A.I

LAB TASK 11

Name: Syed Muzzamil Waseem

SID: 11067

CID: 110089

QUESTION

Implement Neural Network for Malaria prediction.

Note: Data-set is available on Kaggle.

1 fit\_model **=** model.fit( train\_set, validation\_data**=**test\_set, epochs**=**10, steps\_per\_epoch**=**



In [35]:

|  |  |
| --- | --- |
| 1 | model **=** VGG19(input\_shape**=**(224,224,3), weights**=**'imagenet', include\_top**=False**) |

Downloading data from https://storage.googleapis.com/tensorflow/keras-applic ations/vgg19/vgg19\_weights\_tf\_dim\_ordering\_tf\_kernels\_notop.h5 (https://stor age.googleapis.com/tensorflow/keras-applications/vgg19/vgg19\_weights\_tf\_dim\_ ordering\_tf\_kernels\_notop.h5)

|  |  |
| --- | --- |
| 1 | **for** layer **in** model.layers: |
| 2 | layer.trainable **= False** |

80142336/80134624 [==============================] - 1s 0us/step

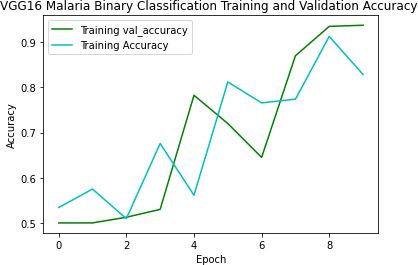
In [37]:

|  |  |
| --- | --- |
| 1 | class\_names **=** os.listdir('/content/drive/MyDrive/final\_project/dataset/testin)g' |
| 2 | class\_names |

Out[37]:

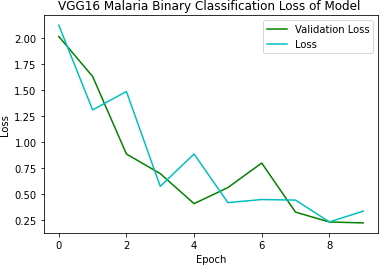
|  |  |  |
| --- | --- | --- |
| 80150528/80134624 [==============================] - 1s 0us/step  In [36]: |  | ['Fake', 'Live'] |
| In [41]: | [42]: |  |
| Epoch 1/10 |  |  |
| 2/2 [==============================] - 61s 41s/step - loss: 2.1255 - accurac y: 0.5341 - |  |  |
| val\_loss: 2.0176 - val\_accuracy: 0.5000 |  |  |
| Epoch 2/10 |  |  |
|  |  |  |
| 2/2 [==============================] - 9s 7s/step - loss: 1.3096 - accuracy: |  |  |
| 0.5749 - val\_loss: 1.6321 - val\_accuracy: 0.5000 |  |  |
| Epoch 3/10 |  |  |
| 2/2 [==============================] - 10s 6s/step - loss: 1.4870 - accurac |  |  |
| y: 0.5095 - val\_loss: 0.8843 - val\_accuracy: 0.5125 |  |  |
| Epoch 4/10 |  |  |
| 2/2 [==============================] - 10s 7s/step - loss: 0.5732 - accurac |  |  |
| y: 0.6757 - val\_loss: 0.6957 - val\_accuracy: 0.5300 |  |  |
| Epoch 5/10 |  |  |
| 2/2 [==============================] - 10s 7s/step - loss: 0.8847 - accurac y: 0.5613 - |  |  |
| val\_loss: 0.4059 - val\_accuracy: 0.7825 |  |  |
| Epoch 6/10 |  |  |
| 2/2 [==============================] - 10s 6s/step - loss: 0.4160 - accurac |  |  |
| y: 0.8120 - val\_loss: 0.5609 - val\_accuracy: 0.7200 |  |  |
| Epoch 7/10 |  |  |
| 2/2 [==============================] - 10s 6s/step - loss: 0.4456 - accurac |  |  |
| y: 0.7657 - val\_loss: 0.7975 - val\_accuracy: 0.6450 |  |  |
| Epoch 8/10 |  |  |
| 2/2 [==============================] - 10s 6s/step - loss: 0.4403 - accurac y: 0.7738 - |  | In [43]: |
| val\_loss: 0.3242 - val\_accuracy: 0.8700 |  |  |
| Epoch 9/10 |  |  |
| 2/2 [==============================] - 10s 7s/step - loss: 0.2299 - accurac |  |  |
|  |  |  |
|  |  |  |
| Epoch 10/10 |  |  |
| 2/2 [==============================] - 10s 6s/step - loss: 0.3333 - accurac y: 0.8283 - |  |  |
| val\_loss: 0.2203 - val\_accuracy: 0.9375 |  |  |

|  |  |
| --- | --- |
| 1 | plt.plot(fit\_model.history['val\_accuracy'],label**=**'Training val\_accuracy',color**=**'green' |
| 2 | plt.plot(fit\_model.history['accuracy'],label**=**'Training Accuracy ',color**=**'c') |
| 3 | plt.title('VGG16 Malaria Binary Classification Training and Validation Accurac)y ' |
| 4 | plt.xlabel('Epoch') |
| 5 | plt.ylabel('Accuracy') |
| 6 | plt.legend() |
| 7 | plt.show() |



y: 0.9128 - val\_loss: 0.2294 - val\_accuracy: 0.9350

|  |  |
| --- | --- |
| 1 | plt.plot(fit\_model.history['val\_loss'],label**=**'Validation Loss',color**=**'green') |
| 2 | plt.plot(fit\_model.history['loss'],label**=**'Loss ',color**=**'c') |
| 3 | plt.title('VGG16 Malaria Binary Classification Loss of Model) ' |
| 4 | plt.xlabel('Epoch') |
| 5 | plt.ylabel('Loss') |
| 6 | plt.legend() |
| 7 | plt.show() |



In [44]:

|  |  |
| --- | --- |
| 1 | **from** tensorflow.keras.models **import** load\_model |
| 2 |  |
| 3 | model.save('Resnet50.h5') |

In [45]:

In [56]:

|  |  |
| --- | --- |
| 1  2  3  4 | **if**(a**==**1):  print("infected")  **else**:  print("Uninfected") |

|  |  |
| --- | --- |
| 1 | y\_pred **=** model.predict(test\_set) |

In [46]:

|  |  |
| --- | --- |
| 1 | **import** numpy **as** np |
| 2 | y\_pred **=** np.argmax(y\_pred, axis**=**1) |

In [47]:

|  |  |
| --- | --- |
| 1 | **from** tensorflow.keras.models **import** load\_model |
| 2 | **from** tensorflow.keras.preprocessing **import** image |

In [48]:

Uninfected In [ ]:

|  |  |
| --- | --- |
| 1 |  |

|  |  |
| --- | --- |
| 1 | model\_resnet**=**load\_model('/content/Resnet50.h5)' |

In [49]:

|  |  |
| --- | --- |
| 1  2 | img**=**image.load\_img('/content/drive/MyDrive/final\_project/dataset/testing/Fake/100\_1 |

In [50]:

|  |  |
| --- | --- |
| 1 | x**=**image.img\_to\_array(img) |

In [51]:

|  |  |
| --- | --- |
| 1 | x**=**np.expand\_dims(x,axis**=**0) |
| 2 | img\_data**=**preprocess\_input(x) |
| 3 | img\_data.shape |

Out[51]:

(1, 224, 224, 3)

In [52]:

|  |  |
| --- | --- |
| 1 | x**=**x**/**255 |

In [53]:

|  |  |
| --- | --- |
| 1 | model\_resnet.predict(img\_data) |

Out[53]:

array([[1.0000000e+00, 1.6674012e-27]],

dtype=float32) In [54]:

|  |  |
| --- | --- |
| 1 | a**=**np.argmax(model\_resnet.predict(img\_data), axis**=**1) |
| 2 | a |

Out[54]:

array([0])

.png