Họ và tên: Nguyễn Hoàng Thắng

MSSV: 2151012009

Bài 1: Detect Pedestrians from image step by step

Source Code:

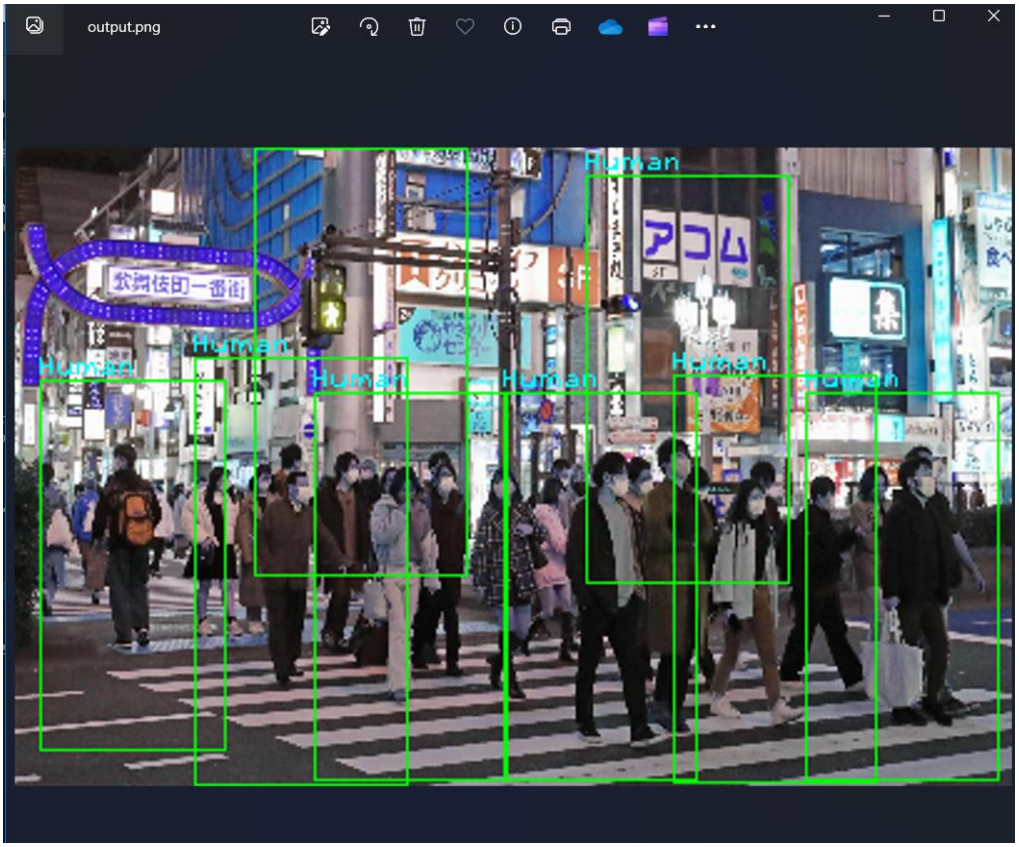
```python
from skimage.feature import hog
from sklearn.svm import LinearSVC
from sklearn.model_selection import train_test_split
from matplotlib import pyplot as plt
from sklearn import metrics
from imutils.object_detection import non_max_suppression
from skimage import color
from skimage.transform import pyramid_gaussian

import joblib
import glob
import os
import cv2
import numpy as np


X = []
Y = []

pos_im_path = "D:/Computer Vision/OpenCV-Practice/Lab5/positive"
neg_im_path = "D:/Computer Vision/OpenCV-Practice/Lab5/negative"

# Load positive features
for filename in glob.glob(os.path.join(pos_im_path, "*.png")):
    fd = cv2.imread(filename, 0)
    fd = cv2.resize(fd, (64, 128))
    fd = hog(
        fd,
        orientations=9,
        pixels_per_cell=(8, 8),
        visualize=False,
        cells_per_block=(2, 2),
    )
    X.append(fd)
    Y.append(1)
```

```
36
37  # Load negative features
38  for filename in glob.glob(os.path.join(neg_im_path, "*.png")):
39      fd = cv2.imread(filename, 0)
40      fd = cv2.resize(fd, (64, 128))
41      fd = hog(
42          fd,
43          orientations=9,
44          pixels_per_cell=(8, 8),
45          visualize=False,
46          cells_per_block=(2, 2),
47      )
48      X.append(fd)
49      Y.append(0)
50
51
52  X = np.float32(X)
53  Y = np.array(Y)
54
55  X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2)
56
57  print("Train data: ", len(X_train))
58  print("Train Labels (1, 0) ", len(y_train))
59
60  model = LinearSVC()
61  model.fit(X_train, y_train)
62
63  # predict
64  y_pred = model.predict(X_test)
65
66  # confusion matrix and accuracy
67  print(
68      f"Classification report for classifier {model}:\n"
69      f"{metrics.classification_report(y_test, y_pred)}\n"
70  )
71
72  joblib.dump(model, "models.dat")
73  print("Model saved : {}".format("models.dat"))
```

```
# pedestrian detection
modelFile = "models.dat"
inputFile = "input.png"
outputFile = "output.png"
image = cv2.imread(inputFile)
image = cv2.resize(image, (400, 256))
size = (64, 128)
step_size = (9, 9)
downscale = 1.05

# List to store the detections
detections = []
scale = 0
model = joblib.load(modelFile)


def sliding_window(image, window_size, step_size):
    for y in range(0, image.shape[0], step_size[1]):
        for x in range(0, image.shape[1], step_size[0]):
            yield (x, y, image[y: y + window_size[1], x: x + window_size[0]])


for im_scaled in pyramid_gaussian(image, downscale=downscale):
    if im_scaled.shape[0] < size[1] or im_scaled.shape[1] < size[0]:
        break
    for x, y, window in sliding_window(im_scaled, size, step_size):
        if window.shape[0] != size[1] or window.shape[1] != size[0]:
            continue
        window = color.rgb2gray(window)
        fd = hog(
            window,
            orientations=9,
            pixels_per_cell=(8, 8),
            visualize=False,
            cells_per_block=(2, 2),
        )

        fd = fd.reshape(1, -1)
        pred = model.predict(fd)
```

```
            if pred == 1:
                if model.decision_function(fd) > 0.5:
                    detections.append(
                        (
                            int(x * (downscale**scale)),
                            int(y * (downscale**scale)),
                            model.decision_function(fd),
                            int(size[0] * (downscale**scale)),
                            int(size[1] * (downscale**scale)),
                        )
                    )
        scale += 1

clone = image.copy()
clone = cv2.cvtColor(clone, cv2.COLOR_BGR2RGB)
rects = np.array([[x, y, x + w, y + h] for (x, y, _, w, h) in detections])
sc = [score[0] for (x, y, score, w, h) in detections]
print("sc: ", sc)
sc = np.array(sc)
pick = non_max_suppression(rects, probs=sc, overlapThresh=0.5)
for x1, y1, x2, y2 in pick:
    cv2.rectangle(clone, (x1, y1), (x2, y2), (0, 255, 0))
    cv2.putText(clone, "Human", (x1 - 2, y1 - 2), 1, 0.75, (255, 255, 0), 1)

cv2.imwrite(outputFile, clone)

plt.imshow(clone)
plt.title("Pedestrian Detection")

plt.show()

```

Result:



Bài 2: Detect Pedestrian in Video

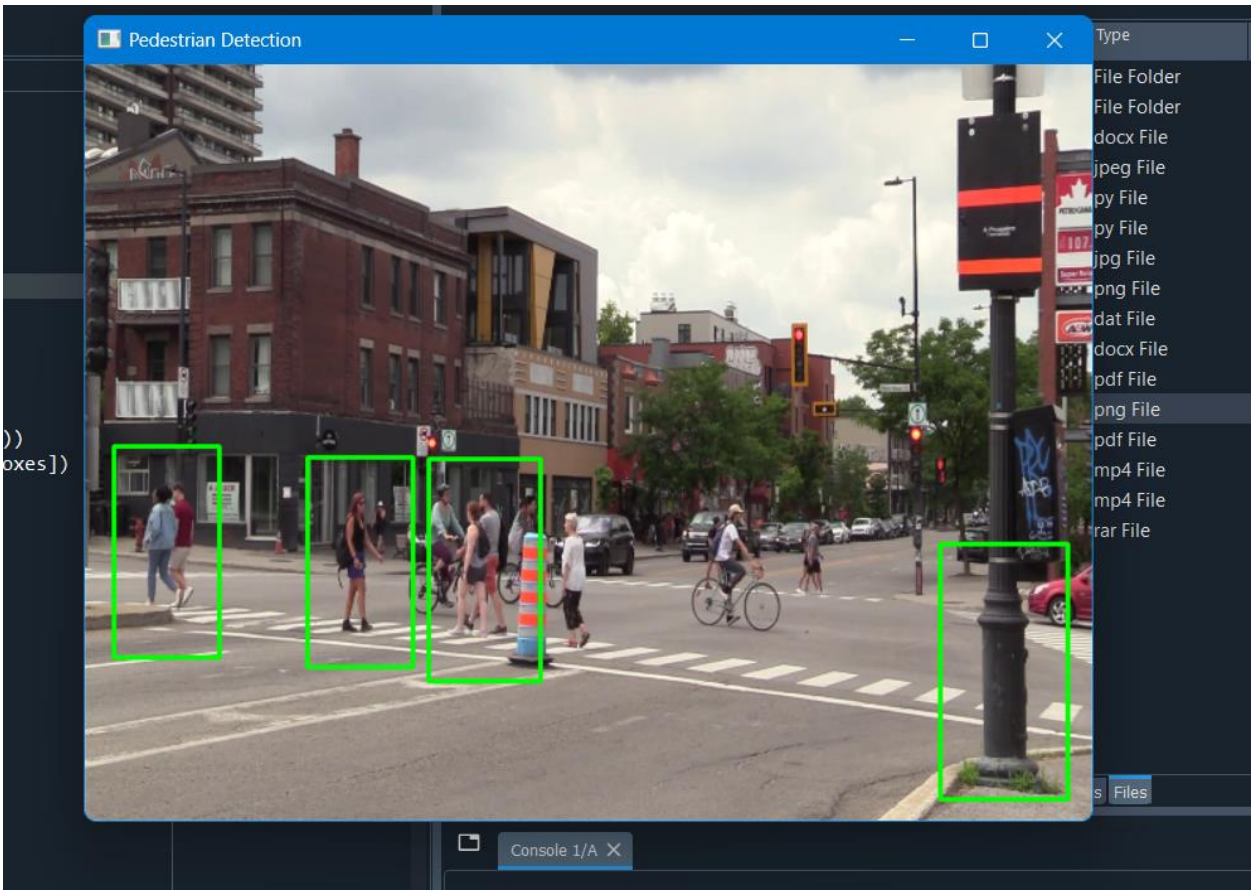
Source Code:

Ex_1_Pedestrian_Recognition_StepByStep.py | Ex_2_Pedestrian_Recognition_UsingCV2.py

<None> | <None>

```python
import numpy as np
import cv2

hog = cv2.HOGDescriptor()
hog.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())
cv2.startWindowThread()

cap = cv2.VideoCapture("video.mp4")
while True:
    ret, frame = cap.read()
    frame = cv2.resize(frame, (640, 480))
    gray = cv2.cvtColor(frame, cv2.COLOR_RGB2GRAY)

    boxes, weights = hog.detectMultiScale(frame, winStride=(8, 8))
    boxes = np.array([[x, y, x + w, y + h] for (x, y, w, h) in boxes])

    for xA, yA, xB, yB in boxes:
        # display the detected boxes in the colour picture
        cv2.rectangle(frame, (xA, yA), (xB, yB), (0, 255, 0), 2)

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

cap.release()
cv2.destroyAllWindows()

```

Result:


Pedestrian Detection
Type
File Folder
File Folder
docx File
jpeg File
py File
py File
jpg File
png File
dat File
docx File
pdf File
png File
pdf File
mp4 File
mp4 File
rar File
Files
Console 1/A