

```

from google.colab import drive
import os

# 1. Mount Drive
drive.mount('/content/drive')

# 2. Check if the file is there
# Note: 'MyDrive' is the standard name for your main Drive folder
zip_path = '/content/drive/MyDrive/AI_Project_Data/archive.zip'

if os.path.exists(zip_path):
    print(f"✅ Success! Found zip file at: {zip_path}")
else:
    print(f"❌ Error: Could not find file at {zip_path}")
    print("Please check your folder name in Google Drive.")

```

Mounted at /content/drive
 ✅ Success! Found zip file at: /content/drive/MyDrive/AI_Project_Data/archive.zip

```

import zipfile

# 1. Define paths
zip_path = '/content/drive/MyDrive/AI_Project_Data/archive.zip'
extract_path = '/content/dataset'

# 2. Unzip
print(f"📁 Unzipping from Drive to {extract_path}...")
print("This might take 1-2 minutes...")

with zipfile.ZipFile(zip_path, 'r') as zip_ref:
    zip_ref.extractall(extract_path)

print("✅ Extraction Complete!")

# 3. Check what's inside to verify the 'ddata' folder
if os.path.exists(extract_path + '/ddata'):
    print("✅ Verified: 'ddata' folder exists.")
else:
    print("⚠️ Check contents:", os.listdir(extract_path))

```

📁 Unzipping from Drive to /content/dataset...
 This might take 1-2 minutes...
 ✅ Extraction Complete!
 ✅ Verified: 'ddata' folder exists.

```

import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D, Dropout
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam

# =====
# 📌 PATH CONFIGURATION
# =====
# We use the unzipped location
BASE_DIR = '/content/dataset/ddata'

TRAIN_DIR = os.path.join(BASE_DIR, 'train')
TEST_DIR = os.path.join(BASE_DIR, 'test')

# Industrial Standard Resolution
IMG_SIZE = (128, 128)
BATCH_SIZE = 32
EPOCHS = 15

# Verify
if os.path.exists(TRAIN_DIR):
    print(f"✅ Ready to train! Found Real/Fake folders in {TRAIN_DIR}")
else:
    print(f"❌ Error: Path issues. Check Cell 2 output.")

```

✅ Ready to train! Found Real/Fake folders in /content/dataset/ddata/train

```

print("🔧 Setting up Data Pipelines...")

train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,

```

```


        width_shift_range=0.2,
        height_shift_range=0.2,
        horizontal_flip=True,
        fill_mode='nearest'
    )

    test_datagen = ImageDataGenerator(rescale=1./255)

    train_generator = train_datagen.flow_from_directory(
        TRAIN_DIR,
        target_size=IMG_SIZE,
        batch_size=BATCH_SIZE,
        class_mode='binary'
    )

    validation_generator = test_datagen.flow_from_directory(
        TEST_DIR,
        target_size=IMG_SIZE,
        batch_size=BATCH_SIZE,
        class_mode='binary'
    )

```

 Setting up Data Pipelines...
 Found 90409 images belonging to 2 classes.
 Found 21776 images belonging to 2 classes.

```

print("🏗 Building MobileNetV2 Model...")

base_model = MobileNetV2(weights='imagenet', include_top=False, input_shape=(128, 128, 3))
base_model.trainable = False


x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dropout(0.2)(x)
predictions = Dense(1, activation='sigmoid')(x)

model = Model(inputs=base_model.input, outputs=predictions)

model.compile(optimizer=Adam(learning_rate=0.0001),
              loss='binary_crossentropy',
              metrics=['accuracy'])

print("✅ Model Built!")

```

 Building MobileNetV2 Model...
 Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/mobilenet_v2/mobilenet_v2_weights_tf_dim_9406464/9406464 ————— 2s 0us/step
 Model Built!


```

print(f"🚀 Starting Training for {EPOCHS} Epochs...")

history = model.fit(
    train_generator,
    epochs=EPOCHS,
    validation_data=validation_generator,
    verbose=1
)

print("🏁 Training Finished!")

```

 Starting Training for 15 Epochs...
 /usr/local/lib/python3.12/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121: UserWarning: Your `PyDatasetAdapter` class does not implement the `warn_if_super_not_called` method.
 self._warn_if_super_not_called()

Epoch	Loss	Accuracy	Val Loss	Val Accuracy
1/15	0.7008	0.5835	0.5357	0.7356
2/15	0.6099	0.6700	0.5051	0.7635
3/15	0.5891	0.6862	0.4856	0.7787
4/15	0.5768	0.6961	0.4845	0.7792
5/15	0.5710	0.7021	0.4859	0.7719
6/15	0.5699	0.7026	0.4780	0.7816
7/15	0.5660	0.7063	0.4656	0.7929
8/15	0.5671	0.7057	0.4634	0.7984
9/15	0.5665	0.7055	0.4691	0.7883
10/15	0.5667	0.7043	0.4607	0.7960
11/15				

```
2826/2826 ————— 493s 174ms/step - accuracy: 0.7076 - loss: 0.5637 - val_accuracy: 0.7937 - val_loss: 0.4661
Epoch 12/15
2826/2826 ————— 493s 175ms/step - accuracy: 0.7094 - loss: 0.5642 - val_accuracy: 0.7999 - val_loss: 0.4585
Epoch 13/15
2826/2826 ————— 505s 176ms/step - accuracy: 0.7079 - loss: 0.5617 - val_accuracy: 0.7895 - val_loss: 0.4631
Epoch 14/15
2826/2826 ————— 499s 176ms/step - accuracy: 0.7114 - loss: 0.5598 - val_accuracy: 0.8005 - val_loss: 0.4539
Epoch 15/15
2826/2826 ————— 498s 176ms/step - accuracy: 0.7097 - loss: 0.5595 - val_accuracy: 0.8021 - val_loss: 0.4539
☒ Training Finished!
```

```
# Save locally
model.save('deepdetect_mobilenet_128.h5')
print("☑ Model Saved.")

# Download
from google.colab import files
files.download('deepdetect_mobilenet_128.h5')
```

```
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file for
☑ Model Saved.
```