DATA AUGMENTATION

```
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Mounted at /content/gdrive
```

```
!pip install augmentor
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting augmentor
  Downloading Augmentor-0.2.12-py2.py3-none-any.whl (38 kB)
Requirement already satisfied: tqdm>=4.9.0 in /usr/local/lib/python3.9/dist-packages (from augmentor) (4.6
Requirement already satisfied: Pillow>=5.2.0 in /usr/local/lib/python3.9/dist-packages (from augmentor) (8
Requirement already satisfied: numpy>=1.11.0 in /usr/local/lib/python3.9/dist-packages (from augmentor) (1
Installing collected packages: augmentor
Successfully installed augmentor-0.2.12
```

```
!pip install augmentor
# Importing necessary library
import Augmentor
# Passing the path of the image directory
p = Augmentor.Pipeline('/content/gdrive/MyDrive/Hidden hunger/Train_dataset')

# Defining augmentation parameters and generating 5 samples
p.zoom(probability = 0.5, min_factor = 0.8, max_factor = 1.5)
p.flip_top_bottom(probability=0.5)
p.sample(7000)
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: augmentor in /usr/local/lib/python3.9/dist-packages (0.2.12)
Requirement already satisfied: numpy>=1.11.0 in /usr/local/lib/python3.9/dist-packages (from augmentor) (1
Requirement already satisfied: Pillow>=5.2.0 in /usr/local/lib/python3.9/dist-packages (from augmentor) (8
Requirement already satisfied: tqdm>=4.9.0 in /usr/local/lib/python3.9/dist-packages (from augmentor) (4.6
Initialised with 6600 image(s) found.
Output directory set to /content/gdrive/MyDrive/Hidden hunger/Train_dataset/output.Processing <PIL.Image.I
```

```
pip install split-folders
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting split-folders
  Downloading split_folders-0.5.1-py3-none-any.whl (8.4 kB)
Installing collected packages: split-folders
Successfully installed split-folders-0.5.1
```

```
import splitfolders

input_folder = r'/content/gdrive/MyDrive/Hidden hunger/original dataset'
splitfolders.ratio(input_folder, output= r'/content/gdrive/MyDrive/Hidden hunger/split dataset_orig',
                   seed=42, ratio=(.8, .2),
                   group_prefix=None)
```

```
Copying files: 5827 files [02:17, 42.52 files/s]
```

```
import matplotlib.pyplot as plt
import numpy as np
import os
import PIL
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.applications.inception_v3 import InceptionV3
```

```
from tensorflow.keras.applications.inception_v3 import preprocess_input
from tensorflow.keras import layers
from tensorflow.python.keras.layers import Dense, Flatten
from tensorflow.keras.models import Sequential
from tensorflow.keras.optimizers import Adam


train_path = '/content/gdrive/MyDrive/Hidden hunger/split dataset_orig/train'
test_path = '/content/gdrive/MyDrive/Hidden hunger/split dataset_orig/val'



from tensorflow.keras.layers import Input,Lambda,Dense,Flatten
from tensorflow.keras.models import Model
from tensorflow.keras.preprocessing import image


IMAGE_SIZE = [299,299]


# Use the Image Data Generator to import the images from the dataset
from tensorflow.keras.preprocessing.image import ImageDataGenerator

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

test_datagen = ImageDataGenerator(rescale = 1./255,
                                  )


# Make sure you provide the same target size as initialied for the image size
train_set=train_datagen.flow_from_directory('/content/gdrive/MyDrive/Hidden hunger/split dataset_orig/train',tar
                                            batch_size=32,
                                            class_mode='categorical')
```

```
Found 4659 images belonging to 6 classes.
```

```
test_set = test_datagen.flow_from_directory('/content/gdrive/MyDrive/Hidden hunger/split dataset_orig/val',
                                            target_size = (299, 299),
                                            batch_size = 32,
                                            class_mode = 'categorical')
```

```
Found 1168 images belonging to 6 classes.
```

```
class_name = train_set.class_indices
print(class_name)
```

```
{'Iodine Deficiency': 0, 'Vitamin - B12  Deficiency': 1, 'Vitamin D deficiency': 2, 'Zinc Deficiency': 3,
```

```
inceptionv3_model = Sequential()
pretrained_model = InceptionV3(input_shape=(299,299,3), weights='imagenet', include_top=False,
                   pooling='avg',classes=6)
for layer in pretrained_model.layers:
        layer.trainable=False

inceptionv3_model.add(pretrained_model)
inceptionv3_model.add(Flatten())
inceptionv3_model.add(Dense(512, activation='relu'))
inceptionv3_model.add(Dense(6, activation='softmax'))
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/inception_v3/inception_
87910968/87910968 [==============================] - 5s 0us/step
```

```
inceptionv3_model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 inception_v3 (Functional)   (None, 2048)              21802784

 flatten (Flatten)           (None, 2048)              0

 dense (Dense)               (None, 512)               1049088

 dense_1 (Dense)             (None, 6)                 3078

=================================================================
Total params: 22,854,950
Trainable params: 1,052,166
Non-trainable params: 21,802,784
_________________________________________________________________
```

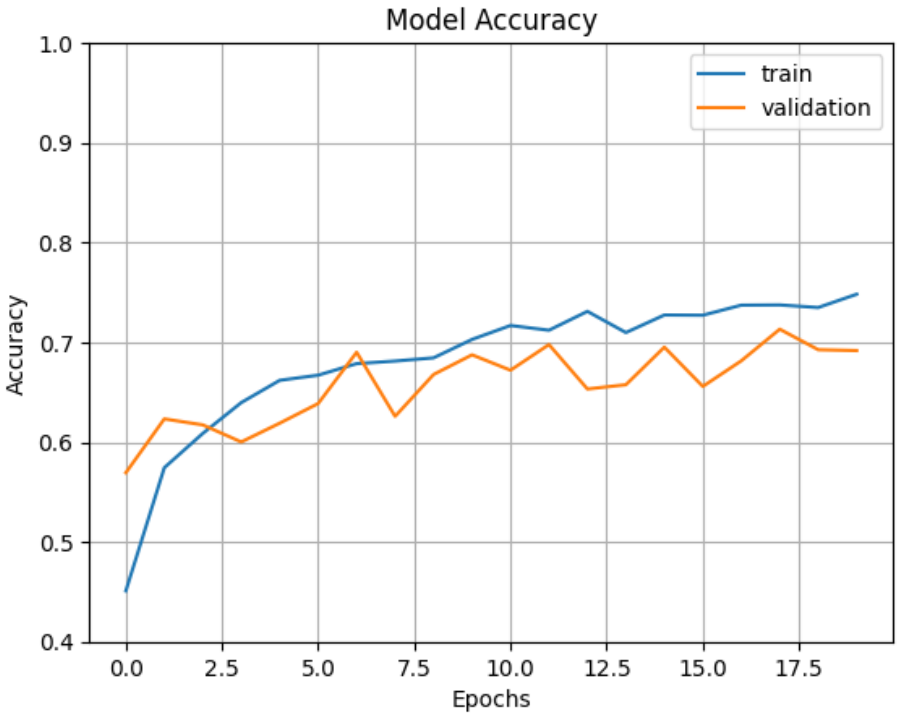
```
inceptionv3_model.compile(optimizer=Adam(learning_rate=0.01),loss='categorical_crossentropy',metrics=['accuracy'
```

```
epochs=20
history = inceptionv3_model.fit(train_set,validation_data=test_set,epochs=epochs)
```

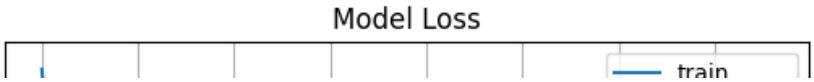
```
Epoch 1/20
146/146 [==============================] - 150s 927ms/step - loss: 2.8875 - accuracy: 0.4510 - val_loss: 1
Epoch 2/20
146/146 [==============================] - 130s 885ms/step - loss: 1.1381 - accuracy: 0.5744 - val_loss: 1
Epoch 3/20
146/146 [==============================] - 131s 897ms/step - loss: 1.0548 - accuracy: 0.6087 - val_loss: 1
Epoch 4/20
146/146 [==============================] - 130s 890ms/step - loss: 0.9788 - accuracy: 0.6396 - val_loss: 1
Epoch 5/20
146/146 [==============================] - 131s 897ms/step - loss: 0.9026 - accuracy: 0.6619 - val_loss: 1
Epoch 6/20
146/146 [==============================] - 132s 900ms/step - loss: 0.8741 - accuracy: 0.6671 - val_loss: 1
Epoch 7/20
146/146 [==============================] - 127s 872ms/step - loss: 0.8702 - accuracy: 0.6787 - val_loss: 0
Epoch 8/20
146/146 [==============================] - 129s 881ms/step - loss: 0.8673 - accuracy: 0.6813 - val_loss: 1
Epoch 9/20
146/146 [==============================] - 127s 871ms/step - loss: 0.8501 - accuracy: 0.6845 - val_loss: 0
Epoch 10/20
146/146 [==============================] - 126s 864ms/step - loss: 0.8136 - accuracy: 0.7029 - val_loss: 0
Epoch 11/20
146/146 [==============================] - 125s 859ms/step - loss: 0.7748 - accuracy: 0.7169 - val_loss: 0
Epoch 12/20
146/146 [==============================] - 126s 863ms/step - loss: 0.7732 - accuracy: 0.7122 - val_loss: 0
Epoch 13/20
146/146 [==============================] - 126s 865ms/step - loss: 0.7521 - accuracy: 0.7311 - val_loss: 0
Epoch 14/20
146/146 [==============================] - 125s 859ms/step - loss: 0.7815 - accuracy: 0.7098 - val_loss: 0
Epoch 15/20
146/146 [==============================] - 125s 858ms/step - loss: 0.7348 - accuracy: 0.7274 - val_loss: 0
Epoch 16/20
146/146 [==============================] - 125s 856ms/step - loss: 0.7490 - accuracy: 0.7272 - val_loss: 0
Epoch 17/20
146/146 [==============================] - 125s 859ms/step - loss: 0.7162 - accuracy: 0.7373 - val_loss: 0
Epoch 18/20
146/146 [==============================] - 125s 858ms/step - loss: 0.7228 - accuracy: 0.7375 - val_loss: 0
Epoch 19/20
146/146 [==============================] - 127s 869ms/step - loss: 0.7247 - accuracy: 0.7349 - val_loss: 0
Epoch 20/20
146/146 [==============================] - 126s 864ms/step - loss: 0.6807 - accuracy: 0.7482 - val_loss: 0
```

```
inceptionv3_model.save("/content/gdrive/MyDrive/Hidden hunger/split dataset_orig/inceptionv3_orig.h5")
```

```
fig1 = plt.gcf()
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.axis(ymin=0.4,ymax=1)
plt.grid()
plt.title('Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epochs')
plt.legend(['train', 'validation'])
plt.show()
```




```
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.grid()
plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epochs')
plt.legend(['train', 'validation'])
plt.show()
```




```
from keras.models import load_model
from tensorflow.keras.preprocessing import image
model=load_model("/content/gdrive/MyDrive/Hidden hunger/split dataset_orig/inceptionv3_orig.h5")
```

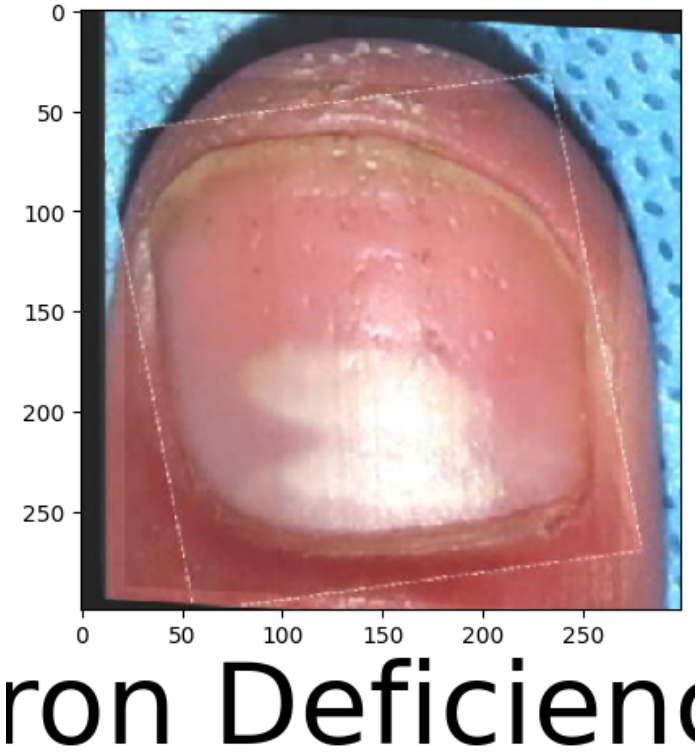
```
import numpy as np
def predictImage(filename,model):
  img1=image.load_img(filename,target_size=(299,299))
  plt.imshow(img1)
  Y=image.img_to_array(img1)
  X=np.expand_dims(Y,axis=0)
  pred=model.predict(X/255)
  print(pred)
  pred = np.array(pred)
  val = np.argmax(pred)
  print(val)
  if (val==0).all():
    plt.xlabel("Iodine Deficiency",fontsize=50)
  elif (val==1).all():
    plt.xlabel("Iron Deficiency",fontsize=50)
  elif (val==2).all():
    plt.xlabel("Vitamin - B12  Deficiency",fontsize=50)
  elif (val==3).all():
    plt.xlabel("Vitamin D - Deficiency",fontsize=25)
  elif (val==4).all():
    plt.xlabel("Zinc Deficiency",fontsize=50)
  elif (val==5).all():
    plt.xlabel("healthy",fontsize=25)
```

```
predictImage("/content/gdrive/MyDrive/Hidden hunger/val/Iodine Deficiency/Iodine Deficiency_original_Screen-Shot
```

```
1/1 [==============================] - 3s 3s/step
[[1.4412683e-01 1.7236900e-02 7.4900180e-01 1.7844034e-02 7.1769923e-02
  2.0544147e-05]]
2
```

```
predictImage("/content/gdrive/MyDrive/Hidden hunger/val/Iron Deficiency/Iron Deficiency_original_112_JPG.rf.4a61
```

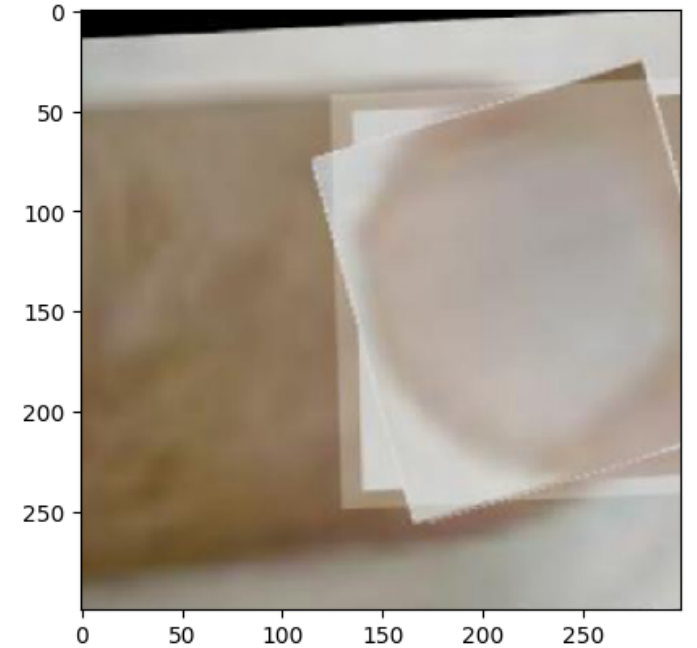
```
1/1 [==============================] - 0s 47ms/step
[[1.2578721e-01 7.1040863e-01 2.5478872e-02 8.6294105e-03 1.2921669e-01
  4.7909268e-04]]
1
```



ron Deficien

```
predictImage("/content/gdrive/MyDrive/Hidden hunger/val/Vitamin - B12  Deficiency/Vitamin - B12  Deficiency_orig
```

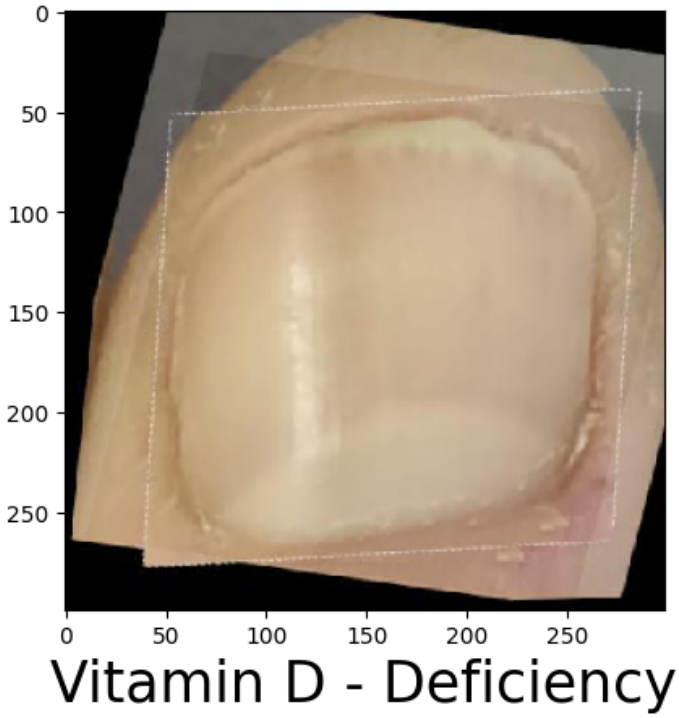
```
1/1 [==============================] - 0s 45ms/step
[[3.1044530e-02 3.8101595e-02 9.2412591e-01 1.9836647e-04 8.3671941e-04
  5.6927935e-03]]
2
```



in - B12 Def

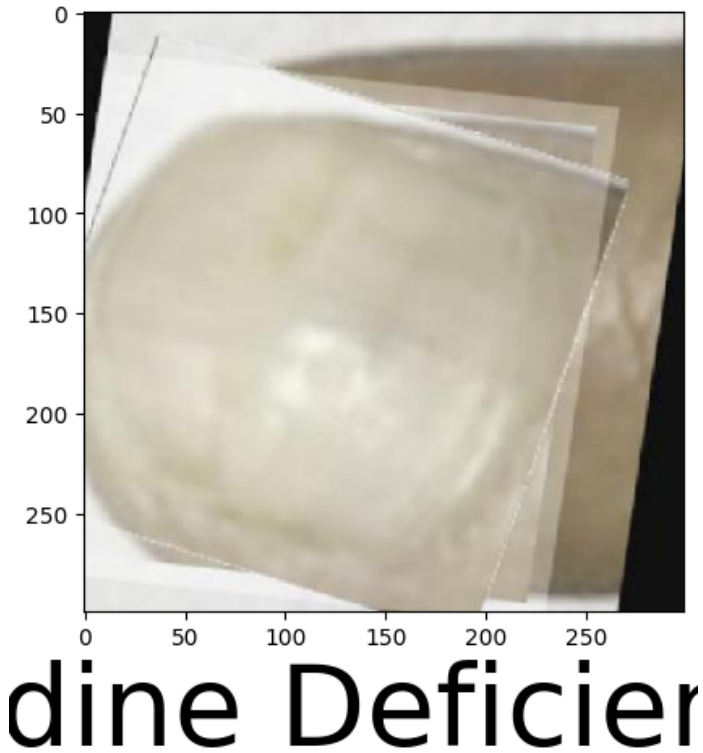
```
predictImage("/content/gdrive/MyDrive/Hidden hunger/val/Vitamin D - Deficiency/Vitamin D - Deficiency_original_S
```

```
1/1 [==============================] - 0s 28ms/step
[[0.00152076 0.29042125 0.03032159 0.5030235  0.1166429  0.05807005]]
3
```



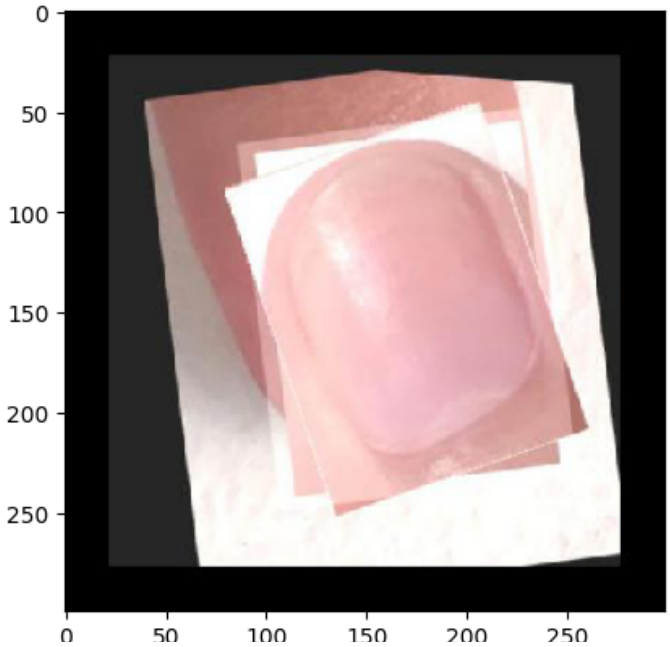

```
predictImage("/content/gdrive/MyDrive/Hidden hunger/val/Zinc Deficiency/Zinc Deficiency_original_Screen-Shot-202
```

```
1/1 [==============================] - 0s 27ms/step
[[0.4563299  0.04907864 0.33956146 0.0005035  0.15108983 0.00343669]]
0
```




```
predictImage("/content/gdrive/MyDrive/Hidden hunger/val/healthy/healthy_original_Screen-Shot-2021-11-15-at-12-52
```

```
1/1 [==============================] - 0s 97ms/step
[[0.00148466 0.02746078 0.01187978 0.00133189 0.2960549  0.6617879 ]]
5
```




```
from sklearn.metrics import ConfusionMatrixDisplay
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
import numpy as np


saved_model = load_model("/content/gdrive/MyDrive/Hidden hunger/split dataset_orig/inceptionv3_orig.h5")
all_y_pred = []
all_y_true = []

for i in range(len(test_set)):
    x, y = test_set[i]
    y_pred = saved_model.predict(x)

    all_y_pred.append(y_pred)
    all_y_true.append(y)

all_y_pred = np.concatenate(all_y_pred, axis=0)
all_y_true = np.concatenate(all_y_true, axis=0)
```

```
1/1 [==============================] - 2s 2s/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 47ms/step
1/1 [==============================] - 0s 46ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 46ms/step
1/1 [==============================] - 0s 57ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 57ms/step
1/1 [==============================] - 0s 45ms/step
1/1 [==============================] - 0s 45ms/step
1/1 [==============================] - 0s 56ms/step
1/1 [==============================] - 0s 44ms/step
1/1 [==============================] - 0s 45ms/step
1/1 [==============================] - 0s 55ms/step
1/1 [==============================] - 0s 43ms/step
1/1 [==============================] - 0s 42ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 56ms/step
1/1 [==============================] - 0s 44ms/step
1/1 [==============================] - 0s 44ms/step
1/1 [==============================] - 0s 45ms/step
```

```
1/1 [==============================] - 0s 44ms/step
1/1 [==============================] - 0s 55ms/step
1/1 [==============================] - 0s 83ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 79ms/step
1/1 [==============================] - 0s 72ms/step
1/1 [==============================] - 0s 74ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 75ms/step
1/1 [==============================] - 0s 73ms/step
1/1 [==============================] - 0s 44ms/step
1/1 [==============================] - 1s 1s/step
```

```
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

# compute confusion matrix
cm = confusion_matrix(all_y_true.argmax(axis=1), all_y_pred.argmax(axis=1))

# create heatmap from confusion matrix
fig, ax = plt.subplots(figsize=(6,6))
sns.heatmap(cm, annot=True, cmap="Greens", fmt="d", xticklabels=train_set.class_indices.keys(),
            yticklabels=train_set.class_indices.keys(), ax=ax)

# set axis labels and title
ax.set_xlabel('Predicted labels')
ax.set_ylabel('True labels')
ax.set_title('Confusion Matrix')
```

```
Text(0.5, 1.0, 'Confusion Matrix')
```

```
from sklearn.metrics import classification_report

# Get the predicted class labels
y_pred = np.argmax(all_y_pred, axis=1)

# Get the true class labels
y_true = np.argmax(all_y_true, axis=1)

# Compute classification report
report = classification_report(y_true, y_pred, target_names=train_set.class_indices.keys())

# Print classification report
print(report)
```

```
                            precision    recall  f1-score   support

          Iodine Deficiency       0.66      0.73      0.69       157
Vitamin - B12  Deficiency       0.59      0.49      0.54       192
     Vitamin D deficiency       0.64      0.77      0.70       161
          Zinc Deficiency       1.00      0.15      0.27        91
                  healthy       0.70      0.75      0.72       148
        iron deficiency         0.75      0.84      0.79       419

                 accuracy                           0.69      1168
                macro avg       0.72      0.62      0.62      1168
             weighted avg       0.71      0.69      0.67      1168
```

```
import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# Load the saved model
model = load_model('/content/gdrive/MyDrive/Hidden hunger/split dataset_orig/inceptionv3_orig.h5')

# Create an image data generator with normalization
test_datagen = ImageDataGenerator(rescale=1./255)

# Load the test data
test_data = test_datagen.flow_from_directory(test_path, target_size=(299, 299), batch_size=32, shuffle=False)

# Use the predict method to obtain predictions on the test data
y_pred = model.predict(test_data)

# Get the predicted class labels
y_pred_classes = np.argmax(y_pred, axis=1)

# Get the true class labels
y_true = test_data.classes

# Compute the test accuracy
test_acc = np.mean(y_pred_classes == y_true)

# Use the predict method to obtain predictions on the training data
y_pred_train = model.predict(train_set)

# Get the predicted class labels
y_pred_classes_train = np.argmax(y_pred_train, axis=1)

# Get the true class labels
y_true_train = train_set.classes

# Compute the train accuracy
train_acc = np.mean(y_pred_classes_train == y_true_train)

# Print the train and test accuracy
```

```
print('Train accuracy:', train_acc)
print('Test accuracy:', test_acc)
```

```
Found 1168 images belonging to 6 classes.
37/37 [==============================] - 9s 206ms/step
146/146 [==============================] - 119s 816ms/step
Train accuracy: 0.22622880446447735
Test accuracy: 0.6917808219178082
```

✓ 1m 57s completed at 12:25 AM