```
!wget -O "butterfly_classification_ai_challenge-dataset.zip" "https://dockship-job-models.s3.
```

```
--2020-10-06 02:54:22--  https://dockship-job-models.s3.ap-south-1.amazonaws.com/d937cfa
Resolving dockship-job-models.s3.ap-south-1.amazonaws.com (dockship-job-models.s3.ap-sou
Connecting to dockship-job-models.s3.ap-south-1.amazonaws.com (dockship-job-models.s3.ap
HTTP request sent, awaiting response... 200 OK
Length: 509734503 (486M) [binary/octet-stream]
Saving to: ‘butterfly_classification_ai_challenge-dataset.zip’

butterfly_classific 100%[===================>] 486.12M  13.0MB/s    in 40s

2020-10-06 02:55:02 (12.2 MB/s) - ‘butterfly_classification_ai_challenge-dataset.zip’ sa
```

```
#unzip file
! unzip -q "butterfly_classification_ai_challenge-dataset.zip"
```

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Input,Dense,Conv2D,Flatten,Dropout,MaxPool2D,Flatten,Acti
from tensorflow.keras.applications.inception_v3 import InceptionV3
from tensorflow.keras.applications.inception_v3 import preprocess_input
from tensorflow.keras.preprocessing import image
from tensorflow.keras.preprocessing.image import ImageDataGenerator,load_img
from tensorflow.keras.models import Model
from tensorflow.keras.applications import InceptionResNetV2
from keras.preprocessing.image import load_img, img_to_array
from glob import glob
import pandas as pd
import numpy as np
import os
import seaborn as sns
import matplotlib.pyplot as plt
```

```
train_path='/content/DATA/TRAIN'
test_path='/content/DATA/TEST'
```

```
folders = glob('DATA/TRAIN/*')
len(folders)
```

```
50
```

```
print("Train dataset size: ", len(os.listdir(train_path)))
print("Test datsest size: ", len(os.listdir(test_path)))
```

```
Train dataset size:  50
Test datsest size:  500
```

```
#image data generator to import the images from the dataset

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

                                fill_mode='nearest')


test_datagen = ImageDataGenerator(rescale = 1./255)


training_set = train_datagen.flow_from_directory(train_path,
                                                 target_size = (224, 224),
                                                 batch_size = 128,
                                                 class_mode = 'categorical')
```

```
Found 4479 images belonging to 50 classes.
```

```
from tensorflow.keras.callbacks import EarlyStopping,ReduceLROnPlateau
early_stopping=EarlyStopping(patience=4,verbose=1,restore_best_weights=True)
reduce_lr=ReduceLROnPlateau(factor=0.1,patience=3,verbose=1)

callbacks = [early_stopping, reduce_lr]
```

```
preds_df=pd.DataFrame(columns=['Filename']+list(folders))
preds_df['Filename']=[path for path in os.listdir(test_path)]
preds_df.head()
```

	Filename	DATA/TRAIN/sixspot burnet	DATA/TRAIN/orange oakleaf	DATA/TRAIN/yellow swallow tail	DATA/TRAIN/strai qu
0	104.jpg	NaN	NaN	NaN	N
1	436.jpg	NaN	NaN	NaN	N
2	480.jpg	NaN	NaN	NaN	N
3	272.jpg	NaN	NaN	NaN	N
4	222.jpg	NaN	NaN	NaN	N

```
import os
```

```
test_datagen = ImageDataGenerator(rescale=1./255)
test_generator = test_datagen.flow_from_dataframe(dataframe=preds_df,
                                                 shuffle=False,
                                                 shuffle=False,
                                                 directory=test_path,
                                                 x_col='Filename',
                                                 y_col=None,
                                                 class_mode=None,
                                                 target_size=(224, 224),
                                                 batch_size=1)
```

```
Found 500 validated image filenames.
```

InceptionV3

```
image_size=[224,224]
inception=InceptionV3(input_shape=image_size+[3],weights='imagenet',include_top=False)


#don't train existing weights
for layer in inception.layers:
    layer.trainable=False


#our layers
x=Flatten()(inception.output)


prediction=Dense(len(folders),activation='softmax')(x)


model=Model(inputs=inception.input,outputs=prediction)
```

```
model.summary()
```

```
conv2d_401 (Conv2D)             (None, 25, 25, 96)   82944       activation_400[0][0]
__________________________________________________________________________________________________
conv2d_402 (Conv2D)             (None, 25, 25, 32)   6144        average_pooling2d_19
__________________________________________________________________________________________________
batch_normalization_396 (BatchN (None, 25, 25, 64)   192         conv2d_396[0][0]
__________________________________________________________________________________________________
batch_normalization_398 (BatchN (None, 25, 25, 64)   192         conv2d_398[0][0]
__________________________________________________________________________________________________
batch_normalization_401 (BatchN (None, 25, 25, 96)   288         conv2d_401[0][0]
__________________________________________________________________________________________________

batch_normalization_402 (BatchN (None, 25, 25, 32)   96          conv2d_402[0][0]
__________________________________________________________________________________________________
activation_396 (Activation)     (None, 25, 25, 64)   0           batch_normalization_
__________________________________________________________________________________________________
activation_398 (Activation)     (None, 25, 25, 64)   0           batch_normalization_
__________________________________________________________________________________________________
activation_401 (Activation)     (None, 25, 25, 96)   0           batch_normalization_
```

```
activation_402 (Activation)     (None, 25, 25, 32)   0           batch_normalization_4

mixed0 (Concatenate)            (None, 25, 25, 256)  0           activation_396[0][0]
                                                                 activation_398[0][0]
                                                                 activation_401[0][0]
                                                                 activation_402[0][0]

conv2d_406 (Conv2D)             (None, 25, 25, 64)   16384       mixed0[0][0]

batch_normalization_406 (BatchN (None, 25, 25, 64)   192         conv2d_406[0][0]

activation_406 (Activation)     (None, 25, 25, 64)   0           batch_normalization_4

conv2d_404 (Conv2D)             (None, 25, 25, 48)   12288       mixed0[0][0]

conv2d_407 (Conv2D)             (None, 25, 25, 96)   55296       activation_406[0][0]

batch_normalization_404 (BatchN (None, 25, 25, 48)   144         conv2d_404[0][0]

batch_normalization_407 (BatchN (None, 25, 25, 96)   288         conv2d_407[0][0]

activation_404 (Activation)     (None, 25, 25, 48)   0           batch_normalization_4

activation_407 (Activation)     (None, 25, 25, 96)   0           batch_normalization_4

average_pooling2d_20 (AveragePo (None, 25, 25, 256)  0           mixed0[0][0]

conv2d_403 (Conv2D)             (None, 25, 25, 64)   16384       mixed0[0][0]

conv2d_405 (Conv2D)             (None, 25, 25, 64)   76800       activation_404[0][0]

conv2d_408 (Conv2D)             (None, 25, 25, 96)   82944       activation_407[0][0]

conv2d_409 (Conv2D)             (None, 25, 25, 64)   16384       average_pooling2d_20

batch_normalization_403 (BatchN (None, 25, 25, 64)   192         conv2d_403[0][0]

batch_normalization_405 (BatchN (None, 25, 25, 64)   192         conv2d_405[0][0]

batch_normalization_408 (BatchN (None, 25, 25, 96)   288         conv2d_408[0][0]
```

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

```
r = model.fit_generator(
  training_set,
  validation_data=test_generator,
  epochs=2,
  steps_per_epoch=len(training_set),
  validation_steps=len(test_generator)
)
```

```
Epoch 1/2
35/35 [==============================] - 73s 2s/step - loss: 0.4996 - accuracy: 0.9455 -
```

```
Epoch 2/2
35/35 [==============================] - 73s 2s/step - loss: 0.5017 - accuracy: 0.9420 -
```

```
v3_test_predictions = model.predict_generator(test_generator, steps = test_generator.n, verbo
```

```
  1/500 [..............................] - ETA: 0sWARNING:tensorflow:Callbacks method `c
500/500 [==============================] - 11s 23ms/step
```

submission file

```
v3_test_predictions
```

```
array([[0.0000000e+00, 0.0000000e+00, 0.0000000e+00, ..., 0.0000000e+00,
        2.9075730e-36, 0.0000000e+00],
       [5.1007542e-27, 7.5714494e-13, 1.4100356e-33, ..., 2.5536914e-19,
        1.3688479e-21, 3.4606312e-23],
       [5.2913056e-15, 1.9073778e-14, 1.8785876e-29, ..., 8.8718500e-24,
        5.0662476e-15, 2.0063254e-21],
       ...,
       [1.2412645e-34, 9.9999964e-01, 1.3642745e-30, ..., 1.6526193e-29,
        6.2130085e-21, 1.1586156e-19],
       [0.0000000e+00, 2.2499756e-10, 0.0000000e+00, ..., 1.1738620e-34,
        5.8283043e-22, 6.7173495e-15],
       [1.6037635e-24, 3.0241080e-22, 0.0000000e+00, ..., 2.3457098e-29,
        1.8951321e-08, 3.3070011e-19]], dtype=float32)
```

```
result_v=np.argmax(v3_test_predictions,axis=1)
result_v=pd.Series(result_v,name='Label')
result_v
```

```
0      20
1       8
2      45
3      35
4       8
       ..
495     7
496    46
497     1
498    13
499    24
Name: Label, Length: 500, dtype: int64
```

```
output=pd.DataFrame()
```

```
output['Filename'] = [path for path in os.listdir(test_path)]
output.head()
```

	Filename
0	104.jpg
1	436.jpg
2	480.jpg
3	272.jpg
4	222.jpg

```
output=pd.concat([output['Filename'],result_v],axis=1)
output.head()
```

	Filename	Label
0	104.jpg	20
1	436.jpg	8
2	480.jpg	45
3	272.jpg	35
4	222.jpg	8

```
class_dict = training_set.class_indices
class_dict
```

```
{'adonis': 0,
 'american snoot': 1,
 'an 88': 2,
 'banded peacock': 3,
 'beckers white': 4,
 'black hairstreak': 5,
 'cabbage white': 6,
 'chestnut': 7,
 'clodius parnassian': 8,
 'clouded sulphur': 9,
 'copper tail': 10,
 'crecent': 11,
 'crimson patch': 12,
 'eastern coma': 13,
 'gold banded': 14,
 'great eggfly': 15,
 'grey hairstreak': 16,
 'indra swallow': 17,
 'julia': 18,
 'large marble': 19,
 'malachite': 20,
 'mangrove skipper': 21,
 'metalmark': 22,
 'monarch': 23,
 'morning cloak': 24,
 'orange oakleaf': 25,
 'orange tip': 26,
 'orchard swallow': 27,
 'painted lady': 28,
 'paper kite': 29,
 'peacock': 30,
 'pine white': 31,
 'pipevine swallow': 32,
 'purple hairstreak': 33,
 'question mark': 34,
 'red admiral': 35,
 'red spotted purple': 36,
 'scarce swallow': 37,
 'silver spot skipper': 38,
 'sixspot burnet': 39,
 'skipper': 40,
 'sootywing': 41,
 'southern dogface': 42,
 'straited queen': 43,
 'two barred flasher': 44,
 'ulyses': 45,
 'viceroy': 46,
 'wood satyr': 47,
 'yellow swallow tail': 48,
 'zebra long wing': 49}
```

```
output['Label']=output['Label'].replace(1,'american snoot')
output['Label']=output['Label'].replace(0,'adonis')
output['Label']=output['Label'].replace(2,'an 88')
output['Label']=output['Label'].replace(3,'banded peacock')
output['Label']=output['Label'].replace(4,'beckers white')
output['Label']=output['Label'].replace(5,'black hairstreak')
output['Label']=output['Label'].replace(6,'cabbage white')
output['Label']=output['Label'].replace(7,'chestnut')
output['Label']=output['Label'].replace(8,'clodius parnassian')
output['Label']=output['Label'].replace(9,'clouded sulphur')
output['Label']=output['Label'].replace(10,'copper tail')
output['Label']=output['Label'].replace(11,'crecent')
output['Label']=output['Label'].replace(12,'crimson patch')
output['Label']=output['Label'].replace(13,'eastern coma')
output['Label']=output['Label'].replace(14,'gold banded')
output['Label']=output['Label'].replace(15,'great eggfly')
output['Label']=output['Label'].replace(16,'grey hairstreak')
output['Label']=output['Label'].replace(17,'indra swallow')
output['Label']=output['Label'].replace(18,'julia')
output['Label']=output['Label'].replace(19,'large marble')
output['Label']=output['Label'].replace(20,'malachite')
output['Label']=output['Label'].replace(21,'mangrove skipper')
output['Label']=output['Label'].replace(22,'metalmark')
output['Label']=output['Label'].replace(23,'monarch')
output['Label']=output['Label'].replace(24,'morning cloak')
output['Label']=output['Label'].replace(25,'orange oakleaf')
output['Label']=output['Label'].replace(26,'orange tip')
```

```
output['Label']=output['Label'].replace(26,'orange tip')
output['Label']=output['Label'].replace(27,'orchard swallow')
output['Label']=output['Label'].replace(28,'painted lady')
output['Label']=output['Label'].replace(29,'paper kite')
output['Label']=output['Label'].replace(30,'peacock')
output['Label']=output['Label'].replace(31,'pine white')
output['Label']=output['Label'].replace(32,'pipevine swallow')
output['Label']=output['Label'].replace(33,'purple hairstreak')
output['Label']=output['Label'].replace(34,'question mark')
output['Label']=output['Label'].replace(35,'red admiral')
output['Label']=output['Label'].replace(36,'red spotted purple')
output['Label']=output['Label'].replace(37,'scarce swallow')
output['Label']=output['Label'].replace(38,'silver spot skipper')
output['Label']=output['Label'].replace(39,'sixspot burnet')
output['Label']=output['Label'].replace(40,'skipper')
output['Label']=output['Label'].replace(41,'sootywing')
output['Label']=output['Label'].replace(42,'southern dogface')
output['Label']=output['Label'].replace(43,'straited queen')
output['Label']=output['Label'].replace(44,'two barred flasher')
output['Label']=output['Label'].replace(45,'ulyses')
output['Label']=output['Label'].replace(46,'viceroy')
output['Label']=output['Label'].replace(47,'wood satyr')
output['Label']=output['Label'].replace(48,'yellow swallow tail')
output['Label']=output['Label'].replace(49,'zebra long wing')
```

```
output
```

	Filename	Label
0	104.jpg	malachite
1	436.jpg	clodius parnassian
2	480.jpg	ulyses
3	272.jpg	red admiral
4	222.jpg	clodius parnassian
...	...	...
495	303.jpg	chestnut
496	447.jpg	viceroy
497	090.jpg	american snoot
498	258.jpg	eastern coma
499	179.jpg	morning cloak

500 rows × 2 columns

```
submission=output
```

```
submission.to_csv('output_butterfly_final7.csv',index=False)
```