Mai Đức Khiêm

Nhóm 11

CODE: HỆ THỐNG TỰ ĐỘNG PHÁT HIỆN DẤU BÀI TRONG ẢNH

**Code giải quyết yêu cầu ban đầu:**

import numpy as np

from PIL import Image

import cv2

from google.colab.patches import cv2\_imshow

# Hàm để tải và chuyển đổi ảnh thành ảnh nhị phân

def load\_binary\_image(image\_path):

img = Image.open(image\_path).convert('L')

img\_array = np.array(img)

binary\_array = (img\_array > 246).astype(int)

return binary\_array

# Hàm để lưu một mảng con của ảnh nhị phân

def save\_subarray(binary\_array, x, y, width, height):

return binary\_array[y:y + height, x:x + width]

# Hàm để thực hiện khung trượt và lưu các mảng con

def sliding\_window\_and\_save\_arrays(image\_path, window\_width, window\_height, step\_size):

binary\_image = load\_binary\_image(image\_path)

img\_height, img\_width = binary\_image.shape

subarrays = []

for y in range(0, img\_height - window\_height + 1, step\_size):

for x in range(0, img\_width - window\_width + 1, step\_size):

subarray = save\_subarray(binary\_image, x, y, window\_width, window\_height)

subarrays.append((subarray, x, y))

return subarrays

# Hàm để tải ảnh nhị phân từ đường dẫn

def load\_train\_image(train\_image\_path):

return load\_binary\_image(train\_image\_path)

# Hàm để so sánh hai mảng nhị phân

def compare\_arrays(arr1, arr2):

if arr1.shape != arr2.shape:

print("Hai mảng có kích thước khác nhau, không thể so sánh!")

return None

difference = np.sum(arr1 != arr2)

return difference

# Hàm để vẽ hình chữ nhật quanh khu vực và hiển thị ảnh

def draw\_rectangle\_and\_show(image\_path, x, y, width, height):

img = cv2.imread(image\_path)

cv2.rectangle(img, (x, y), (x + width, y + height), (0, 255, 0), 2)

cv2\_imshow(img)

# Sử dụng các hàm đã định nghĩa

def main():

image\_path = '/content/drive/MyDrive/ok/original.png'

train\_image\_path = '/content/drive/MyDrive/ok/negative.jpg'

window\_width = 50

window\_height = 50

step\_size = 7

# Lấy các mảng nhị phân từ khung trượt của ảnh

subarrays = sliding\_window\_and\_save\_arrays(image\_path, window\_width, window\_height, step\_size)

# Lưu ảnh nhị phân vào mảng arr\_train

arr\_train = load\_train\_image(train\_image\_path)

min\_difference = float('inf')

min\_index = -1

min\_x, min\_y = 0, 0

for i, (subarray, x, y) in enumerate(subarrays, start=1):

difference = compare\_arrays(subarray, arr\_train)

if difference is not None:

if difference < min\_difference:

min\_difference = difference

min\_index = i

min\_x, min\_y = x, y

if min\_index != -1:

print(f"Số điểm ảnh khác biệt nhỏ nhất là {min\_difference} tại mảng thứ {min\_index}.")

print(f"Vị trí của khung trượt là: ({min\_x}, {min\_y})")

# Vẽ hình chữ nhật và hiển thị ảnh

draw\_rectangle\_and\_show(image\_path, min\_x, min\_y, window\_width, window\_height)

# Chạy hàm chính

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Code cải tiến:**

import numpy as np

from PIL import Image

import cv2

from google.colab.patches import cv2\_imshow

# Hàm để tải ảnh nhị phân

def load\_binary\_image(image\_path):

img = Image.open(image\_path).convert('L')

img\_array = np.array(img)

binary\_array = (img\_array > 246).astype(int)

return binary\_array

# Hàm để cắt một mảng con từ ảnh

def save\_subarray(binary\_array, x, y, width, height):

return binary\_array[y:y + height, x:x + width]

# Hàm trượt cửa sổ để lấy các mảng con từ ảnh

def sliding\_window\_and\_save\_arrays(image\_path, window\_width, window\_height, step\_size):

binary\_image = load\_binary\_image(image\_path)

img\_height, img\_width = binary\_image.shape

subarrays = []

for y in range(0, img\_height - window\_height + 1, step\_size):

for x in range(0, img\_width - window\_width + 1, step\_size):

subarray = save\_subarray(binary\_image, x, y, window\_width, window\_height)

subarrays.append((subarray, x, y))

return subarrays

# Hàm để tải ảnh mẫu

def load\_train\_image(train\_image\_path):

return load\_binary\_image(train\_image\_path)

# Hàm so sánh 2 mảng nhị phân

def compare\_arrays(arr1, arr2):

if arr1.shape != arr2.shape:

print("Hai mảng có kích thước khác nhau, không thể so sánh!")

return None

difference = np.sum(arr1 != arr2)

return difference

# Hàm để vẽ các khung chữ nhật tại nhiều vị trí

def draw\_rectangles\_and\_show(image\_path, rectangles):

img = cv2.imread(image\_path)

for (x, y, width, height) in rectangles:

cv2.rectangle(img, (x, y), (x + width, y + height), (0, 255, 0), 2)

cv2\_imshow(img)

# Sử dụng hàm với Google Colab

image\_path = '/content/drive/MyDrive/ok/original.png'

train\_image\_path = '/content/drive/MyDrive/ok/train1.png'

window\_width = 39

window\_height = 50

step\_size = 1

threshold = 100 # Ngưỡng để xác định sự khác biệt tối thiểu

# Lấy các mảng nhị phân từ khung trượt của ảnh

subarrays = sliding\_window\_and\_save\_arrays(image\_path, window\_width, window\_height, step\_size)

# Lưu ảnh nhị phân mẫu vào mảng arr\_train

arr\_train = load\_train\_image(train\_image\_path)

# Danh sách để lưu các khung có số điểm ảnh khác biệt nhỏ hơn threshold

rectangles = []

for i, (subarray, x, y) in enumerate(subarrays, start=1):

difference = compare\_arrays(subarray, arr\_train)

if difference is not None and difference < threshold:

print(f"Số điểm ảnh khác biệt tại mảng thứ {i} là {difference}. Vị trí: ({x}, {y})")

rectangles.append((x, y, window\_width, window\_height))

if rectangles:

# Vẽ tất cả các khung chữ nhật và hiển thị ảnh

draw\_rectangles\_and\_show(image\_path, rectangles)

else:

print("Không tìm thấy đối tượng nào phù hợp.")

**Code để phát hiện vật thể nhỏ:**

import cv2

import numpy as np

import matplotlib.pyplot as plt

def imshow(img, figsize=(6, 6)):

fig, ax = plt.subplots(1, 1, figsize=figsize)

ax.axis('off')

ax.imshow(img)

img = cv2.imread('/original.png')

img\_gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

template = cv2.imread('/template\_resize.png', 0)

# Resize the template to half its size

template = cv2.resize(template, (0, 0), fx=0.5, fy=0.5)

w, h = template.shape[1], template.shape[0]

res = cv2.matchTemplate(img\_gray,template, cv2.TM\_CCOEFF\_NORMED)

threshold = 0.85

loc = np.where(res >= threshold)

for pt in zip(\*loc[::-1]):

cv2.rectangle(img, pt, (pt[0] + w, pt[1] + h), (0, 255, 0), 2)

imshow(img)