**Міністерство освіти і науки України**

**Національний технічний університет України**

**«Київський політехнічний інститут імені Ігоря Сікорського»**

# Звіт

Лабораторна робота № 4 з дисципліни

«Штучний інтелект в задачах обробки зображень»

**«**Розпізнавання людини на фото з використанням бібліотеки Dlib**»**

**Виконав(ла)**

*ІП-01 Черпак А. В.*

(шифр, прізвище, ім'я, по батькові)

**Перевірив(ла)**

*Нікітін В. А.*

(прізвище, ім'я, по батькові)

Київ 2023

# Завдання

1. Зробити розпізнавання будь-якої “зірки”
2. 2. Зробити розпізнавання себе.

**Мета**

Навчитися розпізнавати обличчя на фото використовуючи навчені нейронні  
мережі

# Хід роботи

1. Імпортуємо необхідні бібліотеки та створимо допоміжні функції для пошуку усіх файлів, що належать заданим класам, пошуку та обробки обличчя, а також аналізу співпадінь:

import os  
import cv2  
import glob  
import dlib  
import numpy  
  
face\_detector = dlib.get\_frontal\_face\_detector()  
shape\_predictor = dlib.shape\_predictor("models/shape\_predictor\_68\_face\_landmarks.dat")  
face\_encoder = dlib.face\_recognition\_model\_v1("models/dlib\_face\_recognition\_resnet\_model\_v1.dat")  
  
VALID\_EXTENSIONS = ['.jpg', 'jpeg']  
  
  
def get\_image\_paths(root\_dir, class\_names):  
 image\_paths = []  
 for class\_name in class\_names:  
 class\_dir = os.path.sep.join([root\_dir, class\_name])  
 class\_file\_paths = glob.glob(os.path.sep.join([class\_dir, '\*.\*']))  
 for file\_path in class\_file\_paths:  
 ext = os.path.splitext(file\_path)[1]  
 if ext.lower() not in VALID\_EXTENSIONS:  
 print("Skipping file: {}".format(file\_path))  
 continue  
 image\_paths.append(file\_path)  
 return image\_paths  
  
  
def face\_rects(image):  
 gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)  
 rects = face\_detector(gray, 1)  
 return rects  
  
  
def face\_landmarks(image):  
 return [shape\_predictor(image, face\_rect) for face\_rect in face\_rects(image)]  
  
  
def face\_encodings(image):  
 return [numpy.array(face\_encoder.compute\_face\_descriptor(image, face\_landmark)) for face\_landmark in  
 face\_landmarks(image)]  
  
  
def nb\_of\_matches(known\_encodings, unknown\_encoding):  
 distances = numpy.linalg.norm(known\_encodings - unknown\_encoding, axis=1)  
 small\_distances = distances <= 0.6  
 return sum(small\_distances)

1. Створимо функцію для генерації вкладень до кожного обличчя:

import os  
import cv2  
from Lab4\_utils import get\_image\_paths, face\_encodings  
import pickle  
  
root\_dir = 'dataset'  
class\_names = os.listdir(root\_dir)  
image\_paths = get\_image\_paths(root\_dir, class\_names)  
name\_encoding\_dict = {}  
  
nb\_current\_image = 1  
for image\_path in image\_paths:  
 print(f"Image processed {nb\_current\_image}/{len(image\_paths)}")  
 image = cv2.imread(image\_path)  
 encodings = face\_encodings(image)  
 name = image\_path.split(os.path.sep)[-2]  
 e = name\_encoding\_dict.get(name, [])  
 e.extend(encodings)  
 name\_encoding\_dict[name] = e  
 nb\_current\_image += 1  
with open("encodings.pickle", "wb") as f:  
 pickle.dump(name\_encoding\_dict, f)

Запустимо описану вище функцію, і поглянемо на вивід:

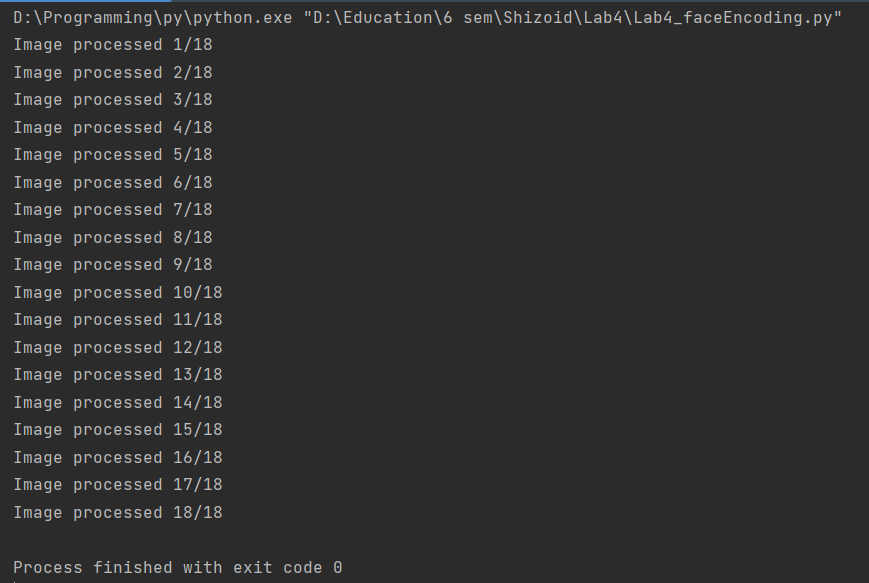


Рисунок 1 – Результати генерації вкладень

Як бачимо, все відпрацювало без помилок. Тепер створимо за стосунок для розпізнавання людей по фото.

from Lab4\_utils import \*  
import pickle  
with open("encodings.pickle", "rb") as f:  
 name\_encodings\_dict = pickle.load(f)  
  
imagePoroshenko = cv2.imread("examples/1.jpg")  
imageArnold = cv2.imread("examples/2.jpg")  
imageMultiple = cv2.imread("examples/3.jpg")  
imagePoroh = cv2.imread("examples/5.jpg")  
imageMe = cv2.imread("examples/4.jpg")  
  
images = [imagePoroshenko, imageArnold, imageMultiple, imageMe, imagePoroh]  
  
for i in range(len(images)):  
 encodings = face\_encodings(images[i])  
 names = []  
 for encoding in encodings:  
 counts = {}  
 for (name, encodings) in name\_encodings\_dict.items():  
 counts [name] = nb\_of\_matches (encodings, encoding)  
 if all (count == 0 for count in counts.values()):  
 name = "Unknown"  
 else: name = max(counts, key=counts.get)  
 names.append(name)  
 for rect, name in zip(face\_rects(images[i]), names):  
 x1, y1, x2, y2 = rect.left(), rect.top(), rect.right(), rect.bottom()  
 cv2.rectangle(images[i], (x1, y1), (x2, y2), (0, 255, 0), 2)  
 cv2.putText(images[i], name, (x1, y1 - 10), cv2.FONT\_HERSHEY\_SIMPLEX, 0.75, (0, 255, 0), 2)  
 cv2.imshow("image", images[i])  
 cv2.waitKey(0)

При навчанні були використані фотографії Порошенка, а також мої фото. Тому протестуємо систему спершу за допомогою фотографії Петра Олексійовича, потім – зображення Шварцнегера, далі – спільного зображення Порошенка, Зеленського та Тимошенко. Очевидно, система повинна розпізнати лише Петра Олексійовича, а інших позначити як Unknown.

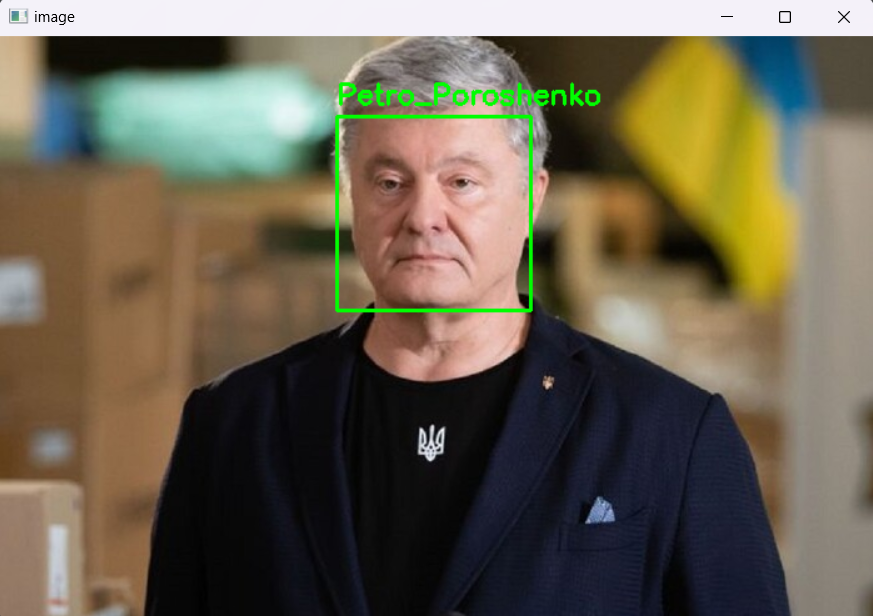


Рисунок 2 – Результат розпізнавання Петра Олексійовича

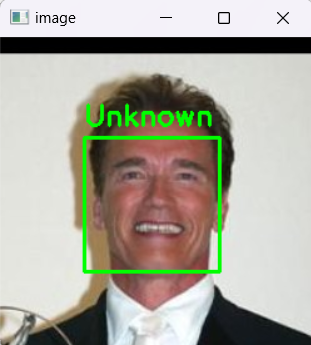


Рисунок 3 – Результат розпізнавання Шварцнегера – не розпізнано

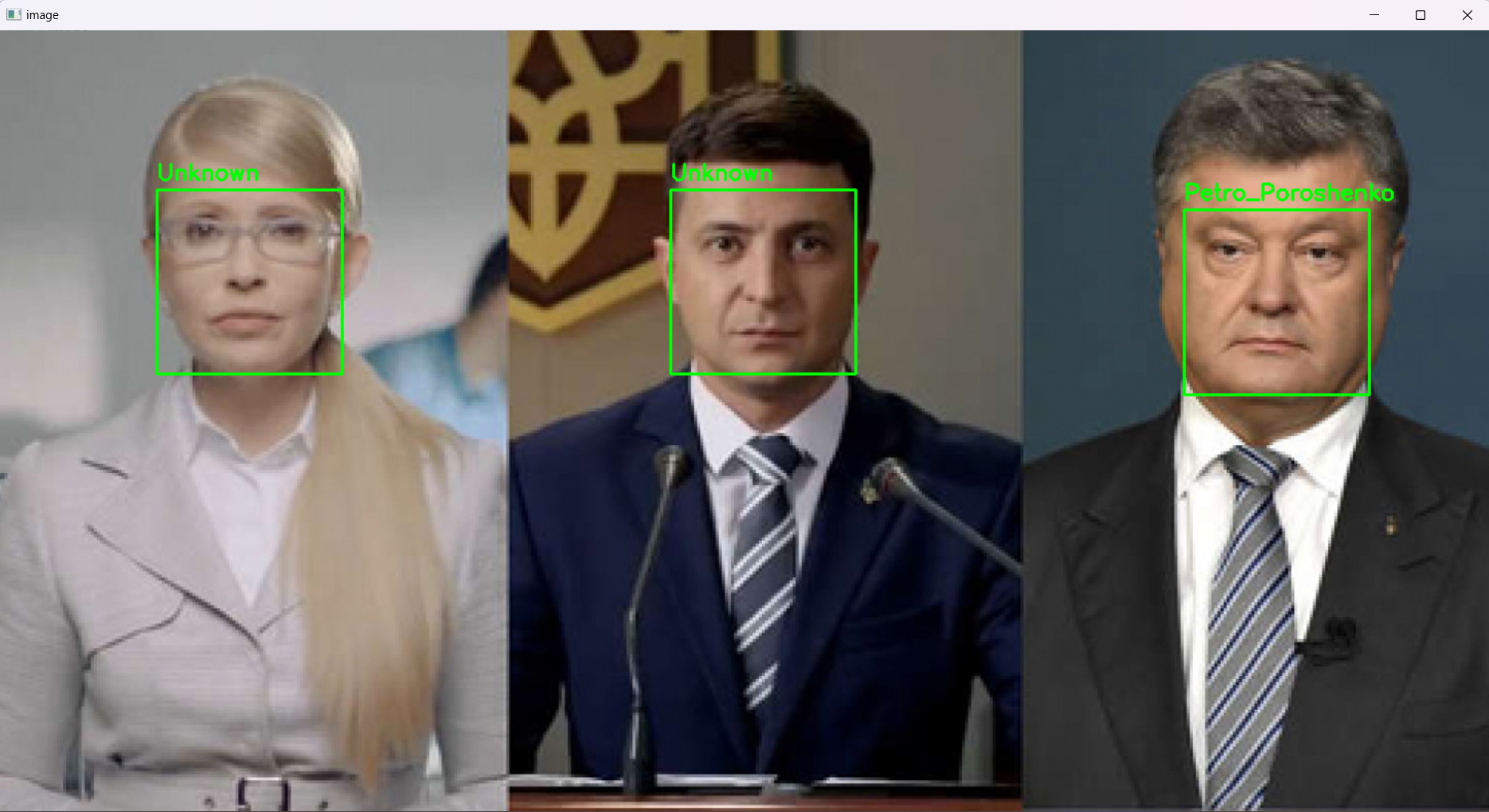


Рисунок 4 – Результат розпізнавання на фото з групою осіб – розпізнано лише тих, на кому система навчалася.

Після цього передамо у систему нашу з дівчиною спільну фотографію. Знову ж, мене повинно розпізнати, а дівчину – ні.

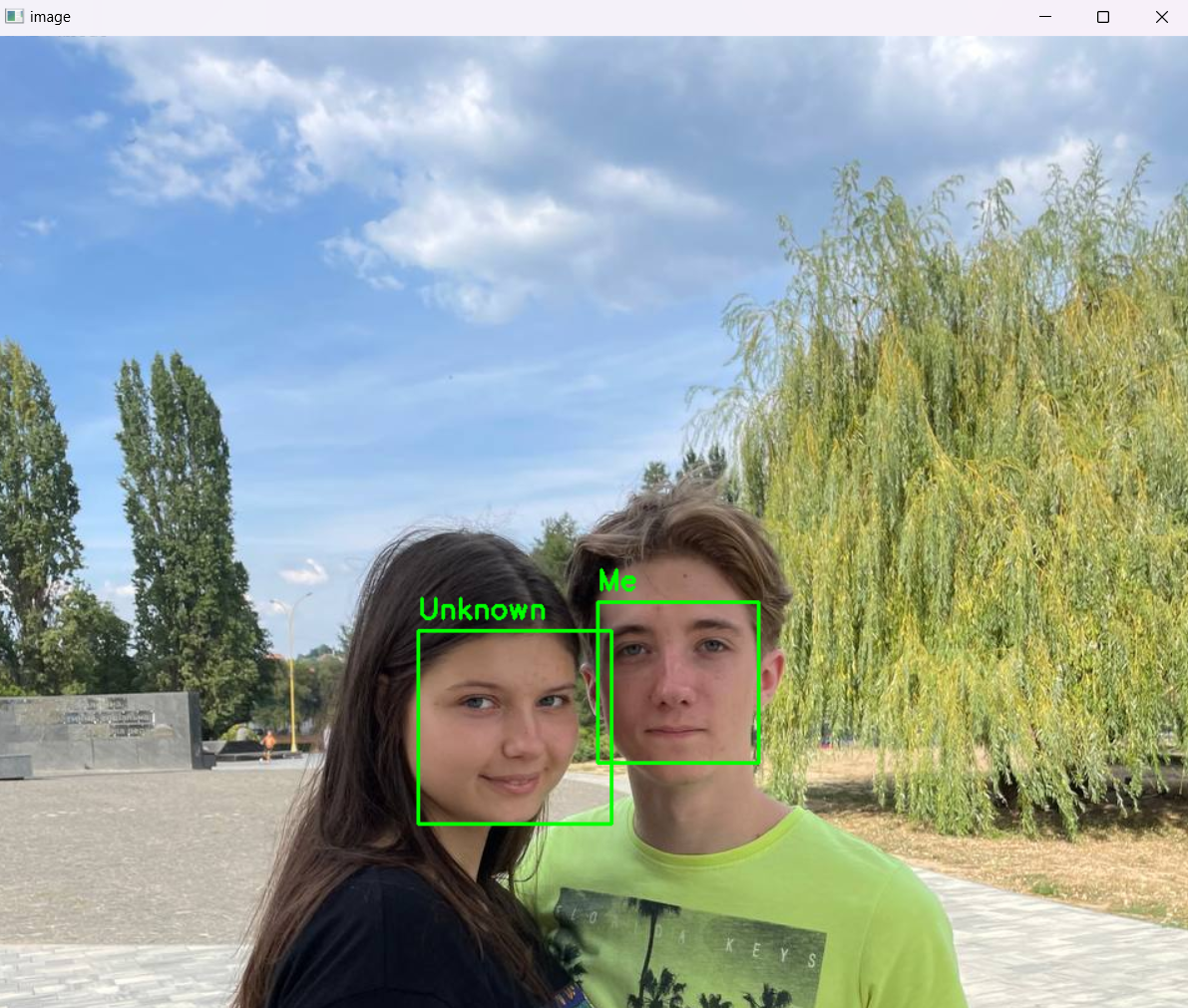


Рисунок 5 – Результати розпізнавання нас з дівчиною

Тепер цікавості заради передамо викривлене зображення і спробуємо його класифікувати.



Рисунок 6 – Розпізнавання викривлених зображень

Як бачимо, навіть на частково викривлених фотографіях система здатна розпізнавати обличчя людини.

# Контрольні запитання:

* 1. Розпізнавання обличчя – це процес ідентифікації людини за рисами її обличчя. Ця технологія використовується при біометричному розблокуванні екрану, ідентифікації при накладанні електронного підпису, а також у криміналістиці для визначення особи правопорушника чи потерпілого.
  2. Для розпізнавання обличчя спершу потрібно знайти область зображення, у якій воно розташоване, потім вирівняти його за допомогою спеціальних орієнтирів, далі кодувати, а вже лиш тоді .
  3. Згорткові нейронні мережі – це різновид нейронних мереж, що містять у тому числі згорткові шари для обробки зображень (двовимірних чи багатовимірних). У даній лабораторній розглядалися сіамські мережі, тобто система з двох абсолютно однакових мереж, через які пропускаються пари зображень і порівнюються результати.
  4. Для навчання мережі необхідно подати на вхід ціле дерево зображень, що міститиме як позитивні, так і негативні семпли. Один з позитивних обирається «якірним», тобто подальше навчання відштовхуватиметься саме від нього. При кодуванні для кожного зображення створюється вкладення, і вже керуючись даними з цих вкладень мережа намагається віднайти такі ваги параметрів, при яких відстань між вкладеннями однієї людини буде мінімальною, а для різних людей – максимальною.
  5. Dlib – кросплатформна бібліотека, написана на С++, що складається з багатьох компонент різного призначення. У даній роботі ми використовували її для отримання детектора облич, вирівнювання за орієнтирами, а також безпосередньо для розпізнавання облич.