MobileNet V2

```
import tensorflow as tf
from tensorflow.keras import layers, models
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import matplotlib.pyplot as plt


print("Num GPUs Available: ", len(tf.config.list_physical_devices('GPU')))


train_data_dir = '/content/drive/MyDrive/dataset'
test_data_dir = '/content/drive/MyDrive/dataset'

img_width, img_height = 128, 128
batch_size = 32

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

test_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
    train_data_dir,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='binary')

validation_generator = test_datagen.flow_from_directory(
    test_data_dir,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='binary')


base_model = MobileNetV2(input_shape=(img_width, img_height, 3), include_top=False, weights='imagenet')
base_model.trainable = False

model = models.Sequential()
model.add(base_model)
model.add(layers.GlobalAveragePooling2D())
model.add(layers.Dense(64, activation='tanh'))  # Reduce the number of neurons
model.add(layers.Dropout(0.5))
model.add(layers.Dense(1, activation='tanh'))

model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),  # Increase the learning rate
              loss='binary_crossentropy',
              metrics=['accuracy'])

epochs = 20

train_datagen = ImageDataGenerator(
    rescale=1./255,
    shear_range=0.3,  # Increased shear intensity
    zoom_range=0.3,   # Increased zoom intensity
    horizontal_flip=True)

train_generator = train_datagen.flow_from_directory(
    train_data_dir,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='binary')

history = model.fit(
    train_generator,
    steps_per_epoch=train_generator.samples // batch_size,
    epochs=epochs,
    validation_data=validation_generator,
    validation_steps=validation_generator.samples // batch_size)


acc = history.history['accuracy']
val_acc = history.history['val_accuracy']
loss = history.history['loss']
val_loss = history.history['val_loss']
```

```
epochs_range = range(1, epochs + 1)

plt.figure(figsize=(15, 6))

plt.subplot(1, 2, 1)
plt.plot(epochs_range, acc, label='Training Accuracy')
plt.plot(epochs_range, val_acc, label='Validation Accuracy')
plt.legend(loc='lower right')
plt.title('Training and Validation Accuracy')

plt.subplot(1, 2, 2)
plt.plot(epochs_range, loss, label='Training Loss')
plt.plot(epochs_range, val_loss, label='Validation Loss')
plt.legend(loc='upper right')
plt.title('Training and Validation Loss')

plt.show()
```

```
Num GPUs Available:  0
Found 1000 images belonging to 2 classes.
Found 1000 images belonging to 2 classes.
Found 1000 images belonging to 2 classes.
Epoch 1/20
31/31 [==============================] - 76s 2s/step - loss: 4.4870 - accuracy: 0.6973 - val_loss: 4.6182 - val_accuracy: 0.7006
Epoch 2/20
31/31 [==============================] - 71s 2s/step - loss: 4.6211 - accuracy: 0.7004 - val_loss: 4.6493 - val_accuracy: 0.6986
Epoch 3/20
31/31 [==============================] - 70s 2s/step - loss: 4.6370 - accuracy: 0.6994 - val_loss: 4.6337 - val_accuracy: 0.6996
Epoch 4/20
31/31 [==============================] - 69s 2s/step - loss: 4.5733 - accuracy: 0.7035 - val_loss: 4.6337 - val_accuracy: 0.6996
Epoch 5/20
31/31 [==============================] - 70s 2s/step - loss: 4.6370 - accuracy: 0.6994 - val_loss: 4.6026 - val_accuracy: 0.7016
Epoch 6/20
31/31 [==============================] - 55s 2s/step - loss: 4.6052 - accuracy: 0.7014 - val_loss: 4.5871 - val_accuracy: 0.7026
Epoch 7/20
31/31 [==============================] - 70s 2s/step - loss: 4.6052 - accuracy: 0.7014 - val_loss: 4.6026 - val_accuracy: 0.7016
Epoch 8/20
31/31 [==============================] - 69s 2s/step - loss: 4.6052 - accuracy: 0.7014 - val_loss: 4.6493 - val_accuracy: 0.6986
Epoch 9/20
31/31 [==============================] - 70s 2s/step - loss: 4.5892 - accuracy: 0.7025 - val_loss: 4.6182 - val_accuracy: 0.7006
Epoch 10/20
31/31 [==============================] - 71s 2s/step - loss: 4.5892 - accuracy: 0.7025 - val_loss: 4.6026 - val_accuracy: 0.7016
Epoch 11/20
31/31 [==============================] - 55s 2s/step - loss: 4.6849 - accuracy: 0.6963 - val_loss: 4.6337 - val_accuracy: 0.6996
Epoch 12/20
31/31 [==============================] - 69s 2s/step - loss: 4.6530 - accuracy: 0.6983 - val_loss: 4.6493 - val_accuracy: 0.6986
Epoch 13/20
31/31 [==============================] - 69s 2s/step - loss: 4.5574 - accuracy: 0.7045 - val_loss: 4.6337 - val_accuracy: 0.6996
Epoch 14/20
31/31 [==============================] - 56s 2s/step - loss: 4.6370 - accuracy: 0.6994 - val_loss: 4.6493 - val_accuracy: 0.6986
Epoch 15/20
31/31 [==============================] - 71s 2s/step - loss: 4.6370 - accuracy: 0.6994 - val_loss: 4.6182 - val_accuracy: 0.7006
Epoch 16/20
31/31 [==============================] - 70s 2s/step - loss: 4.5892 - accuracy: 0.7025 - val_loss: 4.6182 - val_accuracy: 0.7006
Epoch 17/20
31/31 [==============================] - 55s 2s/step - loss: 4.6530 - accuracy: 0.6983 - val_loss: 4.5715 - val_accuracy: 0.7036
Epoch 18/20
31/31 [==============================] - 70s 2s/step - loss: 4.5892 - accuracy: 0.7025 - val_loss: 4.6182 - val_accuracy: 0.7006
Epoch 19/20
31/31 [==============================] - 71s 2s/step - loss: 4.6849 - accuracy: 0.6963 - val_loss: 4.6337 - val_accuracy: 0.6996
Epoch 20/20
31/31 [==============================] - 70s 2s/step - loss: 4.6849 - accuracy: 0.6963 - val_loss: 4.6026 - val_accuracy: 0.7016
```