

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіоелектроніки,

Факультет комп'ютерних наук і технологій

(повне найменування інституту, назва факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ ЗАСТОСУНКУ
РОЗПІЗНАВАННЯ ОСІБ НА ЗОБРАЖЕННЯХ
RESEARCH AND SOFTWARE IMPLEMENTATION OF THE
APPLICATION FOR FACE RECOGNITION IN IMAGES

Виконав: студент(ка) 2 курсу, групи КНТ-220М
Спеціальності 122 Комп'ютерні науки
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Системи штучного інтелекту

Ніколаєвський Д.О.

(прізвище та ініціали)

Керівник Корнієнко С. К.

(прізвище та ініціали)

Рецензент Казурова А.Є.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
 (повне найменування закладу вищої освіти)

Інститут, факультет ІРЕ, ФКНТ
 Кафедра програмних засобів
 Ступінь вищої освіти магістр
 Спеціальність 122 Комп'ютерні науки
 (код і найменування)
 Освітня програма (спеціалізація) Системи штучного інтелекту
 (назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ
 Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
 “ ” 20__ року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Ніколаєвського Данила Олександровича
 (прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація застосунку розпізнавання осіб на зображеннях. Research and Software Implementation of the Application for Face Recognition in Images

керівник проєкту (роботи) Корнієнко Сергій Костянтинівч. к.т.н., доцент
 (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “05” листопада 2021 року № 435

2. Строк подання студентом проєкту (роботи) 13 грудня 2021 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Характеристики об'єкта проєктування. 3. Огляд структури проєкту. 4. Керівництво програміста. 5. Керівництво користувача. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Слайди презентації

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-6 Основна частина	Корнієнко С.К., доцент		
Нормоконтроль	Белова А.В., асистент		

7. Дата видачі завдання “15” вересня 2021року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Постановка завдання роботи	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області	2-4 тижні	Розділ 1
3	Розробка архітектури програми	5 тиждень	Розділ 2
4	Розробка програми	6-8 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми	9-10 тиждень	Розділ 4
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	11-12 тиждень	Додатки
7	Захист роботи	13 тиждень	

Студент

(підпис)

Ніколаєвський Д.О.

(прізвище та ініціали)

Керівник проекту (роботи)

(підпис)

Корнієнко С.К.

(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи магістра: 92 с., 5 табл., 41 рис., 2 дод., 44 джерел.

СИСТЕМИ ШТУЧНОГО ІНТЕЛЕКТУ, СИСТЕМИ РОЗПІЗНАВАННЯ ОСІБ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, НЕЙРОМЕРЕЖИ, ДЕСКТОПНИЙ ЗАСТОСУНОК.

Об'єкт дослідження – програмний застосунок для розпізнавання осіб на зображеннях.

Предмет дослідження – технології нейромереж для обробки зображень.

Метою дипломної роботи є проведення дослідження на тему розпізнавання осіб на зображеннях та програмна реалізація застосунку.

Матеріали, методи та технічні засоби: структурне та об'єктно-орієнтоване програмування, мова програмування та середовище розробки Python, PyCharm, персональний комп'ютер з процесором AMD A4-6300 під управлінням операційної системи Microsoft Windows 7 Ultimate.

Науковою новизною є модернізування методу обробки зображення, шляхом застосування переробленого методу Віола-Джонса за допомогою ознак Хаара.

В результаті створено програмний десктопний застосунок, який оброблює отримане зображення або відеопотік та визначає кількість осіб на ній за допомогою візуального відображення рамкою.

Галузь використання – посилення заходів безпеки в різних сферах життєдіяльності людини.

ABSTRACT

Explanatory note to the diploma qualifying work of the master: 92 pages, 5 tables, 41 figures, 2 appendixes, 44 sources.

ARTIFICIAL INTELLIGENCE SYSTEMS, PERSONAL RECOGNITION SYSTEMS, SOFTWARE, NEURAL NETWORKS, DESKTOP APPLIANCE.

The object of research is a software application for face recognition in images.

The subject of research is neural network technologies for image processing.

The purpose of the thesis is to conduct research on the recognition of faces in images and its software implementation.

Materials, methods and technical means: structural and object-oriented programming, programming language and development environment Python, PyCharm, personal computer with AMD A4-6300 processor running Microsoft Windows 7 Ultimate operating system.

As a result, a desktop application was created that processes the resulting image or video stream and determines the number of people on it using a visual display frame.

Area of use - strengthening security measures in various spheres of human life.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК	8
ВСТУП	10
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ	11
1.1 Особливості предметної області.....	11
1.2 Різновиди нейронних мереж	13
1.3 Принципи розпізнавання та їх аналіз.....	17
1.4 Цілі використання	22
1.5 Висновки за розділом 1	24
2 ХАРАКТЕРИСТИКИ ОБ'ЄКТА ПРОЕКТУВАННЯ.....	25
2.1 Існуючі застосунки.....	25
2.1.1 AutoDraw	25
2.1.2 Shazam	27
2.1.3 Deep Fake або Face Swap	29
2.2 Висновки за розділом 2	32
3 ОГЛЯД СТРУКТУРИ ПРОЄКТУ.....	33
3.1 Особливості технологій.....	33
3.2 Вибір мови програмування	35
3.3 Середовище розробки	39
3.4 Коротко про архітектуру	40
3.5 MVC.....	44
3.6 Висновки за розділом 3	47
4 КЕРІВНИЦТВО ПРОГРАМІСТА.....	48
4.1 Функціонал застосунку.....	48
4.2 Характеристики застосунку, ПЗ та АЗ.....	49
4.3 База Даних.....	49
4.4 Важливі параметри.....	49
5 КЕРІВНИЦТВО КОРИСТУВАЧА	51
5.1 Призначення застосунку.....	51

5.2 Приклад взаємодії з програмою	51
ВИСНОВКИ.....	54
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ	56
ДОДАТОК А Текст програми.....	61
ДОДАТОК Б Слайди презентації	83

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

HM		Нейромережа
DNN	Deep Neural Network	Глибока нейронна мережа
IDE	Integrated Development Environment	Інтегроване середовище розробки
JSON	JavaScript Object Notation	Текстовий формат обміну даними
SD	Secure Digital	Формат карт пам'яті
SDK	Software Development Kit	Набір засобів розробки
UML	Unified Modeling Language	Уніфікована мова моделювання
XML	Extensible Markup Language	Розширювана мова розмітки
OC		Операційна система
ПЗ		Програмне забезпечення
AЗ		Апаратне забезпечення
TЗ		Технічне завдання
FFNN	Feed Forward Neural Networks	Нейронні мережі прямого розповсюдження
RBF	Radial Basis Function	Мережі радіально-базових функцій
DTM	Discrete Time Markov Chains	Ланцюги Маркова
C		
RBM	Restricted Boltzmann Machine	Обмежена машина Больцмана
AE	Autoencoder	Автокодувальник
SAE	Sparse Autoencoder	Розріджений автокодувальник
VAE	Variational Autoencoder	Варіаційні автокодувальники
DAE	Denoising Autoencoder	Шумоподавляючі автокодувальники
DBN	Deep Belief Networks	Мережа типу «deep belief»
CNN	Convolutional Neural Networks	Згорткові нейронні мережі
DCN	Deep Convolutional Neural	Глибинні згорткові нейронні

N	Networks	мережі
MVC	Model View Controller	Схема поділу даних програми

ВСТУП

З моменту появи перших ЕОМ, пройшло багато часу, і технологічний розвиток не стоїть на місці. Те, що у минулому здавалося нам неможливим, розвивається на очах. Ніхто й уявити не міг, наприклад на початку нульових, яких досягнень доб'ється людство.

Розвиток робототехніки, створення нейромереж, які закладають основу для появи справжнього штучного інтелекту. Розробка нових гаджетів, кількість та можливості функцій яких, тільки ростуть: до мережі можливо підключитися завдяки холодильнику, у смартфоні вже є усі документи користувача, годинники слідкують за здоров'ям носія, крок за кроком відтворюється у реальність єдиний світовий швидкісний інтернет та готуються до підкорення космічних просторів!

На деяких конференціях, що посвячені розвитку технологій, є можливим зустрітись з думкою: «Якщо вам розповідатимуть про 2050 рік і це буде виглядати як наукова фантастика, ймовірно, це неправда. Але якщо опис 2050 року не буде звучати як наукова фантастика, це абсолютно точно неправда». На мій погляд, ця цитата несе в собі важливу для людства думку, а саме: розвиток технологій обмежується лише уявою самих вчених та інженерів. Людство має змогу створювати такі технічні шедеври, у існування яких неможливо повірити.

Із огляду на таку величезну кількість тем для обговорення, ми зупинимося на розвитку та впливу нейромереж, і зокрема системи розпізнавання осіб, на сьогодення та найближче майбутнє. Я вважаю дану тему досить важливою у технологічній революції 21 століття.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Перед тим, як почати дане дослідження, треба в першу чергу ознайомитися із існуючими предметними областями та дослідити їх особливості.

1.1 Особливості предметної області

Спочатку потрібно визначити, що таке нейромережі, як вони виглядають, які існують, за якими принципами вони працюють. Коротко, нейромережа – це математична модель у вигляді програмного та апаратного втілення, що будується на принципах функціонування біологічних нейромереж (рис. 1.1) [1]-[3]. Сьогодні такі мережі активно використовують у практичних цілях за рахунок можливості не лише розробки, а й навчання. Їх застосовують для прогнозування, розпізнавання образів, машинного перекладу, розпізнавання аудіо та багато інших можливостей.

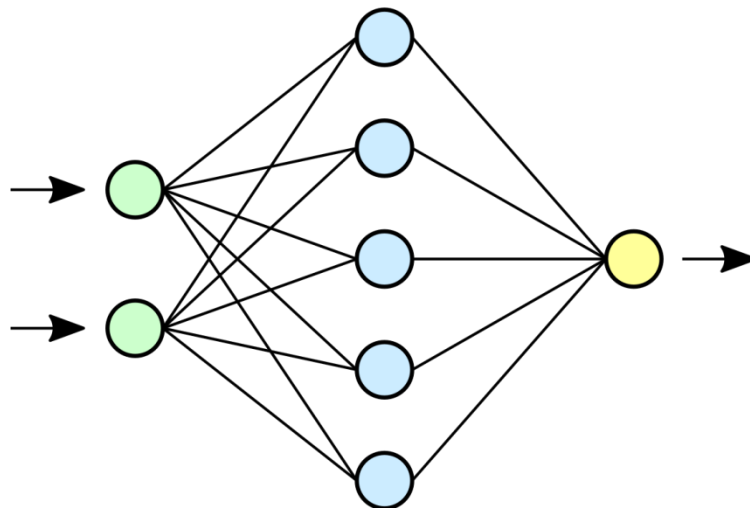


Рисунок 1.1 – Схема простої нейромережі

На рисунку 1.1 зазначено:

- зеленим кольором позначені вхідні нейрони – які приймають вихідний вектор, що кодує вхідний сигнал. Як правило, ці нейрони не виконують обчислювальних операцій, а просто передають отриманий вхідний сигнал на вихід, можливо посиливши або послабивши його;
- блакитним зазначені приховані нейрони - які виконують основні обчислювальні операції ;
- жовтим визначають вихідний нейрон – котрий являє собою вихід з мережі. Вони можуть виконувати будь-які обчислювальні операції.

Безперечно, самим практичним завданням нейромереж є розпізнавання візуальних образів [4]. Сьогодні створюються мережі, в яких прилади здатні успішно розпізнавати символи на паперах та банківських картках, підписи на офіційних документах, визначати об'єкти, тощо. Ці функції дозволяють істотно полегшити робочий процес людини, пошук і обробку інформації, а також дозволяють без проблем працювати з величезними обсягами даних, тобто підвищити надійність і точність різних процесів за рахунок відсутності можливості припущення помилки через людський фактор або інших неврахованих проблем.

Основою всіх архітектур для відеоспостереження є аналіз, першою фазою якого буде розпізнавання зображення (об'єкта). Потім штучний інтелект за допомогою машинного навчання або завдяки заздалегідь заготовлених моделей розпізнає наявність і дії об'єкта та класифікує їх. Для того щоб розпізнати зображення, нейронна мережа має бути раніше навчена на даних, або мати доступ до вже готових баз даних. Такі дані можливо зібрати в мережі інтернет у відкритих джерелах або зібрати власноруч. Це дуже схоже на нейронні зв'язки в людському мозку — ми маємо певні знання, бачимо об'єкт, аналізуємо його та ідентифікуємо [2]-[3].

Об'єктами можуть виступати: символи тексту, зображення, зразки звуків тощо. Під час навчання мережі пропонуються різні зразки образів із зазначенням, якого класу вони ставляться [3]. Зразок, як правило, представляється як вектор значень ознак. При цьому, сукупність всіх ознак

має однозначно визначати клас, до якого належить зразок. Якщо ознак недостатньо, мережа може співвіднести його з кількома класами, що збільшує відсоток помилкових результатів.

1.2 Різновиди нейронних мереж

Нові види архітектури нейронних мереж виникають постійно, і в них стає складно орієнтуватись. Хоча всі вони представлені як унікальні, багато з них мають забагато однакових ознак або є відгалуженням інших існуючих мереж. Також, існує велика ймовірність допустити помилки з їх найменуванням та аббревіатурою.

Для прикладу, варіаційні автокодувальники (VAE) виглядають зовсім як прості автокодувальники (AE), але їх процеси навчання суттєво різняться. Випадки використання відрізняються ще більш, оскільки VAE це генератор, якому для отримання нового зразка подається новий шум. AE просто порівнює отримані дані з найбільш схожим зразком, отриманим під час навчання [4].

Варто зауважити, що хоча більшість цих аббревіатур є загальноприйнятими, також існують винятки. Під RNN іноді мають на увазі рекурсивну нейронну мережу, але зазвичай, мають на увазі рекурентну. Також можна часто зустріти використання аббревіатури RNN, коли йдеться мова про будь-яку рекурентну НМ. Автокодувальники також мають цю проблему. Іноді варіаційні та шумоподавлюючі автокодувальники (VAE, DAE) називають просто автокодувальниками (AE). Крім того, у багатьох аббревіатурах розрізняється кількість букв «N» наприкінці, оскільки в деяких випадках використовується «neural network», а в інших просто «network» [4].

Для подальшого дослідження слід ознайомитись з прикладами нейромереж та їх особливостями, які зазначені у таблиці 1.1.

Таблиця 1.1 – Нейромережі та їх особливості

Назва 1	Опис 2
Нейронні мережі прямого поширення (feed forward neural networks, FF або FFNN) та перцептрони (perceptrons, P)	Вони дуже прямолінійні та передають інформацію від входу до виходу. Нейронні мережі часто описуються у вигляді торта, де кожен шар складається з вхідних, прихованих або вихідних клітин. Клітини одного шару не пов'язані між собою, а сусідні шари зазвичай пов'язані. Найпростіша нейронна мережа має дві вхідні клітини і одну вихідну, і може використовуватися як модель логічних вентилів. FFNN зазвичай навчається методом зворотного поширення помилки, у якому мережа отримує безлічі вхідних і вихідних даних. Цей процес називається навчанням з учителем, і він відрізняється від навчання без вчителя тим, що у другому випадку безліч вихідних даних мережа становить самостійно. Вищезгадана помилка є різницею між введенням та висновком [5].
Мережі радіально-базових функцій (RBF)	Ця мережа є FFNN, яка використовує радіальні базові функції як функції активації. Більш нічим вона не виділяється [6].
Нейронна мережа Хопфілда (Hopfield network, HN)	Це пов'язана із симетричною матрицею зв'язків нейронна мережа. Під час отримання вхідних даних кожен вузол є входом, у процесі навчання стає прихованим, та в кінці стає виходом. Мережа навчається так: значення нейронів встановлюються відповідно до бажаного шаблону, після чого обчислюються ваги, які надалі не змінюються. Після навчання, вона завжди буде зводитися до одного з них. У кожного нейрона є свій поріг активації, який залежить від температури, при проходженні якого нейрон набуває одного з двох значень. Така мережа часто називається мережею з асоціативною пам'яттю; як людина, дивлячись на половину таблиці, може уявити другу половину таблиці, так і ця мережа, отримуючи таблицю, наполовину зашумлену, відновлює її до повної та чистої [7].
Ланцюги Маркова (Markov chains, MC або discrete time Markov Chains, DTMC)	Це попередник машин Больцмана (BM) та мереж Хопфілда (HN). Їхній сенс можливо пояснити так: які мої шанси потрапити в один із наступних вузлів, якщо я перебуваю в цьому? Кожен наступний стан залежить лише від попереднього. Хоча насправді ланцюги Маркова не є НМ, вони дуже схожі. Також ланцюги Маркова не обов'язково можуть бути пов'язані між собою [8].

Продовження таблиці 1.1

1	2
Машина Больцмана (Boltzmann machine, BM)	Машина дуже схожа на мережу Хопфілда, але в ній лише деякі нейрони позначені як вхідні, а деякі як приховані. Вхідні нейрони надалі стають вихідними. Машина Больцмана – це стохастична мережа. Навчання проходить за методом зворотного поширення помилки або алгоритму порівняльної розбіжності. У цілому, нині процес навчання дуже схожий на такий як у Хопфілда [9].
Обмежена машина Больцмана (restricted Boltzmann machine, RBM)	Судячи з назви, нагадує машину Больцмана і, отже, на мережу Хопфілда. Єдиною різницею є її обмеженість. У ній нейрони одного типу пов'язані між собою. Обмежену машину Больцмана можна навчати як FFNN, але з одним нюансом: замість прямої передачі даних та зворотного розповсюдження помилки потрібно передавати дані спершу у прямому напрямку, потім у зворотньому. Після цього вона проходить навчання за методом прямого та зворотного поширення помилки [10].
Автокодувальник (autoencoder, AE)	АЕ чимось схожа на FFNN, тому що це більш інший спосіб використання FFNN, а не інша архітектура. Основною ідеєю є автоматичне кодування інформації. Сама мережа формою нагадує пісочний годинник, у ній приховані шари менше вхідного і вихідного. Мережу можна навчити методом зворотного розповсюдження помилки, подаючи вхідні дані та ставлячи помилку рівною різниці між входом та виходом [11].
Розріджений автокодувальник (sparse autoencoder, SAE)	В деякому сенсі протилежність звичайного АЕ. Замість того, щоб навчати мережу відображати інформацію в меншому обсязі вузлів, ми збільшуємо їх кількість. Замість того, щоб звужуватись до центру, мережа там збільшується. Мережі такого типу корисні до роботи з великою кількістю дрібних властивостей набору даних. Якщо навчати мережу як звичайний автокодувальник, нічого корисного не вийде. Тому крім вхідних даних подається ще й спеціальний фільтр розрідженості, що пропускає лише певні помилки [12].

Продовження таблиці 1.1

1	2
Варіаційні автокодувальники (Variational autoencoder, VAE)	Вони мають схожу з АЕ архітектуру, але навчають їх по іншому: наближенню ймовірнісного розподілу вхідних зразків. У цьому беруть початок від машин Больцмана. Тим не менш, вони спираються на байєсовську математику, коли йдеться про ймовірні висновки та незалежності, які інтуїтивно зрозумілі, але складні в реалізації. Якщо узагальнити, то можна сказати, що ця мережа бере до уваги вплив нейронів. Якщо щось одне відбувається в одному місці, а щось інше — в іншому, ці події не обов'язково пов'язані, і це має враховуватися [13].
Шумопригнічуючі автокодувальники (denoising autoencoder, DAE)	Це АЕ, які вхідні дані подаються в зашумленому стані. Помилки ми обчислюємо так само, і вихідні дані порівнюються із зашумленими. Завдяки цьому мережа вчиться звертати увагу більш широкі властивості, оскільки маленькі можуть змінюватися разом із шумом [14].
Мережа типу «deep belief» (DBN)	Це назва, яку отримав тип архітектури, в якій мережа складається з кількох з'єднаних RBM або VAE. Такі мережі навчаються побічно, причому кожному блоку потрібно лише вміти закодувати попередній. Така техніка називається «жадібним навчанням», яка полягає у виборі локальних оптимальних рішень, що не гарантують оптимальний кінцевий результат. Також мережу можна навчити (методом зворотного поширення помилки) відображати дані як імовірнісної моделі. Якщо використовувати навчання без вчителя, стабілізовану модель можна використовувати для створення нових даних [15].
Згорткові нейронні мережі (convolutional neural networks, CNN) та глибинні скруткові нейронні мережі (deep convolutional neural networks, DCNN)	Такі НМ дуже відрізняються від інших видів мереж. Зазвичай, вони використовуються для обробки зображень, рідше для аудіо. Типовим способом застосування CNN є класифікація зображень: якщо на зображенні є кішка, мережа видасть "кішка", якщо є собака - "собака". Такі мережі зазвичай використовують «сканер», який не парсить всі дані за один раз. Наприклад, якщо у вас є зображення 200x200, ви не відразу оброблятимете всі 40 тисяч пікселів. Замість це мережа рахує квадрат розміру 20x20 (зазвичай з лівого верхнього кута), потім зрушить на 1 піксель і рахує новий квадрат. Ці вхідні дані передаються через згорткові шари, в яких не всі вузли з'єднані між собою. Ці шари мають властивість стискатися з глибиною, причому часто використовуються ступеня двійки: 32, 16, 8, 4, 2, 1. Такі мережі називаються глибинними (DCNN) [16].

Продовження таблиці 1.1

1	2
Розгортаючі нейронні мережі (deconvolutional networks, DN)	Також звані зворотними графічними мережами, є зворотнім до згорткових нейронних мереж. Уявіть, що ви передаєте мережі слово "кішка", а вона генерує картинку з кішками, схожі на реальні зображення котів. DNN також можна поєднувати з FFNN. Варто зауважити, що в більшості випадків мережі передається не рядок, а бінарний вектор: наприклад, $\langle 0, 1 \rangle$ - це кішка, $\langle 1, 0 \rangle$ - собака, а $\langle 1, 1 \rangle$ - і кішка, і собака [17].

Таким чином, ми оглянули деякі існуючі нейромережі та визначили їх опис, особливості, основні принципи їх функціонування.

1.3 Принципи розпізнавання та їх аналіз

Як на практиці працюють дані системи? На практиці це означає, що до певної межі чим більше прихованих шарів нейронної мережі існує, тим точніше буде розпізнане зображення (рис. 1.1) [18]. Як це реалізується?

Зображення розбивається на маленькі ділянки, до кількох пікселів, кожен із яких буде входним нейроном. За допомогою синапсів сигнали передаються від одного шару до іншого. Під час цього процесу сотні тисяч нейронів із мільйонами параметрів порівнюють отримані сигнали з уже обробленими даними [18]-[19].

На приклад, якщо ми просимо нейромережу розпізнати зображення тварини, вона розіб'є фото на маленькі шматочки і буде порівнювати ці шари з мільйонами зображень тварин, які вже мережа вивчила (рис. 1.2, рис. 1.3). Таким чином, момент збільшення числа шарів призводить до простого запам'ятовування вибірки, а не навчання [20].

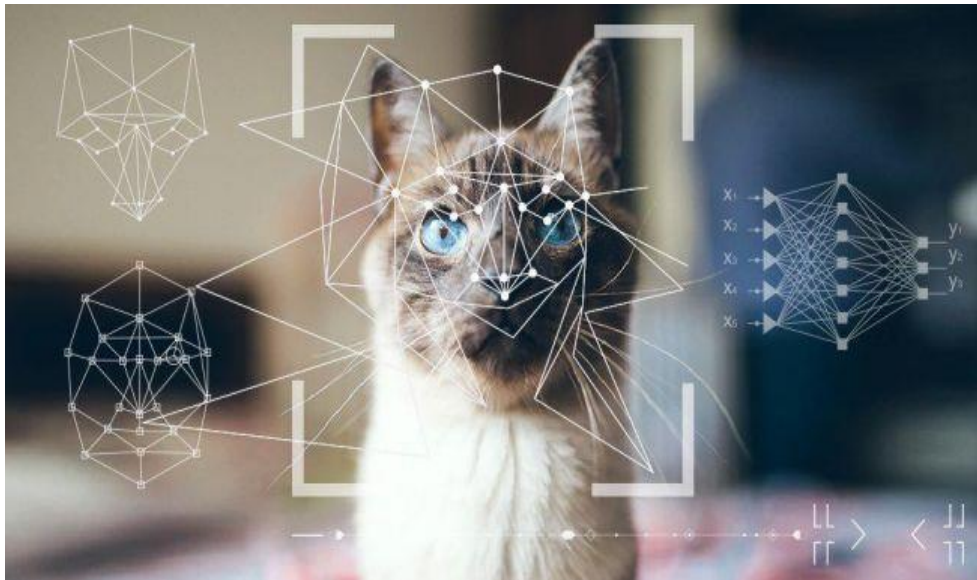


Рисунок 1.2 – Приклад розпізнавання зображення [20]

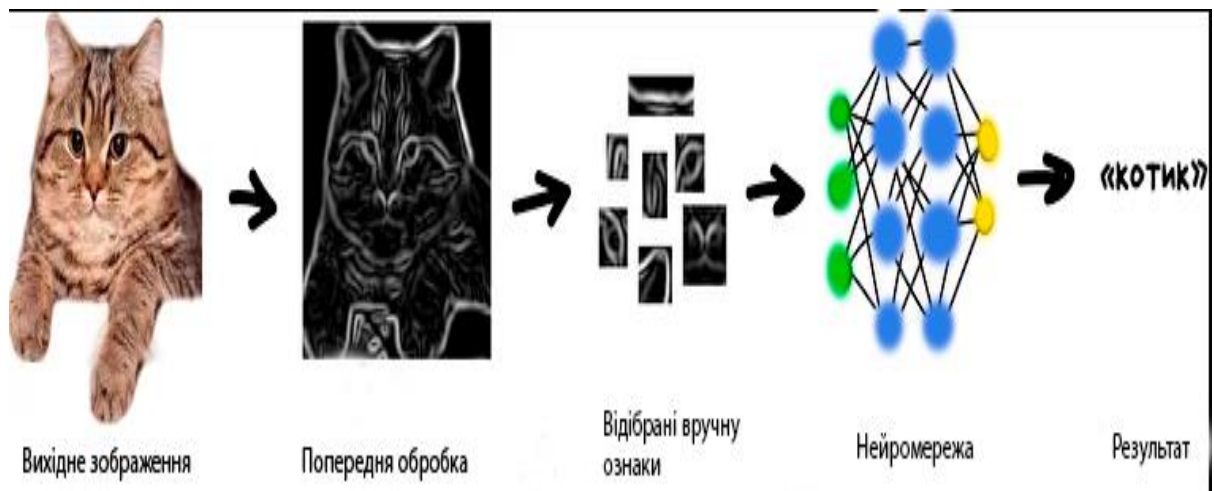


Рисунок 1.3 – Процес розпізнавання зображення [20]

Важливо під час навчання навчити мережу визначати і достатню кількість і значення ознак, щоб видавати високу точність на нових зображеннях, окрім того, важливо не перенавчитися. Після завершення правильного навчання НМ повинна вміти визначати образи (тих класів), із якими вона мала справи у процесі навчання.

Також, треба враховувати, що вихідні дані для нейромережі повинні бути однозначними і несуперечливими, щоб не виникали ситуації, коли НМ видаватиме високі ймовірності приналежності одного об'єкта до кількох різних протиставлених класів.

В цілому, створення нейронної мережі для розпізнавання зображень включає до себе (рис. 1.4):



Рисунок 1.4 – Схема створення нейромережі [18]

Надалі слідкує більш уважно та підрібно оцінити існуючі методи розпізнавання та провести аналіз їх сильних та слабких сторін. Визначимо це у таблиці 1.2.

Таблиця 1.2 – Аналіз методів розпізнавання

Метод	Швидкість	Точність
1	2	3
Метод гнучкого порівняння на графах [21]	Операція деформації графа та процес обробки виконується послідовно всіма вершинами до того часу, поки не буде досягнута найменша сумарна різниця між ознаками деформованого і еталонного графів, що виразно б'є по швидкості роботи даного методу. Для порівняння вхідного зображення особи з еталонними витрачалося приблизно 25 секунд. В інших публікаціях на цю тематику час або не вказується, або говориться, що воно значно велике.	Хоча в окремих публікаціях вказується 95% ефективність розпізнавання осіб навіть за наявності різних емоційних висловлювань і змінних ракурсу особи до 15 градусів, всі переваги методу руйнуються із-за високої обчислювальної складності процедури розпізнавання і низької технологічності при запам'ятовуванні нових еталонів.
Приховані Марківські моделі [8]	При розпізнаванні об'єкта перевіряються згенеровані для заданої бази об'єктів Марківські моделі і шукається максимальна ймовірність того, що послідовність спостережень для даного об'єкта згенерована відповідною моделлю. Крім того, необхідно вибирати параметри моделі для кожної бази даних. ПММ використовує статистичні властивості сигналів та враховують безпосередньо їх просторові характеристики. Так що безперечно, ні про яку високу швидкість обробки даних мова не йдеться.	ПММ не має здатності розрізняти осіб, тобто алгоритм навчання тільки максимізує відгук кожного зображення на свою модель, але не мінімізує відгук на інші моделі. На сьогоднішній день не вдалося знайти приклад застосування ПММ для розпізнавання осіб, тому міркувати про точність даного методу не має сенсу.

Продовження таблиці 1.2

1	2	3
Метод основних компонентів [22]	Суть методу зводиться до наступного: спочатку весь навчальний набір осіб перетворюється на одну загальну матрицю даних, розкладену в рядок, всі особи навчального набору повинні бути приведені до одного розміру та з нормованими гістограмами. Далі проводиться нормування даних та приведення рядків до 0 середнього та 1 дисперсії. Для отриманої матриці вирішується завдання визначення власних значень та відповідних власних векторів. При цьому, швидкість обробки виростає в декілька разів.	Метод головних компонент добре зарекомендував себе у практичних додатках. Однак, у випадках, коли на зображенні особи присутні значні зміни у освітленості або виразі обличчя, ефективність методу значно падає. Вся справа в тому, що обирається підпростір з такою метою, щоб максимально апроксимувати вхідний набір даних, а не виконати дискримінацію між класами осіб.
Метод Віюли-Джонса із ознаками Хаара	Метод не є жорстким фіксованим алгоритмом. Його каскади формуються в результаті навчання. Тому швидкість роботи може сильно змінюватись в залежності від складності об'єкта та розміру об'єкта, який ми хочемо виявити. Найпростіший спосіб прискорення його роботи – це задати обмеження на мінімальний об'єкт або збільшити крок піраміди. Досить наявним способом прискорення є обробка різних частин зображення в паралельних потоках, який вже реалізован у OpenCV.	Хоча метод був розроблений і представлений в 2001, він досі є основним для пошуку об'єктів на зображенні в реальному часі. Він є одним із найкращих за співвідношенням показників ефективності розпізнавання/швидкість роботи. Також цей детектор має вкрай низьку ймовірність помилкового виявлення обличчя. Алгоритм навіть добре працює та розпізнає риси обличчя під невеликим кутом, приблизно до 30 градусів.

Як видно із таблиці 1.2, останній метод Віюли-Джонса є найкращим вибором для роботи над застосунками по розпізнаванні осіб на зображеннях, порівняно з його співвідношення швидкості роботи до точності результату (рис. 1.5, рис. 1.6).

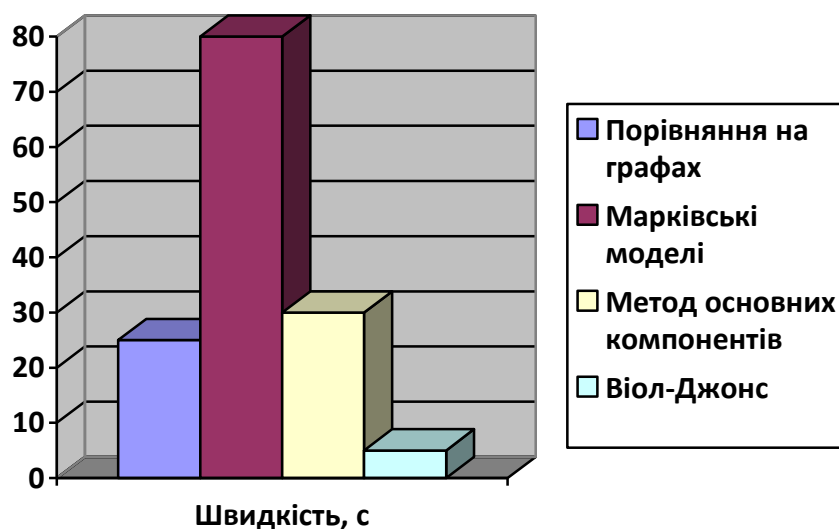


Рисунок 1.5 – Графік швидкості обробки методів

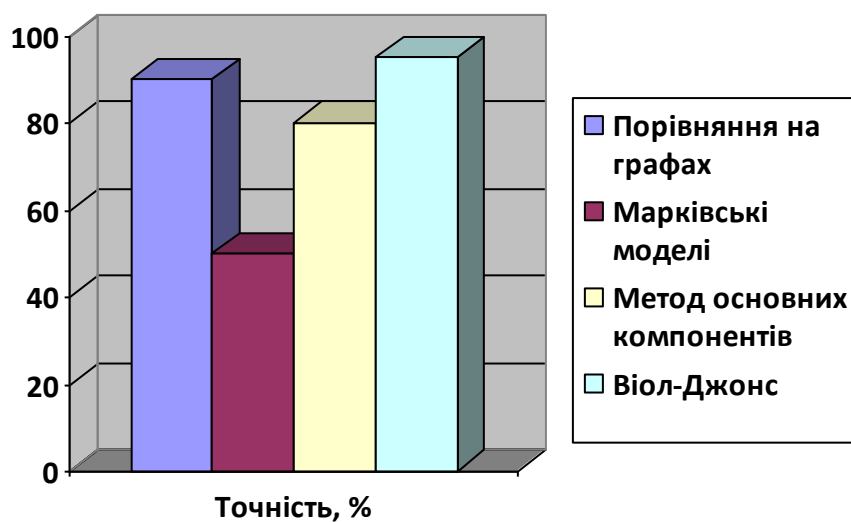


Рисунок 1.6 – Графік точності обробки методів

1.4 Цілі використання

До широкої аудиторії доходять лише гучні випадки застосування нейромереж. Наприклад, коли нейромережа записує власноруч музичний альбом чи знімають короткометражні фільми за сценарієм, який написала машина. Основна мета таких акцій – привернути увагу до технології,

показати всю її незвичайність та можливості. Такі події допомагають людям побачити, як розвивається світ.

Але насправді, неймережі щодня виконують більш важливі завдання. Наприклад, алгоритм Brain на основі неймереж щодня працює над системою рекомендацій YouTube. Він підбирає для користувачів релевантний контент, вивчаючи їхню поведінку та історію переглядів. Тому відео часто набирають високі перегляди не за рахунок прямого трафіку, коли користувач вводить запит у пошуковий рядок, а тоді коли система сама пропонує йому потенційно цікавий контент [23].

Нещодавно, неймережа змогла перетворювати різні фотографії на зразок шедеврів мистецтва (рис. 1.7). Розробники з Білорусі випустили сервіс MSQRD, який потім купив Facebook за досить непогані гроші [24].

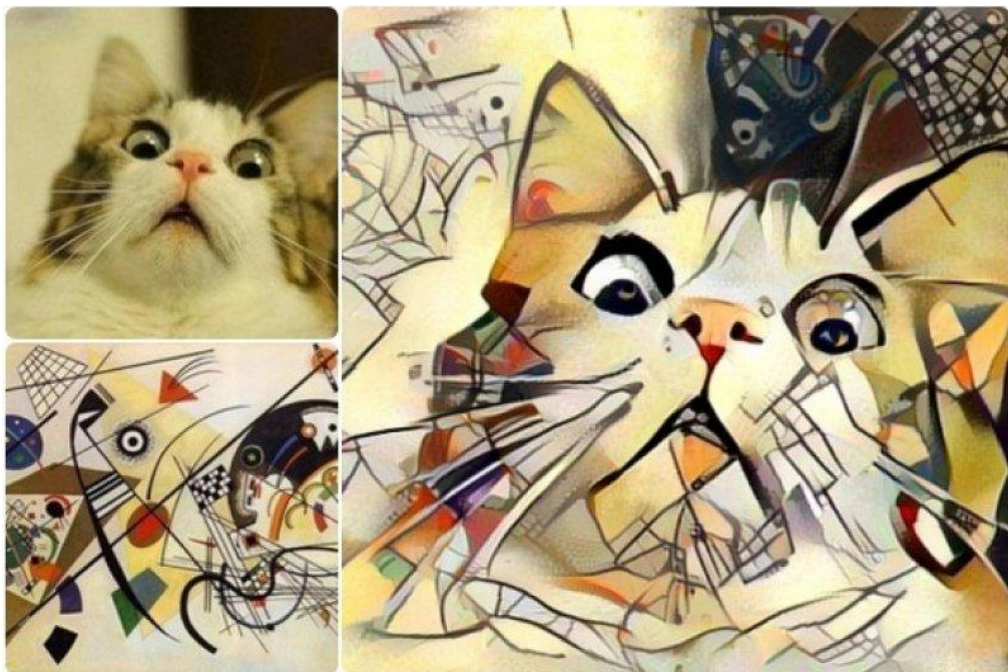


Рисунок 1.7 – Зразок дії MSQRD [24]

Крім того, існують і безліч інших призначень для неймереж. Розпізнавання осіб на допомогу поліції, створення більш спрощених інструментів для полегшення різних видів діяльності, автоматизація різних

трудоу процесів. Окрім того, існує забагато інших можливостей, про які поговоримо надалі.

1.5 Висновки за розділом 1

Були проаналізовані існуючі нейромережі, принципи їх роботи, особливості. Також було проаналізовано і порівняно між собою різні методи розпізнавання осіб на зображеннях. Окрім цього, були вивчені існуючі приклади таких мереж, приклади результатів їх роботи. Ще були зазначені переваги у використанні нейромереж у повсякденному житті людей, їх основні цілі та було проаналізовано області їх можливого використання.

2 ХАРАКТЕРИСТИКИ ОБ'ЄКТА ПРОЕКТУВАННЯ

Існує велика кількість нейромереж, котрі люди використовують для облегшення повсякденного життя. В цьому розділі ми розглянемо вже існуючі застосунки на базі нейромереж, а також виявимо особливості цих систем, алгоритми та функції.

2.1 Існуючі застосунки

Перед тим, як приступити до основної частини проекту, слід зупинитись на прикладах таких мереж, ознайомитися з принципами їхньої роботи та алгоритмами, детально розглянути результати роботи.

2.1.1 AutoDraw

Перший екземпляр застосунку, заснованим на НМ – Auto Draw. Це такий сервіс, який допомагає швидко створити основні контури об'єктів та зробити з цього іконку або деяке зображення. Користувачі можуть накреслити кілька основних ліній, і система зробить почне обробляти це, щоб інтерпретувати їх і замінити професійними лініями запланованого об'єкта. Тобто дозволяє малювати схематично та швидко і отримувати професійний результат. Окрім цього, як виявив CNET, він цілком здатний інтерпретувати мінімалістичні сучасні твори мистецтва, наприклад ескізи Пабло Пікассо із зображенням його улюбленої собаки Лумп.

В офіційному блозі Google, що представляє AutoDraw, він описується як «новий веб-інструмент, який поєднує машинне навчання з малюнками, створеними талановитими художниками, щоб допомогти вам малювати», тому що «малювати на телефоні чи комп'ютері може бути повільним і складним».

Ідея полягає в тому, що ви можете швидко отримати доступ до чистих і простих зображень від професійних художників, що дозволить вам легко створювати флаєри, запрошення та інші матеріали, які в іншому випадку вимагають свіжості і новизну графічного дизайну. Він має простий, мінімалістичний і зрозумілий інтерфейс (рис. 2.1, рис. 2.2) [25].

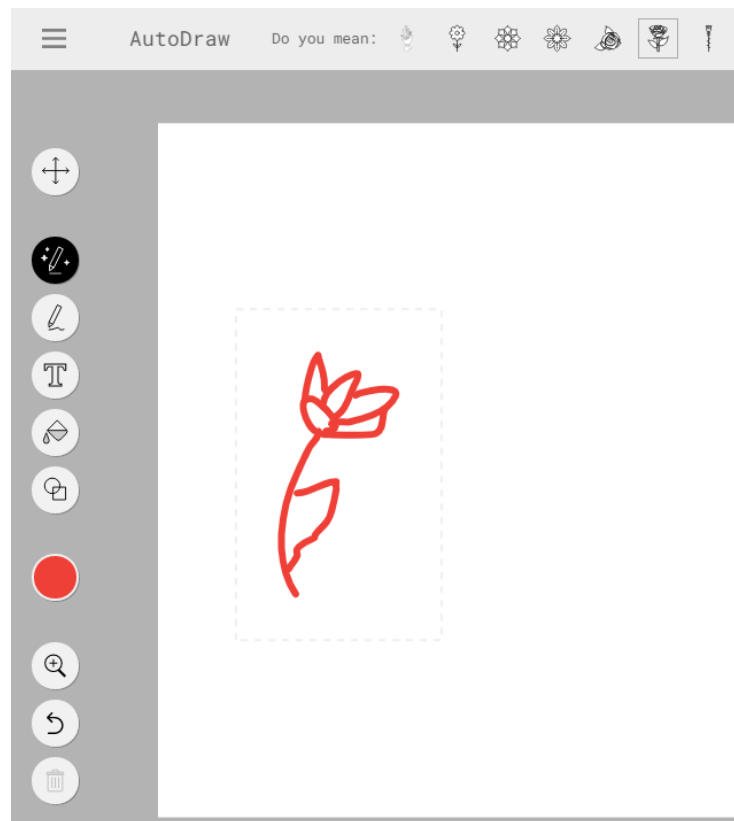


Рисунок 2.1 – Початок роботи [25]

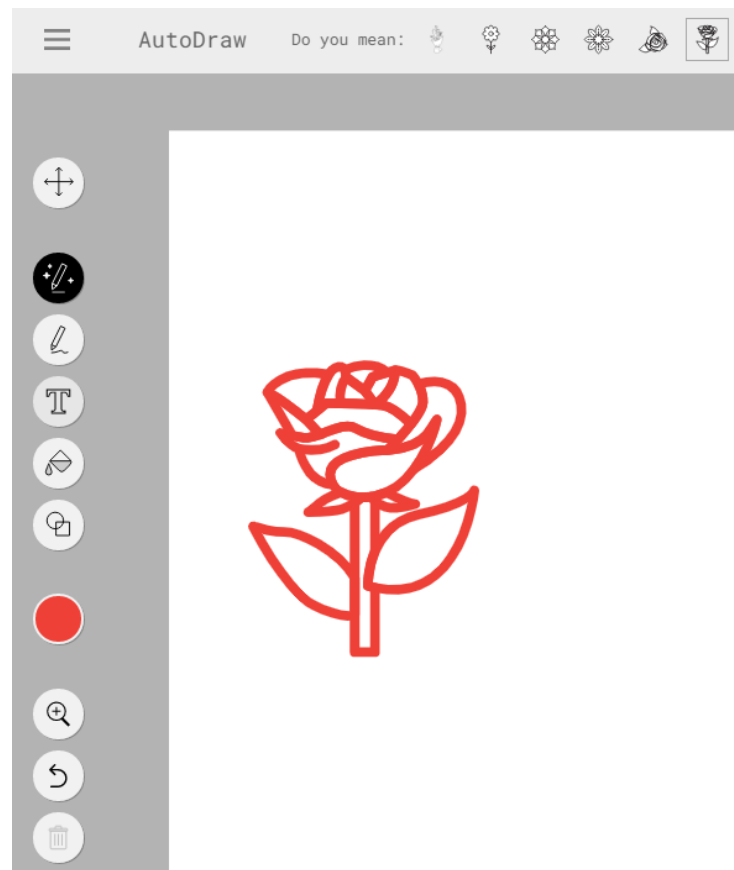


Рисунок 2.2 – Результат роботи Auto Draw [25]

Застосунок підтримується на усіх доступних платформах: смартфонів, планшетів, ноутбуків, комп'ютерів, тощо.

2.1.2 Shazam

Shazam (рис. 2.3) - безкоштовний кросплатформовий проект на основі неймереж, що дозволяє користувачам дізнаватися, що за пісня грає на фоні. На даний момент існує програма для iOS, Android, macOS, WatchOS.

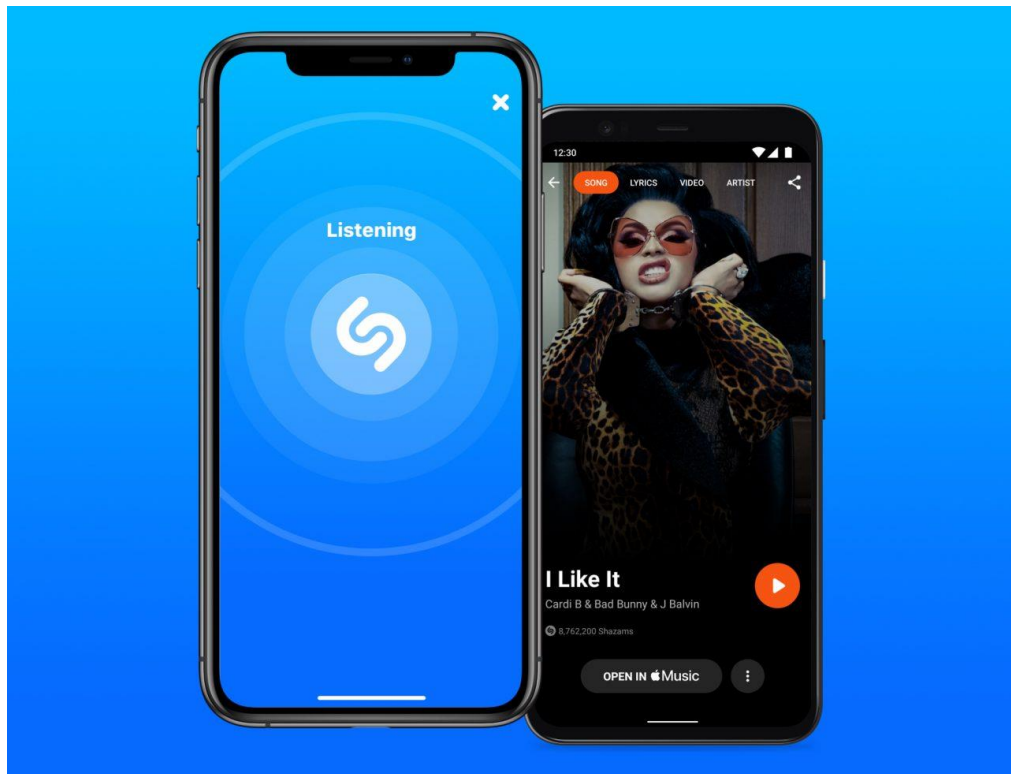


Рисунок 2.3 – Зовнішній вигляд за стосунку [26]

Користувач Shazam використовує мікрофон свого пристрою для запису фрагмента музики, яка грає поруч. Потім, програма порівнює фрагмент із центральною базою даних і при успішному зіставленні видає інформацію про трек. Система створює спектрограму пісні, тобто її аудіо відбиток, а потім алгоритми порівнюють фрагмент із мільйоном інших, очищуючи звук від сторонніх шумів (рис. 2.4). У такий спосіб можна знайти будь-який трек, який ви почули по радіо, у магазині чи на вулиці.

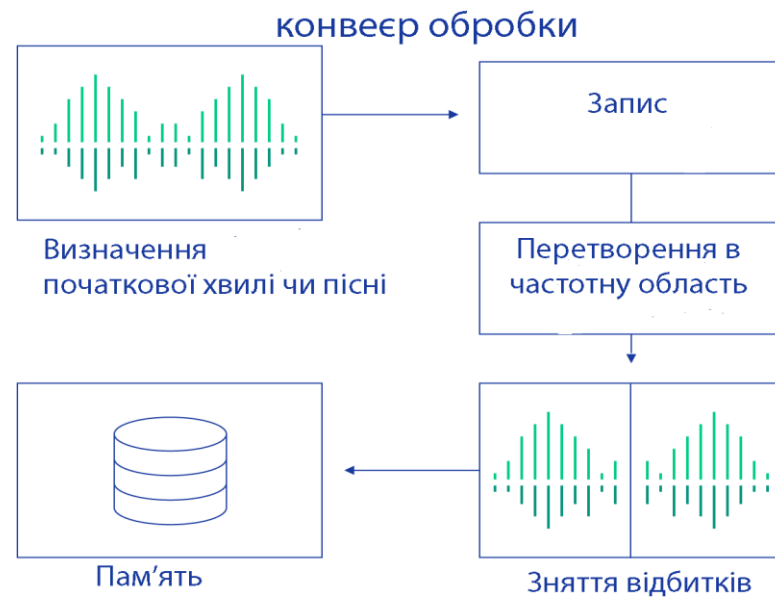


Рисунок 2.4 – Принцип роботи [26]

Таким чином, завдяки такій технології вкупі із зрозумілим дизайном, користувачі отримують змогу розпізнати будь-яку мелодію, незважаючи на зовнішні шуми (рис. 2.3, рис. 2.4) [26].

2.1.3 Deep Fake або Face Swap

Deep Fake – це синтетично вироблений медіа контент, в якому оригінальна людина (та, хто спочатку знаходиться на фото або відео), заміщується іншою людиною (рис. 2.5).



Рисунок 2.5 – Приклад заміни обличчя [27]

Цікаво, що початком історії таких технологій можна вважати 19 століття, коли розробили перші методики маніпуляцій над фото. Далі технологія розвивалася як у наукових установах, так і простими ентузіастами спільноти.

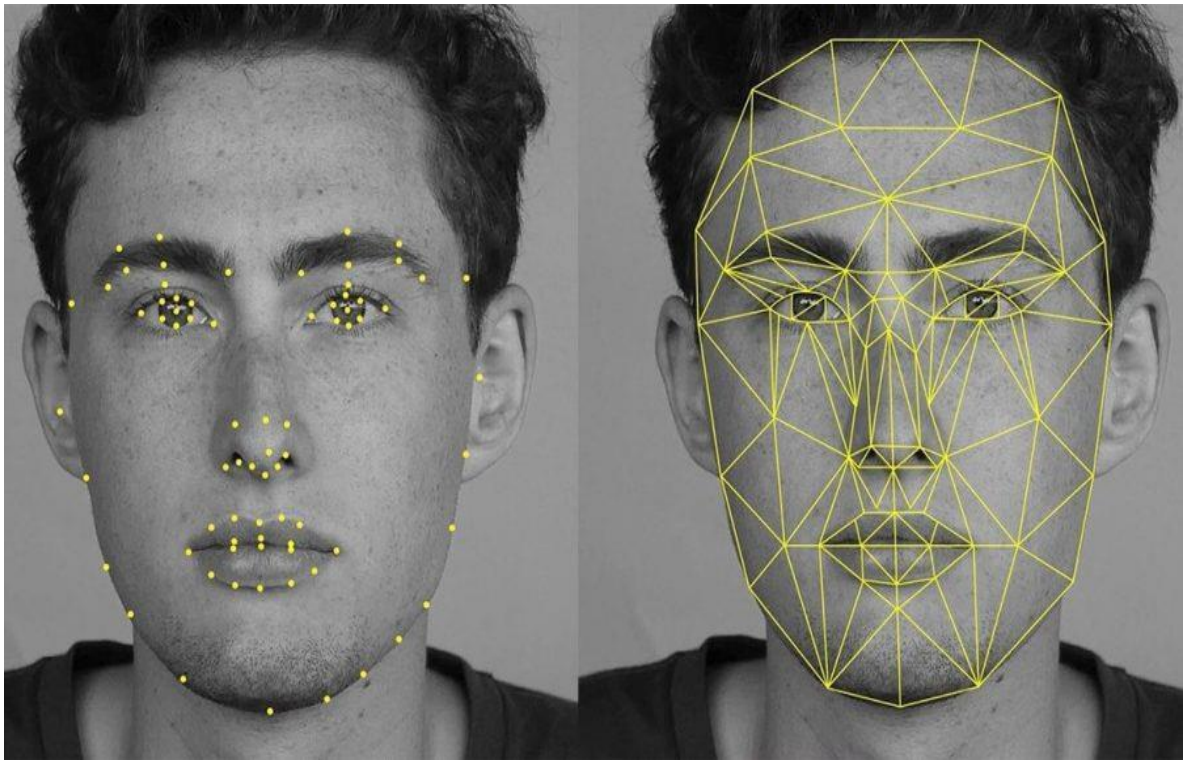


Рисунок 2.6 – Приклад процесу обробки обличчя [27]

В основі роботи таких алгоритмів лежить неймережа-автокодувальник (рис. 2.6) [27]. Пізніше було оновлення до генеративно-змагальної мережі.

Якщо розглядати принцип створення таких фото чи відео загалом, то основні етапи у їх створенні такі:

- неймережі дається великий масив даних;
- вона обробляє масив та вивчає особливості особи та міміки;
- як наслідок, неймережа вчиться відтворювати обличчя людини і її можна використовувати для синтезу контенту.

Але де можливо застосувати цю технологію? Практично скрізь, де можна використовувати відео, фото чи звук.

Як результат, у багатьох фільмах завдяки цієї системі стає можливо «омолодити» акторів, що дає змогу спростити процес кіновиробництва, дати більш творчої свободи та звісно, зекономити бюджет (рис. 2.5) [27].

2.2Висновки за розділом 2

Нейронні мережі можуть знаходити різне застосування, причому не тільки для розпізнавання зображень і текстів, але і в багатьох інших сферах. НМ здатні до навчання, завдяки чому їх можна оптимізувати та максимально збільшувати функціональність.

Дослідження НМ – це одна з найперспективніших областей в даний час, оскільки в майбутньому вони будуть застосовуватися практично повсюдно, у різних галузях науки і техніки, тому що вони здатні значно полегшити робочі процеси, життя суспільства в цілому.

У цій роботі наголос буде зроблено на дослідження систем із розпізнавання образів, тому що вони мають найбільший технологічний потенціал у майбутньому.

Також, на цих прикладах було продемонстровано різноманітність таких нейромереж та можливі галузі їх використання.

3 ОГЛЯД СТРУКТУРИ ПРОЄКТУ

3.1 Особливості технологій

Ми вже визначили, що таке нейромережа для розпізнавання облич і як вона працює, залишилося дослідити її призначення. Отже, де використовується дана нейромережа? Ми зазначимо це у таблиці 3.1 [28]-[29].

Таблиця 3.1 – Області використання НМ

Області використання	Подробиці
1	2
Служби безпеки	Криміналісти, поліція та спецслужби використовують автоматизовані системи біометричної ідентифікації (ABIS) для пошуку злочинців, докази злочинів та їх запобігання — наприклад, терактів чи шахрайства з документами. Камери з розпізнаванням осіб застосовують для забезпечення безпеки на масових заходах, контролю в аеропортах, контролю доступу в різних організаціях. Системи допомагають знаходити пропавших дітей, дезорієнтованих дорослих або людей, яких тримають у рабстві.
Охорона здоров'я та медицина	НМ у лікарнях та будинках для людей похилого віку допомагає відстежувати, чи приймають пацієнти ліки та стежити за їх станом через спеціальний монітор. Нейромережи навіть вміють виявляти по обличчю генетичні захворювання за характерними ознаками, такі як синдром Ді Джорджі, та оцінювати загальний стан пацієнтів.
Торгівля, громадське харчування та банківська справа	Технологія допомагає ідентифікувати клієнта та запобігти шахрайству під час покупки в магазині, аналізувати поведінку людей та оптимізувати сервіс так, щоб продавати більше товарів та отримувати більше грошей. За допомогою онлайн-біометрії можливо відкрити рахунок та отримати кредит, а також зняти гроші у банкоматі. Наприклад, у KFC та Amazon Go діє «оплата по обличчю». У нас біометрію планують запровадити у найбільших банках.

Продовження таблиці 3.1

1	2
Освіта	Сервіси на базі цієї технології допомагають під час онлайн навчання: стежать, щоб учень не відволікався під час іспиту, не списував та не користувався усними підказками. Одна школа у Швеції використовує FRT для перевірки відвідуваності на уроках. Школи в США, особливо в Нью-Йорку, починають тестувати використання технологій розпізнавання осіб як систему раннього оповіщення проти загроз з боку маніяків та інших злочинців.
Інші	Технології розпізнавання осіб застосовують у найрізноманітніших сферах для покращення життя людей, його облегшення та захисту від потенційних загроз. З кожним роком з'являється більше можливостей та областей для впровадження цієї технології.

Однак, було б помилкою не розглянути, які небезпеки або недоліки приховує дана система. Деякі люди вважають, що вона ставить під загрозу безпеку та громадянські права людини, тому її часткове регулювання має бути замінене повною заборонаю [30]-[31]. Поки весь світ захоплений перевагами технології розпізнавання осіб, деякі фахівці безпеки вважають, що вона таїть у собі велике зло для людства. Здавалося б, технологія несе лише переваги, у світі запанує абсолютна справедливість, реалізуються найнеймовірніші ідеї людства. Але жоден із винайдених людством механізмів спостереження не несе у собі такої небезпеки, як технологія розпізнавання осіб. Цей ідеальний інструмент утисків дуже звабливий, аби ним не скористалися тоталітарні уряди для встановлення авторитарного контролю та всеосяжних режимів, які знищують поняття особистого життя.

Американський союз захисту громадянських свобод спільно з 70 іншими правозахисними організаціями вимагає від компанії Amazon припинити надавати технологію уряду, а також закликав Конгрес ввести мораторій на її використання. До них підключилися і деякі ЗМІ, які висловили своє занепокоєння. Наприклад, редколегія Washington Post вважає, що Конгрес зобов'язаний негайно втрутитися у ситуацію. У парламентаріїв

теж є вагомий привід задуматися: деяких із них програма розпізнавання облич Amazon переплутала зі злочинцями [32].

Що за підсумком? Технологія розпізнавання осіб відкриває безмежні можливості відстежувати інформацію про особистість та переміщення людини. А також практично миттєво зберігати, розповсюджувати та аналізувати її. Розвиток цієї технології в майбутньому може призвести до того, що конфіденційність приватної інформації людини постійно порушуватиметься.

Деякі вчені закликають до повної необхідності заборони цих систем. Проте інші розробники, інженери та вчені доводять, що ця позиція є невірною та небезпечною для суспільства. Вони пропонують нейтральну з технологічної точки зору тактику: заборона на конкретні дії, а також конституційно позначити цінності та права, які потрібно захистити цілісно та детальніше.

3.2 Вибір мови програмування

Навіщо потрібно створювати програму на основі технології розпізнавання осіб? На це є декілька основних причин:

- допомога у розвитку систем штучного інтелекту;
- висока заробітна плата у цій сфері;
- робота на покращення якості життя суспільства;
- розвиток власних навичок.

Виходячи з вищенаведених причин було вирішено провести дослідження даної галузі та зайнятися написанням програми на цю тему[33]. Але яку мову та яку IDE слід використовувати?

Написати нейромережу можна будь-якою мовою програмування, таких як Java, Python та інші. Окрім цього, вибір мови може бути продиктований наявністю вже готових спеціалізованих бібліотек для

створення неймереж, таких як: Theano, TensorFlow, Lasagne та багато інших [33]-[34].

Аби обрати мову програмування, треба провести аналіз та порівняння деяких існуючих мов, завдяки яким стає можливим реалізувати потрібний застосунок. Аналізувати будемо мови Python та Java, які зазначені у таблиці 3.2 [35], тому що ці мови є самими популярними у цій галузі.

Таблиця 3.2 – Порівняння та аналіз мов програмування

Критерії	Python	Java
1	2	3
Простота вивчення	Мова загального призначення, яка відрізняється від конкурента потужністю, ефективністю та високим рівнем читабельності коду. Python широко відомий своєю лаконічністю та простотою у вивченні. Те, на що в Java у вас піде кілька рядків коду, в Python може бути реалізовано лише однією. Якщо взяти одну й ту саму задачу і спробувати її реалізувати на Python, то можна відзначити меншу кількість символів у пайтон-кодї, що є великою перевагою.	Це швидка, безпечна та надійна мова програмування загального призначення. Серед недоліків треба зазначити громіздкість, проте яка дозволяє створювати додатки з більшою опрацьованістю та реалізовувати більш складнішу логіку.
Динамічна типізація та строга типізація	Python володіє динамічною типізацією, що означає виключення необхідності оголошувати кінцевий тип змінних та функцій під час написання коду. Отже, їх типи з'ясовуються під час виконання програми автоматично.	У Java ж строга типізація: кінцеві типи змінних та функцій мають бути встановлені на етапі компіляції. Це означає, що вони повинні бути написані вручну під час написання коду.
Продуктивність	Python зазвичай компілює код під час виконання програми. Проте, якщо використати інтерпретатор PyPy, то стає можливим виконувати своєчасну компіляцію всіх програм або їх частин у нативний код, що значно підвищує продуктивність.	Java має змогу виконувати компіляцію заздалегідь та розподіляти байт-код, що також являється значним плюсом у продуктивності програм.

Продовження таблиці 3.2

1	2	3
Табуляція	Одна з головних особливостей, це відсутність звичних для того ж Java точок з комою і фігурних дужок. Замість цих елементів у коді використовується табуляція та пробіли. Хоча табуляція і є більш зручною та вільнішою, треба зазначити і недолік. Під час кодингу можна дуже серйозно застрягти на пошуку помилок у коді, що може привести до критичних багів та невідпрацюванні програми.	Візуально стиль синтаксису з дужками та точками із комами є суворішим. Він визначає правила, за якими повинен грати програміст. Такий зовнішній вид дуже сильно відволікає від самого коду, робе його менш читаючим.
Застосування	Python вважається більш універсальним, ніж Java. Його лаконічність приносить йому помітну славу у багатьох областях. Наприклад, розробка веб додатків з використанням фреймворків, у завданнях системного адміністрування, у тестуванні, у роботі з великими обсягами даних, у машинному навчанні, а також розробка нейронних мереж.	Ця мова чудово почувається в мобільній розробці під Android і iOS, що може принести добру зарплатню.

Резюмуючи все вищевикладене у таблиці 3.2, відзначимо основні особливості обох мов. Таким чином, Java:

- швидка, безпечна та надійна мова програмування загального призначення;
- статично типізований, має суворі правила синтаксису, а також високу продуктивність;
- дуже популярний у мобільній розробці під мобільні платформи.

Python:

- мова загального призначення, що відрізняється потужністю, ефективністю та високим рівнем читабельності коду;

- динамічно типізована мова, код більш читабельний і лаконічний, відсутні дужки та крапки з комою у синтаксисі;
- продуктивність трохи нижча, ніж у Java;
- незамінний у системному адмініструванні, дуже популярний у обчисленнях з великими масивами даних, а також у веб розробці.

Отже, всім відомо, що Python - це найпопулярніша високорівнева мова програмування з динамічною семантикою (рис. 3.1) [35]. Він досить простий для роботи та читання: його використання значно знижує вартість розробки та обслуговування програм. Тому його обір в якості основної мови програмування, є обґрунтованим.

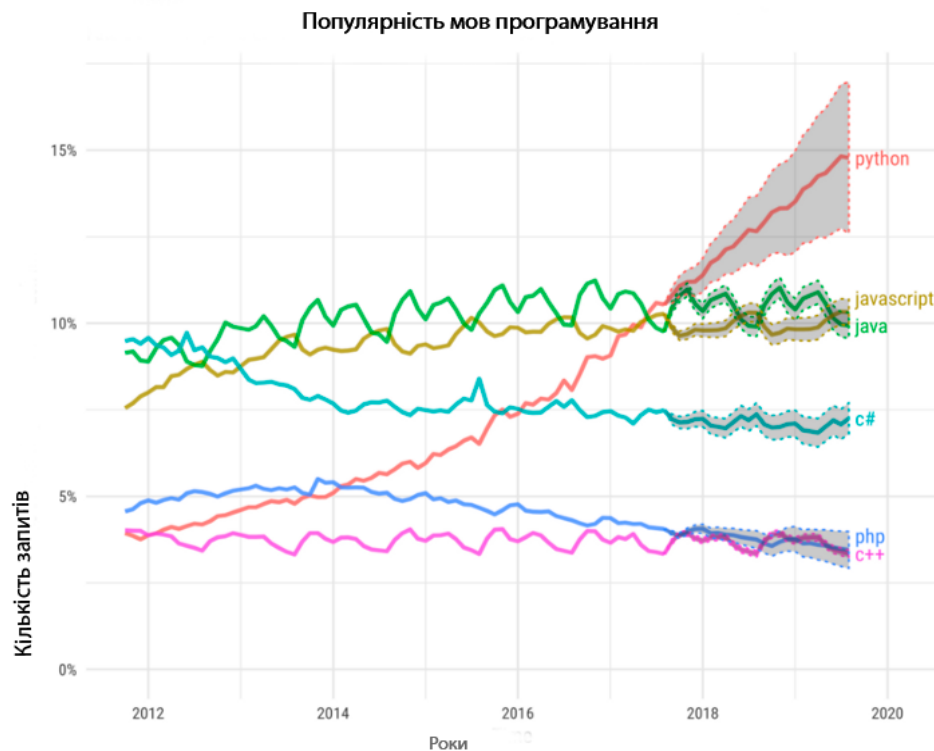


Рисунок 3.1 – Популярність мов програмування [35]

Наступна перевага Python полягає в його гнучкості: наприклад, розробник має вибір між об'єктно-орієнтованим підходом і скриптами. Python допомагає поєднувати різні типи даних. Більш того, він особливо зручний для тих розробників, які більшу частину коду пишуть за допомогою IDE. Крім простоти, Python має ще один плюс — він досить легко взаємодіє

з іншими мовами, особливо з C і C++. Одна з основних причин, чому ця використовується для створення систем штучного інтелекту, полягає в тому, що він має безліч фреймворків, які спрощують процес написання коду і скорочують час на розробку [34]-[35].

3.3 Середи розробки

Для Python найкращою IDE є PyCharm. Він надає засоби для аналізу коду, графічний налагоджувач, інструмент для запуску юніт-тестів та підтримує веб розробку на Django (рис. 3.2, рис. 3.3). PyCharm розроблена компанією JetBrains на основі IntelliJ IDEA, як і PHPStorm. Ця IDE сумісна з Windows, macOS, Linux [36].

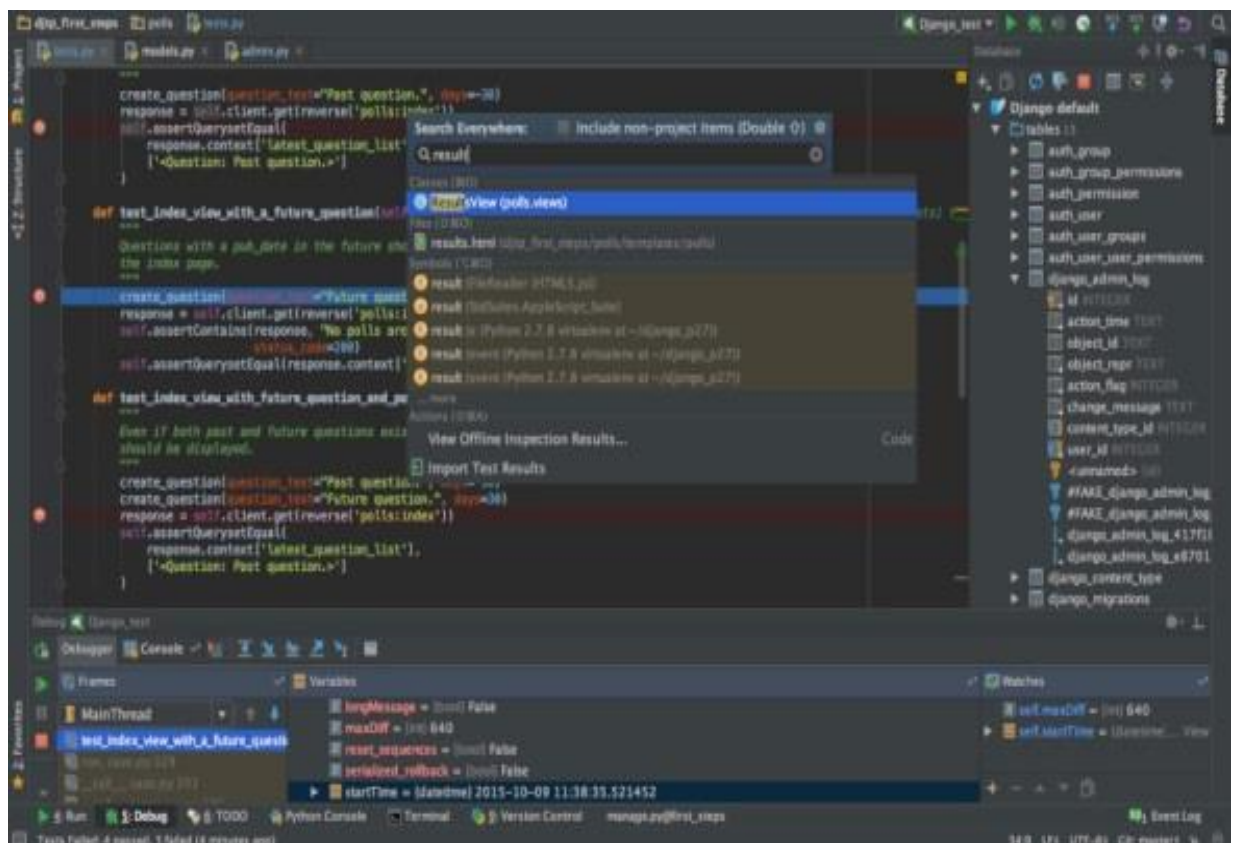


Рисунок 3.2 – Робота у PyCharm

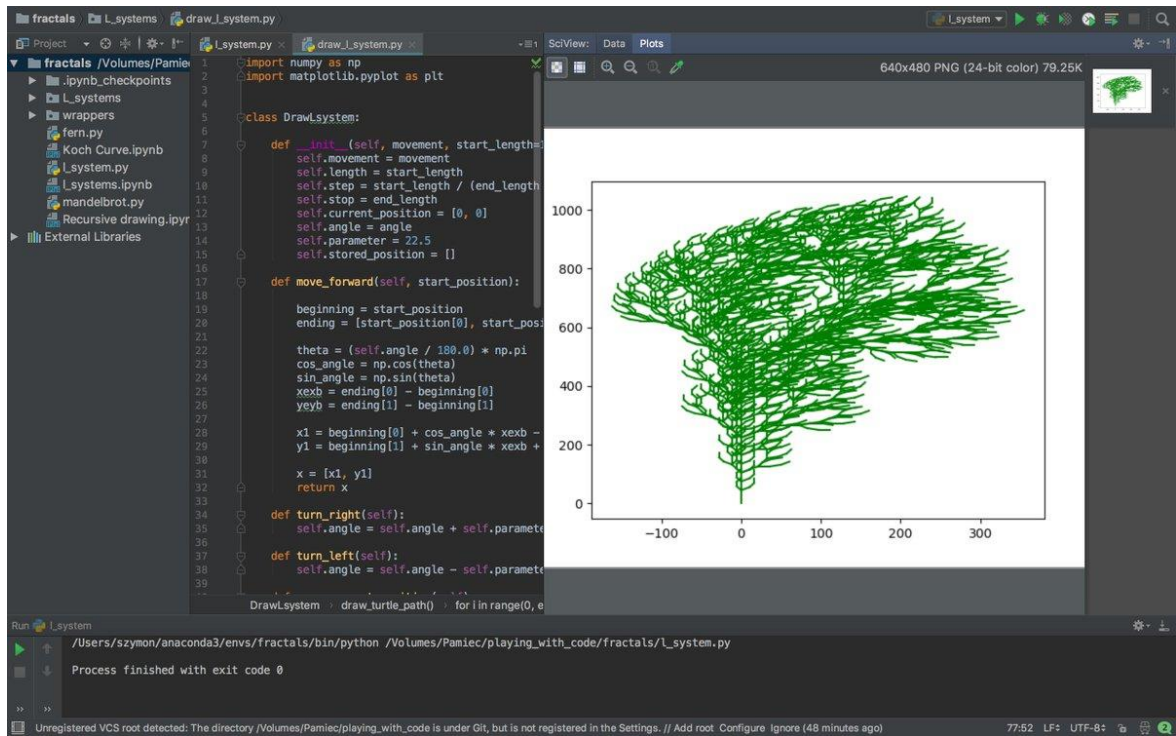


Рисунок 3.3 – Приклад роботи з зображенням

Слід зауважити, що безапеляційно було обрано PyCharm у якості IDE, проте треба згадати, що існує ще декілька середовищ для розробки застосунків на мові Python. Це PyDev, WingWare, Komodo IDE, Eclipse. В порівнянні з PyCharm вони мають не забагато функціоналу, менш популярні та не дуже зручні у використанні, тому, я вважаю, вибір PyCharm у якості IDE є обґрунтованим [36]-[37].

3.4 Коротко про архітектуру

Для початку, пробіжимося за визначенням нейронної мережі. Штучна нейронна мережа – обчислювальна нелінійна модель, в основі якої лежить нейронна структура мозку, здатна навчатися виконання завдань класифікації, передбачення, прийняття рішень, візуалізації та деяких інших лише завдяки розгляду прикладів. Будь-яка архітектура складається зі штучних нейронів - елементів обробки, що мають структуру 3 пов'язаних один з одним шарів: вхідним, що складається з одного або більше шарів прихованим і вихідним

(рис. 1.1) . Кожен нейрон має входи з вагами – синапсами, функцію активації, що визначає вихідну інформацію при заданій вхідній, та один вихід. Синапси - регульовані параметри, що конвертують нейронну мережу в параметризовану систему (рис. 3.4).

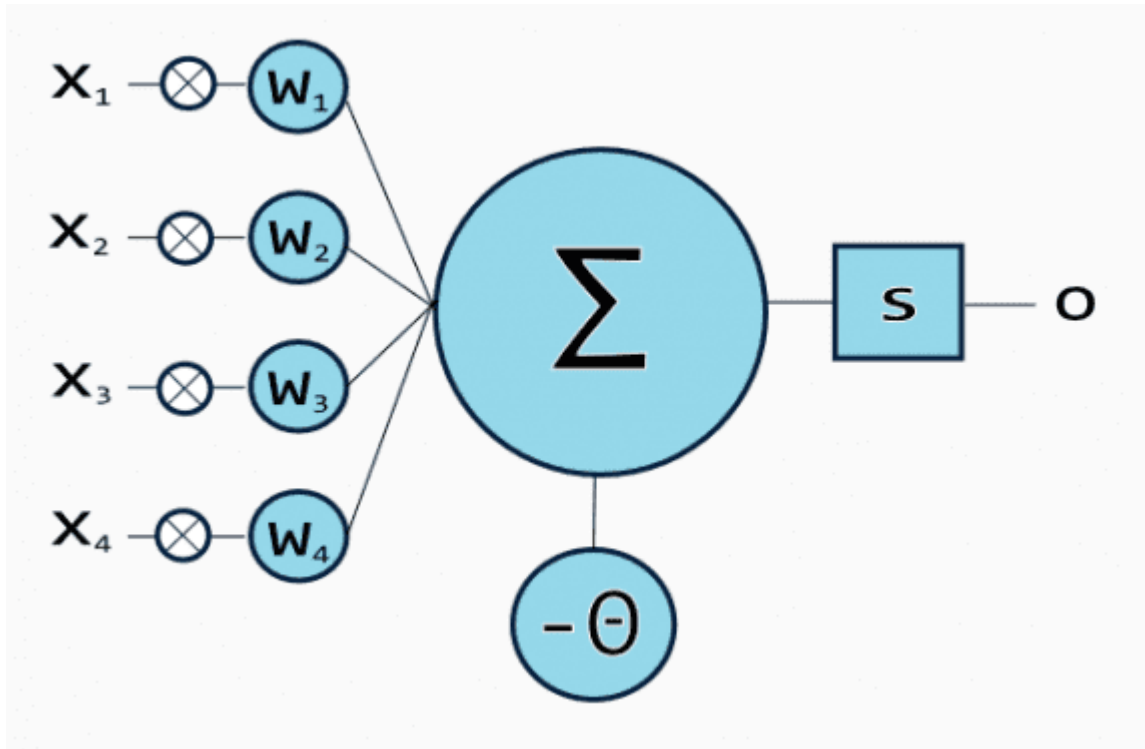


Рисунок 3.4 – Штучна нейронна мережа з 4 входами [38]-[40]

Зважена сума входів – активаційний сигнал – проходить через функцію активації виведення даних з нейрона. Є кілька видів функції активації [38]:

- лінійна: $f(x) = ax$;
- ступінчаста: $f(x) = \begin{cases} 0, & x < 0; \\ 1, & x \geq 0; \end{cases}$
- сигмоїдна: $f(x) = \frac{1}{1+e^{-x}}$;
- тангенціальна: $f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$;
- випрямна: $f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases}$.

За допомогою додаткових прихованих шарів можна зробити систему більш гнучкою і потужною. Нейромережа із багатьма прихованими шарами називаються глибокими нейронними мережами, вони створюють складні нелінійні зв'язки [38]-[40].

Один із способів розпізнати образ – знайти контур об'єкта та дослідити його властивості. За цим принципом працює метод Віола-Джонса з використанням ознак Хаара, який вигадав угорський математик Альфред Хаар. Ознаки - це набір геометричних фігур із чорно-білим візерунком, їх ще називають маски. Вони допомагають знайти межі будь-якої форми, наприклад, обриси обличчя, лінії брів, носа або рота (рис. 3.5, рис. 3.6) [41]-[43].

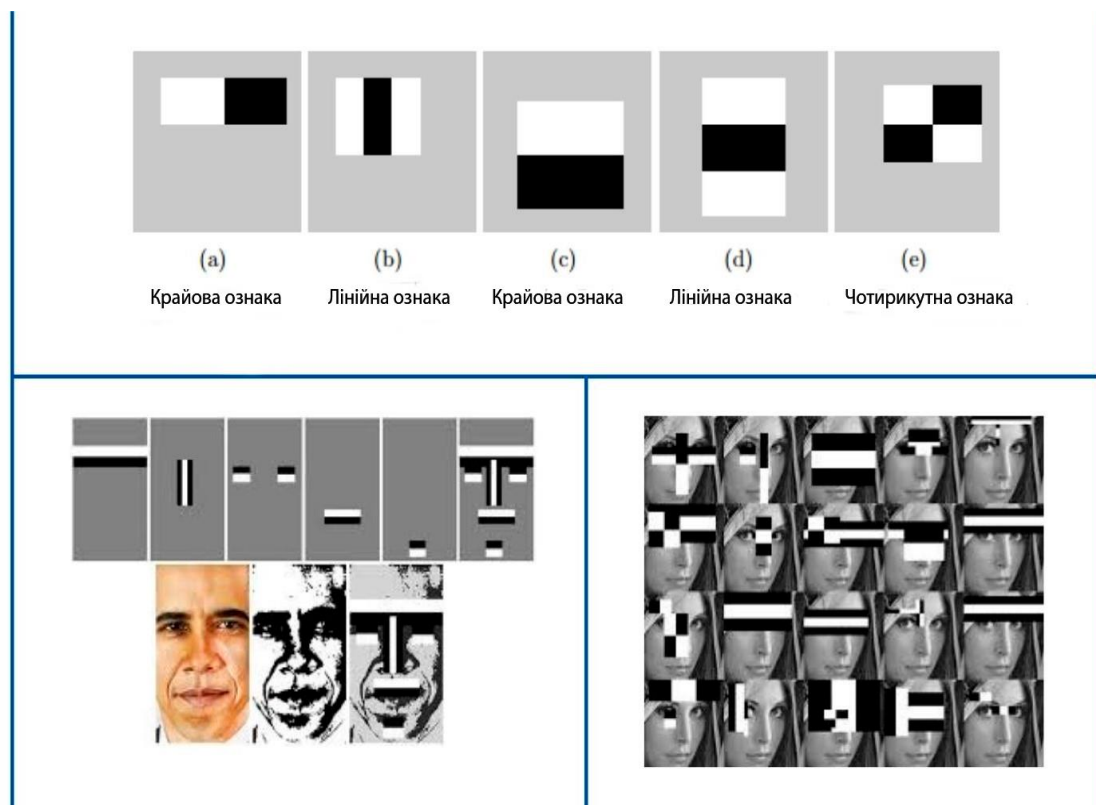


Рисунок 3.5 – Ознака Хаара [42]

У алгоритмі Віоли-Джонса маски накладаються різні частини кадру, а програма визначає, чи може у них перебувати об'єкт. Працює це так:

- класифікатор (алгоритм, який шукатиме об'єкт у кадрі) навчають на фотографіях осіб та набувають граничного значення — його перевищення сигналізуватиме про те, що в кадрі є особа;
- у класифікатор завантажують зображення, на якому шукатимуть обличчя;
- класифікатор накладає на нього маски та окремо складає яскравість пікселів, що потрапили в білу частину маски, та яскравість пікселів, що потрапили у чорну частину маски. Потім із першого значення він віднімає друге.

Результат порівнюється з граничною величиною. Якщо результат менший за порогову величину, значить, у частині кадру немає особи, і алгоритм закінчує свою роботу. А якщо більше, то він переходить до наступної частини кадру (рис. 3.5, рис. 3.6).

Найпростішу прямокутну ознаку Хаара можна визначити як різницю сум пікселів двох суміжних областей усередині прямокутника, який може займати різні положення та масштаби на зображенні. Такий вид ознак називається 2-прямокутним. Віола та Джонс також визначили 3-прямокутні та 4-прямокутні ознаки. Кожна ознака може показати наявність (або відсутність) будь-якої конкретної характеристики зображення, як межі або зміна текстур. Наприклад, 2-прямокутна ознака може показати, де знаходиться межа між темним та світлим регіонами.

Наприклад розглянемо базу даних із людськими особами. Спільним для всіх зображень є те, що область в районі очей темніша, ніж область в районі щік. Отже загальною ознакою Хаара є 2 суміжні прямокутні регіони, що лежать на очах і щоках [41]-[43].

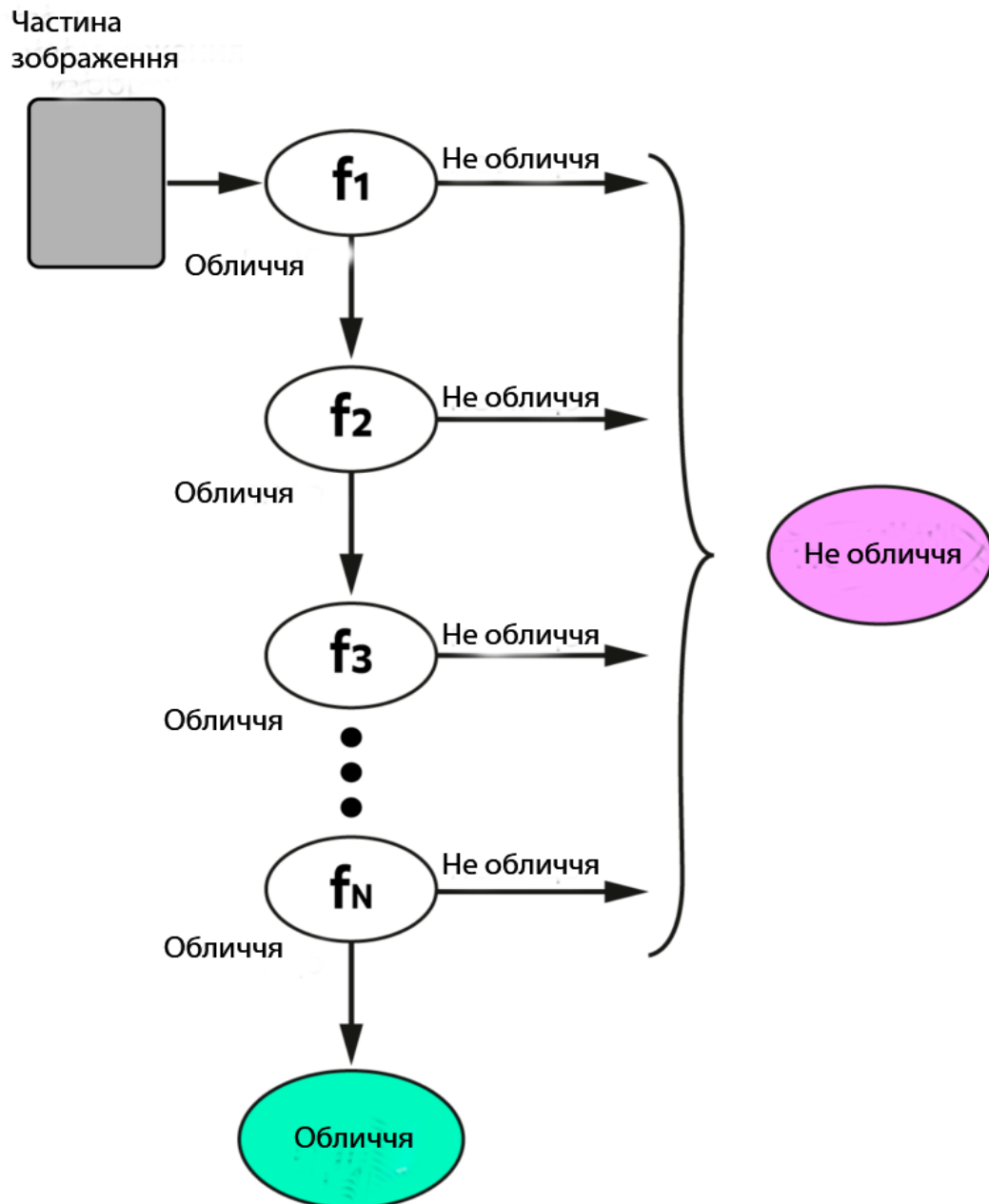


Рисунок 3.6 – Алгоритм класифікації обличчя [41]-[43]

3.5 MVC

Вже існує відполірована архітектура, за якою треба розробляти застосунки. Називається вона MVC (рис. 3.7). Вона складається із декількох незалежних модулів, які можна модифікувати. Треба зауважити, що MVC являється шаблоном проектування, паттерном, яким програмісти часто користуються для розробки застосунків [44].

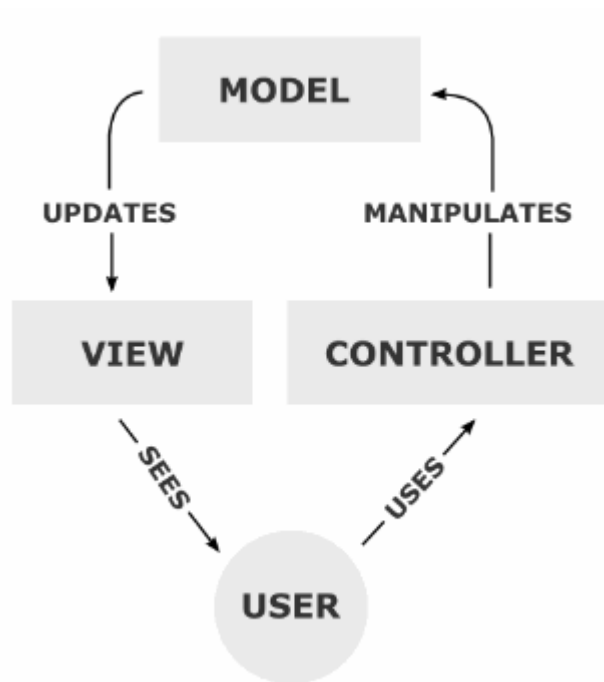


Рисунок 3.7 – Схема MVC [44]

Як виглядає MVC:

- користувач (User) – користується застосунком;
- контролер (Controller) – реагує на дії користувача, передає запити моделі про необхідність змін;
- модель (Model) – відповідає за оброблення запиту Дії, змінює свій стан;
- вигляд (View) – відповідає за відображення даних моделі користувачу, реагує на зміни моделі.

Модель, також іменується бізнес-логікою, відповідає за логіку програми, а також за саме звернення, наприклад, до бази даних. Вона бере необхідну вибірку даних з бази і працює з ними. Для прикладу, модель в додатку «Камера» буде відповідати за здійснення дії, такі як фотографування, відеозйомка. Треба зауважити, що розроблений застосунок є невеликим та не складним, тому були використані данні напряму, без створення та використання моделі. У застосунку буде використатися трохи злегшена архітектура задля оптимізації процесів розробки (рис. 3.8).

Вигляд – це все що бачить і з чим може взаємодіяти користувач, а саме: усілякі кнопки, списки, картинки, текстові поля. У застосунку FaceApp користувач побаче різні кнопки, фільтри, маски, налаштування, тощо. А в десктопному застосунку, що розробляється в цієї роботі, виглядом є все, що баче користувач: інтерфейс вікна, кнопки, початкове та фінальне зображення.

Контролер буде в ролі оброблювача дій користувача на поточному екрані, а після обробки взаємодіючи з моделлю, видає той чи інший результат за допомогою уявлення. Наприклад, у застосунку для прослуховування музики, контролер буде перевіряти доступ до зовнішній або внутрішній пам'яті, визначати формат файлів та включати звук. В деяких проектах кожен екран являє собою контролер. У моєму застосунку 3 головних контролера:

- кнопка обору зображення;
- кнопка початку обробки зображення;
- кнопка закриття вікна;

Модель відповідає за основну логіку застосунку. В даному випадку, модель буде намагатися визначити осіб на доступних зображеннях та виводити їх на головний екран користувачу.

Таким чином, для цієї роботи була використана ця архітектура (рис. 3.7, рис. 3.8) [44].



Рисунок 3.8 – Схема MVC у роботі

3.6 Висновки за розділом 3

Після аналізу предметної області було проаналізовано і архітектуру застосунку. Було порівняно та обрано мову програмування. Яку середу розробки було використано. Також було описано алгоритми роботи програми, як загально, так і по частинам. Були дослідженні методи та формули, використані у роботі.

4 КЕРІВНИЦТВО ПРОГРАМІСТА

4.1 Функціонал застосунку

Десктопний застосунок реалізує такий функціонал:

- взаємодія з файлами зображень у комп'ютері або у зовнішньому флеш-накопичувачі;
- визначення контурів обличчя на зображенні;
- визначення контурів обличчя через відеопотік з веб-камери;
- показ результату.

Застосунок працює за таким алгоритмом (рис. 4.1).



Рисунок 4.1 – Алгоритм роботи програми

4.2 Характеристики застосунку, ПЗ та АЗ

Для тестування застосунку обов'язково потрібний робочий комп'ютер або ноутбук з встановленим на нього ОС Windows, Linux або macOS. Обов'язково треба мати підключеною веб-камеру та знати її ID у системі.

Все, що потрібно розробнику - це встановлена IDE PyCharm, готовий інтерпретатор Python, необхідні плагіни та імпортувати бібліотеки, такі як OpenCV, Dlib, TensorFlow. Окрім цього, для проектування зовнішньої оболонки застосунку, буде необхідно встановити та налагодити VisualStudio.

4.3 База Даних

Уся необхідна інформація, а саме тестові зображення можуть знаходитися на комп'ютері: як у папці проекту, так і в іншому місці. Звертатися до цих файлів можливо напряму через запит у коді (наприклад, для отримання потоку з веб-камери), або через готову кнопку у вікні застосунку.

4.4 Важливі параметри

Для пошуку осіб на зображенні спочатку треба зробити зображення об'єктом, з яким буде можливо працювати, а потім ми будемо використовувати метод із сигнатурою (його ім'ям та списком параметрів).

Також треба перевести зображення в режим відтінків сірого (інформація про сам колір алгоритму не важлива, тільки його інтенсивність).

```
img = cv2.imread('exampl.png')
```

```
img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

```
cv2.CascadeClassifier.detectMultiScale (image [, scaleFactor [, minNeighbors [, flags [, minSize [, maxSize]]]]) (таблиця 4.1)
```

Таблиця 4.1 – Існуючі параметри та їх значення

Параметр	Значення
image	У цей параметр треба буде передавати 8-бітну матрицю зображення.
scaleFactor	Параметр показує, у скільки разів ми зменшуватимемо вихідне зображення, намагаючись виявити об'єкт: це робиться тому, що ми спочатку не знаємо, якого розміру буде обличчя. Чим меншим буде цей коефіцієнт, тим довше працюватиме алгоритм.
minNeighbors	Цей параметр впливає на якість виявлення: чим більше його значення, тим менше образів зможе виявити алгоритм, але водночас його робота буде точніше.

Якщо обличчя будуть виявлені, функція поверне набір об'єктів типу Rect (x, y, w, h) - прямокутників, початок яких задано парою координат (x, y), а ширина та висота - як w і h. У циклі додамо ці прямокутники у вихідне зображення image за допомогою cv2.rectangle(image, start_point, end_point, color, thickness) за координатами їх протилежних вершин (x, y) і (x+w, y+h).

5 КЕРІВНИЦТВО КОРИСТУВАЧА

5.1 Призначення застосунку

Застосунок призначений познайомити користувача із особливостями можливості визначення осіб на різних зображеннях.

Для того, аби користуватися цим десктопним застосунком, користувач повинен мати комп'ютер або ноутбук з будь-якою встановленою ОС. Також, необов'язково мати веб-камеру, аби була можливість роботи з відеопотоком.

5.2 Приклад взаємодії з програмою

При запуску застосунку, користувач повинен обрати та завантажити у програму файл зображення, після цього натиснути на кнопку обробки та очікувати результат (рис. 5.1, рис. 5.2).

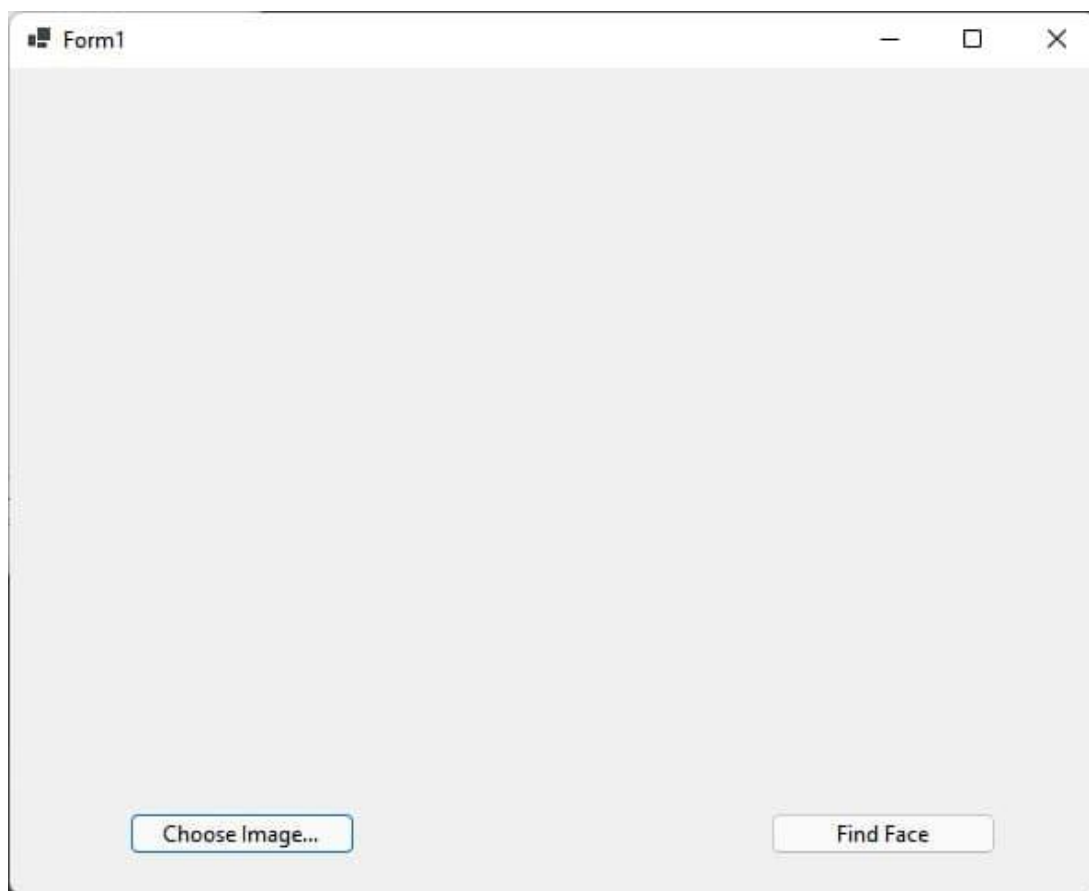


Рисунок 5.1 – Приклад початкового екрану

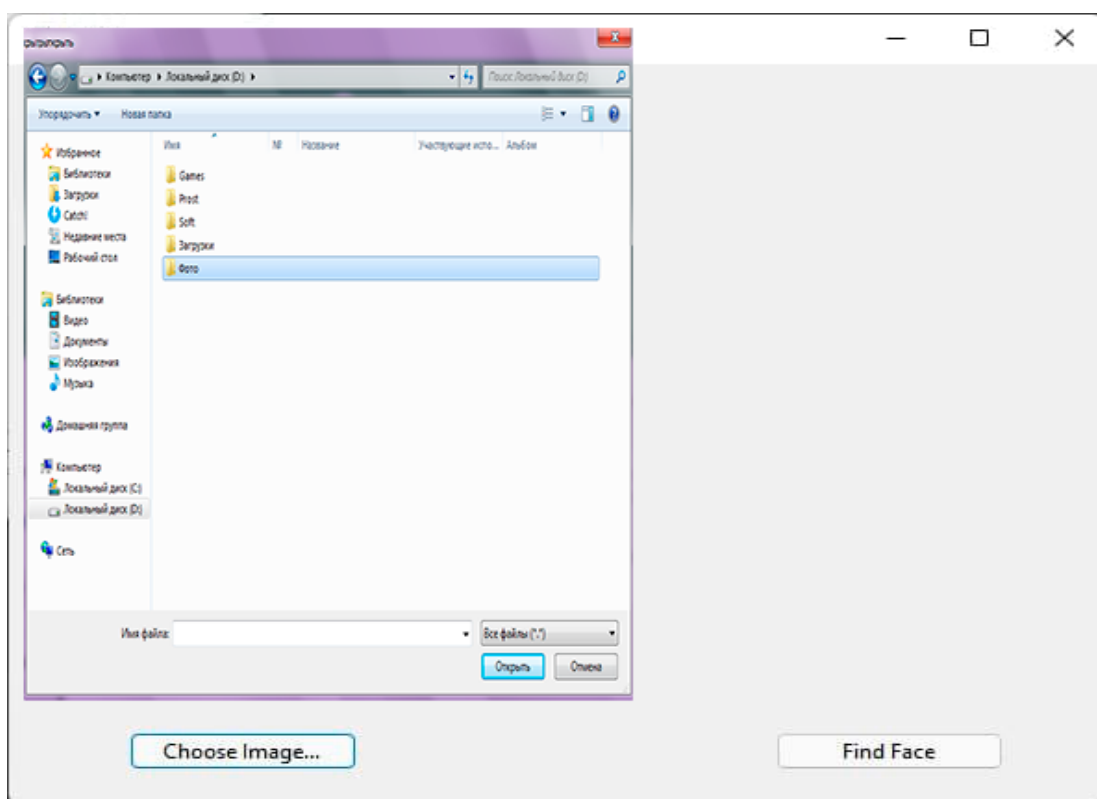


Рисунок 5.2 – Пошук зображення

Далі, користувач отримує вже оброблене зображення з виділеним у рамку обличчя людини, звісно, якщо програма змогла правильно проаналізувати файл та визначити на ньому людей. Якщо визначити контури обличчя не вдалося, користувачу повернеться його зображення без спеціальних рамок або відміток (рис. 5.3, рис. 5.4).

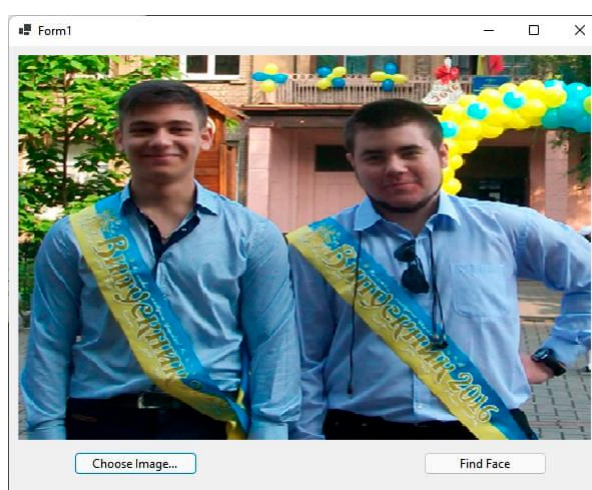


Рисунок 5.3 – Приклад роботи програми

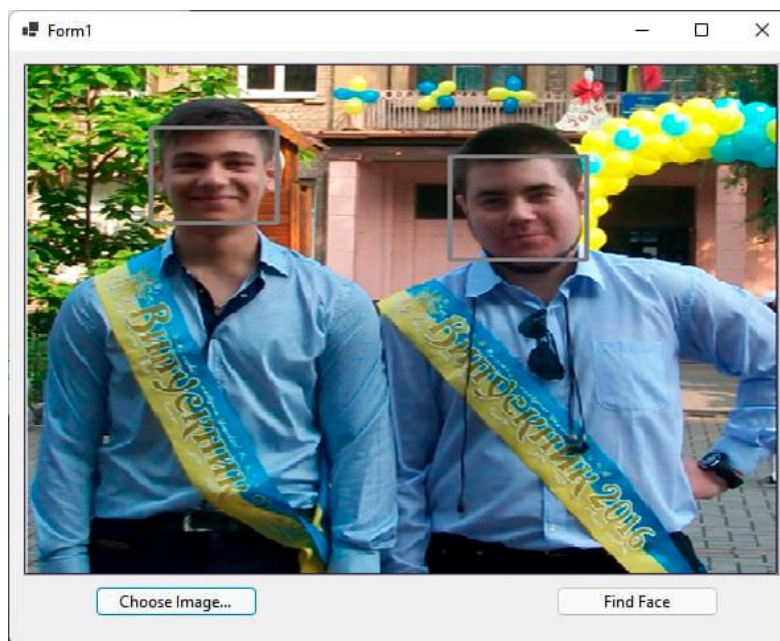


Рисунок 5.4 – Результат

Після цього, користувач має вибір: завершити роботу або перейти до обробки інших зображень.

Також, застосунок має змогу обробити вхідний потік з веб-камери, якщо користувач її має. В такому разі користувач отримує декілька секундний фрагмент з веб-камери, на якому буде відмічено у рамці обличчя людини, яка знаходиться у об'єктиві камери.

ВИСНОВКИ

При виконанні випускної кваліфікаційної роботи було проаналізовано предметну область системи, що розроблюється, а точніше, проектування застосунку розпізнавання осіб на зображенні. Аналіз отриманих даних дав можливість визначити головні вимоги до програмного продукту та сформулювати основні його функції. За допомогою системи потенційний користувач має змогу визначити контури обличчя на зображеннях та відеопотоках.

Задачі, що були виконані в роботі:

- був виконаний аналіз предметної області, вивчені аналогічні існуючі застосунки, підходи і програмні засоби для рішення поставленої мети;
- вивченні особливості обраної платформи, мови програмування, середовища розробки;
- розроблена програмна система може успішно використовуватися на деяких платформах.

Для розробки десктопного застосунку було обрано середовище розробки PyCharm, який підтримує мову Python.

Для того, щоб користуватися застосунком, необхідно мати звичайний персональний комп'ютер або ноутбук. Пристрій мусить мати вільне місце для встановлення застосунку. Для користувача не обов'язково, але не буде зайвим мати робочу веб-камеру, щоб мати змогу визначати осіб і на відео.

Для готової системи було розроблено керівництва користувача і програміста, а також надан вихідний код застосунку.

При розробці виконувалось тестування системи на наявність помилок, за допомогою реального пристрою.

Можна вважати, що програма відповідає усім стандартам якості та заявленим вимогам. Слід зауважити, що програма буде знаходитись відтепер на стадії підтримки готового продукту. Очікуються нові оновлення, можливі

створення нових функцій та можливостей застосунку, можлива повністю дизайнерська переробка.

Усі завдання, котрі були зазначені за для виконання у випускній кваліфікаційній роботі, були повністю виконані.

Також, був оптимізований та зменшений код програми, що зводить появу критичних багів до мінімуму, завдяки цьому було модифіковано алгоритм роботи програми, що зменшило навантаження на системи.

Наукова новизна у цій роботі полягає в оптимізації робочого простору та основного коду програми. Для досягнення цієї мети був модифікований код та принципи роботи, було покращено алгоритм роботи застосунку. Була проведена робота по запобіганні можливим критичним багам.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. О будущем человечества [Электрон. ресурс]. – Режим доступа: <https://integral-russia.ru/2018/07/16/razvitii-tehnologij-i-tom-kak-eto-mozhet-povliyat-na-razvitie-cheloveka/>.
2. Нейронная сеть [Электрон. ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%B9%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C.
3. Как работает нейронная сеть [Электрон. ресурс]. – Режим доступа: <https://neurohive.io/ru/osnovy-data-science/osnovy-nejronnyh-setej-algoritmy-obuchenie-funkcii-aktivacii-i-poteri/>.
4. Разновидности нейронных сетей. Элементарные конфигурации [Электрон. ресурс]. – Режим доступа: <https://tproger.ru/translations/neural-network-zoo-1/>.
5. The perceptron: a probabilistic model for information storage and organization in the brain [Electronic resource]. – Access mode: <https://www.ling.upenn.edu/courses/cogs501/Rosenblatt1958.pdf>.
6. Radial basis function network [Electronic resource]. – Access mode: https://en.wikipedia.org/wiki/Radial_basis_function_network.
7. Нейронная сеть Хопфилда [Электрон. ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%B9%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C_%D0%A5%D0%BE%D0%BF%D1%84%D0%B8%D0%BB%D0%B4%D0%B0.
8. Что такое цепи Маркова и как они работают [Электрон. ресурс]. – Режим доступа: <https://thecode.media/markov-chain/>.
9. Learning and relearning in Boltzmann machines network [Electronic resource]. – Access mode: https://www.researchgate.net/publication/242509302_Learning_and_relearning_in_Boltzmann_machines.

10. Restricted Boltzmann Machine and Its Application [Electronic resource]. – Access mode: <https://www.latentview.com/blog/restricted-boltzmann-machine-and-its-application/>.

11. Auto-association by multilayer perceptrons and singular value decomposition [Electronic resource]. – Access mode: <https://www.semanticscholar.org/paper/Auto-association-by-multilayer-perceptrons-and-Bourlard-Kamp/f5821548720901c89b3b7481f7500d7cd64e99bd?p2df>.

12. Sparse autoencoder [Electronic resource]. – Access mode: <https://paperswithcode.com/method/sparse-autoencoder>.

13. Variational autoencoder [Electronic resource]. – Access mode: https://en.wikipedia.org/wiki/Variational_autoencoder.

14. Autoencoders and the Denoising Feature: From Theory to Practice [Electronic resource]. – Access mode: <https://towardsdatascience.com/autoencoders-and-the-denoising-feature-from-theory-to-practice-db7f7ad8fc78>.

15. Глубокая сеть доверия [Электрон. ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/%D0%93%D0%BB%D1%83%D0%B1%D0%BE%D0%BA%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C_%D0%B4%D0%BE%D0%B2%D0%B5%D1%80%D0%B8%D1%8F.

16. Свёрточная нейронная сеть [Электрон. ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D1%91%D1%80%D1%82%D0%BE%D1%87%D0%BD%D0%B0%D1%8F_%D0%BD%D0%B5%D0%B9%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C.

17. Deconvolutional Neural Network [Electronic resource]. – Access mode: <https://habr.com/ru/company/nordavind/blog/253859/>.

18. Распознавание образов нейросетями [Электрон. ресурс]. – Режим доступа: <https://center2m.ru/ai-recognition>.

19. How facial recognition works [Electronic resource]. – Access mode: <https://us.norton.com/internetsecurity-iot-how-facial-recognition-software-works.html>.
20. What is Facial Recognition - Definition and Explanation [Electronic resource]. – Access mode: <https://www.kaspersky.com/resource-center/definitions/what-is-facial-recognition>.
21. Метод гибкого сравнения на графах [Электрон. ресурс]. – Режим доступа: <http://masters.donntu.org/2016/fknt/lychagin/library/article1.html>
22. Метод главных компонент [Электрон. ресурс]. – Режим доступа: http://www.machinelearning.ru/wiki/index.php?title=%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%B3%D0%BB%D0%B0%D0%B2%D0%BD%D1%8B%D1%85_%D0%BA%D0%BE%D0%BC%D0%BF%D0%BE%D0%BD%D0%B5%D0%BD%D1%82.
23. Google Brain gave YouTube new life [Electronic resource]. – Access mode: <https://www.theverge.com/2017/8/30/16222850/youtube-google-brain-algorithm-video-recommendation-personalized-feed>.
24. MSQRD [Электрон. ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/MSQRD>.
25. AutoDraw [Electronic resource]. – Access mode: <https://experiments.withgoogle.com/autodraw>.
26. Shazam [Электрон. ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/Shazam>.
27. Deepfake: Everything You Need to Know About What It Is & How It Works [Electronic resource]. – Access mode: <https://recfaces.com/articles/what-is-deepfake>.
28. Где применяют распознавание лиц? [Электрон. ресурс]. – Режим доступа: <https://trends.rbc.ru/trends/industry/6050ac809a794712e5ef39b7>.
29. Facial recognition: top 7 trends [Electronic resource]. – Access mode: <https://www.thalesgroup.com/en/markets/digital-identity-and-security/government/biometrics/facial-recognition>.

30. Чем опасно развитие технологий по распознаванию лиц [Электрон. ресурс]. – Режим доступа:

https://webznam.ru/blog/tekhnologij_po_raspoznavaniju_lic/2018-08-27-633

31. Facial Recognition Is the Perfect Tool for Oppression [Electronic resource]. – Access mode: <https://medium.com/s/story/facial-recognition-is-the-perfect-tool-for-oppression-bc2a08f0fe66>.

32. Amazons Facial Recognition Technology is Supercharging Local Police [Electronic resource]. – Access mode: <https://www.washingtonpost.com/technology/2019/04/30/amazons-facial-recognition-technology-is-supercharging-local-police/>.

33. Бум нейросетей [Электрон. ресурс]. – Режим доступа: <https://old.sk.ru/news/b/press/archive/2016/07/13/bum-neyrosetey-kto-delaet-neyronnye-seti-zachem-oni-nuzhny-i-skolko-deneg-mogut-prinosit.aspx#:~:text=%D0%>.

34. Лучший язык для программирования ИИ! [Электрон. ресурс]. – Режим доступа: <https://www.zfort.com.ua/blog/luchshii-yazyk-dlya-programmirovaniya-ii-rukovodstvo-polzovatelya>.

35. Что выбрать? [Электрон. ресурс]. – Режим доступа: <https://itvdn.com/ru/blog/article/java-vs-python>.

36. PyCharm [Электрон. ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/PyCharm>.

37. Лучшие IDE и редакторы кода для Python [Электрон. ресурс]. – Режим доступа: <https://tproger.ru/translations/python-ide/>.

38. Функции активации нейросетей [Электрон. ресурс]. – Режим доступа: <https://neurohive.io/ru/osnovy-data-science/activation-functions/>.

39. Нейросеть [Электрон. ресурс]. – Режим доступа: <https://basegroup.ru/deductor/function/algorithm/neuronet>.

40. Искусственный нейрон [Электрон. ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/%D0%B2%D0%BB%D1%8F%D1%8E%D1%82%20%D0%B8%D0%B7%20%D1%81%D0%B5%D0%B1%D1%8F%20%D0%B2>

D1%8B%D1%85%D0%BE%D0%B4%D1%8B%20%D1%81%D0%B5%D1%82%D0%B8.

41. Метод Виолы-Джонса [Электрон. ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%92%D0%B8%D0%BE%D0%BB%D1%8B_%E2%80%94%D0%94%D0%B6%D0%BE%D0%BD%D1%81%D0%B0.

42. Признаки Хаара [Электрон. ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%B7%D0%BD%D0%B0%D0%BA%D0%B8_%D0%A5%D0%B0%D0%B0%D1%80%D0%B0.

43. Viola Jones Algorithm and Haar Cascade Classifier [Electronic resource]. – Access mode: <https://towardsdatascience.com/viola-jones-algorithm-and-haar-cascade-classifier-ee3bfb19f7d8>.

44. MVC what is it? [Electronic resource]. – Access mode: https://www.tutorialspoint.com/mvc_framework/mvc_framework_introduction.html.

ДОДАТОК А
Текст програми

A1 Текст face.py

```

#Import OpenCV
import cv2.cv2 as cv2
face_cascade = cv2.CascadeClassifier ('haarcascade_frontalface_default.xml')

#Image as object
img = cv2.imread ('exampl.png')

#GrayScreen regime
img_gray = cv2.cvtColor (img, cv2.COLOR_BGR2GRAY)
faces = face_cascade.detectMultiScale (img_gray, 1.3, 5)

#Recognition faces
for (x, y, w, h) in faces:
    cv2.rectangle (img, (x, y), (x + w, y + h), (128, 128, 128), 2)

#Show image in window with frame
cv2.imshow ('Result', img)
cv2.waitKey (0)
cv2.destroyAllWindows ()

print (img.shape)

```

A2 Текст cam.py

```

#Impotn OpenCV
import cv2.cv2 as cv2
faceCascade = cv2.CascadeClassifier ('haarcascade_frontalface_default.xml')

#Initial web-cam in win7
video_capture = cv2.VideoCapture (0, cv2.CAP_DSHOW)

```

```

# Capture frame-by-frame, work with gray regime
while True:

    ret, frame = video_capture.read ()

    gray = cv2.cvtColor (frame, cv2.COLOR_BGR2GRAY)

    faces = faceCascade.detectMultiScale (gray, 1.1, 5, (30, 30),
cv2.CASCADE_SCALE_IMAGE)

    # Draw a rectangle around the faces
    for (x, y, w, h) in faces:

        cv2.rectangle (frame, (x, y), (x + w, y + h), (0, 255, 0), 2)

    # Display the resulting frame
    cv2.imshow ('Video', frame)

    if cv2.waitKey (10) & 0xFF == ord ('q'):
        break

# When everything is done, release the capture
video_capture.release ()
cv2.destroyAllWindows ()

```

A3 Текст haarcascade_frontalface_default.xml

```

<? xml version = "1.0"?>
<opencv_storage>
<cascade type_id = "opencv-cascade-classifier"><stageType> BOOST
</stageType>

<featureType> HAAR </featureType>

<height> 24 </height>

<width> 24 </width>

<stageParams>

```

```

<maxWeakCount> 211 </maxWeakCount></stageParams>

<featureParams>

<maxCatCount> 0 </maxCatCount></featureParams>

<stageNum> 25 </stageNum>

<stages>

<_>

<maxWeakCount> 9 </maxWeakCount>

<stageThreshold> -5.0425500869750977e + 00 </stageThreshold>

<weakClassifiers>

<_>

<internalNodes>

    0 -1 0 -3.1511999666690826e-02 </internalNodes>

<leafValues>

    2.0875380039215088e    +    00    -2.2172100543975830e    +    00
</leafValues></_>

<_>

<internalNodes>

    0 -1 1 1.2396000325679779e-02 </internalNodes>

<leafValues>

    -1.8633940219879150e    +    00    1.3272049427032471e    +    00
</leafValues></_>

<_>

<internalNodes>

    0 -1 2 2.1927999332547188e-02 </internalNodes>

<leafValues>

    -1.5105249881744385e    +    00    1.0625729560852051e    +    00
</leafValues></_>

<_>

<internalNodes>

```

```

    0 -1 3 5.7529998011887074e-03 </internalNodes>
<leafValues>
    -8.7463897466659546e-01 1.1760339736938477e + 00 </leafValues></_>
<_>
<internalNodes>
    0 -1 4 1.5014000236988068e-02 </internalNodes>
<leafValues>
    -7.7945697307586670e-01 1.2608419656753540e + 00 </leafValues></_>
<_>
<internalNodes>
    0 -1 5 9.9371001124382019e-02 </internalNodes>
<leafValues>
    5.5751299858093262e-01 -1.8743000030517578e + 00 </leafValues></_>
<_>
<internalNodes>
    0 -1 6 2.7340000960975885e-03 </internalNodes>
<leafValues>
    -1.6911929845809937e + 00 4.4009700417518616e-01 </leafValues></_>
<_>
<internalNodes>
    0 -1 7 -1.8859000876545906e-02 </internalNodes>
<leafValues>
    -1.4769539833068848e + 00 4.4350099563598633e-01 </leafValues></_>
<_>
<internalNodes>
    0 -1 8 5.9739998541772366e-03 </internalNodes>
<leafValues>

```

```

-8.5909199714660645e-01      8.5255599021911621e-01
</leafValues></_></weakClassifiers></_>

<_>

<maxWeakCount> 16 </maxWeakCount>

<stageThreshold> -4.9842400550842285e + 00 </stageThreshold>

<weakClassifiers>

<_>

<internalNodes>

  0 -1 2908 5.2420001477003098e-02 </internalNodes>

<leafValues>

  1.1297199875116348e-01 -1.0833640098571777e + 00 </leafValues></_>

<_>

<internalNodes>

  0 -1 2909 -1.9160000607371330e-02 </internalNodes>

<leafValues>

  -7.9880100488662720e-01 -3.4079000353813171e-02 </leafValues></_>

<_>

<internalNodes>

  0 -1 2910 -3.7730000913143158e-03 </internalNodes>

<leafValues>

  -1.9124099612236023e-01 2.1535199880599976e-01 </leafValues></_>

<_>

<internalNodes>

  0 -1 2911 7.5762003660202026e-02 </internalNodes>

<leafValues>

  -1.3421699404716492e-01 1.6807060241699219e + 00 </leafValues></_>

<_>

<internalNodes>

```

0 -1 2912 -2.2173000499606133e-02 </internalNodes>

<leafValues>

4.8600998520851135e-01

3.6160000599920750e-03

</leafValues></_></weakClassifiers></_></stages>

<features>

<_>

<rects>

<_>

6 4 12 9 -1. </_>

<_>

7 12 3 3. </_></rects></_>

<_>

<rects>

<_>

17 16 6 -1. </_>

<_>

3 19 16 2 3. </_></rects></_>

<_>

<rects>

<_>

13 6 6 9 -1. </_>

<_>

13 9 6 3 3. </_></rects></_>

<_>

<rects>

<_>

6 14 6 -1. </_>

<_>

6 7 3 2.</_>

<_>

12 9 7 3 2.</_></rects></_>

<_>

<rects>

<_>

13 5 8 10 -1.</_>

<_>

17 5 4 5 2.</_>

<_>

13 10 4 5 2.</_></rects></_>

<_>

<rects>

<_>

2 2 20 3 -1.</_>

<_>

2 3 20 1 3.</_></rects></_>

<_>

<rects>

<_>

9 2 9 6 -1.</_>

<_>

12 2 3 6 3.</_></rects></_>

<_>

<rects>

<_>

6 6 9 -1.</_>

<_>

10 6 2 9 3.</_></rects></_>

<_>

<rects>

<_>

12 3 4 11 -1.</_>

<_>

12 3 2 11 2.</_></rects></_>

<_>

<rects>

<_>

3 4 11 -1.</_>

<_>

3 2 11 2.</_></rects></_>

<_>

<rects>

<_>

3 8 10 -1.</_>

<_>

12 3 4 5 2.</_>

<_>

8 4 5 2.</_></rects></_>

<_>

<rects>

<_>

1 2 18 -1.</_>

<_>

1 1 18 2.</_></rects></_>

<_>

<rects>

<_>

2 9 6 -1.</_>

<_>

2 3 6 3.</_></rects></_>

<_>

<rects>

<_>

0 2 19 3 -1.</_>

<_>

0 3 19 1 3.</_></rects></_>

<_>

<rects>

<_>

14 9 6 -1.</_>

<_>

9 16 9 2 3.</_></rects></_>

<_>

<rects>

<_>

1 8 18 5 -1.</_>

<_>

7 8 6 5 3.</_></rects></_>

<_>

<rects>

<_>

0 6 9 -1.</_>

<_>

0 2 9 3.</_></rects></_>

<_>

<rects>

<_>

0 6 9 -1.</_>

<_>

8 0 2 9 3.</_></rects></_>

<_>

<rects>

<_>

13 6 4 15 -1.</_>

<_>

13 11 4 5 3.</_></rects></_>

<_>

<rects>

<_>

1 5 18 3 -1.</_>

<_>

1 6 18 1 3.</_></rects></_>

<_>

<rects>

<_>

9 7 14 6 -1.</_>

<_>

9 9 14 2 3.</_></rects></_>

<_>

<rects>

<_>

2 16 18 3 -1.</_>

<_>

2 17 18 1 3.</_></rects></_>

<_>

<rects>

<_>

17 9 6 -1.</_>

<_>

19 9 2 3.</_></rects></_>

<_>

<rects>

<_>

0 8 12 6 -1.</_>

<_>

0 8 6 3 2.</_>

<_>

11 6 3 2.</_></rects></_>

<_>

<rects>

<_>

9 13 7 8 -1.</_>

<_>

9 17 7 4 2.</_></rects></_>

<_>

<rects>

<_>

2 17 20 3 -1.</_>

<_>

2 18 20 1 3.</_></rects></_>

<_>

<rects>

<_>

15 17 9 6 -1.</_>

<_>

15 19 9 2 3.</_></rects></_>

<_>

<rects>

<_>

0 15 4 -1.</_>

<_>

2 15 2 2.</_></rects></_>

<_>

<rects>

<_>

2 6 6 -1.</_>

<_>

5 6 3 2.</_></rects></_>

<_>

<rects>

<_>

0 3 6 9 -1.</_>

<_>

0 6 6 3 3.</_></rects></_>

<_>

<rects>

<_>

15 17 9 6 -1.</_>

<_>

15 19 9 2 3.</_></rects></_>

<_>

<rects>

<_>

0 17 9 6 -1.</_>

<_>

0 19 9 2 3.</_></rects></_>

<_>

<rects>

<_>

9 18 12 6 -1.</_>

<_>

15 18 6 3 2.</_>

<_>

9 21 6 3 2.</_></rects></_>

<_>

<rects>

<_>

3 15 6 9 -1.</_>

<_>

3 18 6 3 3.</_></rects></_>

<_>

<rects>

<_>

16 13 8 10 -1.</_>

<_>

13 4 5 2.</_>

<_>

16 18 4 5 2.</_></rects></_>

<_>

<rects>

<_>

0 14 24 4 -1.</_>

<_>

14 8 4 3.</_></rects></_>

<_>

<rects>

<_>

13 11 9 4 -1. </_>

<_>

13 13 9 2 2. </_></rects></_>

<_>

<rects>

<_>

3 10 17 9 -1. </_>

<_>

3 13 17 3 3. </_></rects></_>

<_>

<rects>

<_>

13 8 8 15 -1. </_>

<_>

13 13 8 5 3. </_></rects></_>

<_>

<rects>

<_>

3 8 8 15 -1. </_>

<_>

3 13 8 5 3. </_></rects></_>

<_>

<rects>

<_>

11 14 10 8 -1. </_>

<_>

16 14 5 4 2. </_>

<_>

11 18 5 4 2. </_></rects></_>

<_>

<rects>

<_>

0 18 22 6 -1. </_>

<_>

0 18 11 3 2. </_>

<_>

11 21 11 3 2. </_></rects></_>

<_>

<rects>

<_>

0 16 24 4 -1. </_>

<_>

0 16 12 4 2. </_></rects></_>

<_>

<rects>

<_>

6 20 12 3 -1. </_>

<_>

12 20 6 3 2. </_></rects></_>

<_>

<rects>

<_>

18 12 6 12 -1. </_>

<_>

21 12 3 6 2. </_>

<_>

18 18 3 6 2. </_></rects></_>

<_>

<rects>

<_>

0 12 6 12 -1. </_>

<_>

0 12 3 6 2. </_>

<_>

3 18 3 6 2. </_></rects></_>

<_>

<rects>

<_>

15 17 9 6 -1. </_>

<_>

15 19 9 2 3. </_></rects></_>

<_>

<rects>

<_>

1 6 22 10 -1. </_>

<_>

1 6 11 5 2. </_>

<_>

12 11 11 5 2. </_></rects></_>

<_>

<rects>

<_>

15 17 9 6 -1. </_>

<_>

15 19 9 2 3. </_></rects></_>

<_>

<rects>

<_>

0 18 18 2 -1. </_>

<_>

0 19 18 1 2. </_></rects></_>

<_>

<rects>

<_>

3 15 19 3 -1. </_>

<_>

3 16 19 1 3. </_></rects></_>

<_>

<rects>

<_>

0 13 18 3 -1. </_>

<_>

0 14 18 1 3. </_></rects></_>

<_>

<rects>

<_>

15 17 9 6 -1. </_>

<_>

15 19 9 2 3. </_></rects></_>

<_>

<rects>

<_>

0 17 9 6 -1. </_>

<_>

0 19 9 2 3. </_></rects></_>

<_>

<rects>

<_>

12 17 9 6 -1. </_>

<_>

12 19 9 2 3. </_></rects></_>

<_>

<rects>

<_>

3 17 9 6 -1. </_>

<_>

3 19 9 2 3. </_></rects></_>

<_>

```

<rects>
<_>
    16 2 3 20 -1. </_>
<_>
    17 2 1 20 3. </_></rects></_>
<_>
<rects>
<_>
    0 13 24 8 -1. </_>
<_>
    0 17 24 4 2. </_></rects></_>
<_>
<rects>
<_>
    9 1 6 22 -1. </_>
<_>
    12 1 3 11 2. </_>
<_>
    9 12 3 11 2. </_></rects></_></features></cascade>
</opencv_storage>

```

A4 Текст untitled.ui

```

<? xml version = "1.0" encoding = "UTF-8"?>
<ui version = "4.0">
<class> Form </class>
<widget class = "QWidget" name = "Form">
<property name = "geometry">
<rect>

```

```

<x> 0 </x>
<y> 0 </y>
<width> 779 </width>
<height> 585 </height>
</rect>
</property>
<property name = "windowTitle">
<string> Form </string>
</property>
<widget class = "QPushButton" name = "pbi">
<property name = "geometry">
<rect>
<x> 320 </x>
<y> 530 </y>
<width> 111 </width>
<height> 41 </height>
</rect>
</property>
<property name = "styleSheet">
<string notr = "true"> QPushButton
{
background-color: # DBD7D2;
width: 75px;
height: 50px;
font-size: 10px;
font-weight: lighter ;;
border: 2px;
text-align: center;

```

```
}
```

```
QPushButton: hover
```

```
{
```

```
background-color: silver;
```

```
}
```

```
QPushButton: pressed
```

```
{
```

```
background-color: gray;
```

```
} </string>
```

```
</property>
```

```
<property name = "text">
```

```
<string> Choose Image </string>
```

```
</property>
```

```
</widget>
```

```
</widget>
```

```
<resources />
```

```
<connections />
```

```
</ui>
```

ДОДАТОК Б
Слайди презентації

Дипломна кваліфікаційна робота

На тему “Дослідження та програмна реалізація застосунку
розпізнавання осіб на зображеннях”

Виконав студент групи
Кнт-220м

Ніколаєвський Д. О.

Керівник

Корнієнко С. К.

Рисунок Б1 – Слайд 1

Метою дипломної роботи є проведення дослідження на тему розпізнавання осіб на зображеннях та програмна реалізація застосунку.

Предмет дослідження – технології неймереж для обробки зображень.

Головна задача - провести аналіз технології розпізнавання осіб, та на основі висновків створити застосунок, який допоможе у різних галузях життя суспільства посилити заходи безпеки.

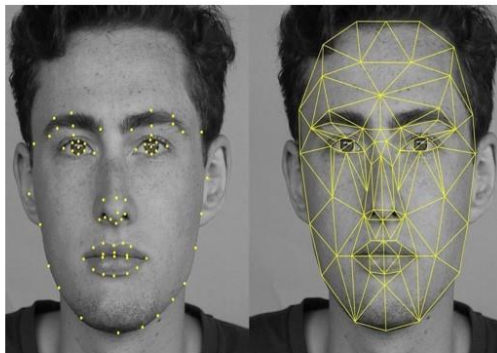


Рисунок Б2 – Слайд 2

Нейромережі.

Опис.

- Нейромережі - це обчислювальні системи, натхнені біологічними нейронними мережами, що складають мозок живих істот. Такі системи навчаються задач (поступально покращують свою продуктивність на них), розглядаючи приклади моделей. Наприклад, у розпізнаванні зображень вони можуть навчатися ідентифікувати зображення, які містять котів, аналізуючи приклади зображень, мічені як «кіт» і «не кіт», і використовуючи результати для ідентифікування котів в інших зображеннях. Вони роблять це без жодного апріорного знання про котів, наприклад, що вони мають хутро, хвости, вуса та котоподібні писки. Натомість, вони розвивають свій власний набір доречних характеристик з навчального матеріалу, який вони оброблюють.

Рисунок Б3 – Слайд 3

Нейромережі.

Опис.

- Вони ґрунтуються на сукупності з'єднаних вузлів, що називають штучними нейронами (аналогічно до біологічних нейронів у головному мозку). Кожне з'єднання між штучними нейронами може передавати сигнал від одного до іншого. Штучний нейрон, що отримує сигнал, може обробляти його, а потім сигналізувати іншим нейронам, приєднаним до нього.
- В поширених реалізаціях, сигнал на з'єднанні між штучними нейронами є дійсним числом, а вихід кожного штучного нейрону обчислюється нелінійною функцією суми його входів. Штучні нейрони та з'єднання зазвичай мають вагу, яка підлаштовується в перебігу навчання. Вага збільшує або зменшує силу сигналу на з'єднанні. Нейрони можуть мати такий поріг, що сигнал надсилається лише якщо сукупний сигнал перетинає цей поріг. Ці нейрони зазвичай організовані в шари. Різні шари можуть виконувати різні види перетворень своїх входів. Сигнали проходять від вхідного до виходового шару.
- Первинною метою НМ було розв'язання задач таким же способом, як це робив би людський мозок. НМ використовували в ряді різноманітних задач, включно з комп'ютерним баченням, розпізнаванням мовлення, машинним перекладом, соціально-мережним фільтруванням, грою в настільні та відео ігри, та медичним діагностуванням.

Рисунок Б4 – Слайд 4

Нейрон

Кожен нейрон має входи з вагами – синапсами, функцію активації, що визначає вихідну інформацію при заданій вхідній, та один вихід. Синапси – регульовані параметри, що конвертують нейронну мережу в параметризовану систему

Зважена сума входів – активаційний сигнал – проходить через функцію активації виведення даних з нейрона. Є кілька видів функції активації:

- лінійна: $f(x) = ax$;
- ступінчаста: $f(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$;
- сигмоїдна: $f(x) = \frac{1}{1+e^{-x}}$;
- тангенціальна: $f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$;
- випрямна: $f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases}$.

За допомогою додаткових прихованих шарів можна зробити систему більш гнучкою і потужною. Нейромережа із багатьма прихованими шарами називаються глибокими нейронними мережами, вони створюють складні нелінійні зв'язки.

Рисунок Б5 – Слайд 5

Розпізнавання осіб

Дані системи використовують повсюдно у багатьох галузях. В банківській, освітній сферах, у медицині, у службах безпеки.

Зображення розбивається на маленькі ділянки, до кількох пікселів, кожен із яких буде вхідним нейроном. За допомогою синапсів сигнали передаються від одного шару до іншого. Під час цього процесу сотні тисяч нейронів із мільйонами параметрів порівнюють отримані сигнали з уже обробленими даними.

Як на практиці працюють дані системи? На практиці це означає, що до певної межі чим більше прихованих шарів нейронної мережі існує, тим точніше буде розпізнане зображення



Рисунок Б6 – Слайд 6

Методи розпізнавання

Метод	Швидкість	Точність
Метод гнучкого порівняння на графах	Процес обробки виконується послідовно всіма вершинами до того часу, поки не буде досягнута найменша сумарна різниця між ознаками деформованого і еталонного графів, що виразно б'є по швидкості роботи даного метода.	Всі переваги методу руйнуються із-за високої обчислювальної складності процедури і низької технологічності при запам'ятовуванні нових еталонів.
Приховані Марківські моделі	ПММ використовує статистичні властивості сигналів та враховують безпосередньо їх просторові характеристики. Так що безперечно, ні про яку високу швидкість обробки даних мова не йдеться.	ПММ не має здатності розрізняти осіб, тобто алгоритм навчання тільки максимізує відгук кожного зображення на свою модель, але не мінімізує відгук на інші моделі.

Рисунок Б7 – Слайд 7

Методи розпізнавання

Метод	Швидкість	Точність
Метод основних компонентів	Весь навчальний набір осіб перетворюється на одну загальну матрицю даних, розкладену в рядок одного розміру. Для отриманої матриці вирішується завдання визначення власних значень та відповідних власних векторів. Завдяки цьому, швидкість обробки виростає в декілька разів.	У випадках, коли на зображенні особи присутні значні зміни у освітленості або виразі обличчя, ефективність методу значно падає.
Метод Віолі-Джонса із ознаками Хаара	Швидкість роботи може сильно змінюватись в залежності від складності об'єкта та розміру об'єкта.	Він є одним із найкращих за співвідношенням показників ефективності розпізнавання/швидкості роботи. Також цей детектор має вкрай низьку ймовірність помилкового виявлення обличчя.

Рисунок Б8 – Слайд 8

Порівняння методів розпізнавання

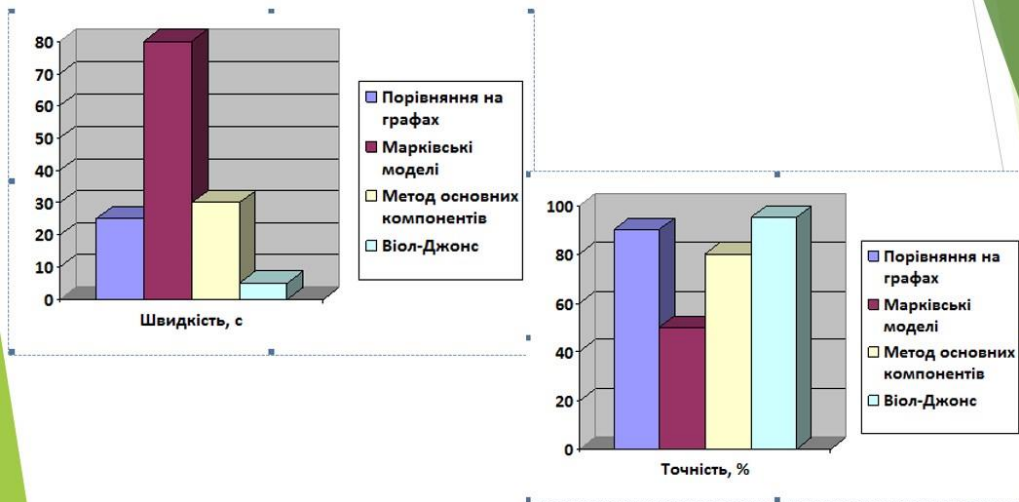


Рисунок Б9 – Слайд 9

Обґрунтування мови

Критерії	Python	Java
Простота вивчення	Широко відомий своєю лаконічністю та простотою у вивченні. Мова загального призначення, яка відрізняється від конкурента потужністю, ефективністю та високим рівнем читабельності коду.	Швидка, безпечна та надійна мова програмування загального призначення. Серед недоліків треба зазначити громіздкість коду, що може привести до криптичних багів.
Продуктивність	Зазвичай компілює код під час виконання програми. Проте, якщо використати правильний інтерпретатор, то стає можливим виконувати своєчасну компіляцію всіх програм або їх частин у нативний код.	Має змогу виконувати компіляцію заздалегідь.
Застосування	Вважається більш універсальним, що дозволяє використовувати у: розробці веб додатків з використанням фреймворків, у завданнях системного адміністрування, у тестуванні, у роботі з великими обсягами даних, у машинному навчанні, а також розробка нейронних мереж.	Чудово відчувається в мобільній розробці під Android і iOS

Рисунок Б10 – Слайд 10

Алгоритм роботи та архітектура MVC

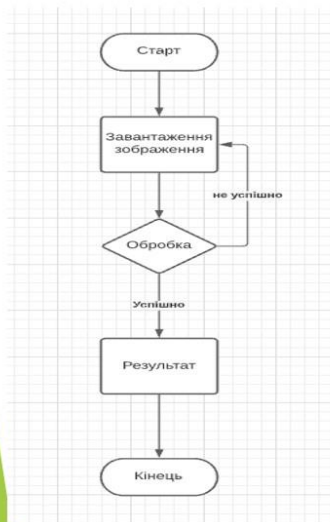


Рисунок Б11 – Слайд 11

Функціонал застосунку

Десктопний застосунок реалізує такий функціонал:

- взаємодія з файлами зображень у комп'ютері або у зовнішньому флеш-накопичувачі;
- визначення контурів обличчя на зображенні;
- визначення контурів обличчя через відео потік з веб-камери;
- показ результату.



Для пошуку осіб спочатку зображення перетворюється на об'єкт, з яким буде можливо працювати. Після цього зображення треба перевести в режим відтінків сірого, однак інформація про сам колір алгоритму не важлива, тільки його інтенсивність. Тому що в районі щік, очей і рота вона темніша, та нейромережа здатна завдяки цьому розпізнавати обличчя.

Рисунок Б12 – Слайд 12

Головні параметри

Параметр	Значення
image	У цей параметр треба буде передавати 8-бітну матрицю зображення.
scaleFactor	Параметр показує, у скільки разів ми зменшуватимемо вихідне зображення, намагаючись виявити об'єкт: це робиться тому, що ми спочатку не знаємо, якого розміру буде обличчя. Чим меншим буде цей коефіцієнт, тим довше працюватиме алгоритм.
minNeighbors	Цей параметр впливає на якість виявлення: чим більше його значення, тим менше образів зможе виявити алгоритм, але водночас його робота буде точніше.

Рисунок Б13 – Слайд 13

Побудова рамок на обличчї

- Якщо обличчя будуть виявлені, функція поверне набір об'єктів типу Rect (x, y, w, h) - прямокутників, початок яких задано парою координат (x, y), а ширина та висота - як w і h. У циклі додамо ці прямокутники у вихідне зображення image за координатами їх протилежних вершин (x, y) і (x+w, y+h). Таким чином, на обличчї осіб буде побудован прямокутник, визначаючи осіб на зображенні.

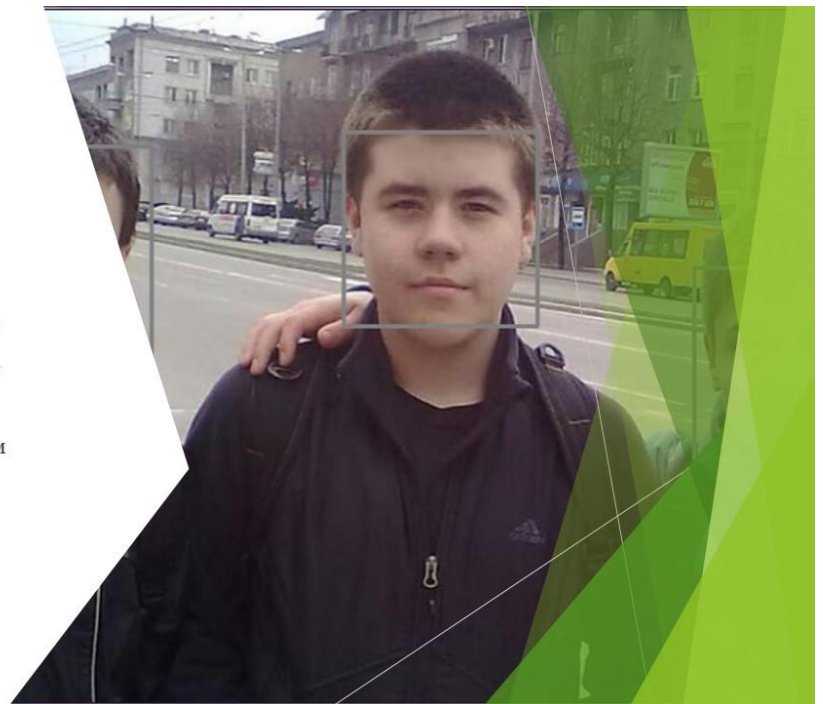


Рисунок Б14 – Слайд 14

Завантаження зображення

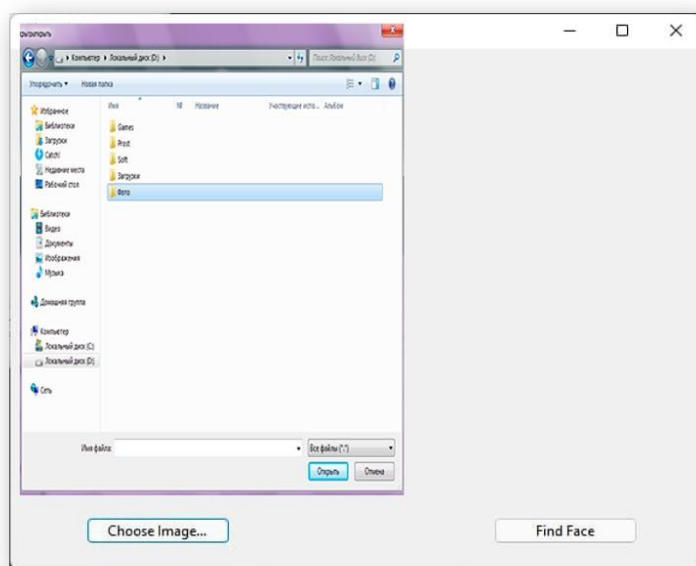


Рисунок Б15 – Слайд 15

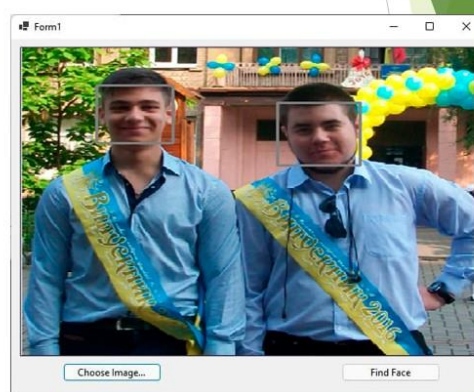
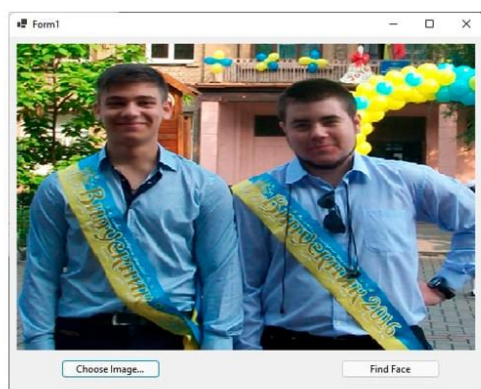


Рисунок Б16 – Слайд 16

Висновки:

При виконанні випускної кваліфікаційної роботи було проаналізовано предметну область системи, що розроблюється. Аналіз отриманих даних дав можливість визначити головні вимоги до програмного продукту та сформулювати основні його функції. За допомогою системи потенційний користувач має змогу визначити контури обличчя на зображеннях та відео потоках.

Задачі, що були виконані в роботі:

- ✓ був виконаний аналіз предметної області, вивчені аналогічні існуючі застосунки, підходи і програмні засоби для рішення поставленої мети;
- ✓ вивчені особливості обраної платформи, мови програмування, середовища розробки;
- ✓ розроблена програмна система може успішно використовуватися на деяких платформах.

Можна вважати, що програма відповідає усім стандартам якості та заявленим вимогам. Слід зауважити, що програма буде знаходитись відтепер на стадії підтримки готового продукту.

Рисунок Б17 – Слайд 17

Дякую за увагу!

Рисунок Б18 – Слайд 18