Gender detection

```
import numpy as np
import tensorflow as tf
from tensorflow.keras.applications import (
    VGG16, ResNet50, MobileNetV2, InceptionV3
)
from tensorflow.keras.applications.vgg16 import preprocess_input as vgg_preprocess
from tensorflow.keras.applications.resnet50 import preprocess_input as resnet_preprocess
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input as
mobilenet_preprocess
from tensorflow.keras.applications.inception_v3 import preprocess_input as
inception_preprocess
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D
from tensorflow.keras.utils import load_img, img_to_array

# ----------------------------
# Utility: load and preprocess image
# ----------------------------
def load_and_preprocess(img_path, target_size, preprocess_func):
    img = load_img(img_path, target_size=target_size)
    img_array = img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0)
    return preprocess_func(img_array)

# ----------------------------
# Build gender classifier from a base model
# ----------------------------
def build_gender_model(base_model_class, preprocess_func, target_size=(224, 224)):
    base_model = base_model_class(
        weights="imagenet",
        include_top=False,
        input_shape=(target_size[0], target_size[1], 3)
    )
    x = base_model.output
    x = GlobalAveragePooling2D()(x)
    predictions = Dense(1, activation="sigmoid")(x)  # Binary classification
    model = Model(inputs=base_model.input, outputs=predictions)
    return model, preprocess_func, target_size

# ----------------------------
# Load all four models
# ----------------------------
vgg_model, vgg_func, vgg_size = build_gender_model(VGG16, vgg_preprocess, (224, 224))
resnet_model, resnet_func, resnet_size = build_gender_model(ResNet50, resnet_preprocess,
(224, 224))
mobilenet_model, mobilenet_func, mobilenet_size = build_gender_model(MobileNetV2,
```

```
mobilenet_preprocess, (224, 224))
inception_model, inception_func, inception_size = build_gender_model(InceptionV3,
inception_preprocess, (299, 299))

# ----------------------------
# Prediction function
# ----------------------------
def predict_gender(model, preprocess_func, target_size, img_path):
    img_array = load_and_preprocess(img_path, target_size, preprocess_func)
    pred = model(img_array, training=False).numpy()[0][0]  # Eager mode compatible
    gender = "Male" if pred > 0.5 else "Female"
    return gender, float(pred)

# ----------------------------
# Example usage
# ----------------------------
if __name__ == "__main__":
    img_path = "test_face.jpg"  # <-- Replace with your face image

    print("VGG16 Prediction:", predict_gender(vgg_model, vgg_func, vgg_size, img_path))
    print("ResNet50 Prediction:", predict_gender(resnet_model, resnet_func, resnet_size,
img_path))
    print("MobileNetV2 Prediction:", predict_gender(mobilenet_model, mobilenet_func,
mobilenet_size, img_path))
    print("InceptionV3 Prediction:", predict_gender(inception_model, inception_func,
inception_size, img_path))
```