**Міністерство освіти і науки України**

**Національний технічний університет України «КПІ» імені Ігоря Сікорського**

**Кафедра обчислювальної техніки ФІОТ**

**ЗВІТ**

**з лабораторної роботи №9**

**з навчальної дисципліни «Вступ до технології Data Science»**

**Тема:**

**ДОСЛІДЖЕННЯ ТЕХНОЛОГІЙ COMPUTER VISION ДЛЯ ІДЕНТИФІКАЦІЇ ОБ'ЄКТІВ У ВИРОБНИЧИХ CRM СИСТЕМАХ ТА НА ОБ’ЄКТАХ КРИТИЧНОЇ ІНФРАСТРУКТУРИ**

**Виконав:**

Студент X курсу кафедри ОТ ФІОТ,

Навчальної групи ІА-73

Петренко П.П.

**Перевірив:**

Професор кафедри ОТ ФІОТ

Писарчук О.О.

**Київ 2023**

**І. Мета:**

виявити дослідити та узагальнити особливості застосування методів Computer Vision для ідентифікації об'єктів у виробничих CRM системах та на об’єктах критичної інфраструктури з використанням спеціалізованих пакетів мови програмування Python.

**ІІ. Завдання:**

| 7, 22 | Розробити програмний скрипт з функціоналом:  самостійно обрати файл цифрового зображення, або відеопотоку;  здійснити спотворення вхідного зображення: інверсія + випадковий шум;  реалізувати сукупність операцій з покращення якості спотвореного зображення;  здійснити побудову контуру обраного об’єкту на відновленому зображенні.  здійснити розпізнавання обраного об’єкту. |
| --- | --- |

| 7, 22 | Розробити універсальний програмний скрипт, що здійснює комплекс операцій із сегментації об’єктів відеопотоку. |
| --- | --- |



Рисунок 1. Початкова картинка



Рисунок 2. Шум + інверсія



Рисунок 3. Сепія

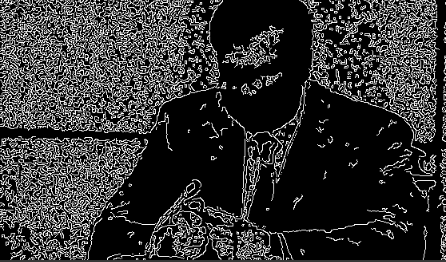


Рисунок 4. Контури

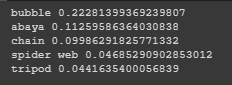


Рисунок 5. Визначення мережі

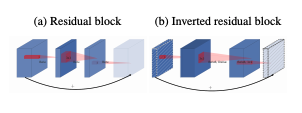


Рисунок 6. Схема мережі

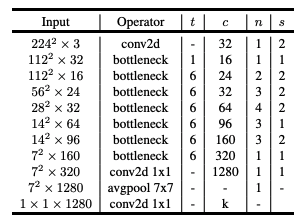


Рисунок 7. Опис шарів мережі

Через наявність шуму та відсутність чіткого контуру мережа не може ефективно розпізнати об'єкт у вигляді людини. Ймовірно, її тренували на зображеннях без обробки, де відсутні виділені межі.

**ІІІ. Довести адекватність сформованих моделей та працездатність розробленого скріпта.**

**Розроблений код повинен бути раціональним та відповідати вимогам до чистого коду.**

**import cv2**

**import numpy as np**

**from google.colab.patches import cv2\_imshow**

**# Load an image**

**def load\_image(path):**

**return cv2.imread(path)**

**# Invert image colors and add random noise**

**def distort\_image(image):**

**# Invert colors**

**inverted\_image = cv2.bitwise\_not(image)**

**# Add random noise**

**noise = np.random.normal(0, 1, inverted\_image.shape).astype(np.uint8)**

**return cv2.add(inverted\_image, noise)**

**# Enhance image quality**

**def enhance\_image(image):**

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

**img\_blur = cv2.GaussianBlur(img\_gray, (5,5), 0)**

**return cv2.equalizeHist(img\_blur)**

**# Detect edges**

**def detect\_edges(image):**

**return cv2.Canny(image, 100, 200)**

**# Load and process an image**

**image = load\_image('variant7.jpg')**

**distorted\_image = distort\_image(image)**

**enhanced\_image = enhance\_image(distorted\_image)**

**edges = detect\_edges(enhanced\_image)**

**cv2\_imshow(image)**

**cv2\_imshow(distorted\_image)**

**cv2\_imshow(enhanced\_image)**

**cv2\_imshow(edges)**

**import torch**

**model = torch.hub.load('pytorch/vision:v0.10.0', 'mobilenet\_v2', pretrained=True)**

**model.eval()**

**from PIL import Image**

**from torchvision import transforms**

**im\_pil = Image.fromarray(edges)**

**input\_image = im\_pil.convert('RGB')**

**preprocess = transforms.Compose([**

**transforms.Resize(256),**

**transforms.CenterCrop(224),**

**transforms.ToTensor(),**

**transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),**

**])**

**input\_tensor = preprocess(input\_image)**

**input\_batch = input\_tensor.unsqueeze(0) # create a mini-batch as expected by the model**

**# move the input and model to GPU for speed if available**

**if torch.cuda.is\_available():**

**input\_batch = input\_batch.to('cuda')**

**model.to('cuda')**

**with torch.no\_grad():**

**output = model(input\_batch)**

**# Tensor of shape 1000, with confidence scores over ImageNet's 1000 classes**

**print(output[0])**

**# The output has unnormalized scores. To get probabilities, you can run a softmax on it.**

**probabilities = torch.nn.functional.softmax(output[0], dim=0)**

**print(probabilities)**

**# Download ImageNet labels**

**!wget https://raw.githubusercontent.com/pytorch/hub/master/imagenet\_classes.txt**

**with open("imagenet\_classes.txt", "r") as f:**

**categories = [s.strip() for s in f.readlines()]**

**top5\_prob, top5\_catid = torch.topk(probabilities, 5)**

**for i in range(top5\_prob.size(0)):**

**print(categories[top5\_catid[i]], top5\_prob[i].item())**

**III. Висновок:**

виявив дослідив та узагальнив особливості застосування методів Computer Vision для ідентифікації об'єктів у виробничих CRM системах та на об’єктах критичної інфраструктури з використанням спеціалізованих пакетів мови програмування Python.

Виконав: студент Петренко П.П.