```
import datetime
cur_date=datetime.datetime.now()
print("current date and time",cur_date)
```

```
current date and time 2024-02-18 17:58:14.706974
```

```
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.applications import MobileNet
from tensorflow.keras.layers import AveragePooling2D
from tensorflow.keras.layers import Dropout
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Input
from tensorflow.keras.models import Model
from keras.models import Sequential

from sklearn.preprocessing import MultiLabelBinarizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report

import matplotlib.pyplot as plt
import numpy as np
import random
import cv2
import os
import h5py

from imutils import paths

imagePaths = sorted(list(paths.list_images("F:\\Paper6_vit+cnn\\
Groundnut_Leaf_dataset\\train1")))
#F:\Paper6_vit+cnn\Groundnut_Leaf_dataset\train1
#D:\G_Dataset_final\Train
# random shuffle
random.seed(42)
random.shuffle(imagePaths)

data = []
labels = []
image_dims = (224, 224, 3)# 212 for xception model

for imagePath in imagePaths:
    image = cv2.imread(imagePath)
    image = cv2.resize(image, (image_dims[1], image_dims[0]))
    image = img_to_array(image)
    data.append(image)
    l = label = imagePath.split(os.path.sep)[-2].split("_")
    labels.append(l)

data = np.array(data, dtype="float") / 255.0
labels = np.array(labels)
```

```
print("{} images ({:.2f}MB)".format(len(imagePaths), data.nbytes /
(1024 * 1000.0)))
```

```
7910 images (9302.16MB)
```

```
data = np.array(data)
label = np.array(labels)
print(data.shape)
```

```
(7910, 224, 224, 3)
```

```
mlb = MultiLabelBinarizer()
labels = mlb.fit_transform(labels)
# total 4 labels
print("class labels:")
for (i, label) in enumerate(mlb.classes_):
    print("{}. {}".format(i + 1, label))
```

```
class labels:
1. ELS
2. ER
3. HL
4. LLS
5. ND
6. RUST
```

```
trainX, testX, trainY, testY = train_test_split(data, labels,
test_size=0.45) # (0.20)
```

```
from keras.applications import MobileNet #InceptionResNetV2
NASNetMobile InceptionV3 VGG19
inc=MobileNet(input_shape=(224,224,3),weights='imagenet',include_top=F
alse)
```

```
for i in inc.layers:
    i.trainable=False
```

```
from tensorflow.keras import layers
x=Flatten()(inc.output)
#x=layers.Dropout(0.2)(x)
```

```
pred=Dense(6,activation='softmax')(x)
```

```
from keras.models import Model
model=Model(inputs=inc.input,outputs=pred)
```

```
model.summary()
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
```

```
 input_1 (InputLayer)        [(None, 224, 224, 3)]     0

 conv1 (Conv2D)              (None, 112, 112, 32)      864

 conv1_bn (BatchNormalizatio  (None, 112, 112, 32)     128
 n)

 conv1_relu (ReLU)           (None, 112, 112, 32)      0

 conv_dw_1 (DepthwiseConv2D)  (None, 112, 112, 32)     288

 conv_dw_1_bn (BatchNormaliz  (None, 112, 112, 32)     128
 ation)

 conv_dw_1_relu (ReLU)       (None, 112, 112, 32)      0

 conv_pw_1 (Conv2D)          (None, 112, 112, 64)      2048

 conv_pw_1_bn (BatchNormaliz  (None, 112, 112, 64)     256
 ation)

 conv_pw_1_relu (ReLU)       (None, 112, 112, 64)      0

 conv_pad_2 (ZeroPadding2D)  (None, 113, 113, 64)      0

 conv_dw_2 (DepthwiseConv2D)  (None, 56, 56, 64)       576

 conv_dw_2_bn (BatchNormaliz  (None, 56, 56, 64)       256
 ation)

 conv_dw_2_relu (ReLU)       (None, 56, 56, 64)        0

 conv_pw_2 (Conv2D)          (None, 56, 56, 128)       8192

 conv_pw_2_bn (BatchNormaliz  (None, 56, 56, 128)      512
 ation)

 conv_pw_2_relu (ReLU)       (None, 56, 56, 128)       0

 conv_dw_3 (DepthwiseConv2D)  (None, 56, 56, 128)      1152

 conv_dw_3_bn (BatchNormaliz  (None, 56, 56, 128)      512
 ation)

 conv_dw_3_relu (ReLU)       (None, 56, 56, 128)       0

 conv_pw_3 (Conv2D)          (None, 56, 56, 128)       16384

 conv_pw_3_bn (BatchNormaliz  (None, 56, 56, 128)      512
```

```
 ation)

 conv_pw_3_relu (ReLU)       (None, 56, 56, 128)       0

 conv_pad_4 (ZeroPadding2D)  (None, 57, 57, 128)       0

 conv_dw_4 (DepthwiseConv2D)  (None, 28, 28, 128)      1152

 conv_dw_4_bn (BatchNormaliz  (None, 28, 28, 128)      512
 ation)

 conv_dw_4_relu (ReLU)       (None, 28, 28, 128)       0

 conv_pw_4 (Conv2D)          (None, 28, 28, 256)       32768

 conv_pw_4_bn (BatchNormaliz  (None, 28, 28, 256)      1024
 ation)

 conv_pw_4_relu (ReLU)       (None, 28, 28, 256)       0

 conv_dw_5 (DepthwiseConv2D)  (None, 28, 28, 256)      2304

 conv_dw_5_bn (BatchNormaliz  (None, 28, 28, 256)      1024
 ation)

 conv_dw_5_relu (ReLU)       (None, 28, 28, 256)       0

 conv_pw_5 (Conv2D)          (None, 28, 28, 256)       65536

 conv_pw_5_bn (BatchNormaliz  (None, 28, 28, 256)      1024
 ation)

 conv_pw_5_relu (ReLU)       (None, 28, 28, 256)       0

 conv_pad_6 (ZeroPadding2D)  (None, 29, 29, 256)       0

 conv_dw_6 (DepthwiseConv2D)  (None, 14, 14, 256)      2304

 conv_dw_6_bn (BatchNormaliz  (None, 14, 14, 256)      1024
 ation)

 conv_dw_6_relu (ReLU)       (None, 14, 14, 256)       0

 conv_pw_6 (Conv2D)          (None, 14, 14, 512)       131072

 conv_pw_6_bn (BatchNormaliz  (None, 14, 14, 512)      2048
 ation)

 conv_pw_6_relu (ReLU)       (None, 14, 14, 512)       0
```

```
 conv_dw_7 (DepthwiseConv2D)  (None, 14, 14, 512)       4608

 conv_dw_7_bn (BatchNormaliz  (None, 14, 14, 512)       2048
 ation)

 conv_dw_7_relu (ReLU)       (None, 14, 14, 512)       0

 conv_pw_7 (Conv2D)          (None, 14, 14, 512)       262144

 conv_pw_7_bn (BatchNormaliz  (None, 14, 14, 512)       2048
 ation)

 conv_pw_7_relu (ReLU)       (None, 14, 14, 512)       0

 conv_dw_8 (DepthwiseConv2D)  (None, 14, 14, 512)       4608

 conv_dw_8_bn (BatchNormaliz  (None, 14, 14, 512)       2048
 ation)

 conv_dw_8_relu (ReLU)       (None, 14, 14, 512)       0

 conv_pw_8 (Conv2D)          (None, 14, 14, 512)       262144

 conv_pw_8_bn (BatchNormaliz  (None, 14, 14, 512)       2048
 ation)

 conv_pw_8_relu (ReLU)       (None, 14, 14, 512)       0

 conv_dw_9 (DepthwiseConv2D)  (None, 14, 14, 512)       4608

 conv_dw_9_bn (BatchNormaliz  (None, 14, 14, 512)       2048
 ation)

 conv_dw_9_relu (ReLU)       (None, 14, 14, 512)       0

 conv_pw_9 (Conv2D)          (None, 14, 14, 512)       262144

 conv_pw_9_bn (BatchNormaliz  (None, 14, 14, 512)       2048
 ation)

 conv_pw_9_relu (ReLU)       (None, 14, 14, 512)       0

 conv_dw_10 (DepthwiseConv2D  (None, 14, 14, 512)       4608
 )

 conv_dw_10_bn (BatchNormali  (None, 14, 14, 512)       2048
 zation)
```

```
 conv_dw_10_relu (ReLU)      (None, 14, 14, 512)       0

 conv_pw_10 (Conv2D)         (None, 14, 14, 512)       262144

 conv_pw_10_bn (BatchNormali  (None, 14, 14, 512)      2048
 zation)

 conv_pw_10_relu (ReLU)      (None, 14, 14, 512)       0

 conv_dw_11 (DepthwiseConv2D  (None, 14, 14, 512)      4608
 )

 conv_dw_11_bn (BatchNormali  (None, 14, 14, 512)      2048
 zation)

 conv_dw_11_relu (ReLU)      (None, 14, 14, 512)       0

 conv_pw_11 (Conv2D)         (None, 14, 14, 512)       262144

 conv_pw_11_bn (BatchNormali  (None, 14, 14, 512)      2048
 zation)

 conv_pw_11_relu (ReLU)      (None, 14, 14, 512)       0

 conv_pad_12 (ZeroPadding2D)  (None, 15, 15, 512)      0

 conv_dw_12 (DepthwiseConv2D  (None, 7, 7, 512)        4608
 )

 conv_dw_12_bn (BatchNormali  (None, 7, 7, 512)        2048
 zation)

 conv_dw_12_relu (ReLU)      (None, 7, 7, 512)         0

 conv_pw_12 (Conv2D)         (None, 7, 7, 1024)        524288

 conv_pw_12_bn (BatchNormali  (None, 7, 7, 1024)       4096
 zation)

 conv_pw_12_relu (ReLU)      (None, 7, 7, 1024)        0

 conv_dw_13 (DepthwiseConv2D  (None, 7, 7, 1024)       9216
 )

 conv_dw_13_bn (BatchNormali  (None, 7, 7, 1024)       4096
 zation)

 conv_dw_13_relu (ReLU)      (None, 7, 7, 1024)        0
```

```
 conv_pw_13 (Conv2D)         (None, 7, 7, 1024)        1048576

 conv_pw_13_bn (BatchNormali  (None, 7, 7, 1024)       4096
 zation)

 conv_pw_13_relu (ReLU)      (None, 7, 7, 1024)        0

 flatten (Flatten)           (None, 50176)             0

 dense (Dense)               (None, 6)                 301062

=================================================================
Total params: 3,529,926
Trainable params: 301,062
Non-trainable params: 3,228,864
_________________________________________________________________
```

```
from tensorflow import keras
opt = keras.optimizers.Adam(learning_rate=0.001)
model.compile(optimizer=opt,loss='categorical_crossentropy',metrics=['
accuracy'])
#model.compile(optimizer='adam',loss='categorical_crossentropy',metric
s=['accuracy'])
```

```
history=model.fit(trainX, trainY,
          batch_size=16,
          epochs=10,
          verbose=1,
          validation_data=(testX, testY))
```

```
Epoch 1/10
272/272 [==============================] - 103s 375ms/step - loss:
1.8610 - accuracy: 0.8285 - val_loss: 0.9048 - val_accuracy: 0.9270
Epoch 2/10
272/272 [==============================] - 99s 365ms/step - loss:
0.3354 - accuracy: 0.9646 - val_loss: 0.9264 - val_accuracy: 0.9219
Epoch 3/10
272/272 [==============================] - 101s 370ms/step - loss:
0.3731 - accuracy: 0.9655 - val_loss: 1.0201 - val_accuracy: 0.9449
Epoch 4/10
272/272 [==============================] - 100s 368ms/step - loss:
0.4369 - accuracy: 0.9694 - val_loss: 1.3725 - val_accuracy: 0.9365
Epoch 5/10
272/272 [==============================] - 100s 367ms/step - loss:
0.2205 - accuracy: 0.9857 - val_loss: 1.8999 - val_accuracy: 0.9135
Epoch 6/10
272/272 [==============================] - 100s 367ms/step - loss:
0.2275 - accuracy: 0.9828 - val_loss: 1.2530 - val_accuracy: 0.9458
Epoch 7/10
272/272 [==============================] - 101s 373ms/step - loss:
```

```
0.1672 - accuracy: 0.9894 - val_loss: 1.0155 - val_accuracy: 0.9508
Epoch 8/10
272/272 [==============================] - 101s 371ms/step - loss:
0.1538 - accuracy: 0.9908 - val_loss: 3.0135 - val_accuracy: 0.9129
Epoch 9/10
272/272 [==============================] - 102s 374ms/step - loss:
0.2772 - accuracy: 0.9855 - val_loss: 1.1140 - val_accuracy: 0.9551
Epoch 10/10
272/272 [==============================] - 100s 370ms/step - loss:
0.1629 - accuracy: 0.9908 - val_loss: 2.0874 - val_accuracy: 0.9272
```

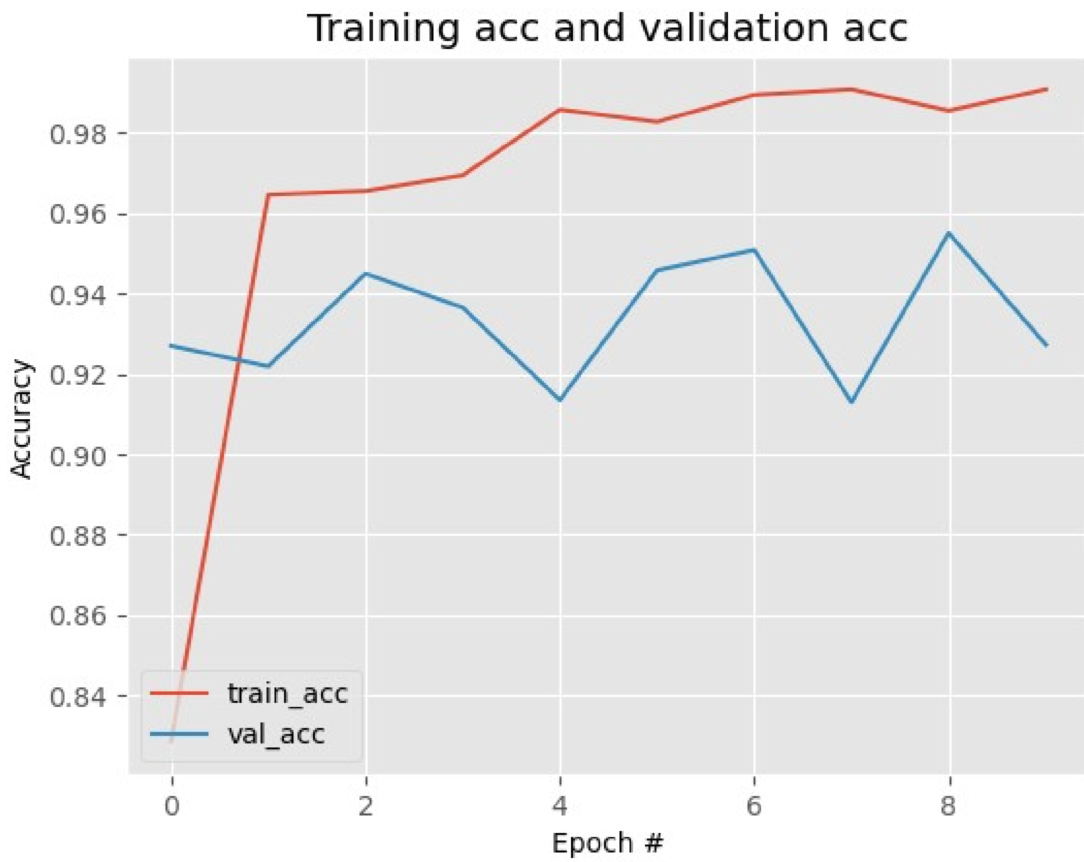
```
print("[INFO] evaluating network...")
predIdxs = model.predict(testX, batch_size=16)

# for each image in the testing set we need to find the index of the
label with corresponding largest predicted probability
predIdxs = np.argmax(predIdxs, axis=1)

# show a nicely formatted classification report
print(classification_report(testY.argmax(axis=1),
predIdxs,target_names=mlb.classes_))
```

```
[INFO] evaluating network...
223/223 [==============================] - 55s 243ms/step
              precision    recall  f1-score   support

         ELS       0.97      0.72      0.83       592
          ER       1.00      0.96      0.98       478
          HL       0.94      0.95      0.95       677
         LLS       0.83      0.96      0.89       655
          ND       0.89      0.97      0.93       557
        RUST       0.99      1.00      0.99       601

    accuracy                           0.93      3560
   macro avg       0.94      0.93      0.93      3560
weighted avg       0.93      0.93      0.93      3560
```

```
N = 10
plt.style.use("ggplot")
plt.figure()
#plt.plot(np.arange(0, N), H.history["loss"], label="train_loss")
#plt.plot(np.arange(0, N), H.history["val_loss"], label="val_loss")
plt.plot(np.arange(0, N), history.history["accuracy"],
label="train_acc")
plt.plot(np.arange(0, N), history.history["val_accuracy"],
label="val_acc")
plt.title("Training acc and validation acc")
plt.xlabel("Epoch #")
```

```
plt.ylabel("Accuracy")
plt.legend(loc="lower left")
```

```
<matplotlib.legend.Legend at 0x210c2fee4a0>
```




```
N = 10
plt.style.use("ggplot")
plt.figure()
plt.plot(np.arange(0, N), history.history["loss"], label="train_loss")
plt.plot(np.arange(0, N), history.history["val_loss"],
label="val_loss")
#plt.plot(np.arange(0, N), H.history["accuracy"], label="train_acc")
#plt.plot(np.arange(0, N), H.history["val_accuracy"], label="val_acc")
plt.title("Training Loss and validation loss")
plt.xlabel("Epoch #")
plt.ylabel("Loss")
plt.legend(loc="lower left")
```

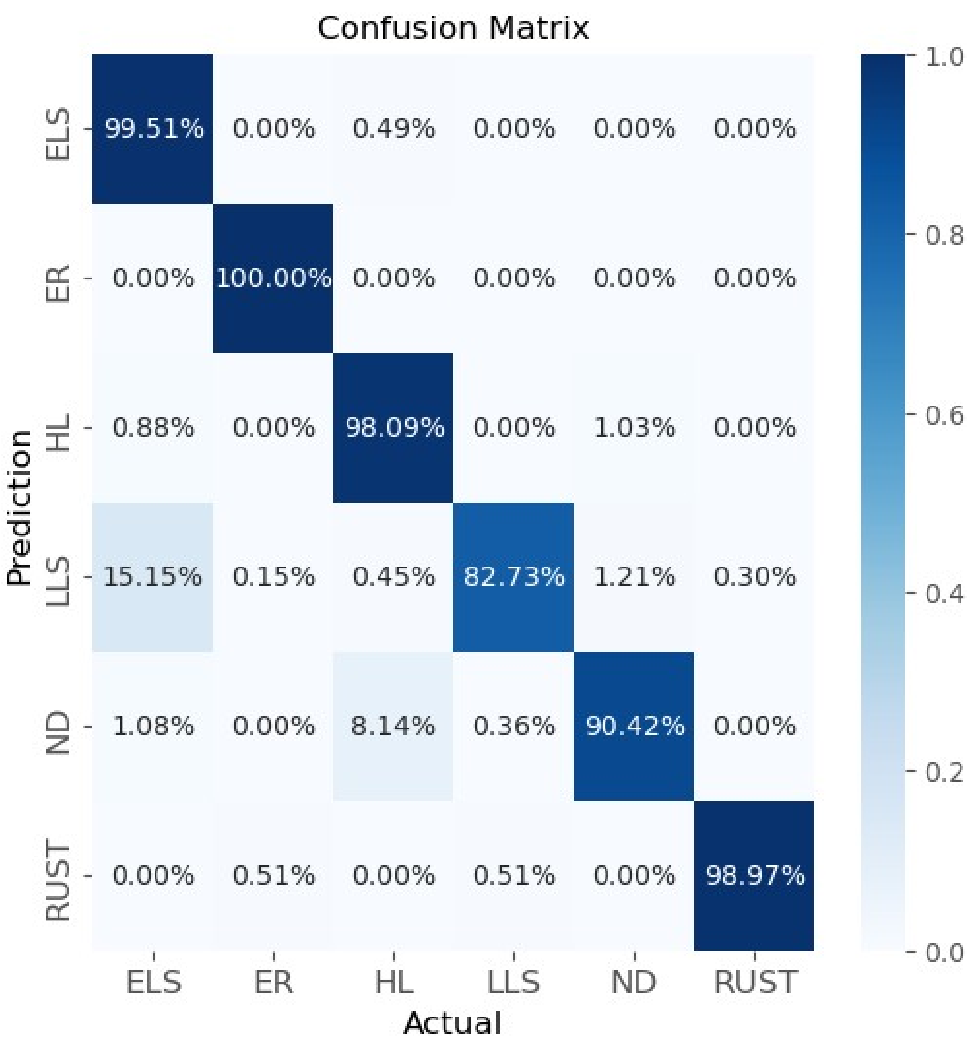
```
<matplotlib.legend.Legend at 0x210c38a8b20>
```


```
from sklearn.metrics import confusion_matrix
print(confusion_matrix(testY.argmax(axis=1), predIdxs))
```

```
[[426   0  23 120  21   2]
 [  0 461   0   9   5   3]
 [  1   0 646   0  30   0]
 [ 10   0   3 628  10   4]
 [  1   0  14   0 542   0]
 [  0   2   0   1   0 598]]
```

```
#Import the necessary libraries
import numpy as np
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
plt.figure(figsize=(6, 6))
#compute the confusion matrix.
#cm = confusion_matrix(testY.argmax(axis=1), predIdxs)
cm=np.array([[612,0,3,0,0,0],
            [0,468,0,0,0,0],
             [6,0,667,0,7,0],
```

```
            [100,1,3,546,8,2],
            [6,0,45,2,500,0],
           [0,3,0,3,0, 578]])
#Plot the confusion matrix.
sns.heatmap(cm/np.sum(cm,axis=1).reshape(-
1,1),cmap='Blues',annot=True,fmt='.2%',xticklabels=['ELS','ER','HL','L
LS','ND','RUST'],yticklabels=['ELS','ER','HL','LLS','ND','RUST'])
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)
plt.ylabel('Prediction',fontsize=12)
plt.xlabel('Actual',fontsize=12)
plt.title('Confusion Matrix',fontsize=12)
#plt.savefig('F:/paper_3_code_files_final/results/results1/mobilenet_C
M.jpg',dpi=600)
plt.show()
```



```python
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
precision_score(testY.argmax(axis=1), predIdxs, average='micro')
```

```
0.927247191011236
```

```python
recall = recall_score(testY.argmax(axis=1), predIdxs, average='micro')
print('recall=',recall)
```

```
recall= 0.927247191011236
```

```python
from sklearn.metrics import f1_score
f1_score(testY.argmax(axis=1), predIdxs, average='micro')
```

```
0.927247191011236
```

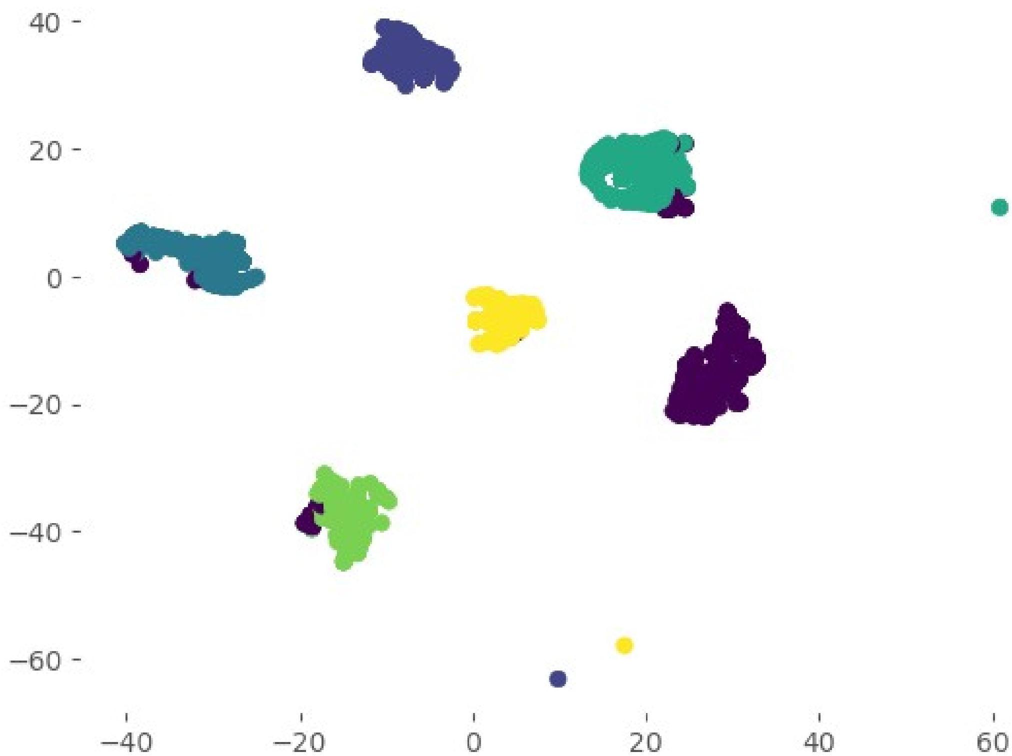
```python
from sklearn.metrics import matthews_corrcoef
from sklearn.metrics import balanced_accuracy_score
a=matthews_corrcoef(testY.argmax(axis=1), predIdxs)
b=balanced_accuracy_score(testY.argmax(axis=1), predIdxs)
print(a)
print(b)
```

```
0.914045809949668
0.9275160755508924
```

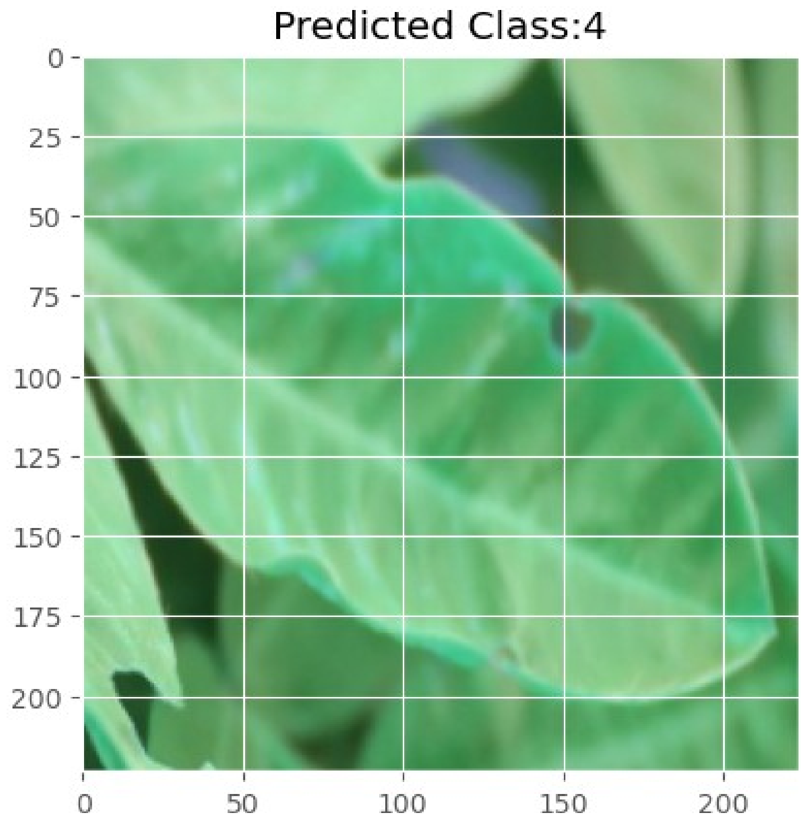
```python
from sklearn.manifold import TSNE
# Extract the features from the last layer of the CNN
features = model.predict(trainX)
# Reduce the dimensionality of the features using t-SNE
tsne = TSNE(n_components=2, perplexity=30.0, early_exaggeration=12.0,
learning_rate=200.0, n_iter=1000, n_iter_without_progress=300,
min_grad_norm=1e-07, metric='euclidean', metric_params=None,
init='pca', verbose=0, random_state=None, method='barnes_hut',
angle=0.5, n_jobs=None)
features_embedded = tsne.fit_transform(features)
```

```
136/136 [==============================] - 66s 482ms/step
```

```python
# Plot the results
ax=plt.axes()
ax.set_facecolor("white")
plt.scatter(features_embedded[:, 0], features_embedded[:, 1], c=
trainY.argmax(axis=1))
#plt.axis('off')
#plt.savefig('F:/Paper6_vit+cnn/results/P6_Xception_TSNE.pdf',dpi=600)
#plt.savefig('F:/Paper6_vit+cnn/results/updated_results/P6_Xception_TS
NE.png',dpi=600)
#plt.legend()
plt.grid(False)

plt.show()
```




```python
import lime
from lime import lime_image

# Choose an image from the test set for explanation
image_index = 20
img = testX[image_index]
img_array = np.expand_dims(img, axis=0)

# Make predictions
predictions = model.predict(img_array)
predicted_class = np.argmax(predictions)

# Display the original image and predictions
plt.imshow(img.reshape(224, 224,3))
plt.title(f'Predicted Class:{predicted_class}')
plt.show()

1/1 [==============================] - 0s 73ms/step
```




```python
# Create a LIME explainer for image classification
explainer = lime_image.LimeImageExplainer()

# Explain predictions
explanation = explainer.explain_instance(img_array[0], model.predict,
top_labels=1, hide_color=0, num_samples=1000)
```

{"model_id":"5af440789aa44dbd8bdd78c37aa5c399","version_major":2,"vers
ion_minor":0}

```
1/1 [==============================] - 0s 157ms/step
1/1 [==============================] - 0s 159ms/step
1/1 [==============================] - 0s 174ms/step
1/1 [==============================] - 0s 218ms/step
1/1 [==============================] - 0s 167ms/step
1/1 [==============================] - 0s 172ms/step
1/1 [==============================] - 0s 156ms/step
1/1 [==============================] - 0s 203ms/step
1/1 [==============================] - 0s 231ms/step
1/1 [==============================] - 0s 183ms/step
1/1 [==============================] - 0s 200ms/step
1/1 [==============================] - 0s 146ms/step
1/1 [==============================] - 0s 231ms/step
1/1 [==============================] - 0s 156ms/step
```

```
1/1 [==============================] - 0s 138ms/step
1/1 [==============================] - 0s 135ms/step
1/1 [==============================] - 0s 175ms/step
1/1 [==============================] - 0s 159ms/step
1/1 [==============================] - 0s 152ms/step
1/1 [==============================] - 0s 158ms/step
1/1 [==============================] - 0s 168ms/step
1/1 [==============================] - 0s 156ms/step
1/1 [==============================] - 0s 140ms/step
1/1 [==============================] - 0s 142ms/step
1/1 [==============================] - 0s 156ms/step
1/1 [==============================] - 0s 167ms/step
1/1 [==============================] - 0s 159ms/step
1/1 [==============================] - 0s 137ms/step
1/1 [==============================] - 0s 159ms/step
1/1 [==============================] - 0s 170ms/step
1/1 [==============================] - 0s 172ms/step
1/1 [==============================] - 0s 148ms/step
1/1 [==============================] - 0s 153ms/step
1/1 [==============================] - 0s 145ms/step
1/1 [==============================] - 0s 156ms/step
1/1 [==============================] - 0s 137ms/step
1/1 [==============================] - 0s 144ms/step
1/1 [==============================] - 0s 150ms/step
1/1 [==============================] - 0s 156ms/step
1/1 [==============================] - 0s 187ms/step
1/1 [==============================] - 0s 142ms/step
1/1 [==============================] - 0s 139ms/step
1/1 [==============================] - 0s 153ms/step
1/1 [==============================] - 0s 172ms/step
1/1 [==============================] - 0s 149ms/step
1/1 [==============================] - 0s 138ms/step
1/1 [==============================] - 0s 145ms/step
1/1 [==============================] - 0s 158ms/step
1/1 [==============================] - 0s 144ms/step
1/1 [==============================] - 0s 140ms/step
1/1 [==============================] - 0s 130ms/step
1/1 [==============================] - 0s 144ms/step
1/1 [==============================] - 0s 156ms/step
1/1 [==============================] - 0s 145ms/step
1/1 [==============================] - 0s 134ms/step
1/1 [==============================] - 0s 145ms/step
1/1 [==============================] - 0s 139ms/step
1/1 [==============================] - 0s 156ms/step
1/1 [==============================] - 0s 149ms/step
1/1 [==============================] - 0s 142ms/step
1/1 [==============================] - 0s 146ms/step
1/1 [==============================] - 0s 154ms/step
1/1 [==============================] - 0s 141ms/step
```

```
1/1 [==============================] - 0s 144ms/step
1/1 [==============================] - 0s 140ms/step
1/1 [==============================] - 0s 147ms/step
1/1 [==============================] - 0s 151ms/step
1/1 [==============================] - 0s 137ms/step
1/1 [==============================] - 0s 137ms/step
1/1 [==============================] - 0s 133ms/step
1/1 [==============================] - 0s 145ms/step
1/1 [==============================] - 0s 148ms/step
1/1 [==============================] - 0s 145ms/step
1/1 [==============================] - 0s 138ms/step
1/1 [==============================] - 0s 148ms/step
1/1 [==============================] - 0s 140ms/step
1/1 [==============================] - 0s 156ms/step
1/1 [==============================] - 0s 140ms/step
1/1 [==============================] - 0s 130ms/step
1/1 [==============================] - 0s 129ms/step
1/1 [==============================] - 0s 152ms/step
1/1 [==============================] - 0s 142ms/step
1/1 [==============================] - 0s 132ms/step
1/1 [==============================] - 0s 131ms/step
1/1 [==============================] - 0s 141ms/step
1/1 [==============================] - 0s 155ms/step
1/1 [==============================] - 0s 144ms/step
1/1 [==============================] - 0s 137ms/step
1/1 [==============================] - 0s 132ms/step
1/1 [==============================] - 0s 146ms/step
1/1 [==============================] - 0s 145ms/step
1/1 [==============================] - 0s 142ms/step
1/1 [==============================] - 0s 133ms/step
1/1 [==============================] - 0s 143ms/step
1/1 [==============================] - 0s 140ms/step
1/1 [==============================] - 0s 152ms/step
1/1 [==============================] - 0s 143ms/step
1/1 [==============================] - 0s 143ms/step
1/1 [==============================] - 0s 143ms/step
1/1 [==============================] - 0s 144ms/step
```

```python
# explain predictions
explanation=explainer.explain_instance(img_array[0],model.predict,top_labels=1,hide_color=0,num_samples=1000)
```


{"model_id":"38e09d9565ef41299869c41a966823ce","version_major":2,"version_minor":0}


```
1/1 [==============================] - 0s 141ms/step
1/1 [==============================] - 0s 156ms/step
1/1 [==============================] - 0s 150ms/step
1/1 [==============================] - 0s 146ms/step
1/1 [==============================] - 0s 141ms/step
```

```
1/1 [==============================] - 0s 158ms/step
1/1 [==============================] - 0s 161ms/step
1/1 [==============================] - 0s 167ms/step
1/1 [==============================] - 0s 201ms/step
1/1 [==============================] - 0s 220ms/step
1/1 [==============================] - 0s 186ms/step
1/1 [==============================] - 0s 202ms/step
1/1 [==============================] - 0s 185ms/step
1/1 [==============================] - 0s 197ms/step
1/1 [==============================] - 0s 160ms/step
1/1 [==============================] - 0s 143ms/step
1/1 [==============================] - 0s 140ms/step
1/1 [==============================] - 0s 135ms/step
1/1 [==============================] - 0s 156ms/step
1/1 [==============================] - 0s 132ms/step
1/1 [==============================] - 0s 134ms/step
1/1 [==============================] - 0s 132ms/step
1/1 [==============================] - 0s 143ms/step
1/1 [==============================] - 0s 164ms/step
1/1 [==============================] - 0s 130ms/step
1/1 [==============================] - 0s 143ms/step
1/1 [==============================] - 0s 139ms/step
1/1 [==============================] - 0s 145ms/step
1/1 [==============================] - 0s 156ms/step
1/1 [==============================] - 0s 137ms/step
1/1 [==============================] - 0s 131ms/step
1/1 [==============================] - 0s 145ms/step
1/1 [==============================] - 0s 158ms/step
1/1 [==============================] - 0s 156ms/step
1/1 [==============================] - 0s 139ms/step
1/1 [==============================] - 0s 145ms/step
1/1 [==============================] - 0s 142ms/step
1/1 [==============================] - 0s 140ms/step
1/1 [==============================] - 0s 142ms/step
1/1 [==============================] - 0s 140ms/step
1/1 [==============================] - 0s 134ms/step
1/1 [==============================] - 0s 146ms/step
1/1 [==============================] - 0s 143ms/step
1/1 [==============================] - 0s 140ms/step
1/1 [==============================] - 0s 145ms/step
1/1 [==============================] - 0s 142ms/step
1/1 [==============================] - 0s 149ms/step
1/1 [==============================] - 0s 144ms/step
1/1 [==============================] - 0s 133ms/step
1/1 [==============================] - 0s 146ms/step
1/1 [==============================] - 0s 140ms/step
1/1 [==============================] - 0s 150ms/step
1/1 [==============================] - 0s 142ms/step
1/1 [==============================] - 0s 140ms/step
```

```
1/1 [==============================] - 0s 146ms/step
1/1 [==============================] - 0s 138ms/step
1/1 [==============================] - 0s 141ms/step
1/1 [==============================] - 0s 141ms/step
1/1 [==============================] - 0s 150ms/step
1/1 [==============================] - 0s 151ms/step
1/1 [==============================] - 0s 143ms/step
1/1 [==============================] - 0s 156ms/step
1/1 [==============================] - 0s 129ms/step
1/1 [==============================] - 0s 143ms/step
1/1 [==============================] - 0s 136ms/step
1/1 [==============================] - 0s 146ms/step
1/1 [==============================] - 0s 172ms/step
1/1 [==============================] - 0s 134ms/step
1/1 [==============================] - 0s 154ms/step
1/1 [==============================] - 0s 138ms/step
1/1 [==============================] - 0s 145ms/step
1/1 [==============================] - 0s 142ms/step
1/1 [==============================] - 0s 137ms/step
1/1 [==============================] - 0s 156ms/step
1/1 [==============================] - 0s 151ms/step
1/1 [==============================] - 0s 172ms/step
1/1 [==============================] - 0s 153ms/step
1/1 [==============================] - 0s 154ms/step
1/1 [==============================] - 0s 151ms/step
1/1 [==============================] - 0s 158ms/step
1/1 [==============================] - 0s 156ms/step
1/1 [==============================] - 0s 137ms/step
1/1 [==============================] - 0s 156ms/step
1/1 [==============================] - 0s 161ms/step
1/1 [==============================] - 0s 159ms/step
1/1 [==============================] - 0s 149ms/step
1/1 [==============================] - 0s 148ms/step
1/1 [==============================] - 0s 150ms/step
1/1 [==============================] - 0s 156ms/step
1/1 [==============================] - 0s 149ms/step
1/1 [==============================] - 0s 177ms/step
1/1 [==============================] - 0s 143ms/step
1/1 [==============================] - 0s 153ms/step
1/1 [==============================] - 0s 146ms/step
1/1 [==============================] - 0s 134ms/step
1/1 [==============================] - 0s 145ms/step
1/1 [==============================] - 0s 140ms/step
1/1 [==============================] - 0s 155ms/step
1/1 [==============================] - 0s 151ms/step
1/1 [==============================] - 0s 139ms/step
```

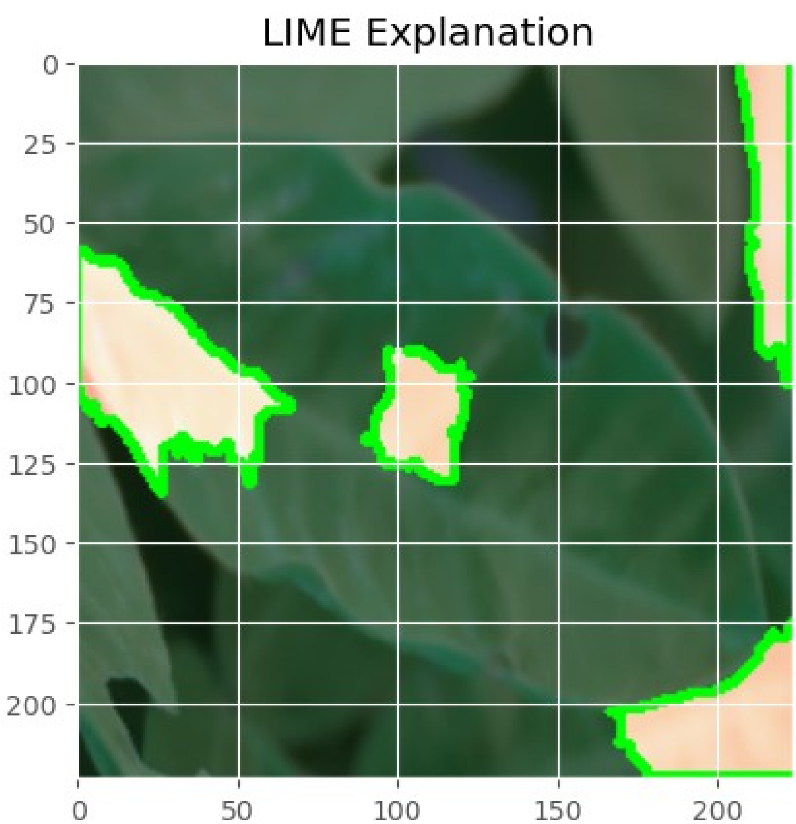
```python
#visualise LIME explanations with cv2.polylines
temp,mask=explanation.get_image_and_mask(explanation.top_labels[0],
positive_only=False,num_features=5,hide_rest=False)
```

```
#convert temp to uint8 for cv2.polylines
temp=(temp/2+0.5*mask[:,:,np.newaxis])*255
temp=temp.astype(np.uint8)

#find contours in the mask
contours,_=cv2.findContours(mask.astype(np.uint8),cv2.RETR_EXTERNAL,cv
2.CHAIN_APPROX_SIMPLE)

#Draw contours on the image
cv2.polylines(temp,contours,isClosed=False,color=(0,255,0),thickness=2
)
```

```
array([[[ 66,  98,  73],
        [ 66,  98,  73],
        [ 66,  97,  72],
        ...,
        [249, 213, 184],
        [  0, 255,   0],
        [  0, 255,   0]],

       [[ 66,  97,  72],
        [ 66,  97,  72],
        [ 66,  97,  72],
        ...,
        [249, 213, 185],
        [  0, 255,   0],
        [  0, 255,   0]],

       [[ 66,  97,  72],
        [ 66,  97,  72],
        [ 66,  97,  72],
        ...,
        [249, 214, 185],
        [  0, 255,   0],
        [  0, 255,   0]],

       ...,

       [[ 11,  33,  16],
        [ 13,  33,  17],
        [ 17,  37,  20],
        ...,
        [249, 213, 180],
        [249, 212, 180],
        [249, 212, 180]],

       [[ 13,  32,  16],
        [ 15,  34,  18],
        [ 17,  36,  19],
        ...,
```

```
        [  0, 255,   0],
        [  0, 255,   0],
        [  0, 255,   0]],

       [[ 16,  33,  17],
        [ 15,  33,  17],
        [ 17,  36,  19],
        ...,
        [  0, 255,   0],
        [  0, 255,   0],
        [  0, 255,   0]]], dtype=uint8)
```

```
#display the image
plt.imshow(temp)
plt.title('LIME Explanation ')
#plt.savefig('F:/Paper6_vit+cnn/results/P6_proposed_lime1.jpg',dpi=600
)
plt.show()
```




```
# Code for turning off x and y labels
plt.imshow(temp)
plt.title('LIME Explanation ')
plt.xticks([])  # Turn off x labels
plt.yticks([])  # Turn off y labels
```

```
#plt.savefig('F:/Paper6_vit+cnn/results/P6_Xception_lime.jpg',
dpi=600)
plt.show()
```

LIME Explanation

```
import datetime
cur_date=datetime.datetime.now()
print("current date and time",cur_date)
```

```
current date and time 2024-02-18 18:26:07.315914
```