conv2D)           (None, 256, 256, 32)      320

 max_pooling2d_3 (MaxPoolin  (None, 128, 128, 32)      0
 g2D)

 conv2d_4 (Conv2D)           (None, 128, 128, 64)      18496

 max_pooling2d_4 (MaxPoolin  (None, 64, 64, 64)        0
 g2D)

 conv2d_5 (Conv2D)           (None, 64, 64, 128)       73856

 max_pooling2d_5 (MaxPoolin  (None, 32, 32, 128)       0
 g2D)

 flatten_1 (Flatten)         (None, 131072)            0

 dense_2 (Dense)             (None, 64)                8388672

 dense_3 (Dense)             (None, 2)                 130

=================================================================
Total params: 8481474 (32.35 MB)
Trainable params: 8481474 (32.35 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
None
```

[25]:

```
model.compile(loss = 'categorical_crossentropy',optimizer = 'adam' , metrics =␣
 ↪['accuracy'])
```

[26]:

```
model_path="eye_detection.h5"

checkpoint = ModelCheckpoint(model_path, monitor='val_accuracy', verbose=1,
                              save_best_only=True, mode='max')

callbacks_list = [checkpoint]
```

[27]:

```
num_epochs = 20
training_steps=x_train.n//x_train.batch_size
validation_steps =x_test.n//x_test.batch_size

history = model.fit_generator(x_train, epochs=num_epochs,␣
 ↪steps_per_epoch=training_steps,validation_data=x_test,
                    validation_steps=validation_steps, callbacks =␣
 ↪callbacks_list)
```

```
C:\Users\ajars\AppData\Local\Temp\ipykernel_12804\87117149.py:5: UserWarning:
`Model.fit_generator` is deprecated and will be removed in a future version.
Please use `Model.fit`, which supports generators.
  history = model.fit_generator(x_train, epochs=num_epochs,
steps_per_epoch=training_steps,validation_data=x_test,

Epoch 1/20
9/9 [==============================] - ETA: 0s - loss: 1.1552 - accuracy: 0.5741
Epoch 1: val_accuracy improved from -inf to 0.51562, saving model to
eye_detection.h5
9/9 [==============================] - 47s 5s/step - loss: 1.1552 - accuracy:
0.5741 - val_loss: 0.6810 - val_accuracy: 0.5156
Epoch 2/20
9/9 [==============================] - ETA: 0s - loss: 0.6333 - accuracy: 0.7098
Epoch 2: val_accuracy improved from 0.51562 to 0.89844, saving model to
eye_detection.h5
9/9 [==============================] - 42s 5s/step - loss: 0.6333 - accuracy:
0.7098 - val_loss: 0.4861 - val_accuracy: 0.8984
Epoch 3/20
9/9 [==============================] - ETA: 0s - loss: 0.3825 - accuracy: 0.8644
Epoch 3: val_accuracy improved from 0.89844 to 0.92969, saving model to
eye_detection.h5
9/9 [==============================] - 41s 5s/step - loss: 0.3825 - accuracy:
0.8644 - val_loss: 0.2423 - val_accuracy: 0.9297
Epoch 4/20
9/9 [==============================] - ETA: 0s - loss: 0.2992 - accuracy: 0.8924
Epoch 4: val_accuracy did not improve from 0.92969
9/9 [==============================] - 44s 5s/step - loss: 0.2992 - accuracy:
0.8924 - val_loss: 0.2611 - val_accuracy: 0.9141
```

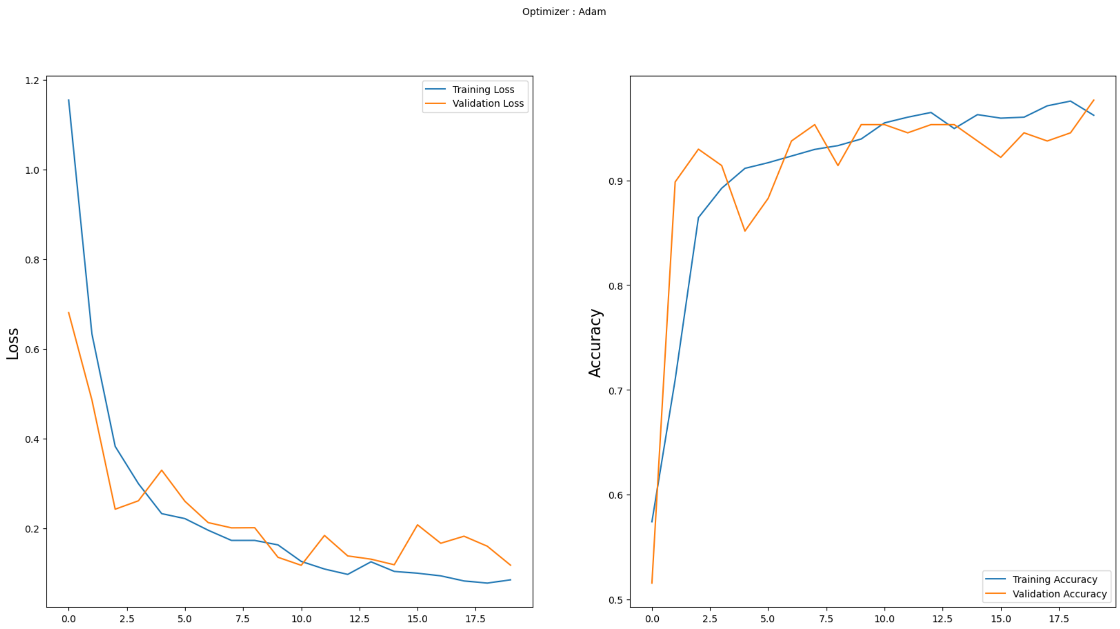
```
Epoch 5/20
9/9 [==============================] - ETA: 0s - loss: 0.2324 - accuracy: 0.9114
Epoch 5: val_accuracy did not improve from 0.92969
9/9 [==============================] - 42s 5s/step - loss: 0.2324 - accuracy:
0.9114 - val_loss: 0.3293 - val_accuracy: 0.8516
Epoch 6/20
9/9 [==============================] - ETA: 0s - loss: 0.2213 - accuracy: 0.9168
Epoch 6: val_accuracy did not improve from 0.92969
9/9 [==============================] - 44s 5s/step - loss: 0.2213 - accuracy:
0.9168 - val_loss: 0.2600 - val_accuracy: 0.8828
Epoch 7/20
9/9 [==============================] - ETA: 0s - loss: 0.1954 - accuracy: 0.9231
Epoch 7: val_accuracy improved from 0.92969 to 0.93750, saving model to
eye_detection.h5
9/9 [==============================] - 44s 5s/step - loss: 0.1954 - accuracy:
0.9231 - val_loss: 0.2124 - val_accuracy: 0.9375
Epoch 8/20
9/9 [==============================] - ETA: 0s - loss: 0.1726 - accuracy: 0.9295
Epoch 8: val_accuracy improved from 0.93750 to 0.95312, saving model to
eye_detection.h5
9/9 [==============================] - 45s 5s/step - loss: 0.1726 - accuracy:
0.9295 - val_loss: 0.2007 - val_accuracy: 0.9531
Epoch 9/20
9/9 [==============================] - ETA: 0s - loss: 0.1727 - accuracy: 0.9331
Epoch 9: val_accuracy did not improve from 0.95312
9/9 [==============================] - 49s 6s/step - loss: 0.1727 - accuracy:
0.9331 - val_loss: 0.2010 - val_accuracy: 0.9141
Epoch 10/20
9/9 [==============================] - ETA: 0s - loss: 0.1627 - accuracy: 0.9394
Epoch 10: val_accuracy did not improve from 0.95312
9/9 [==============================] - 47s 5s/step - loss: 0.1627 - accuracy:
0.9394 - val_loss: 0.1350 - val_accuracy: 0.9531
Epoch 11/20
9/9 [==============================] - ETA: 0s - loss: 0.1258 - accuracy: 0.9548
Epoch 11: val_accuracy did not improve from 0.95312
9/9 [==============================] - 46s 5s/step - loss: 0.1258 - accuracy:
0.9548 - val_loss: 0.1172 - val_accuracy: 0.9531
Epoch 12/20
9/9 [==============================] - ETA: 0s - loss: 0.1088 - accuracy: 0.9602
Epoch 12: val_accuracy did not improve from 0.95312
9/9 [==============================] - 47s 5s/step - loss: 0.1088 - accuracy:
0.9602 - val_loss: 0.1837 - val_accuracy: 0.9453
Epoch 13/20
9/9 [==============================] - ETA: 0s - loss: 0.0969 - accuracy: 0.9647
Epoch 13: val_accuracy did not improve from 0.95312
9/9 [==============================] - 45s 5s/step - loss: 0.0969 - accuracy:
0.9647 - val_loss: 0.1381 - val_accuracy: 0.9531
Epoch 14/20
```

```
9/9 [==============================] - ETA: 0s - loss: 0.1250 - accuracy: 0.9494
Epoch 14: val_accuracy did not improve from 0.95312
9/9 [==============================] - 46s 5s/step - loss: 0.1250 - accuracy:
0.9494 - val_loss: 0.1308 - val_accuracy: 0.9531
Epoch 15/20
9/9 [==============================] - ETA: 0s - loss: 0.1035 - accuracy: 0.9627
Epoch 15: val_accuracy did not improve from 0.95312
9/9 [==============================] - 46s 5s/step - loss: 0.1035 - accuracy:
0.9627 - val_loss: 0.1184 - val_accuracy: 0.9375
Epoch 16/20
9/9 [==============================] - ETA: 0s - loss: 0.0995 - accuracy: 0.9593
Epoch 16: val_accuracy did not improve from 0.95312
9/9 [==============================] - 47s 5s/step - loss: 0.0995 - accuracy:
0.9593 - val_loss: 0.2074 - val_accuracy: 0.9219
Epoch 17/20
9/9 [==============================] - ETA: 0s - loss: 0.0936 - accuracy: 0.9602
Epoch 17: val_accuracy did not improve from 0.95312
9/9 [==============================] - 46s 5s/step - loss: 0.0936 - accuracy:
0.9602 - val_loss: 0.1663 - val_accuracy: 0.9453
Epoch 18/20
9/9 [==============================] - ETA: 0s - loss: 0.0822 - accuracy: 0.9711
Epoch 18: val_accuracy did not improve from 0.95312
9/9 [==============================] - 48s 5s/step - loss: 0.0822 - accuracy:
0.9711 - val_loss: 0.1820 - val_accuracy: 0.9375
Epoch 19/20
9/9 [==============================] - ETA: 0s - loss: 0.0775 - accuracy: 0.9756
Epoch 19: val_accuracy did not improve from 0.95312
9/9 [==============================] - 48s 5s/step - loss: 0.0775 - accuracy:
0.9756 - val_loss: 0.1598 - val_accuracy: 0.9453
Epoch 20/20
9/9 [==============================] - ETA: 0s - loss: 0.0847 - accuracy: 0.9620
Epoch 20: val_accuracy improved from 0.95312 to 0.97656, saving model to
eye_detection.h5
9/9 [==============================] - 45s 5s/step - loss: 0.0847 - accuracy:
0.9620 - val_loss: 0.1174 - val_accuracy: 0.9766
```

[28]:

```
plt.figure(figsize=(20,10))
plt.subplot(1, 2, 1)
plt.suptitle('Optimizer : Adam', fontsize=10)
plt.ylabel('Loss', fontsize=16)
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.legend(loc='upper right')

plt.subplot(1, 2, 2)
plt.ylabel('Accuracy', fontsize=16)
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.legend(loc='lower right')
plt.show()
```



[32]:

```
img=image.load_img(r"C:\Users\ajars\Downloads\photo_2023-11-20_00-10-25.jpg",␣
 ↪target_size= (256,256), color_mode="grayscale")
x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)
```

[33]:

```
pred=np.argmax(model.predict(x))
pred
```

```
1/1 [==============================] - 0s 30ms/step
```

[33]: 0

[34]:

```
if pred == 0:
    print("closed eye")
else:
    print("open eye")
```

```
closed eye
```

[ ]:

```

```