```
# Step 1 : Data Preparation
pip install tensorflow opencv-python numpy matplotlib scikit-learn
```

```
Requirement already satisfied: tensorflow in /usr/local/lib/python3.12/dist-packages (2.19.0)
Requirement already satisfied: opencv-python in /usr/local/lib/python3.12/dist-packages (4.12.0.88)
Requirement already satisfied: numpy in /usr/local/lib/python3.12/dist-packages (2.0.2)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.12/dist-packages (3.10.0)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.12/dist-packages (1.6.1)
Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.12/dist-packages (from tensorflow) (1.4.0)
Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.12/dist-packages (from tensorflow) (1.6
Requirement already satisfied: flatbuffers>=24.3.25 in /usr/local/lib/python3.12/dist-packages (from tensorflow) (
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in /usr/local/lib/python3.12/dist-packages (fro
Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.12/dist-packages (from tensorflow) (0
Requirement already satisfied: libclang>=13.0.0 in /usr/local/lib/python3.12/dist-packages (from tensorflow) (18.1
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.12/dist-packages (from tensorflow) (3.4
Requirement already satisfied: packaging in /usr/local/lib/python3.12/dist-packages (from tensorflow) (25.0)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<6.0.0dev,>=3.20.3 in
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.12/dist-packages (from tensorflow) (2
Requirement already satisfied: setuptools in /usr/local/lib/python3.12/dist-packages (from tensorflow) (75.2.0)
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.12/dist-packages (from tensorflow) (1.17.0)
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.12/dist-packages (from tensorflow) (3.1.
Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/python3.12/dist-packages (from tensorflo
Requirement already satisfied: wrapt>=1.11.0 in /usr/local/lib/python3.12/dist-packages (from tensorflow) (1.17.3)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.12/dist-packages (from tensorflow) (1
Requirement already satisfied: tensorboard~=2.19.0 in /usr/local/lib/python3.12/dist-packages (from tensorflow) (2
Requirement already satisfied: keras>=3.5.0 in /usr/local/lib/python3.12/dist-packages (from tensorflow) (3.10.0)
Requirement already satisfied: h5py>=3.11.0 in /usr/local/lib/python3.12/dist-packages (from tensorflow) (3.14.0)
Requirement already satisfied: ml-dtypes<1.0.0,>=0.5.1 in /usr/local/lib/python3.12/dist-packages (from tensorflow
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (1.3.
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (4.5
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (1.4
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (11.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (3.2.
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.12/dist-packages (from scikit-learn) (1.16.1
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.12/dist-packages (from scikit-learn) (1.5.2
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.12/dist-packages (from scikit-learn)
Requirement already satisfied: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.12/dist-packages (from astunparse>=1.6
Requirement already satisfied: rich in /usr/local/lib/python3.12/dist-packages (from keras>=3.5.0->tensorflow) (13
Requirement already satisfied: namex in /usr/local/lib/python3.12/dist-packages (from keras>=3.5.0->tensorflow) (0
Requirement already satisfied: optree in /usr/local/lib/python3.12/dist-packages (from keras>=3.5.0->tensorflow) (
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/python3.12/dist-packages (from requests<
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-packages (from requests<3,>=2.21.0->
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-packages (from requests<3,>=2.
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-packages (from requests<3,>=2.
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.12/dist-packages (from tensorboard~=2.19.
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in /usr/local/lib/python3.12/dist-packages (f
Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.12/dist-packages (from tensorboard~=2.19.
Requirement already satisfied: MarkupSafe>=2.1.1 in /usr/local/lib/python3.12/dist-packages (from werkzeug>=1.0.1-
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.12/dist-packages (from rich->keras>
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.12/dist-packages (from rich->kera
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.12/dist-packages (from markdown-it-py>=2.2.0->
```

```
# Step 2: Train the model
import os, json, math
import numpy as np
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras import layers, models
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint, ReduceLROnPlateau
import matplotlib.pyplot as plt
import json
import cv2
import numpy as np
import tensorflow as tf
import os

DATA_DIR = "data"
IMG_SIZE = 128
```

```
BATCH_SIZE = 32
VAL_SPLIT = 0.2
EPOCHS = 20
CLASS_ORDER = ["without_mask", "with_mask"]
SEED = 42
tf.random.set_seed(SEED)
np.random.seed(SEED)
datagen = ImageDataGenerator(
    rescale=1/255.0,
    validation_split=VAL_SPLIT,
    rotation_range=20,
    zoom_range=0.2,
    horizontal_flip=True
)

train_gen = datagen.flow_from_directory(
    DATA_DIR,
    target_size=(IMG_SIZE, IMG_SIZE),
    batch_size=BATCH_SIZE,
    class_mode="binary",
    subset="training",
    shuffle=True,
    seed=SEED,
    classes=CLASS_ORDER
)

val_gen = datagen.flow_from_directory(
    DATA_DIR,
    target_size=(IMG_SIZE, IMG_SIZE),
    batch_size=BATCH_SIZE,
    class_mode="binary",
    subset="validation",
    shuffle=False,
    seed=SEED,
    classes=CLASS_ORDER
)

print("\nClass indices:", train_gen.class_indices)
def build_model(img_size=IMG_SIZE):
    inputs = layers.Input(shape=(img_size, img_size, 3))
    x = layers.Conv2D(32, 3, activation="relu")(inputs)
    x = layers.MaxPooling2D()(x)
    x = layers.Conv2D(64, 3, activation="relu")(x)
    x = layers.MaxPooling2D()(x)
    x = layers.Conv2D(128, 3, activation="relu")(x)
    x = layers.MaxPooling2D()(x)
    x = layers.Dropout(0.3)(x)
    x = layers.Flatten()(x)
    x = layers.Dense(128, activation="relu")(x)
    x = layers.Dropout(0.4)(x)
    outputs = layers.Dense(1, activation="sigmoid")(x)  # binary
    model = models.Model(inputs, outputs)
    model.compile(optimizer="adam", loss="binary_crossentropy", metrics=["accuracy"])
    return model

model = build_model()
os.makedirs("checkpoints", exist_ok=True)
ckpt_path = "checkpoints/mask_detector.best.keras"
callbacks = [
    EarlyStopping(monitor="val_accuracy", patience=5, restore_best_weights=True),
    ReduceLROnPlateau(monitor="val_loss", factor=0.5, patience=2),
    ModelCheckpoint(ckpt_path, monitor="val_accuracy", save_best_only=True)
]

steps_per_epoch = math.ceil(train_gen.samples / BATCH_SIZE)
val_steps = math.ceil(val_gen.samples / BATCH_SIZE)

history = model.fit(
```

```
        train_gen,
        validation_data=val_gen,
        epochs=EPOCHS,
        callbacks=callbacks,
        steps_per_epoch=steps_per_epoch,
        validation_steps=val_steps,
        verbose=1
    )

    model.save("mask_detector.keras")
    with open("class_indices.json", "w") as f:
        json.dump(train_gen.class_indices, f)
    plt.figure()
    plt.plot(history.history["accuracy"], label="train_acc")
    plt.plot(history.history["val_accuracy"], label="val_acc")
    plt.title("Accuracy")
    plt.xlabel("Epoch"); plt.ylabel("Accuracy"); plt.legend(); plt.tight_layout()
    plt.savefig("accuracy.png")

    plt.figure()
    plt.plot(history.history["loss"], label="train_loss")
    plt.plot(history.history["val_loss"], label="val_loss")
    plt.title("Loss")
    plt.xlabel("Epoch"); plt.ylabel("Loss"); plt.legend(); plt.tight_layout()
    plt.savefig("loss.png")

    print("\nSaved: mask_detector.keras, class_indices.json, accuracy.png, loss.png")
```

```
Found 1757 images belonging to 2 classes.
Found 439 images belonging to 2 classes.

Class indices: {'without_mask': 0, 'with_mask': 1}
/usr/local/lib/python3.12/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121: UserWarning:
  self._warn_if_super_not_called()
/usr/local/lib/python3.12/dist-packages/PIL/Image.py:1047: UserWarning: Palette images with Transparency expresse
  warnings.warn(
Epoch 1/20
55/55 ━━━━━━━━━━━━━━━━━━━━ 88s 2s/step - accuracy: 0.6859 - loss: 0.6062 - val_accuracy: 0.8383 - val_loss: 0.369
Epoch 2/20
55/55 ━━━━━━━━━━━━━━━━━━━━ 139s 1s/step - accuracy: 0.8702 - loss: 0.2926 - val_accuracy: 0.8292 - val_loss: 0.33
Epoch 3/20
55/55 ━━━━━━━━━━━━━━━━━━━━ 87s 2s/step - accuracy: 0.8953 - loss: 0.2699 - val_accuracy: 0.8565 - val_loss: 0.295
Epoch 4/20
55/55 ━━━━━━━━━━━━━━━━━━━━ 79s 1s/step - accuracy: 0.8896 - loss: 0.2486 - val_accuracy: 0.8292 - val_loss: 0.334
Epoch 5/20
55/55 ━━━━━━━━━━━━━━━━━━━━ 66s 1s/step - accuracy: 0.9073 - loss: 0.2198 - val_accuracy: 0.8633 - val_loss: 0.300
Epoch 6/20
55/55 ━━━━━━━━━━━━━━━━━━━━ 67s 1s/step - accuracy: 0.9087 - loss: 0.2210 - val_accuracy: 0.8815 - val_loss: 0.282
Epoch 7/20
55/55 ━━━━━━━━━━━━━━━━━━━━ 80s 1s/step - accuracy: 0.9032 - loss: 0.2062 - val_accuracy: 0.8747 - val_loss: 0.264
Epoch 8/20
55/55 ━━━━━━━━━━━━━━━━━━━━ 66s 1s/step - accuracy: 0.9362 - loss: 0.1614 - val_accuracy: 0.8975 - val_loss: 0.250
Epoch 9/20
55/55 ━━━━━━━━━━━━━━━━━━━━ 84s 1s/step - accuracy: 0.9304 - loss: 0.1741 - val_accuracy: 0.8861 - val_loss: 0.274
Epoch 10/20
55/55 ━━━━━━━━━━━━━━━━━━━━ 67s 1s/step - accuracy: 0.9309 - loss: 0.1629 - val_accuracy: 0.8884 - val_loss: 0.251
Epoch 11/20
55/55 ━━━━━━━━━━━━━━━━━━━━ 81s 1s/step - accuracy: 0.9310 - loss: 0.1560 - val_accuracy: 0.8838 - val_loss: 0.256
Epoch 12/20
```

```
# Step 3: Real-time detection
import json
import cv2
import numpy as np
import tensorflow as tf

MODEL_PATH = "mask_detector.keras"
CLASS_MAP_PATH = "class_indices.json"
IMG_SIZE = 128
model = tf.keras.models.load_model(MODEL_PATH)
with open(CLASS_MAP_PATH, "r") as f:
    class_indices = json.load(f)
idx_to_class = {v: k for k, v in class_indices.items()}
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + "haarcascade_frontalface_default.xml")
cap = cv2.VideoCapture(0)
def predict_label(face_img_bgr):
    face = cv2.resize(face_img_bgr, (IMG_SIZE, IMG_SIZE))
    face = face.astype("float32") / 255.0
    face = np.expand_dims(face, axis=0)
    prob = model.predict(face, verbose=0)[0][0]
    pred_idx = 1 if prob >= 0.5 else 0
    pred_name = idx_to_class[pred_idx]
    return pred_name, prob

while True:
    ok, frame = cap.read()
    if not ok:
        break

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, 1.1, 5)

    for (x, y, w, h) in faces:
        roi = frame[y:y+h, x:x+w]
        pred_name, prob = predict_label(roi)

        if pred_name == "with_mask":
            label = f"Mask ({prob:.2f})"
            color = (0, 200, 0)
```

```
            else:
                label = f"No Mask ({1-prob:.2f})"
                color = (0, 0, 255)

            cv2.rectangle(frame, (x, y), (x+w, y+h), color, 2)
            cv2.putText(frame, label, (x, y-10),
                        cv2.FONT_HERSHEY_SIMPLEX, 0.8, color, 2)

        cv2.imshow("Face Mask Detection", frame)
        if cv2.waitKey(1) & 0xFF == ord("q"):
            break

cap.release()
cv2.destroyAllWindows()

# Step 4: Sanity check
import os
from collections import Counter
```