```
import tensorflow as tf
import numpy as np
import os
import matplotlib.pyplot as plt
import zipfile

PATH = os.path.join('/content/drive/MyDrive/Colab Notebooks/archive(2).zip (Unzipped Files
```

```
trainDir = os.path.join(PATH, 'seg_train/seg_train')
validDir = os.path.join(PATH, 'seg_test/seg_test')

trainData = tf.keras.utils.image_dataset_from_directory(trainDir, shuffle = True, batch_si

validData = tf.keras.utils.image_dataset_from_directory(validDir, shuffle = True, batch_si
```

```
     Found 14034 files belonging to 6 classes.
     Found 3000 files belonging to 6 classes.
```

```
valBatches = tf.data.experimental.cardinality(validData)
testData = validData.take(valBatches // 5)
validData = validData.skip(valBatches // 5)
```

```
trainData = trainData.prefetch(buffer_size = tf.data.AUTOTUNE)
validData = validData.prefetch(buffer_size = tf.data.AUTOTUNE)
testData = testData.prefetch(buffer_size = tf.data.AUTOTUNE)
```

```
data_augmentation = tf.keras.Sequential([
    tf.keras.layers.RandomFlip('horizontal'),
    tf.keras.layers.RandomRotation(0.2),
])
```

```
preprocess_input = tf.keras.applications.mobilenet_v2.preprocess_input
IMAGE_SHAPE = (150, 150) + (3,)

baseModel = tf.keras.applications.MobileNetV2(input_shape = IMAGE_SHAPE, include_top = Fal
```

```
     WARNING:tensorflow:`input_shape` is undefined or non-square, or `rows` is not in [96
```

```
imgBatch, labelBatch = next(iter(trainData))
featureBatch = baseModel(imgBatch)
```

```
baseModel.trainable = False
globalAvgLayer = tf.keras.layers.GlobalAveragePooling2D()
featureBatchAvg = globalAvgLayer(featureBatch)
```

✓ 23s completed at 8:38 PM

```
predictLayer = tf.keras.layers.Dense(1)
predictBatch = predictLayer(featureBatchAvg)


inputs = tf.keras.Input(shape = (150, 150, 3))
x = data_augmentation(inputs)
x = preprocess_input(x)
x = baseModel(x, training = False)
x = globalAvgLayer(x)
x = tf.keras.layers.Dropout(0.2)(x)
outputs = predictLayer(x)
model = tf.keras.Model(inputs, outputs)


model.compile(optimizer = tf.keras.optimizers.Adam(learning_rate = 0.0001), loss = tf.kera


loss0, acc0 = model.evaluate(validData)

     76/76 [==============================] - 587s 6s/step - loss: -1.4457 - accuracy: 0.


history = model.fit(trainData, epochs = 10, validation_data = validData)

     Epoch 1/10
     439/439 [==============================] - 1872s 4s/step - loss: -30.0407 - accuracy
     Epoch 2/10
     439/439 [==============================] - 304s 691ms/step - loss: -76.4642 - accura
     Epoch 3/10
     439/439 [==============================] - 303s 688ms/step - loss: -122.6972 - accur
     Epoch 4/10
     439/439 [==============================] - 303s 690ms/step - loss: -169.5753 - accur
     Epoch 5/10
     439/439 [==============================] - 307s 699ms/step - loss: -215.8738 - accur
     Epoch 6/10
     439/439 [==============================] - 310s 706ms/step - loss: -262.6495 - accur
     Epoch 7/10
     439/439 [==============================] - 306s 695ms/step - loss: -308.9842 - accur
     Epoch 8/10
     439/439 [==============================] - 306s 695ms/step - loss: -355.9136 - accur
     Epoch 9/10
     439/439 [==============================] - 304s 692ms/step - loss: -401.7234 - accur
     Epoch 10/10
     439/439 [==============================] - 305s 693ms/step - loss: -448.2376 - accur


baseModel.trainable = True
fineTuneAt = 100

for layer in baseModel.layers[:fineTuneAt]:
    layer.trainable = False

model.compile(optimizer = tf.keras.optimizers.RMSprop(learning_rate = 0.00001), loss = tf.
```

```
fineHistory = model.fit(trainData, epochs = 15, initial_epoch = history.epoch[-1], validat
```

```
Epoch 10/15
439/439 [==============================] - 501s 1s/step - loss: -4500.2979 - accurac
Epoch 11/15
439/439 [==============================] - 488s 1s/step - loss: -4911.9355 - accurac
Epoch 12/15
439/439 [==============================] - 488s 1s/step - loss: -5006.9990 - accurac
Epoch 13/15
439/439 [==============================] - 492s 1s/step - loss: -5083.9121 - accurac
Epoch 14/15
439/439 [==============================] - 491s 1s/step - loss: -5153.7998 - accurac
Epoch 15/15
439/439 [==============================] - 553s 1s/step - loss: -5213.1162 - accurac
```

```
fineHistory = model.fit(trainData, epochs = 20, initial_epoch = fineHistory.epoch[-1], val
```

```
Epoch 15/20
439/439 [==============================] - 505s 1s/step - loss: -5275.1484 - accurac
Epoch 16/20
439/439 [==============================] - 508s 1s/step - loss: -5333.9043 - accurac
Epoch 17/20
439/439 [==============================] - 506s 1s/step - loss: -5392.3682 - accurac
Epoch 18/20
439/439 [==============================] - 510s 1s/step - loss: -5456.2480 - accurac
Epoch 19/20
439/439 [==============================] - 538s 1s/step - loss: -5515.3237 - accurac
Epoch 20/20
439/439 [==============================] - 546s 1s/step - loss: -5572.9893 - accurac
```

```
loss, accuracy = model.evaluate(testData)

imageBatch, labelBatch = testData.as_numpy_iterator().next()
predict = model.predict_on_batch(imageBatch).flatten()

predict = tf.nn.sigmoid(predict)
predict = tf.where(predict < 0.5, 0, 1)
```

```
15/15 [==============================] - 10s 598ms/step - loss: -5903.7285 - accurac
```

Colab paid products - Cancel contracts here