

```
# Import necessary libraries
import zipfile
import os
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from sklearn.model_selection import train_test_split
import shutil
```

```
# Upload the detection.zip file
from google.colab import files
uploaded = files.upload()
```

```
# Unzip the detection.zip
with zipfile.ZipFile('detection.zip', 'r') as zip_ref:
    zip_ref.extractall('detection_data')
```

 Choose files detection.zip

- **detection.zip**(application/x-zip-compressed) - 126465236 bytes, last modified: 28/04/2025 - 100% done

Saving detection.zip to detection.zip

```
# Create train and test folders for real and fake images
os.makedirs('dataset/train/real', exist_ok=True)
os.makedirs('dataset/train/fake', exist_ok=True)
os.makedirs('dataset/test/real', exist_ok=True)
os.makedirs('dataset/test/fake', exist_ok=True)
```

```
# Function to split data into train and test
def split_data(source_dir, train_dir, test_dir, split_ratio=0.8):
    files = os.listdir(source_dir)
    train_files, test_files = train_test_split(files, train_size=split_ratio, random_state=42)


    for file in train_files:
        shutil.copy(os.path.join(source_dir, file), train_dir)
    for file in test_files:
        shutil.copy(os.path.join(source_dir, file), test_dir)
```

```
# Splitting real and fake folders
split_data('detection_data/real', 'dataset/train/real', 'dataset/test/real')
split_data('detection_data/fake', 'dataset/train/fake', 'dataset/test/fake')
```

```
# Create Image Data Generators for training and testing
train_datagen = ImageDataGenerator(rescale=1./255)
test_datagen = ImageDataGenerator(rescale=1./255)
```

```
train_generator = train_datagen.flow_from_directory(
    'dataset/train',
    target_size=(224, 224),
    batch_size=32,
    class_mode='binary'
)
```

```
test_generator = test_datagen.flow_from_directory(
    'dataset/test',
    target_size=(224, 224),
    batch_size=32,
    class_mode='binary'
)
```

 Found 1224 images belonging to 2 classes.  
Found 306 images belonging to 2 classes.

```
# Load MobileNetV2 as base model
base_model = tf.keras.applications.MobileNetV2(input_shape=(224,224,3),
                                                include_top=False,
                                                weights='imagenet')
```

```
# Freeze the base model
```

```
base_model.trainable = False
```

Downloading data from [https://storage.googleapis.com/tensorflow/keras-applications/mobilenet\\_v2/mobilenet\\_v2\\_weights\\_tf\\_dim\\_ordering\\_tf\\_9406464/9406464](https://storage.googleapis.com/tensorflow/keras-applications/mobilenet_v2/mobilenet_v2_weights_tf_dim_ordering_tf_9406464/9406464) 0s 0us/step

```
# Build the full model
model = tf.keras.Sequential([
    base_model,
    tf.keras.layers.GlobalAveragePooling2D(),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
```

```
# Compile the model
model.compile(optimizer='adam',
              loss='binary_crossentropy',
              metrics=['accuracy'])
```

```
# Train the model
history = model.fit(
    train_generator,
    epochs=5,
    validation_data=test_generator
)
```

/usr/local/lib/python3.11/dist-packages/keras/src/trainers/data\_adapters/py\_dataset\_adapter.py:121: UserWarning: Your `PyDataset` class `self.warn_if_super_not_called()`

```
Epoch 1/5
39/39 ————— 93s 2s/step - accuracy: 0.6021 - loss: 0.6660 - val_accuracy: 0.7582 - val_loss: 0.5468
Epoch 2/5
39/39 ————— 81s 2s/step - accuracy: 0.7434 - loss: 0.5342 - val_accuracy: 0.7843 - val_loss: 0.4969
Epoch 3/5
39/39 ————— 93s 2s/step - accuracy: 0.7986 - loss: 0.4772 - val_accuracy: 0.7843 - val_loss: 0.4639
Epoch 4/5
39/39 ————— 87s 2s/step - accuracy: 0.8288 - loss: 0.4322 - val_accuracy: 0.8105 - val_loss: 0.4398
Epoch 5/5
39/39 ————— 84s 2s/step - accuracy: 0.8226 - loss: 0.4212 - val_accuracy: 0.7974 - val_loss: 0.4287
```

```
# Evaluate the model on test data
loss, accuracy = model.evaluate(test_generator)
print(f"Test Accuracy: {accuracy*100:.2f}%")
```

10/10 ————— 17s 2s/step - accuracy: 0.8137 - loss: 0.3970  
Test Accuracy: 79.74%

```
model.save('deepfake_detector_model.h5')
```

WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save\_model(model)`. This file format is consi

```
from google.colab import files
files.download('deepfake_detector_model.h5')
```



```
# from tensorflow.keras.models import load_model
# model = load_model('deepfake_detector_model.h5')
```

