International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Apple Disease Detection Using Convolutional Neural Networks

[1]A. S. Lalitha, [2]K. Nageswararao

Submitted: 06/02/2024 **Revised**: 14/03/2024 **Accepted**: 20/03/2024

Abstract: Cultivation of a crop is a pertinent aspect in the agricultural sector. The infection of crops with a disease is one of the bottlenecks leading to reduction in the yield of the crop. This decreases the production rate and thereby creating problems in maintaining the crop throughout the year. Manual identification of the disease is laborious and time consuming. It is laborious in the sense that, it requires lot of expertise and monitoring and this problem is going to be much more, especially, when the crop is big enough to manage. In order to minimize the effort needed, A deep learning model has been developed to identify particular disease. Using this model it facilitates the farmer to detect the disease accurately in a real time. The model developed in this research is a Residual Neural Network model as it helps us in implementing the feature extraction and classification of a particular disease of a fruit. A database of 505 apples was considered and 385 apples were used for training the model and 120 apples were considered for testing. The proposed model has generated an accuracy of 78.76% with a loss value of 0.6818 in detecting the disease of an apple. This computer based model will enhance usability of the model in detecting the disease of a fruit in a feasible manner and at an early stage of the disease. This in turn enables the farmer to detect the disease at an early stage of the disease and can take the precautionary measures to cure the disease in a more effortless manner.

Keywords: Residual Neural Network (CNN), Deep learning, rotten fruit,Blotch fruit, Normal Fruit, Scab fruit, Epoch

1. Introduction

Recent advances in technologies like Image processing, Machine and Deeplearning has varied applications in biological and in medical domain. It is observed that, Many farmers in india are equipped with minimal resources and hence they are not in a position to use the resources what they have in increasing the productivity. Many software's have been developed by earlier researchers for the detection of the disease of fruit and these software's helped the farmers identifying the disease even for a novice farmer.

Disease detection/identification can be done using image processing techniques for a given image. By detecting the disease with the image processing technology, the farmer can get to know the type of disease and can take precautionary measures. With this technology in vogue, The farmer can instantly identify the disease of the fruits in bulk quantity. It is imperative that, Agriculture is one of the biggest sector in Indian economy and this sector is undoubtedly the best employer of Indian industries. Infact, the industry of agriculture plays a vital role in enhancing the economy of the country. Comparing the cost of Indian agricultural product with the products in abroad, It is surprised to see that the cost of daily product like, rice, wheat, etc… in India is much cheaper than the cost of the same product outside the world. Hence, there is a lot of scope for export making the farmer to earn more money, thereby increasing the agricultural income for the nation.

This proposed research talks about disease detection of the fruits by developing a CNN model. The productivity of the cash crops like Apple, Grapes and Mango can be enhanced by using this model as this model can be used to detect the disease of the fruits instantly. In this proposed research, Apple fruit database has been considered for the purpose, to detect the disease as a means for research. The different types of diseases an apple can get affected with the diseases are Blotch apple, Rot apple and Scab apple. If the apple has not affected with any of the disease, It is regarded as Normal apple. The fig:1 below shows different types of diseases of an apple.

a) Blotch Apple

[1]*Research Scholar, Department of Computer Science and Engineering, Andhra University, Visakhapatnam Email: santha.lalitha@gmail.com*
[2]*Professor, Department of Computer Science and Systems Engineering, Andhra University, Visakhapatnam Email: hodcsse.au@gmail.com*

b) Rotten Apple

c) Scab Apple

d) Normal Apple

Fig:1.Different types of diseases of an apple

The Sooty blotch apple is a disease affected apple, in which small dots will appear on the surface of the apple and the fruit will be in light black in colour, whereas the Scab apple will have dark patches on the its surface. Rotten apple is one, that has fungal disease with small dots appear on its surface.

2. Literature Survey

Earlier researchers have done a lot of research in detecting the disease of a fruit. Research that was carried out earlier has been given here. Hongjun wang, Qisong Mou, Youjun Yue, Hui Zhao[1] have developed a model R-CNN in identifying the spots on the fruits. Prior to this research, Nivedita.R. Kakade, Dnyaneswar.D. Ahire. Ahire[2], have done research on grape leaf detection using deep learning techniques. Kulkarni Anand H, Ashwin Patil RK [3] have developed a model on plant leaf disease detection using DL techniques. Recently, In the year 2020, Jamil Ahmad, Bilal Jan, Haleem Farman, Wakeel Ahmad, and Atta Ullah[4] have developed a CNN model for disease detection in PLUM under certain conditions. All these researchers have developed various models in trying to find the disease either for a fruit or for leaves. Our recent paper, A S Lalitha, K.Nagaswararao[5] titled " Disease detection in fruits using deep learning" have developed a CNN model fruit disease classification. There, we have taken into consideration of three different fruits. This current paper, we have taken only one fruit for disease detection by using RESNET model.

3.Convolution Neural Network

A Convolutional Neural Network is a deep learning model and it is a type of feed forward neural network and is used for image recognition and processing because of its ability in identifying the patterns in images. By identifying the different patterns in the image, it can distinguish one image from another. It is a multilayered neural network with different layers like input layer, outputlayer, convolution layer, max pooling layer and fully connected layer. The convolution layer is an important layer and it is responsible for computation. These layers are responsible for detecting the patterns in the input image. An important feature of CNN is that, it can detect the features automatically. Because of these features, CNN's have been used widely image recognition tasks. CNN's also have the capability to optimize the filters or kernels. The knowledge gained by the CNN model by giving images as input will be represented in the form of synaptic weights.

4. Tensor Flow

Tensor Flow is a free and open source software library for machine and deep learning applications especially using dataflow graphs. Image classification can be done using neural network models using Tensor flow. Many different architectures of CNN can be developed using this framework. It has its applications in wide range of tasks and has a particular focus on training and inference on deep neural network. It uses Python as frontend API for building applications with the framework but it has several wrapper classes implemented in C++ and Java languages.

5. Imagenet

In this current research, Image net has been used as a database for the images. Different types of datasets have been considered for each type of the disease for the same fruit.., an Apple. ImageNet database has been considered for procuring images. The database consists of 507 images consisting of Blotch, Normal, Rot and Scab apple fruits. Blotch apple consists of 149 fruits, Normal consists of 91 fruits, Rot consists of 152 fruits and finally, Scab fruits consists of 113 fruits. This ImageNet database has been used by earlier researchers and played a very vital role in ascertaining the accuracy of the predictions by the model they have developed or used. This database is available for free for performing research.

6. Convolution

Convolutional Neural Network is a deep learning model and it performs an important function called convolution.

Convolution is a method to be performed on the pixels of the input image to cutshort the outer layer of the image with a value using a filter called kernel. It performs the scalar product of the input image with the filter. This filter or the kernel is basically a 3x3, 4x4 or 5x5 image template. This filter is moved to different regions and compute all the scalar product and eventually we get one image from this scalar product called output image. The filter/kernel is moved along the surface of the original image by one column or shift over to two columns.

7. Softmax Function

Activation function is very important to get to know the behavior of the network. Many activation functions can be used and some of the activation function are Sigmoidal, Relu,Tanh, Softmax and stepfunction. In this proposed research softmax activation function has been used for the purpose. These activation functions will make the neural network to perform machine learning tasks which involve nonlinear problems. The formula for softmax activation function is given below.

$$F(Z)=\frac{ezi}{\sum_{J=1}^{K} ezj}$$

This function takes a vector of K real values and outputs a vector K real values that sum to one. The output values generated by this function lie between 0 and 1. Hence, these values can be interpreted as probabilities. This function considers the maximum probability value and this value will be propagated to the next layers of the network.

8. Max Pooling

Coordinating the result of each and every neuron in a group of neurons into a solitary result is called pooling. There are various types of pooling components: Min pooling and Max Pooling. Essentially a pooling activity chooses the most extreme component from the locale of the element map covered by the channel. The result of maxpooling layer would be a component map containing the most unmistakable elements of the past element map. This decreases the size of the information picture prompting the decrease in the intricacy of the organization as well as computational time.

9. Different Layers Used in CNN Model

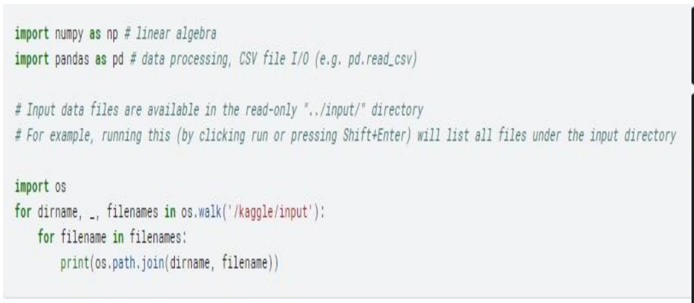
Various layers have been utilized in the model and the conspicuous layers among them are Convolution layer, Maxpooling layer, Standardization, Actuation and afterward the outcome produced by the organization. A portion of the above layers have been tediously utilized in the model. The convolution of the picture lattice and the piece is viewed as the result signal. Max pooling is a significant layer and it is utilized to overlook the with lesser importance in their incentive for the impending layers. Standardization is a course of changing the scope of values the elements take. In picture handling, the pixels might take on values from 0 to 255. To make the scope of values the pixels take from 0 to 1, we partition every one of the upsides of the pixels by 255. This course of doing standardization is called bunch standardization and it makes the brain networks more steady and quicker. The enactment layer is a definitive layer utilized in the proposed model. In this layer, The first picture is separated into different pixels and portion of size 2x2 is thought of. This part is put on the first picture pixel values and the actuation capability is utilized to create the result from the neuron that will be engendered to different neurons in other layer.

There are different layers thought about in the proposed model. They are smooth and thick layers. The smooth layer is utilized to change over complex clusters or lattices into single layered exhibits. The straighten capability is utilized for the reason. Thick layer is a layer where it comprises of set of hubs and it conveys the contribution to different hubs. This layer is capable to keep the information in various aspects. Every hub will place the information in various aspects. One hub can draw one choice limit, two hubs can draw two choice limit, etc. On the off chance that the result is more prominent than 0.5, the result is valid in any case the result is bogus. For Instance: On the off chance that the information network is: [[3,4], [5,6]] and the piece grid is : [[1,0.5], [0.5,0.6]] with an inclination of 0 with enactment as straight. Consequently the result lattice is only the framework increase of the two grids. The result grid is: [[5, 3.9], [8, 6.1]].The worth 6.1 is gotten by: 5*0.5 +6*0.6 = 6.1, which is general framework increase of 2*2 lattices.

10. Results and Discussions

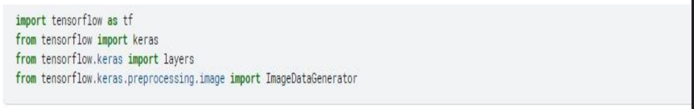
The below snippets of code in Jupiter Notebook (Python Programming) in Anaconda Navigator IDE. The local host server has been used to upload the images of the apples. The results have been given below. For training the model, 119 blotch, 67 Normal, 114 rot and 85 Scab apples were used for training the model. For testing the model, 30 apples of Blotch, 24 apples of Normal, 38 apples of Rot and 28 apples of Scab apples were considered for testing. Totally, 505 images were considered both for training and testing purposes.

The code snippet for listing of files in the input directory is given below.

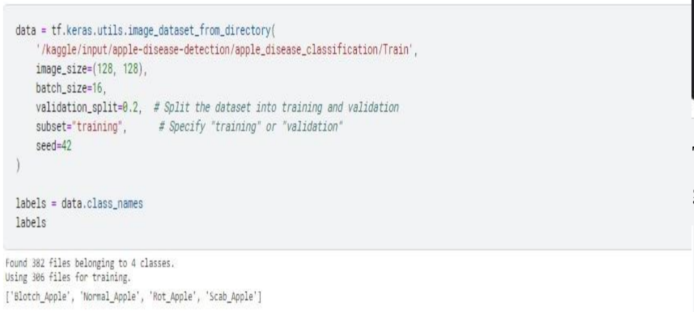


```python
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

The code snippet for identifying the class names for the model is given below.

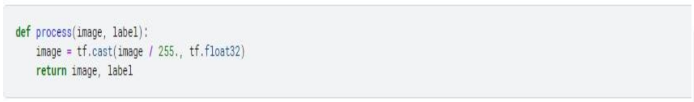


```python
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```



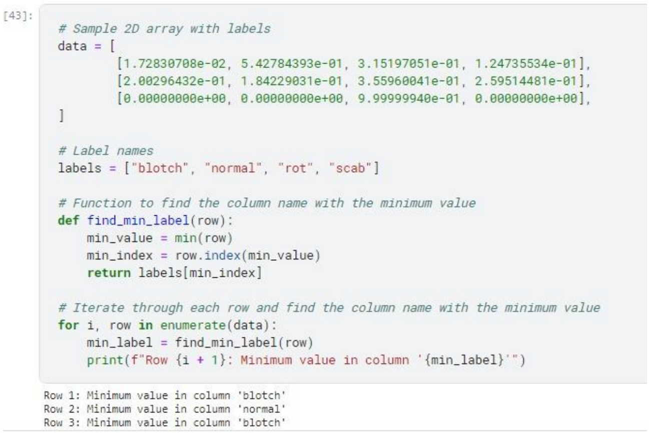
```python
data = tf.keras.utils.image_dataset_from_directory(
    '/kaggle/input/apple-disease-detection/apple_disease_classification/Train',
    image_size=(128, 128),
    batch_size=16,
    validation_split=0.2,  # Split the dataset into training and validation
    subset="training",      # Specify "training" or "validation"
    seed=42
)

labels = data.class_names
labels
```

```
Found 382 files belonging to 4 classes.
Using 306 files for training.
['Blotch_Apple', 'Normal_Apple', 'Rot_Apple', 'Scab_Apple']
```



```python
def process(image, label):
    image = tf.cast(image / 255., tf.float32)
    return image, label
```

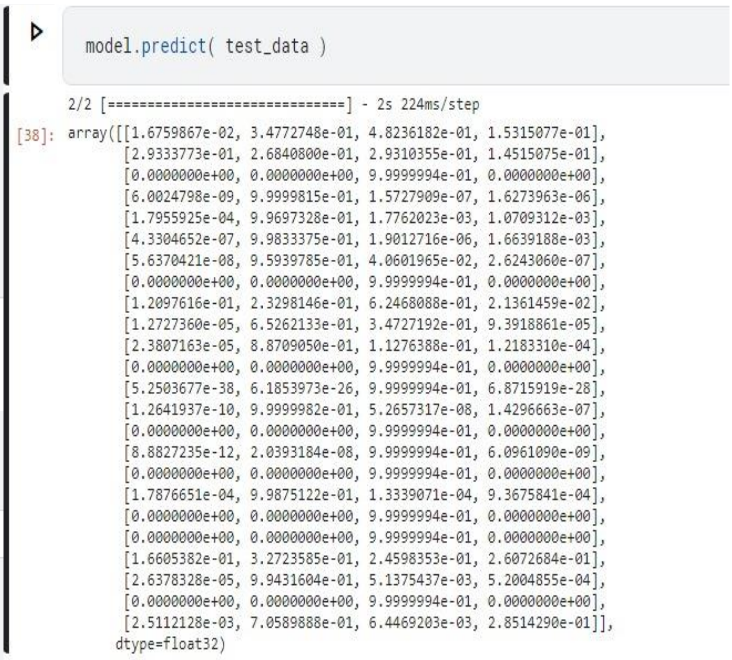
The code snippet for identifying the minimum value of the model is given below.



```python
[43]:
# Sample 2D array with labels
data = [
        [1.72830708e-02, 5.42784393e-01, 3.15197051e-01, 1.24735534e-01],
        [2.00296432e-01, 1.84229031e-01, 3.55960041e-01, 2.59514481e-01],
        [0.00000000e+00, 0.00000000e+00, 9.99999940e-01, 0.00000000e+00],
]

# Label names
labels = ["blotch", "normal", "rot", "scab"]

# Function to find the column name with the minimum value
def find_min_label(row):
    min_value = min(row)
    min_index = row.index(min_value)
    return labels[min_index]

# Iterate through each row and find the column name with the minimum value
for i, row in enumerate(data):
    min_label = find_min_label(row)
    print(f"Row {i + 1}: Minimum value in column '{min_label}'")
```

```
Row 1: Minimum value in column 'blotch'
Row 2: Minimum value in column 'normal'
Row 3: Minimum value in column 'blotch'
```

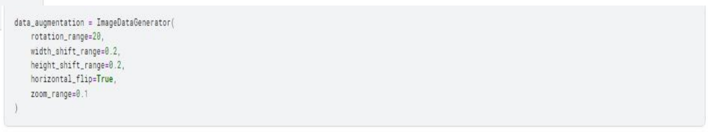
The code below gives how the data is fed to the model.



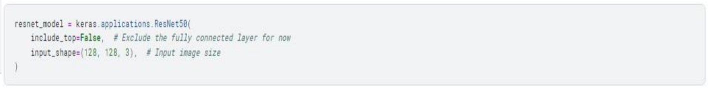
```python
model.predict( test_data )
```

```
2/2 [==============================] - 2s 224ms/step
[38]: array([[1.6759867e-02, 3.4772748e-01, 4.8236182e-01, 1.5315077e-01],
       [2.9333773e-01, 2.6840800e-01, 2.9310355e-01, 1.4515075e-01],
       [0.0000000e+00, 0.0000000e+00, 9.9999994e-01, 0.0000000e+00],
       [6.0024798e-09, 9.9999815e-01, 1.5727909e-07, 1.6273963e-06],
       [1.7955925e-04, 9.9697328e-01, 1.7762023e-03, 1.0709312e-03],
       [4.3304652e-07, 9.9833375e-01, 1.9012716e-06, 1.6639188e-03],
       [5.6370421e-08, 9.5939785e-01, 4.0601965e-02, 2.6243060e-07],
       [0.0000000e+00, 0.0000000e+00, 9.9999994e-01, 0.0000000e+00],
       [1.2097616e-01, 2.3298146e-01, 6.2468088e-01, 2.1361459e-02],
       [1.2727360e-05, 6.5262133e-01, 3.4727192e-01, 9.3918861e-05],
       [2.3807163e-05, 8.8709050e-01, 1.1276388e-01, 1.2183310e-04],
       [0.0000000e+00, 0.0000000e+00, 9.9999994e-01, 0.0000000e+00],
       [5.2503677e-38, 6.1853973e-26, 9.9999994e-01, 6.8715919e-28],
       [1.2641937e-10, 9.9999982e-01, 5.2657317e-08, 1.4296663e-07],
       [0.0000000e+00, 0.0000000e+00, 9.9999994e-01, 0.0000000e+00],
       [8.8827235e-12, 2.0393184e-08, 9.9999994e-01, 6.0961090e-09],
       [0.0000000e+00, 0.0000000e+00, 9.9999994e-01, 0.0000000e+00],
       [1.7876651e-04, 9.9875122e-01, 1.3339071e-04, 9.3675841e-04],
       [0.0000000e+00, 0.0000000e+00, 9.9999994e-01, 0.0000000e+00],
       [0.0000000e+00, 0.0000000e+00, 9.9999994e-01, 0.0000000e+00],
       [1.6605382e-01, 3.2723585e-01, 2.4598353e-01, 2.6072684e-01],
       [2.6378328e-05, 9.9431604e-01, 5.1375437e-03, 5.2004855e-04],
       [0.0000000e+00, 0.0000000e+00, 9.9999994e-01, 0.0000000e+00],
       [2.5112128e-03, 7.0589888e-01, 6.4469203e-03, 2.8514290e-01]],
      dtype=float32)
```

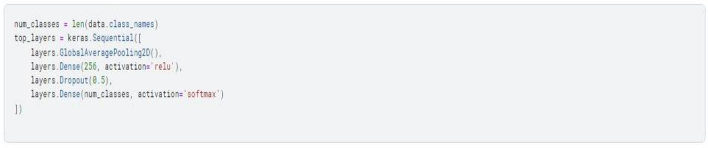
The code for building of Resnet and Sequential model is given below.



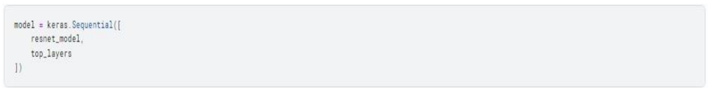
```python
data_augmentation = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    horizontal_flip=True,
    zoom_range=0.1
)
```



```python
resnet_model = keras.applications.ResNet50(
    include_top=False,  # Exclude the fully connected layer for now
    input_shape=(128, 128, 3),  # Input image size
)
```

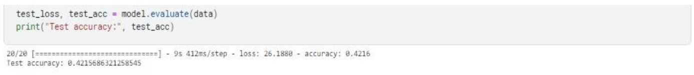


```python
num_classes = len(data.class_names)
top_layers = keras.Sequential([
    layers.GlobalAveragePooling2D(),
    layers.Dense(256, activation='relu'),
    layers.Dropout(0.5),
    layers.Dense(num_classes, activation='softmax')
])
```



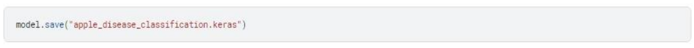
```python
model = keras.Sequential([
    resnet_model,
    top_layers
])
```

The source code below for training and validating the model is given below

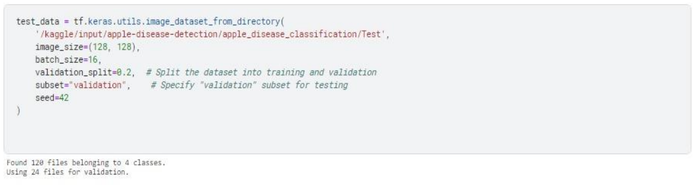


```python
test_loss, test_acc = model.evaluate(data)
print("Test accuracy:", test_acc)
```

```
20/20 [==============================] - 9s 412ms/step - loss: 26.1880 - accuracy: 0.4216
Test accuracy: 0.4215686321258545
```

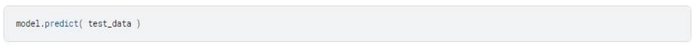


```python
model.save("apple_disease_classification.keras")
```



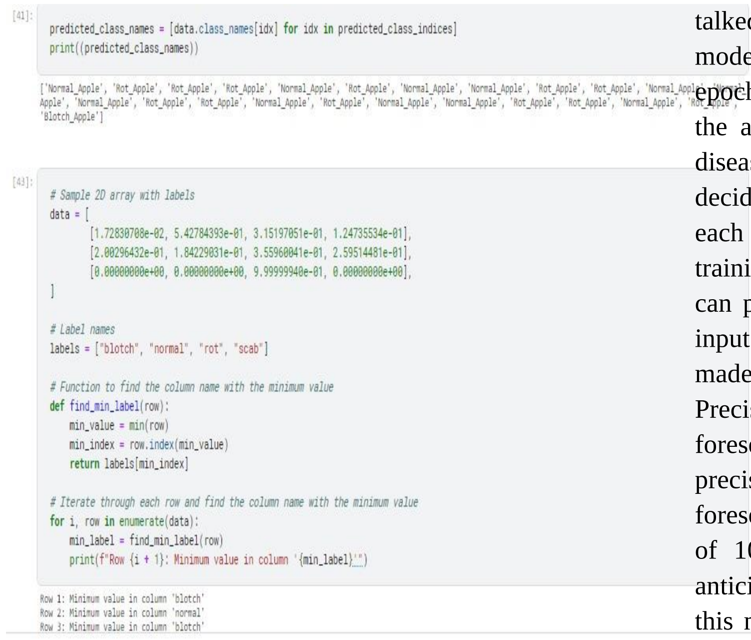
```python
test_data = tf.keras.utils.image_dataset_from_directory(
    '/kaggle/input/apple-disease-detection/apple_disease_classification/Test',
    image_size=(128, 128),
    batch_size=16,
    validation_split=0.2,  # Split the dataset into training and validation
    subset="validation",    # Specify "validation" subset for testing
    seed=42
)
```

```
Found 120 files belonging to 4 classes.
Using 24 files for validation.
```



```python
model.predict( test_data )
```

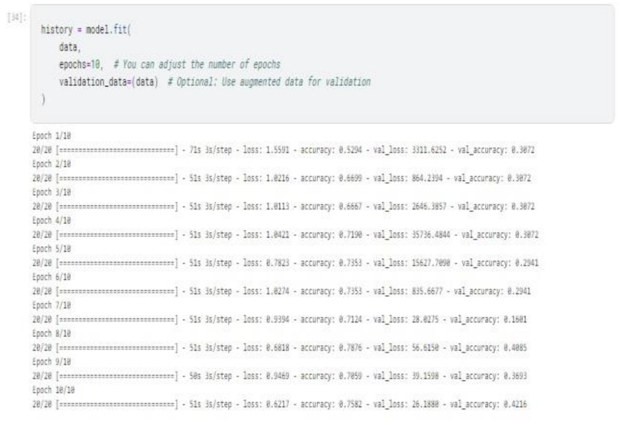
The source code for prediction of classnames has been given below



```
[41]:
predicted_class_names = [data.class_names[idx] for idx in predicted_class_indices]
print((predicted_class_names))

['Normal_Apple', 'Rot_Apple', 'Rot_Apple', 'Rot_Apple', 'Normal_Apple', 'Rot_Apple', 'Normal_Apple', 'Normal_Apple', 'Rot_Apple', 'Rot_Apple', 'Normal_Apple', 'Normal_
Apple', 'Normal_Apple', 'Rot_Apple', 'Rot_Apple', 'Normal_Apple', 'Rot_Apple', 'Normal_Apple', 'Normal_Apple', 'Rot_Apple', 'Rot_Apple', 'Normal_Apple', 'Rot_Apple',
'Blotch_Apple']
```

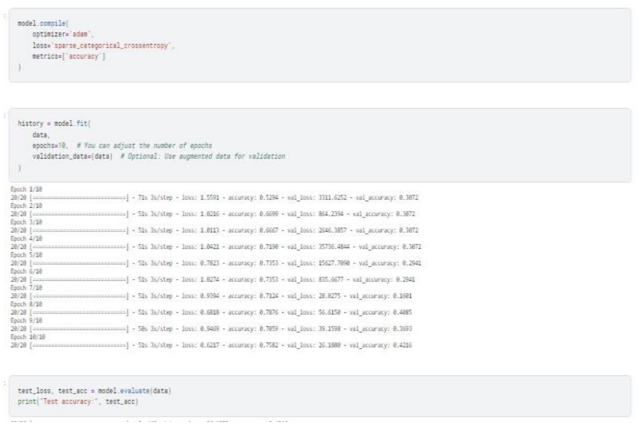
```
[43]:
# Sample 2D array with labels
data = [
        [1.72830708e-02, 5.42784393e-01, 3.15197051e-01, 1.24735534e-01],
        [2.00296432e-01, 1.84229031e-01, 3.55960041e-01, 2.59514481e-01],
        [0.00000000e+00, 0.00000000e+00, 9.99999940e-01, 0.00000000e+00],
]

# Label names
labels = ["blotch", "normal", "rot", "scab"]

# Function to find the column name with the minimum value
def find_min_label(row):
    min_value = min(row)
    min_index = row.index(min_value)
    return labels[min_index]

# Iterate through each row and find the column name with the minimum value
for i, row in enumerate(data):
    min_label = find_min_label(row)
    print(f"Row {i + 1}: Minimum value in column '{min_label}'")

Row 1: Minimum value in column 'blotch'
Row 2: Minimum value in column 'normal'
Row 3: Minimum value in column 'blotch'
```

Similarly, the source code for finding the accuracy of the model is given below.



```
[34]:
history = model.fit(
    data,
    epochs=10,  # You can adjust the number of epochs
    validation_data=(data)  # Optional: Use augmented data for validation
)

Epoch 1/10
20/20 [==============================] - 71s 3s/step - loss: 1.5591 - accuracy: 0.5294 - val_loss: 3311.6252 - val_accuracy: 0.3072
Epoch 2/10
20/20 [==============================] - 51s 3s/step - loss: 1.0216 - accuracy: 0.6699 - val_loss: 864.2394 - val_accuracy: 0.3072
Epoch 3/10
20/20 [==============================] - 51s 3s/step - loss: 1.0113 - accuracy: 0.6667 - val_loss: 2646.3857 - val_accuracy: 0.3072
Epoch 4/10
20/20 [==============================] - 51s 3s/step - loss: 1.0421 - accuracy: 0.7190 - val_loss: 35736.4844 - val_accuracy: 0.3072
Epoch 5/10
20/20 [==============================] - 51s 3s/step - loss: 0.7823 - accuracy: 0.7353 - val_loss: 15627.7090 - val_accuracy: 0.2941
Epoch 6/10
20/20 [==============================] - 51s 3s/step - loss: 1.0274 - accuracy: 0.7353 - val_loss: 835.6677 - val_accuracy: 0.2941
Epoch 7/10
20/20 [==============================] - 51s 3s/step - loss: 0.9394 - accuracy: 0.7124 - val_loss: 28.0275 - val_accuracy: 0.1601
Epoch 8/10
20/20 [==============================] - 51s 3s/step - loss: 0.6818 - accuracy: 0.7876 - val_loss: 56.6150 - val_accuracy: 0.4085
Epoch 9/10
20/20 [==============================] - 50s 3s/step - loss: 0.9469 - accuracy: 0.7059 - val_loss: 39.1598 - val_accuracy: 0.3693
Epoch 10/10
20/20 [==============================] - 51s 3s/step - loss: 0.6217 - accuracy: 0.7582 - val_loss: 26.1880 - val_accuracy: 0.4216
```

The source code for test accuracy of the model is given below.



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

```
history = model.fit(
    data,
    epochs=10,  # You can adjust the number of epochs
    validation_data=(data)  # Optional: Use augmented data for validation
)
Epoch 1/10
20/20 [==============================] - 71s 3s/step - loss: 1.5591 - accuracy: 0.5294 - val_loss: 3311.6252 - val_accuracy: 0.3072
Epoch 2/10
20/20 [==============================] - 51s 3s/step - loss: 1.0216 - accuracy: 0.6699 - val_loss: 864.2394 - val_accuracy: 0.3072
Epoch 3/10
20/20 [==============================] - 51s 3s/step - loss: 1.0113 - accuracy: 0.6667 - val_loss: 2646.3857 - val_accuracy: 0.3072
Epoch 4/10
20/20 [==============================] - 51s 3s/step - loss: 1.0421 - accuracy: 0.7190 - val_loss: 35736.4844 - val_accuracy: 0.3072
Epoch 5/10
20/20 [==============================] - 51s 3s/step - loss: 0.7823 - accuracy: 0.7353 - val_loss: 15627.7090 - val_accuracy: 0.2941
Epoch 6/10
20/20 [==============================] - 51s 3s/step - loss: 1.0274 - accuracy: 0.7353 - val_loss: 835.6677 - val_accuracy: 0.2941
Epoch 7/10
20/20 [==============================] - 51s 3s/step - loss: 0.9394 - accuracy: 0.7124 - val_loss: 28.0275 - val_accuracy: 0.1601
Epoch 8/10
20/20 [==============================] - 51s 3s/step - loss: 0.6818 - accuracy: 0.7876 - val_loss: 56.6150 - val_accuracy: 0.4085
Epoch 9/10
20/20 [==============================] - 50s 3s/step - loss: 0.9469 - accuracy: 0.7059 - val_loss: 39.1598 - val_accuracy: 0.3693
Epoch 10/10
20/20 [==============================] - 51s 3s/step - loss: 0.6217 - accuracy: 0.7582 - val_loss: 26.1880 - val_accuracy: 0.4216
```

```
test_loss, test_acc = model.evaluate(data)
print("Test accuracy:", test_acc)
```

11. Conclusions:

This paper is basically about how a deeplearning model can be used to detect the disease of an apple. The model is able to generate an accuracy of 78.76% with a loss value of 0.6818 in detecting the disease of an apple. This paper has talked about an intelligent development of the prototype model for detecting a fruit diseases. The total number of epochs considered are 20 and it is assumed that, by using the above said model, The farmer can easily detect the disease in an easy and faster way. The loss value will decide how exactly or incorrectly the model behaves after each iteration. The loss has been calculated for both training and test sets and it is observed that loss value, we can predict the behavior of the model when it is fed the input to the model. It is basically the summation of errors made for each image in training and validation sets. Precision of the model is a level of how well the model can foresee the outcomes. The proposed model has created a precision of 78.76% as in out of 100apples, the model can foresee sicknesses of approaching 79apples accurately out of 100 apples and the rest ie., 21 apples have been anticipated as off-base outcomes. At long last, by fostering this model, we can affirm that the profound organization can be utilized in foreseeing the natural product sickness in a simple and effective manner. This model can be most certainly be coordinated into a portable application so every rancher can recognize the illness easily and in a helpful manner.

References:

[1] Hongjun wang, Qisong Mou, Youjun Yue, Hui Zhao: "Research on Detection Technology of various Fruit Disease Spots Based on Mask R-CNN", Tianjin University of Technology, Tianjin China, 2016

[2] Nivedita.R. Kakade, Dnyaneswar.D. Ahire. Ahire: "Real Time Grape Leaf Disease Detection", International Journal of Advance Research and Innovative Ideas in Education, Vol-1 Issue-4, 2015

[3] Kulkarni Anand H, Ashwin Patil RK: "Applying image processing technique to detect plant diseases". Int J Mod Eng Res, 2012.

[4] Jamil Ahmad, Bilal Jan, Haleem Farman, Wakeel Ahmad, and Atta Ullah: "Disease Detection in Plum using Convolutional Neural Network under true field conditions".28 September 2020.

[5] A S Lalitha, K.Nagaswararao., " Fruit Disease Categorization based on Convolutional Neural Networks", Journal of Data Acquisition and Processing", 38(3), 2023.