Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра обчислювальної техніки

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему «СИСТЕМА РОЗПІЗНАВАННЯ ОБЛИЧЧЯ НА ФОТОГРАФІЇ З ВИКОРИСТАННЯМ НЕЙРОННОЇ МЕРЕЖІ»

Виконав: студент 2 курсу, групи 1КІ-22мз
спеціальності 123 Комп'ютерна інженерія
_____ Чміхаленко В.Є.
Керівник к.т.н., доц., доц. каф. ОТ
_____ Муращенко О.Г
Опонент Голова секції УБ
д.т.н., проф, каф. МБІС
_____ Яремчук Ю.Є.

Допущено до захисту
Завідувач кафедри ОТ
д.т.н., проф. Азаров О.Д.
«19» 06 2024 р.

Вінниця 2024

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра обчислювальної техніки
Освітньо-кваліфікаційний рівень магістр
Спеціальність 123 Комп'ютерна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедри
обчислювальної техніки
проф., д.т.н. О. Д. Азаров
_____
06" ____02___2024 р.

З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Чміхаленку Володимиру Євгенійовичу

1 Тема роботи «Система розпізнавання обличчя на фотографії з використанням нейронної мережі», керівник роботи Муращенко Олександр Геннадійович,к.т.н., доцент,затверджені наказом вищого навчального закладу від 11.03.24 року № 81

2 Строк подання студентом роботи 12.06.24р..

3 Вихідні дані до роботи: розпізнавання проводиться з веб-камери, навчання проводиться з графічних файлів типу jpg розмірністю 250x250 пікселів.

4 Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити): вступ, постановка задачі, аналіз предметної області розпізнавання облич на фотографії, обґрунтування вибору типу нейронної мережі для розв'язання задачі детектування облич, проектування основного компонента програми для розпізнавання облич, програмна реалізація модуля для розпізнавання облич, аналіз результатів тестування програми для

розпізнавання облич, висновки, перелік використаних джерел, додатки

5 Перелікграфічного матеріалу (з точним зазначенням обов'язкових креслень):схема алгоритму роботи програми, структура згорткової нейронної мережі, результати роботи програми.

6 Консультанти розділів роботи представленов табл.1.

Таблиця 1— Консультантирозділівроботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1,2,3,4	Муращенко О.Г., к.т.н., доц. каф. ОТ		
5	Небава М.І., к.е.н., проф. каф. ЕП і ВМ		

7 Дата видачі завдання 12.03.2024р.

8 Календарний план наведенов табл.2.

Таблиця 2—Календарний план

№ з/п	Назва етапів виконання магістерської роботи	Строк виконання етапів роботи	Прим.
1	Постановка мети та задач роботи	27.02.24	вик
2	Аналіз предметної області	06.03–20.03.24	вик
3	Обгрунтування методу розв'язання задачі	27.03–7.04.24	вик
4	Проектування інтелектуального модуля розпізнавання облич на фотографії	10.04–21.04.24	вик
5	Програмна реалізація модуля розпізнавання облич на фотографії	24.04–12.05.24	вик
6	Аналіз результатів тестування модуля розпізнавання облич на фотографі	15.05–26.05.24	вик
7	Розрахунок економічної частини роботи	29.05–09.06.24	вик
8	Оформлення пояснювальної записки та ілюстративного матеріалу	06.06.24	вик
9	Аналіз виконання роботи, висновки, додатки	07.06.24	вик
10	Перевірка якості виконання магістерської роботи та усунення недоліків	8.06.24	вик

Студент _____ Чміхаленко В.Є.

Керівник роботи _____ Муращенко О.Г.

АНОТАЦІЯ

УДК 004.9

Чміхаленко В.Є.

Система розпізнавання обличчя на фотографії з використанням нейронної мережі. Магістерська кваліфікаційна робота зі спеціальності 123 — комп'ютерна інженерія, освітня програма — комп'ютерна інженерія. Вінниця: ВНТУ, 2024, 122 с.

На укр.мові. Бібліогр.: 39 назв, рис. 45, табл. 7.

Магістерська кваліфікаційна робота присвячена розв'язанню задачі розпізнавання облич. Були розглянуті та проаналізовані існуючі методи розпізнавання облич і наоснові об'єктивних переваг був вибраний метод розпізнавання за допомогою нейронних мереж. Відповідно до поставленої задачі і на основі аналізу відомих структур нейронних мереж була обрана згорткова нейронна мережа, як така, що повністю виконує поставлені задачі.

Практичне значення полягає в накопиченні рекомендацій щодо розробки та навчання згорткових нейронних мереж в цілях використання в системах розпізнавання обличь на основі аналізу сучасних архітектур та принципів роботи існуючих моделей.

Ключові слова: згорткова, нейронна мережа, дослідження, розпізнавання, детекція, обличь, глибоке навчання.

ANNOTATION

Chmihalenko V.Y.

Facial recognition system in a photo using a neural network.Master's qualification route in the specialty 123 — computer engineering, educational program computer engineering. Vinnitsa, VTNU, 2022, 124 p.

In the Ukr.leng. Libr.name 39, figure 45, table 7.

The master's thesis is devoted to solving the problem of face recognition. The existing methods of face recognition were considered and analyzed, and the method of recognition using neural networks was chosen based on objective advantages. In accordance with the task and based on the analysis of known structures of neural networks, a convolutional neural network was chosen as one that fully fulfills the tasks.

The practical significance lies in the accumulation of recommendations for the development and training of convolutional neural networks for use in face recognition systems based on the analysis of modern architectures and the principles of operation of existing models.

Keywords: convolution, neural network, research, recognition, detection, face, deep learning..

ЗМІСТ

ВСТУП

Сучасний етап розвитку інформаційних технологій характеризується розвитком, створенням і широким застосуванням відеоінформаційних технологій, заснованих на обробці та використанні зображень. Актуальність відеоорієнтаційної інформації на даний час часто визначається потребами розробки систем штучного інтелекту, які повинні мати можливість візуального орієнтування в просторі та бути оптимізованими для візуального аналізу сцен, візуального пошуку кадрів. Рухомі предмети. Такі можливості є важливими характеристиками для інтелектуальних систем не тільки промислового, але й звичайного побутового призначення [1] .

Комп’ютерний зір — це теорія створення машин, які можуть виявляти, спостерігати та класифікувати об’єкти в зображенні чи відеопотоці.

Використання штучного інтелекту для розпізнавання образів призвело до створення функціональних систем для розпізнавання візуальних об’єктів на основі подібних ознак. Будь-які характеристики об’єктів, які потрібно ідентифікувати, можна розглядати як ознаки. Позначення повинні бути незмінними щодо орієнтації, розміру та форми предметів. Символи символів формуються розробником системи. Якість розпізнавання багато в чому залежить від того, наскільки добре сформовані характери персонажів. Розпізнавання полягає у знаходженні апріорного вектора ознак для об’єкта, виділеного на зображенні, а потім у визначенні того, яким рівням ознак відповідає цей вектор.

Розпізнавання облич є одним із підрозділів ширшої категорії розпізнавання образів. Насправді методи та алгоритми розпізнавання на практиці досить схожі за відмінністю функції розпізнавання, а точніше, її параметрів.

Основними завданнями, в яких використовується розпізнавання обличь є:

— комп'ютерна графіка;

— криміналістика;

— охоронні системи.

Зі збільшенням обчислювальної потужності та вдосконаленням алгоритмів розпізнавання перелік завдань, які вони вирішують, зріс у геометричній прогресії: спілкування між комп’ютером і людиною; віртуальна реальність, комп'ютерні ігри; основи доступу до інформації; імміграційний контроль; персоналізувати побутову техніку; шифрування даних; електронна комерція; Суспільні послуги.

Яскравим прикладом останнього є інтеграція алгоритмів автоматичного розпізнавання облич на фото в популярні соціальні сервіси Яндекс Фото і Google Picasa. [2].

Мета дослідження — пошук та реалізація підходів удосконалення системи розпізнавання облич, які дозволяють розпізнавати об’єкти з підвищеною точністю розпізнавання без зниження швидкості.

Об’єкт дослідження даної роботи — процес розпізнавання облич за допомогою згорткових нейронних мереж.

Предмет дослідження — архітектури та можливості згорткових нейронних мереж для вирішення задачі розпізнавання обличь на двовимірних зображеннях.

Задачі дослідження:

— проаналізувати відомі методи розпізнавання облич та обрати напрямок досліджень;

— розробити метод розпізнавання облич на основі нейронної мережі;

— розробити структуру програмного модуля;

— розробити алгоритм роботи програмного модуля розпізнавання облич,

— розробити програмне забезпечення розпізнавання облич нейронною мережею;

— здійснити програмну реалізацію модуля розпізнавання облич;

— провести тестування програмного модуля розпізнавання облич;

— провести економічний розрахунок системи, що розробляється.

Наукова новизна отриманих результатів магістерської роботи полягає в удосконаленні методу розпізнавання обличчя на фотографії з використанням нейронної мережі.

Об'єкт дослідження — процес комп'ютеризованого розпізнавання облич.

Предмет дослідження — програмні засоби розпізнавання облич на основі нейронних мереж.

Апробацію результатів наукової роботи було проведено на науковій конференції:

«Молодь в науці: дослідження, проблеми, перспективи (МН–2024)», доповідь на тему "Система розпізнавання обличчя на фотографії з використанням нейронної мережі"

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ДЕТЕКТУВАННЯ ОБЛИЧ НА ФОТОГРАФІЇ

1.1 Постановка завдання детектування обличь

Задача розпізнавання в зображеннях людських облич була однією з перших практичних завдань, яка стимулювала розвиток теорії розпізнавання образів. Незважаючи на численні дослідження в цій галузі, які проводилися в усьому світі за останні кілька десятиліть, не було розроблено жодних методів надійного виявлення та розпізнавання людських облич за будь-яких обставин.

Завдання розпізнавання обличчя складається з двох підзадач: автоматичний вибір обличчя на зображенні (розпізнавання обличчя) і розпізнавання обличчя людини, якщо необхідно (розпізнавання). Виділене обличчя на зображенні можна використовувати для подальшого аналізу, який можна використовувати для таких завдань, як моделювання обличчя, відстеження та розпізнавання виразу обличчя, визначення статі та віку людини, верифікація та автентифікація.

Завдання розпізнавання осіб зводиться в першу чергу до організації інтерфейсу людина-комп'ютер, встановлення систем доступу до важливих приміщень. У цьому випадку наявність обличчя в кадрі необхідно швидко виявити, тому розробка ефективних методів і алгоритмів аналізу відеопотоку є дуже актуальною і актуальною задачею.

Задачу розпізнавання обличчя, яка буде вирішена в ході виконання бакалаврської дипломної роботи можна сформулювати наступним чином. При аналізі відеопотоку з веб-камери слід видати одну з наступних відповідей:

— у відеопотоці немає облич;

— у відеопотоці є нерозпізнане обличчя;

— у відеопотоці є розпізнане обличчя;

В ході виконання бакалаврської дипломноїроботи буде розроблена програма, що буде в реальному часі виявляти та розпізнавати обличчя з

відеопотоку, який передається веб-камерою. Розпізнавання буде можливим завдяки попередньому навчанні на попередньо підготовленому тренувальному наборі зображень.

1.2 Типовий процес детектування облич

Розпізнавання облич відноситься до завдань розпізнавання образів. Такі задачі не мають точного аналітичного вирішення. Основна ідея розпізнавання обличчя людини полягає у виділенні інформативних ознак із зображення, кодуванні та порівнянні закодованого обличчя з базою даних.

У найзагальнішому випадку алгоритм вирішення задачі розпізнавання за зображенням обличчя людини складається з наступних кроків:

— виявлення присутності обличчя людини на зображенні та виділення цього обличчя;

— визначте та опишіть ключові риси обличчя (наприклад, очі, ніс, брови, рот, вуха тощо);

— представлення поверхні в деякому просторі ознак (моделювання);

— порівняння (класифікація) зі стандартами та просування;

Витяг із зображень обличчя є ключовим кроком у всіх методах, пов’язаних з аналізом людських облич. Його призначення полягає в наступному: визначити, чи присутні фасети на довільному зображенні, і якщо так, то вказати положення та межі кожної виявленої грані. [2].

1.3 Огляд відомих методів розв’язання задачі детектування облич

Основними методами розв’язання задачі детектування обличь є наведені нижче методи.

Традиційний метод передбачає, що деякі алгоритми розпізнавання обличчя визначають риси обличчя за ключовими точками або рисами на зображенні обличчя суб’єкта. Наприклад, алгоритм може визначати взаємне

розташування, розмір і/або форму очей, носа, вилиць і щелепи .. Потім ці функції використовуються для пошуку інших зображень із подібними характеристиками.

Інші алгоритми нормалізують галерею зображень обличчя, а потім стискають дані обличчя, зберігаючи на зображенні лише дані, необхідні для розпізнавання обличчя. Потім тестове зображення порівнюється з даними, що відповідають поверхні. Одна з перших успішних систем була заснована на методах зіставлення шаблонів, застосованих до набору характеристик поверхні, забезпечуючи рівномірний опис стиснення поверхні.

Алгоритми розпізнавання можна розділити на два основні підходи. Геометричний підхід розглядає конкретні властивості. Фотометричний, який є статистичним методом, який перетворює зображення на набір значень і порівнює ці значення з шаблонами для усунення відмінностей. Деякі класифікують ці алгоритми на дві великі категорії: цілісні та на основі функцій. Перші намагаються повністю ідентифікувати обличчя, а ті, що ґрунтуються на ознаках, розкладають обличчя на такі елементи, як відповідні ознаки, і аналізують кожну, а також її просторове розташування по відношенню до інших ознак [.

Популярні алгоритми розпізнавання включають аналіз головних компонент із використанням методу власних граней, лінійний дискримінантний аналіз, еластичне зіставлення] з використанням алгоритму риб’ячого обличчя, приховану марковську модель, полілінійне підпросторове навчання, тензорне представлення та динамічне зіставлення за допомогою нейронів.

Тривимірне виявлення включає виявлення поверхонь у просторі, яке використовує 3D-сенсори для отримання інформації про форму поверхні. Потім ця інформація використовується для визначення поверхневих особливостей, таких як форма повік, носа та підборіддя.

Одна з переваг розпізнавання обличчя в просторі полягає в тому, що на нього не впливають зміни освітленості, як в інших методах. Він також може

розпізнавати обличчя під різними кутами огляду, включаючи профіль. Використання тривимірних точок обличчя значно підвищує точність розпізнавання обличчя. Освоєння космосу було підштовхнуто завдяки розробці складних датчиків, які краще справляються із захопленням 3D-зображень поверхонь. Робота датчиків заснована на правильно організованій оптичній поверхні. До дюжини або більше таких датчиків зображення можна розмістити на одній мікросхемі CMOS — кожен датчик займає власну частину спектру.

Навіть ідеальна техніка зіставлення 3D поверхні може бути чутливою до наявності емоцій. Для цього дослідницька група з Technion використовувала інструменти метричної геометрії, щоб видалити вираз поверхні за допомогою ізометрії.

Новий метод полягає у впровадженні техніки захоплення тривимірного зображення за допомогою трьох камер стеження, орієнтованих під різними кутами; Одна камера лобова, друга в профіль і третя під кутом. Разом усі ці камери зможуть відстежувати та розпізнавати обличчя глядача в реальному часі.

Аналіз текстури шкіри — це ще один новий напрямок, який використовує візуальні деталі шкіри, отримані за допомогою стандартних цифрових або сканованих зображень. Цей метод, який називається аналізом текстури шкіри, перетворює чіткі лінії, візерунки та плями на шкірі людини в математичний простір.

Аналіз текстури обличчя працює так само, як і розпізнавання обличчя. Малюнок ділянки шкіри називається відбитком шкіри. Потім цю публікацію поділено на менші розділи. Використовуючи алгоритми для перекладу друку в математичний вимірюваний простір, система починає розрізняти будь-які лінії, пори та фактичну текстуру шкіри. Він може знаходити відмінності між ідентичними парами, чого неможливо зробити лише за допомогою програми розпізнавання облич.

Дослідження показали, що додатковий аналіз текстури шкіри може збільшити результати розпізнавання обличчя на 20-25%.

Метод Віоли-Джонса (Viola-Jones). В даний час метод Віоли-Джонса є найпопулярнішим методом отримання області обличчя на зображеннях завдяки високій швидкості та ефективності. Такий підхід дозволив реалізувати розпізнавання обличчя на зображеннях у реальному часі в практичних додатках, таких як, наприклад, цифрові камери та програмне забезпечення для керування фотоальбомами [3].

Алгоритм Віоли-Джонса виявляє лише повністю видимі (без розмиття), обличчя на передньому плані (без обертання голови), вертикальні (без інверсії), добре освітлені та повнорозмірні (що займають більшу частину кадру) зображення з фіксованою роздільною здатністю. .

Ці обмеження не такі суворі, як може здатися, оскільки можна нормалізувати картину для наближення факторів Віоли-Джонса, докладно наведених нижче;будь-яке зображення можливо перемасштабувати до фіксованої роздільності.

Для загального зображення з поверхнею невідомого розміру та орієнтації може бути виконано виявлення крапок для виявлення потенційних поверхонь з подальшим масштабуванням і поворотом у вертикальний простір на всю ширину;

Яскравість зображення можна відкоригувати за допомогою балансу білого, а обмежувальні рамки можна отримати, пересуваючи вікно по всьому зображенню та позначаючи вікно, що містить кожне обличчя.

Як правило, це буде багаторазово виявляти те саме обличчя, де можна застосувати методи зменшення повторення, наприклад немаксимальне придушення.

Вимога «переднього плану» не підлягає обговоренню, оскільки немає простої зміни зображення, яка може перетворити сторінку з профілю на повну. Однак можна відпрацьовувати кілька категорій Віоли-Джонса: одну для

кожного кута, одну для виду спереду, одну для виду 3/4, одну для профілю та деякі інші кути між ними. Тоді, під час виконання, можна запускати всі ці класифікатори паралельно, щоб виявляти обличчя під різними кутами огляду.

Вимога «повної видимості» також не підлягає обговоренню, і її неможливо виконати, просто навчивши більше класифікаторів Віоли-Джонса, оскільки існує забагато можливих способів розмити обличчя.

Детектор обличчя Віоли-Джонса базується на трьох основних принципах:

— інтегроване представлення зображень на основі властивостей Хаара, що дозволяє дуже швидко обчислювати необхідні властивості;

— метод побудови класифікатора на основі алгоритму адаптивного підвищення (AdaBoost);

— метод об’єднання партій у каскадний формат.

Ці ідеї дозволяють швидко шукати обличчя на зображенні в реальному часі.

Переваги:

— висока швидкість виявлення об'єктів;

— висока ймовірність точного визначення обличчя (більше 90%) для фронтальних зображень і спостереження об'єкта під невеликим кутом, приблизно до 30°;

— низька ймовірність помилкового розпізнавання обличчя.

Недоліки:

— дуже тривалий період навчання;

— при великому куті відхилення голови ймовірність розпізнавання обличчя значно зменшується;

— чутливість до освітленості.

Метод власних облич (Eigenfaces) [4] використовує аналіз головних компонент для зменшення розмірності даних без істотної втрати інформації. Простір власних облич генерується шляхом застосування методу головних компонент до навчального набору зображень. Потім навчальні зображення

проектуються на власні поверхні в просторі. Потім тестове зображення проектується в нове місце, і розраховується відстань між проектованим тестовим зображенням і зображеннями з навчального набору. Подане навчальне зображення сприймається як визнання. а метод основного тіла використовується лише для виявлення поверхні на зображенні. Для облич значення елементів у власному просторі мають більші значення та наближаються до нуля в доповненні власного простору. Цей факт можна використати, щоб визначити, чи є вхід зображенням обличчя чи ні.

Переваги:

— при дотриманні ідеальних умов точність виявлення за допомогою цього методу може сягати понад 90% значення;

— зберігання та пошук зображень у великих базах даних, реконструкція зображень.

Недоліки:

— обчислення набору власних векторів займає дуже багато часу;

— зображення в умовах близького освітлення, однорідна перспектива (визначається додаванням зображень з різних точок зору до навчальної вибірки);

— необхідно провести якісну попередню обробку, привівши зображення до стандартного стану;

— відсутність перешкод, таких як окуляри чи бороди.

Лінійний дискримінантний аналіз (дискримінант Фішера) [5] використовує таку проекцію простору зображення на властивості простору, щоб групувати зображення зі схожими гранями якомога ближче, уникаючи зображень з різними гранями одне від одного. Як і попередній, цей підхід дозволяє зменшити кількість ознак, при цьому значно покращуючи кластеризацію зображень (відокремлення один від одного). Алгоритм передбачає, що для кожної людини в базі даних є багато зображень людей в різних умовах освітлення. Цей алгоритм стійкий до змін умов освітлення. Цей

алгоритм дозволяє підвищити рівень розпізнавання до 99% навіть у дуже різних ситуаціях.

Методи на основі штучних нейронних мереж [6]. Водночас у класифікаційних завданнях відбувається неявний вибір ключових ознак у межах мережі, визначення значення ознак і послідовності взаємозалежностей між ними. Серед нейронних мереж для вирішення задач розпізнавання образів найчастіше використовують багатошарові персептрони з помилкою зворотного поширення, мережі з радіальними базисними функціями та згорточні нейронні мережі (ConvolutionalNeuralNetworks) [7]. Нейронні мережі можна успішно використовувати для виділення облич на зображенні та верифікації особи за обличчям. У той же час якість розпізнавання знижується зі збільшенням кількості оцінюваних класів.

Методи сегментації кольору шкіри людини. Локалізація обличчя також використовує методи, які використовують сегментацію кольорового зображення для виділення ділянок зображення кольорами, схожими на колір людської шкіри [8,9]. Основною метою таких методів є створення набору правил, які розрізняють «звичайні» пікселі зображення та пікселі шкіри.

Для реалізації в цій роботі ми виберемо методи на основі штучних нейронних мереж.

1.4 Програми детектування облич

При проведенні аналізу протестовано наступні функціональні блоки:

— виявлення обличчя на фото (Face Detection);

— порівняння облич (Face Comparison);

— визначення атрибутів обличчя (стать, вік, етнічна приналежність, емоції);

— пошук схожих облич — чи може система розпізнати обличчя певної людини серед групи зображень (знайти обличчя, яке найбільше схоже на

цільове).При тестуванні кожного з перерахованих вище функціональних блоків зверталася увага на такі критерії:

— точність (відсоток співпадіння);

— швидкість відпрацювання сервісу — час від початку відправки запиту до отримання результату;

— вартість розпізнавання по кожній функції;

— додатковий функціонал і можливості розпізнавання (в тому, що має відношення до розпізнавання облич).

1.4.1 Хмарний сервіс для розпізнавання облич Azure Face API

Azure Face API — це хмарна служба розпізнавання облич від Microsoft, яка є частиною Azure Cognitive Services. Azure Face API використовує розширені алгоритми розпізнавання, призначені для виявлення, перевірки, ідентифікації та аналізу облич на зображеннях і відео.

Коли служба Azure Face виявляє обличчя, воно виявляє обличчя на малюнку та повертає координати прямокутника, у якому воно розташоване. При необхідності функція розпізнавання визначає ряд ознак, пов'язаних з особами: положення голови, стать, вік, емоції, волосся на обличчі, окуляри, а також відсоток ймовірності впізнання даної людини. Як ви можете бачити нижче, система визначила, що на фотографії зображений усміхнений 36-річний чоловік, а відомою емоцією було щастя.

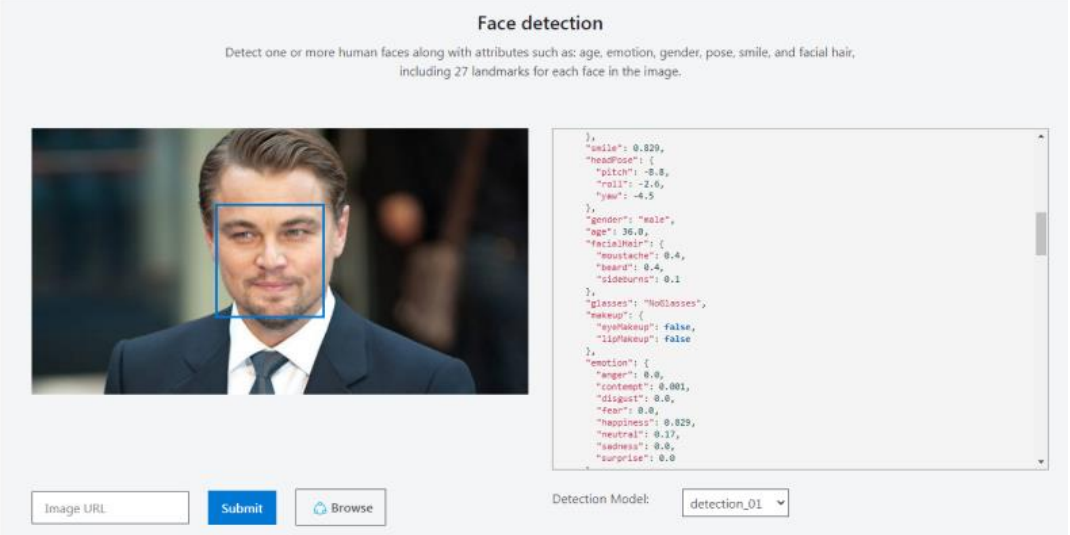


Рисунок 1.1 — Виявленні обличчя на фото

Face ID може визначати наявність і положення облич на зображенні та повертати точні рамки навколо них. Система також розпізнає ключові точки обличчя (підборіддя, очі, брови, кути рота, форму носа), а також особливості, стать, вік, інтенсивність посмішки, положення голови, розмите зображення та очі (відкриті/закриті, чи є окуляри). Весь звіт буде у форматі JSON.

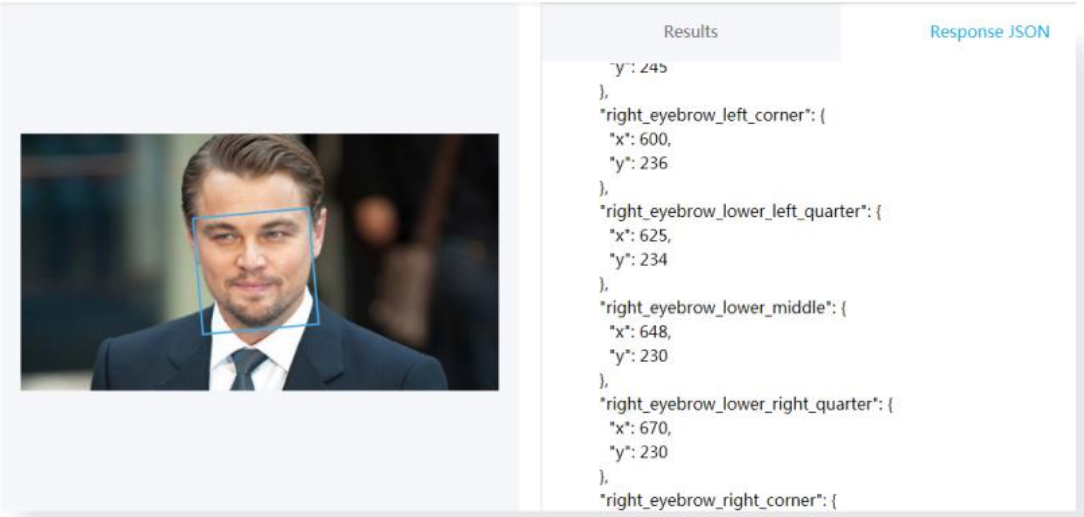


Рисунок 1.2 — Визначення координат та атрибутів

Для Face ID середня швидкість визначення наявності на фото людського обличчя для більшої вибірки фото складає 1,4 с.

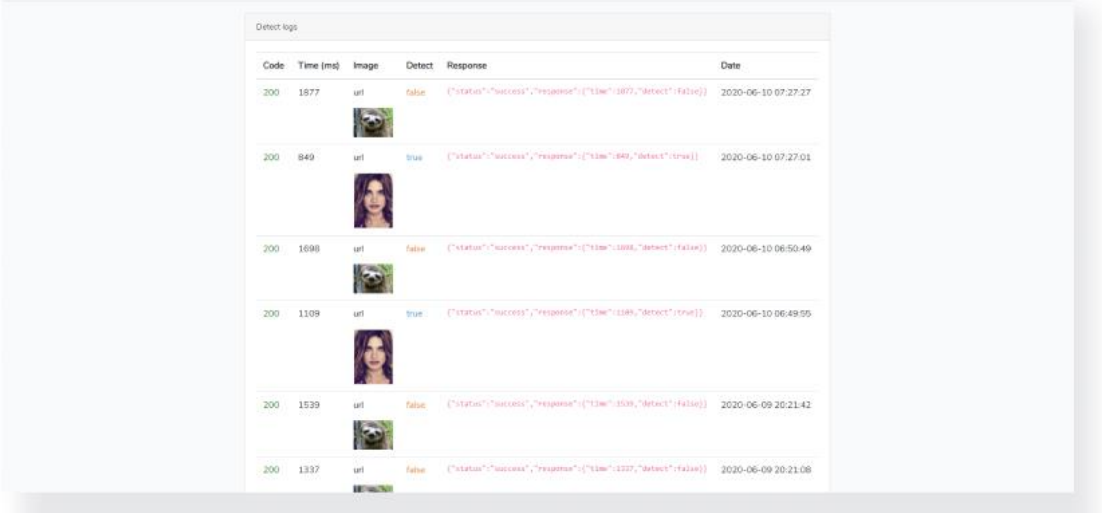


Рисунок 1.3 — Середня швидкість визначення наявності на фото людського обличчя

Під час порівняння сторінок у Azure Face використовується функція «перевірка обличчя». Насправді система оцінює, чи належать два обличчя на фотографії одній людині, і повертає прямокутник із межами навколо виявлених облич. Як ви можете бачити в прикладі нижче, Azure Face гарантує, що на фотографії є різні люди.Швидкість відповіді — приблизно 394 мс.

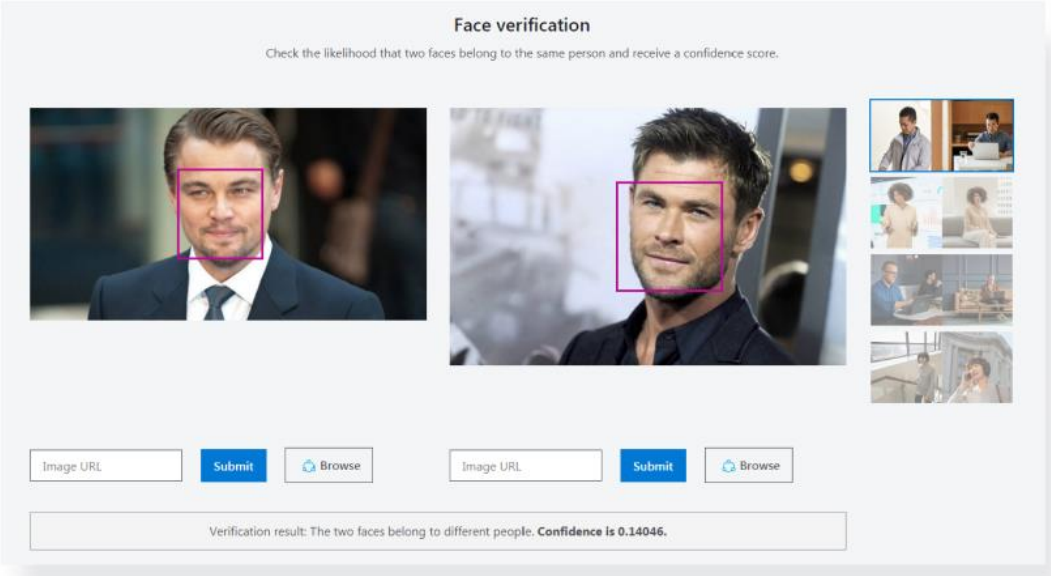


Рисунок 1.4 — Порівняння облич

FaceID також порівнює дві фотографії, щоб визначити, чи описують вони одну людину. Ви можете завантажити зображення зі свого комп'ютера або скористатися URL-адресою, щоб перевірити зображення. Крім того, система оцінює схожість обличчя (59,2% в даному випадку різні люди). Час обробки — 454 мс.

Azure Face API може розпізнавати вирази обличчя: нейтральні, щасливі, злі, презирство, огиду, смуток, страх і здивування. Ця функція називається «розпізнавання емоцій». Azure визначає атрибути сторінки на етапі розпізнавання (перед дією). У цьому режимі програма створює прямокутник навколо виявленої сторінки та призначає рівні точності, що відповідають кожній емоції.

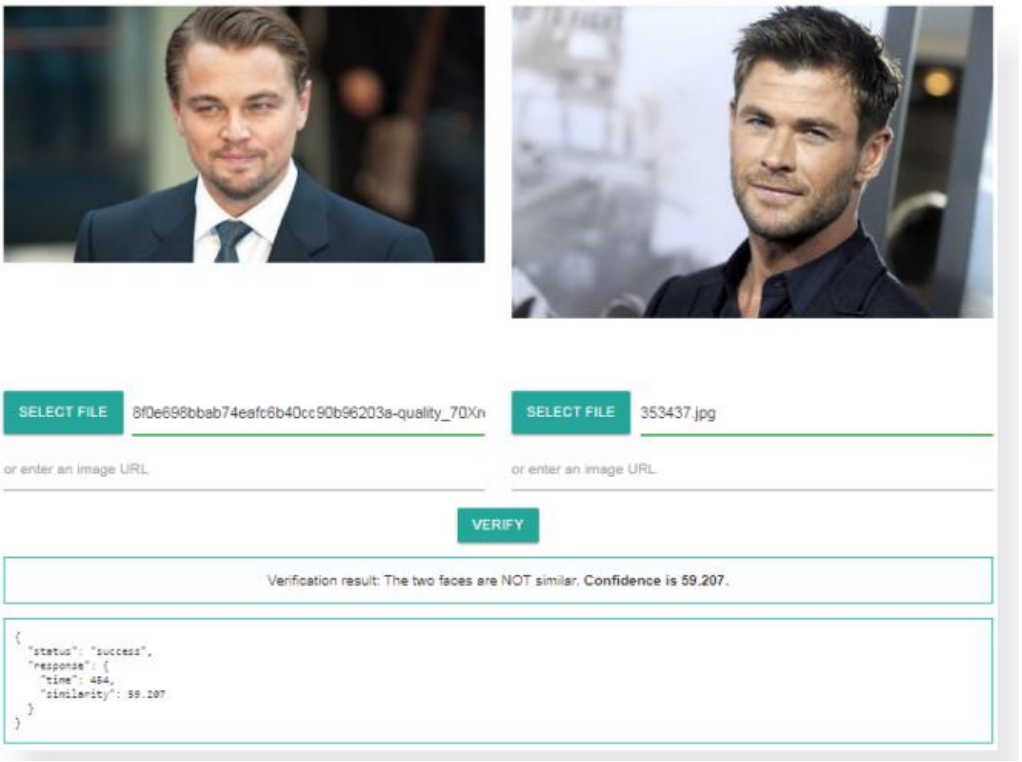


Рисунок 1.5 — Порівняння подібності облич

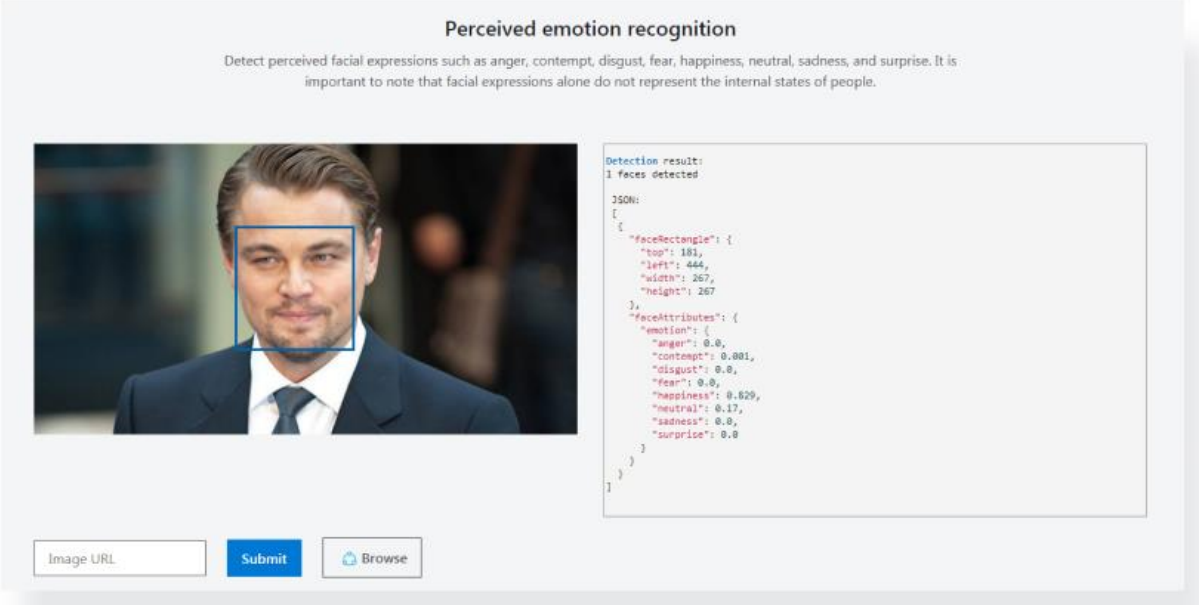


Рисунок 1.5 — Оцінка достовірності для кожної емоції

Розпізнавання емоцій наразі розробляється, оскільки воно не є частиною пакета послуг ідентифікації обличчя. Ми можемо створити індивідуальне рішення, яке може розпізнавати такі емоції: щастя, нейтральне вираз обличчя, подив, смуток, огиду, гнів і страх.

Сервіс Azure Face може порівнювати цільове обличчя та набір облич, доступних для пошуку, щоб знайти дуже схожі обличчя. Програма підтримує два режими роботи: matchPerson повертає схожі обличчя після фільтрації для того самого користувача, а режим matchFace ігнорує такий фільтр і повертає список виявлених облич, які можуть належати або не належати людині, яку ми шукаємо.

Наразі ми не включили можливість пошуку однакових облич у пакет Face ID. Але ми можемо реалізувати цей функціонал у вашому проекті.

Рисунок 1.5 — Пошук подібних облич

Щоб порівняти повну сторінку та колекцію фотографій після отримання, API пошуку поверне вибірку найбільш схожих сторінок і відповідних тегів подібності. Це буде виглядати так:

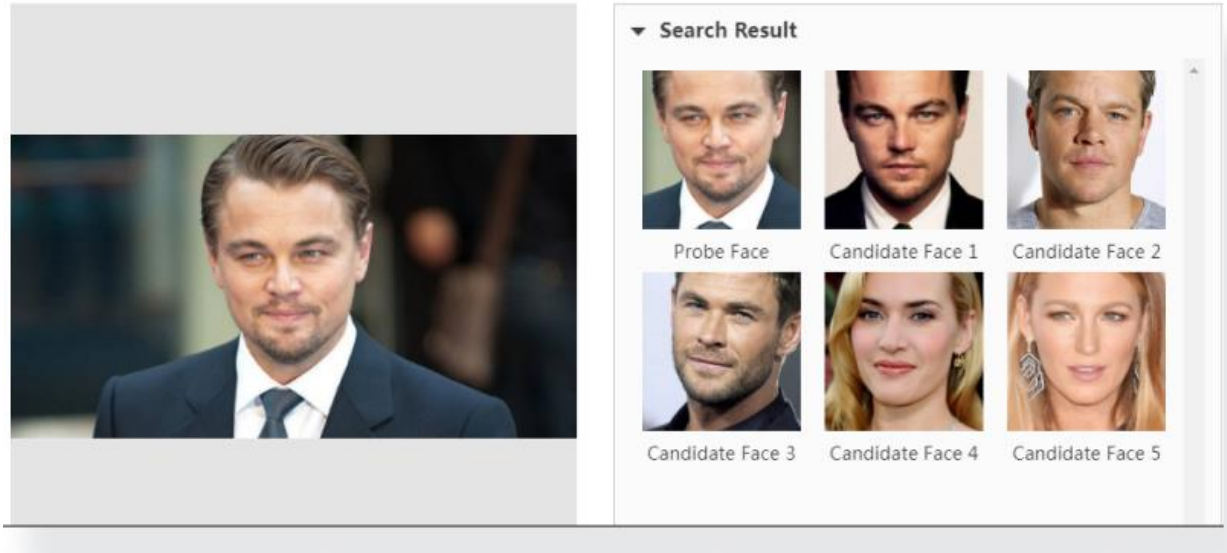


Рисунок 1.6 — Порівняння з групою кандидатів

Рисунок 1.7 — Результати пошуку і відповідь системи в форматі JSON:

На цьому прикладі бачимо, що система розпізнала ту саму людину на фото 1 (його ми використовували як запит на пошук) і фото 2 з найбільшою вірогідністю. Для інших кандидатів ймовірність співпадіння низька.

1.4.2 Комплексний продукт по розпізнаванню Cloud Vision AP

Cloud Vision AP — це комплексний продукт розпізнавання від Google, який використовує попередньо навчені моделі. Він розпізнає об’єкти та обличчя на фото, може розпізнавати текст, автоматично призначати метадані тощо. Для цілей цього огляду ми зосередимося на функціях, пов’язаних із розпізнаванням і порівнянням справжніх облич.

Функція розпізнавання облич у Google Cloud Vision API може виявляти одне чи кілька облич на фотографії. Програма також визначає загальні властивості фотографії (недоекспонування, розмитість). Однак розпізнавання в контексті особи не підтримується.

Cloud Vision виявляє лише наявність облич на зображенні — так само, як розпізнає об’єкти. Система повертає такі дані: текстовий опис, рівень достовірності та нормаль вершин [0, 1] прямокутника, в якому розташований об’єкт (у JSON).

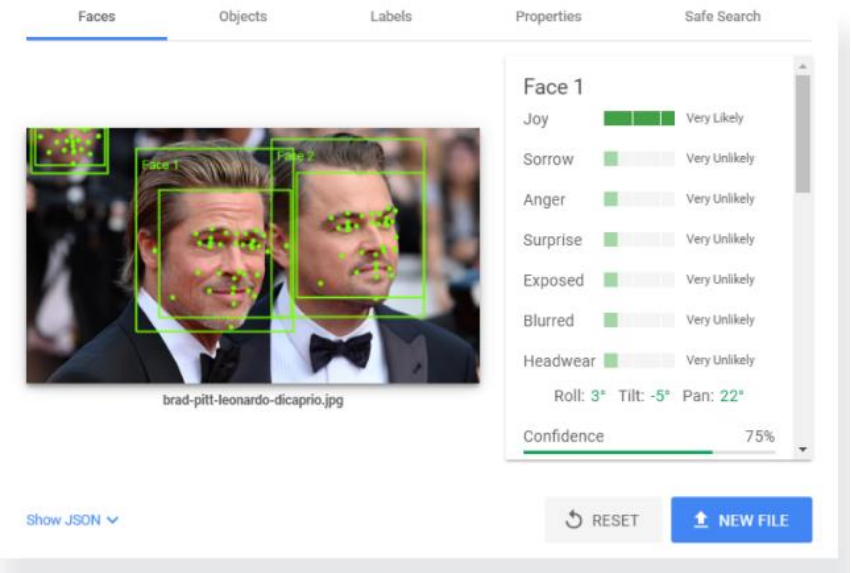


Рисунок 1.**8** — Розпізнавання облич в Google Cloud Vision API

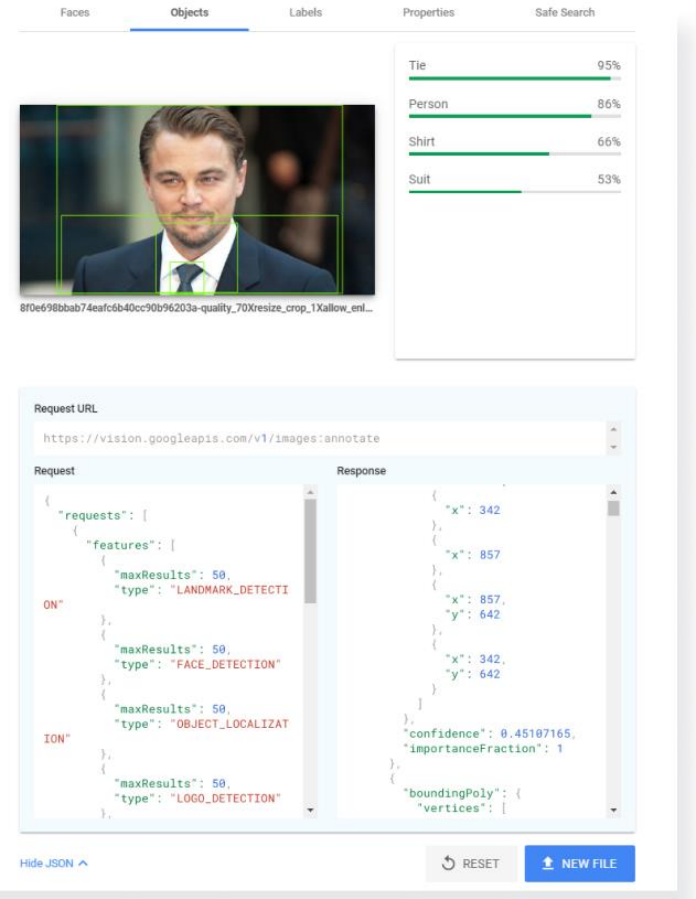


Рисунок 1.**9** — Визначення наявності облич на зображенні

Face ID повертає точні прямокутники та визначає координати ключових точок обличчя (підборіддя, очі, брови, куточки рота, контур носа). Алгоритм також повертає додаткові атрибути: стать, вік, наявність усмішки, положення голови, розмиття, наявність окулярів, закриті/відкриті очі — у формі JSON.

Cloud Vision API не підтримує порівняння та перевірку сторінок. Система просто перевіряє наявність обличчя одного або кількох людей на знімку.

За допомогою FaceID ви можете порівняти два зображення, щоб визначити, чи належать ідентифіковані обличчя одній людині. Якщо схожість <80%, система визначає, що на фотографії зображені різні люди.

Два фото однієї людини, с хожість 93,1% (збіги обличчя), час відповіді 1,163 с.

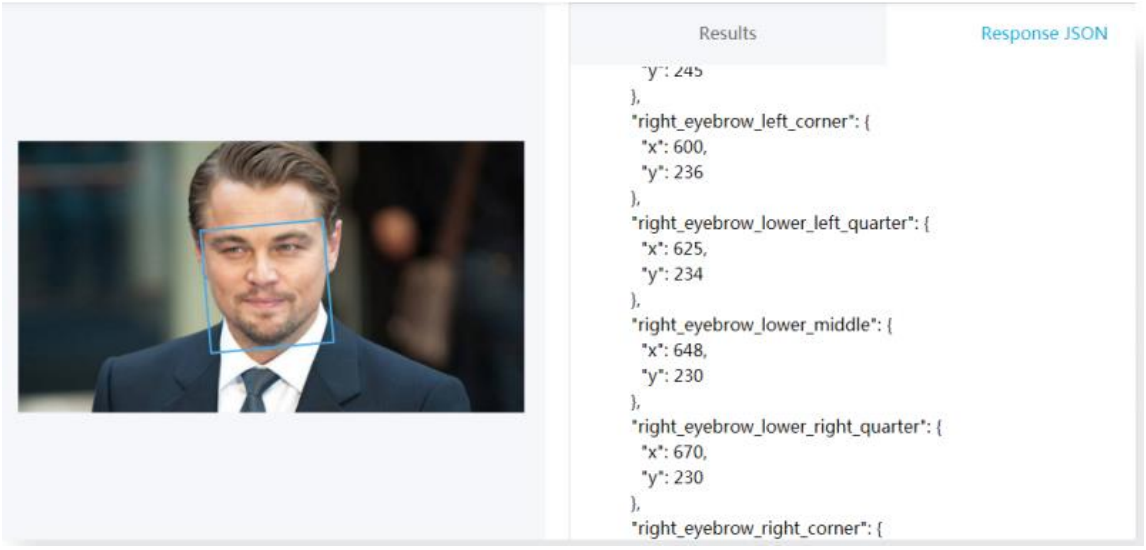


Рисунок 1.10 — Визначення координат і атрибутів в Cloud Vision API

Хоча ці актори дуже схожі, система оцінила схожість на рівні 57,6% (обличчя не збігаються), а час обробки склав 0,459 с.

Cloud Vision API повертає прямокутники навколо виявлених облич і визначає ключові точки (очі, вуха, ніс, рот тощо) з координатами достовірних точок. Система також повертає імовірнісні бали для емоцій (щастя, смуток,

гнів, здивування) і загальних характеристик зображення (низька експозиція, розмиття, наявність масок).

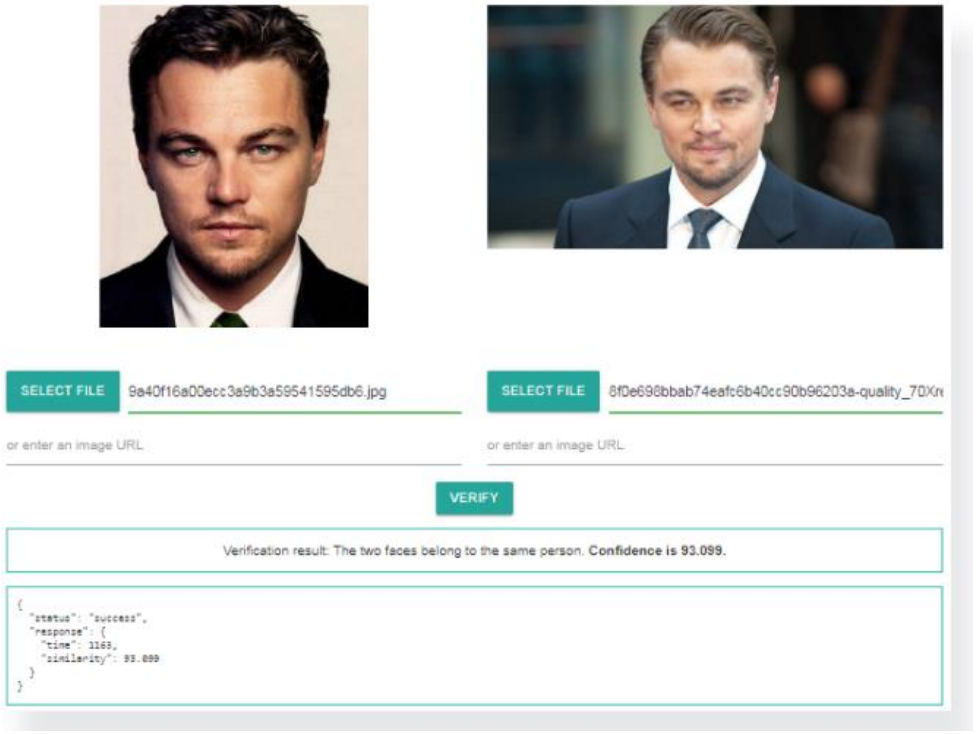


Рисунок 1.11 — Фото однієї людини

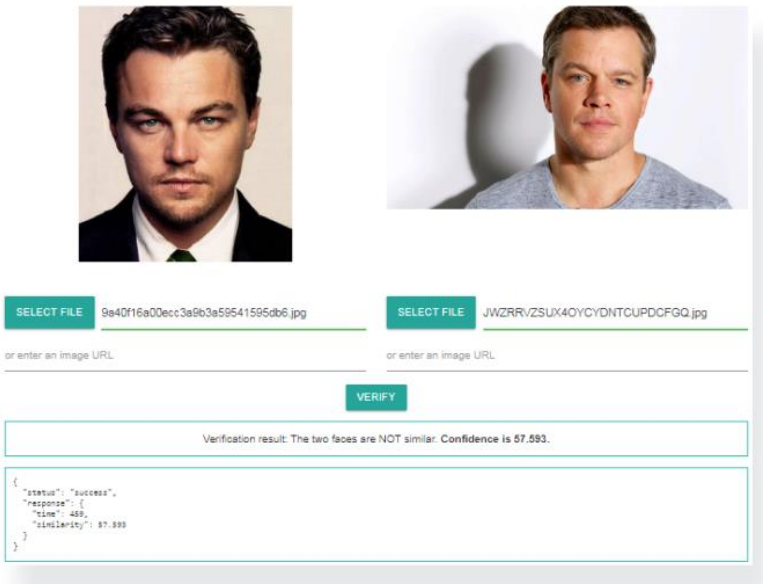


Рисунок 1.12 — Фото двох різних людей

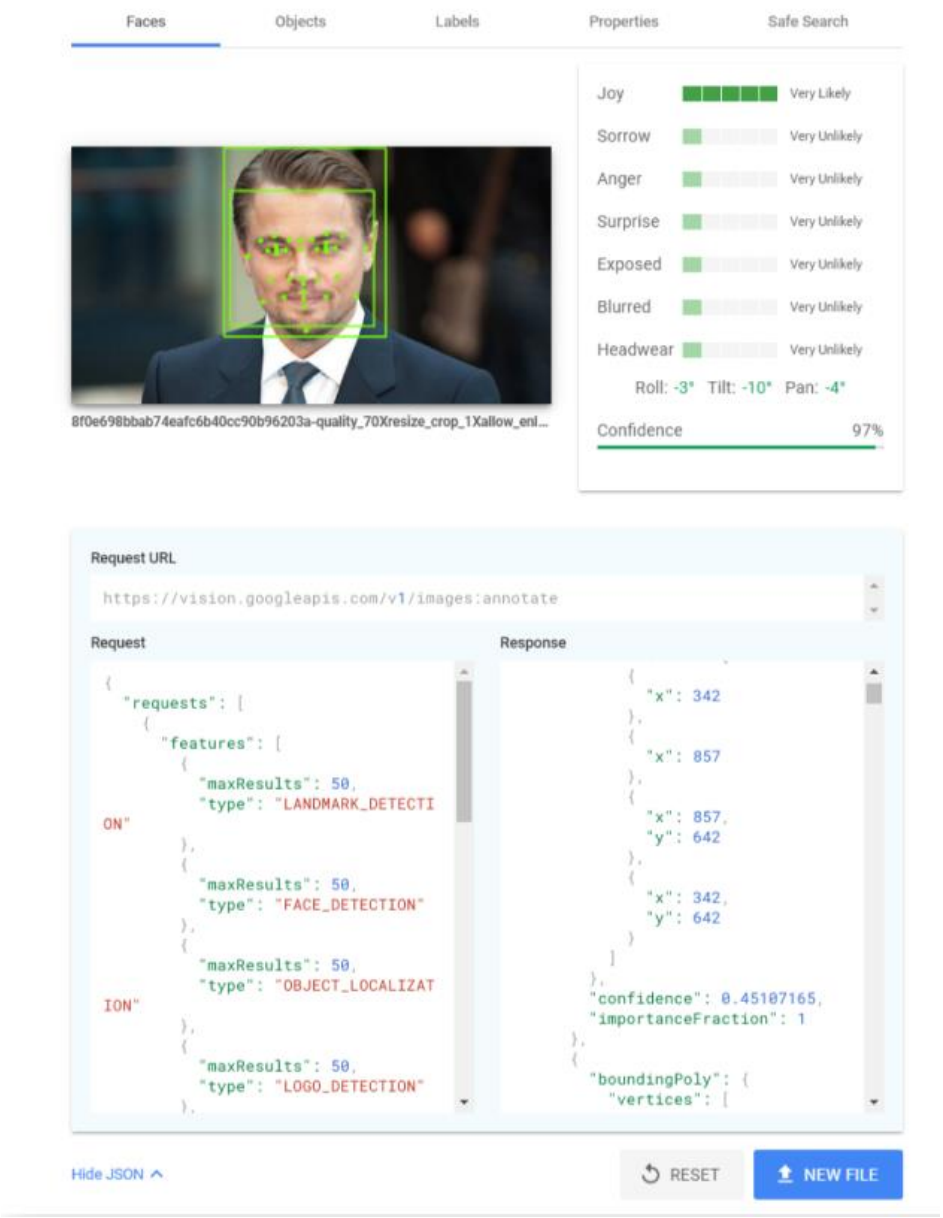


Рисунок 1.13 — Імовірнісні оцінки для емоцій

1.4.3 Програма KLIK

Програма KLIK [11] розпізнаванняобличь під iOS, EmguCVRecognizer[12].

Однією з програм-аналогів, яка вирішує поставлену задачу є програма під назвою EmguCVRecognizer. Вона написана мовою C#, тому швидкодія її відносно менша за аналоги, написані на C++. Ще одна характеристика яку можна покращити — достовірність розпізнавання, яка складає ~ 80%.. Нижче

показано розпізнавання облич під iOS програмою KLIK, скріншот зображено на рис. 1.14.

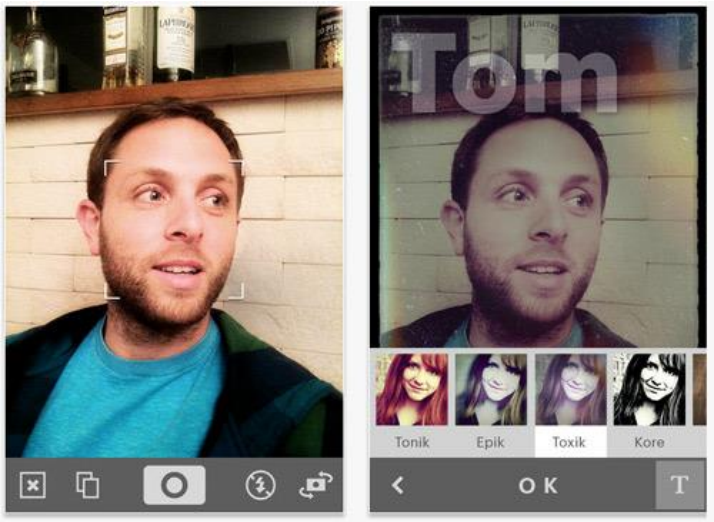


Рисунок 1.14 — Скріншот програми KLIK.

Компанія Face.com — один із провідних світових розробників систем розпізнавання облич, постачальник відповідної технології для Facebook – кілька місяців працювала над секретним «Проектом Badass» (це не справжня назва) і нещодавно показала результати. : безкоштовна програма KLIK для iOS 4.3+. Він в режимі реального часу розпізнає обличчя людей, на яких спрямована камера смартфона. Нехай люди знають про це серед тих, кого ви фотографували та позначили раніше, а також серед ваших друзів у Facebook.

Інколи програма працює настільки швидко, що навіть не потрібно натискати кнопку спуску затвора, щоб ідентифікувати людину чи кількох людей у кадрі. Обробка інших малюнків може тривати до хвилини. Розробники стверджують, що це вся цінність версії 0.9, поки програма знаходиться в експериментальній стадії. Система працює навіть в умовах поганого освітлення. Точність розпізнавання обличчя Face.com в середньому становить 91,3%.

2 МАТЕМАТИЧНИЙ ОПИС ПРИНЦИПІВ ДЕТЕКТУВАННЯ ОБЛИЧ

Задача розпізнавання обличь є однією з задач розпізнавання образів, яку можна представити наступним чином. Нехай на вхід системи розпізнавання надходить деякий об’єкт *x*, який необхідно розпізнати. Цей об’єкт, підлеглий до розпізнавання, називають образом, що у контексті задачі розпізнавання обличь безпосередньо є його зображенням. При цьому вважається, що *x* належить до одного з класів з множини всіх класів $A = \{a_1, \ldots, a_n\}$ – кожен клас відповідає ідентифікатору однієї людини, наприклад, її імені. Передбачається, що ці класи є непересічними.

При завданні будь-якого *x* через функцію *f(x)*, яка являє собою систему або алгоритм розпізнавання обличь, на виході потрібно отримати номер або значення відповідного елемента множини *A*, якому належить вхідне зображення *x*, з вказівкою ступеню достовірності результату розпізнавання, або отримати інформацію, що вхідний образ не належить жодному з класів [3].

2.1 Нейронна мережа як сучасний метод машинного навчання

Спочатку ідея нейронних мереж замислювалася як модель роботи людського мозку. Відомо, що в мозку людини присутні нейрони, які весь час передають один одному інформацію у вигляді електричних сигналів по зв’язкам – синапсах. При цьому зв’язки є не між усіма нейронами і відрізняються за силою. Коли людина вчиться чомусь новому, між нейронами, які відповідають за відповідні навички, утворюються нові зв’язки або стають сильнішими, якщо вони вже є. Чим сильніше зв’язок між нейронами, тим простіше інформація передається від одного іншому і, отже, людині з кожним разом простіше виконувати дії, за які відповідають ці нейрони.

Наприклад, дуже міцні зв’язки між тими нейронами, що відповідають за навички, набутими ще в дитинстві, такими, як здатність ходити або говорити. У

процесі життя людини зв’язки між нейронами то утворюються і стають сильнішими, то стають слабкішими в залежності від того, набуває людина нових навичок або втрачає їх. Подібна ідея і лягла в основу першої моделі штучних нейронних мереж – персептрону, що також називають повнозв’язною нейронною мережею [4].

Пізніше, на основі відкриттів в області зорової кори людини, було розроблено модель згорткової нейронної мережі, націленої саме на розпізнавання образів [2]. Зазвичай ЗНМ включають в себе повнозв’язну мережу, тому далі розглянуто алгоритм роботи персептрону, а потім згорткової мережі.

2.1.4 Опис архітектури повнозв’язної нейронної мережі

Персептрон складається з послідовних шарів, які, в свою чергу, складаються з нейронів. Кожен нейрон попереднього шару пов’язаний з кожним нейроном наступного шару. Саме через це такий вид нейронних мереж і називається також повнозв’язною (рис. 2.1). При цьому зв’язки мають певні числові ваги, які і показують наскільки міцний зв’язок між нейронами. Чим більше вага у зв’язку, тим більший вплив надають нейрони один на одного.

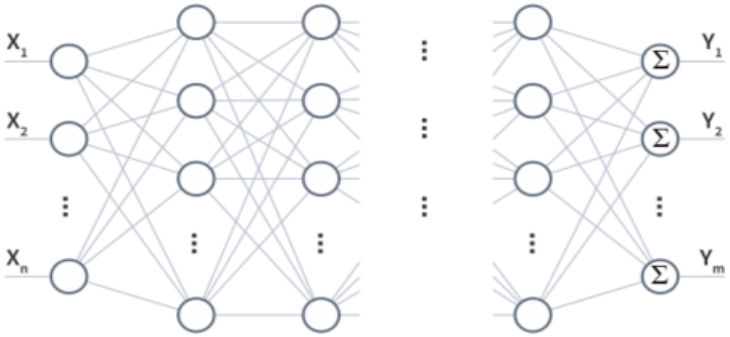


Рисунок 2.1 — Загальний вигляд структури персептрону

З точки зору використання, повнозв’язну нейронну мережу можна представити у вигляді чорного ящика. Перший шар називається вхідним,

останній вихідним, а шари між ними – прихованими. На вхід кожного нейрону вхідного шару подається якісь числові значення X і на виходах кожного нейрону вихідного шару можна отримати нові значення Y.

Нейрони першого шару, по суті, просто приймають вхідні значення, які задаються мережі. Нейрони наступних шарів мають більш складну структуру, наведену на рисунку 1.2. У її структурі можна виділити два основних елементи: суматор і функція активації.

Суматор приймає значення з усіх нейронів попереднього шару, з якими він пов’язаний, перемножує їх на ваги відповідних зв’язків і підсумовує їх. При цьому, до суми також додається порогове значення, яке також називають зміщенням. Його можна уявити як вагу додаткового нейрона з фіксованим значенням +1. Воно робить нейрон більш гнучким, додаючи ефект афінного перетворення.

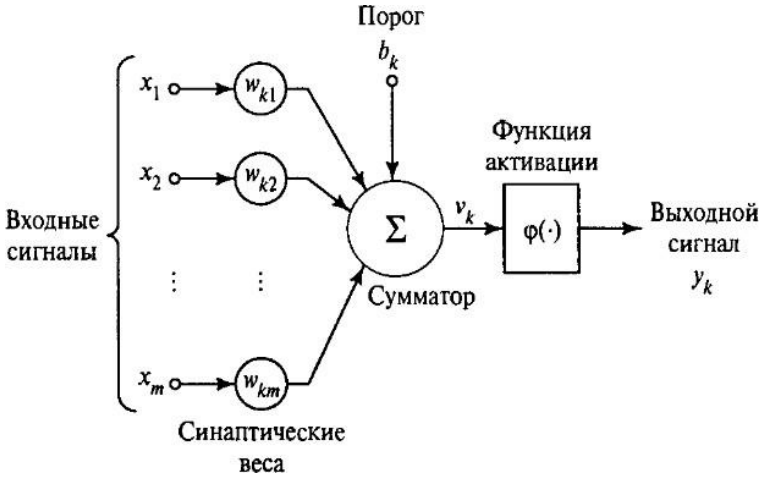


Рисунок 2.2 — Формальна модель нейрону

Функція активації згладжує значення на суматорі, обмежуючи значення вихідного сигналу. Зазвичай, значення на виходах нейронів лежать в інтервалі [0, 1] або [-1, 1] в залежності від використовуваної функції активації. Математично, розрахунок вихідного значення наведено у формулі (2.1).

$$f(w_1x_1 + w_2x_2 + \cdots + w_mx_m + b), \quad (2.1),$$

де f — використовувана функція активації;

w_m — вага між поточним нейроном і нейроном m попереднього шару;

x_m — значення на нейроні m попереднього шару;

b — зміщення на поточному нейроні.

Для всієї мережі в цілому або для кожного шару окремо може бути обрана єдина функція активації, але частіше за все для прихованих шарів вибирається одна функція, а для останнього шару її вибирають інший залежно від завдання.

Існує безліч функцій активації, одними з найбільш поширених є логістична функція або сигмоїда, гіперболічний тангенс, ReLU, PReLU. На питання, яку функцію вибрати немає однозначної відповіді, зазвичай вона підбирається в ході експериментів або з огляду на вже відомий досвід.

Сигмоїда (див. формулу (2.2)) застосовується в мережах з невеликою кількістю прихованих шарів (як правило, не більше шести). Це викликано тим, що у цих функцій значення зазвичай менше одиниці і при перемножуванні безлічі значень нейронів з такою функцією активації виходять дуже маленькі значення, через що нейронна мережа навчається повільніше — по суті більше навчаються останні шари, так як на них в такому випадку помилка впливає більше, а перші майже не змінюються.

$$f(x) = \frac{1}{1+e^{-x}} \quad (2.2)$$

Гіперболічний тангенс (див. формулу (2.3)) за властивостями і застосування збігається з сигмоїдою, але відрізняється від неї тим, що має діапазон значень (-1; 1). Він буває корисний в задачах, де потрібно на виході мережі також отримувати негативні значення.

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (2.3)$$

Для мереж з великою кількістю шарів на прихованих шарах використовують функцію активації ReLU – для таких випадків вона вважається незамінною. Але при малій кількості шарів, її також використовують замість інших функцій активації. Раніше, замість цієї функції, використовували гіперболічний тангенс, але в певний момент виявили, що ReLU дає більш кращі результати, завдяки чому вона стала більш розповсюдженою [6].

$$f(x) = \begin{cases} 0, x < 0 \\ x, x \geq 0 \end{cases}$$

Іноді, кращі результати в порівнянні з ReLU дає її модифікація PReLU. В цю функцію додається параметр, який навчається разом з усіма іншими параметрами мережі.

$$f(x) = \begin{cases} \alpha x, x < 0 \\ x, x \geq 0 \end{cases}$$

У задачах класифікації при непересічних класах використовується функція softmax на вихідному шарі (див. формулу (1.6)). Вона переводить значення вихідних нейронів в значення, ближче до імовірнісних, враховуючи значення на всіх вихідних нейронах.

$$f_i(\vec{x}) = \frac{e^{x_i}}{\sum_{j=1}^{m} e^{x_j}}, i = 1, \dots, m$$

де $\vec{x}$ — вихідний вектор нейронної мережі;

m — кількість значень в вихідному векторі.

У завданнях регресії, тобто в задачах, де вихідне число це і є відповідь і його не потрібно якось інтерпретувати, як, наприклад, в задачах класифікації, на вихідному шарі може використовуватися лінійна функція активації;

$$f(x) = x$$

Таким чином, проходячи по шарам і розраховуючи значення на кожному нейроні, нейронна мережа видає результат на останньому шарі. Такий процес називається прямим ходом або процесом прямого поширення.

Навчання нейронної мережі є завданням оптимізації, в якій підбираються параметри ваг і зміщень для досягнення бажаного результату. За допомогою заздалегідь обраної функції помилки (критерію якості) обчислюються помилки на нейронах останнього шару L і за допомогою методу градієнтного спуску виконується пошук нового значення ваг зв'язків, що ведуть до відповідних нейронів [7].

Щоб знайти величину зміни для даної ваги, використовується похідна від функції помилки по цій вазі, яка показує нахил, який вказує в напрямку зменшення помилки. Саме цей нахил і дає напрямок, в якому потрібно рухатися, щоб досягти локальних мінімумів. А для того, щоб підштовхнути вагу в знайденому напрямку потрібно використовувати коефіцієнт швидкості навчання. Його можна розглядати як крок зміни ваги, який дорівнює:

$$\Delta_{ij}^{(l)} = -\eta * g_i^{(l)} * y_j^{(l-1)},$$

де η — коефіцієнт швидкості навчання ($0 < \eta < 1$);

g — градієнт для нейрону i поточного шару;

y — сигнал нейрону j з попереднього шару, з яким пов'язаний нейрон i.

Градієнт залежить від функції помилки, яка, в свою чергу, залежить від помилок сигналів на виходах нейронів. Помилки на нейронах вихідного шару є різницею між отриманими і очікуваним сигналами, тоді градієнт для нейрону вихідного шару:

$$g_j^{(l)} = -\eta * f'\left(S_j^{(l)}\right) * \sum_{q=1}^{N^{(l+1)}} w_{jq}^{(l+1)} * g_q^{(l+1)},$$

де g_j — градієнт нейрону j;

S_j — значення суматору цього нейрону;

$f'(x)$ — похідна функції активації;

y_j — отриманий сигнал на нейроні;

d_j — очікуваний сигнал на нейроні.

Для прихованих шарів неможливо безпосередньо задати критерій якості, тому помилки на нейронах шару $L-1$ обчислюються як зважена сума градієнтів наступного шару L і далі, таким же чином, коригуються ваги нейронів цього шару. В такому випадку градієнт виглядає наступним чином:

$$g_j^{(l)} = -\eta * f'\left(S_j^{(l)}\right) * \sum_{q=1}^{N^{(l+1)}} w_{jq}^{(l+1)} * g_q^{(l+1)},$$

де g_j — градієнт нейрону j прихованого шару l;

η — коефіцієнт швидкості навчання;

S_j — значення суматору нейрону j; $f'(x)$ – похідна функції активації;

N — кількість нейронів у наступному шарі;

w_{jq} — вага між нейроном j шару l та нейроном q наступного шару;

g_q — градієнт нейрону q наступного шару.

Залежно від використовуваної функції активації, її похідна буде відрізнятися.

При навчанні нейронної мережі все перераховане вище виконується для всіх ваг і зміщень, перебираючи всі приклади навчання. Можна перебирати ті ж приклади багато разів, а як сигнал зупинки вважати досягнення необхідної точності помилки нейронної мережі або ліміту кількості епох – ітерацій по всім прикладам.

Функцію помилки у нейронній мережі, як і функцію активації, вибирають

експериментальним шляхом або беруть до уваги вже відомий досвід використання в конкретних завданнях.

Для розпізнавання при класифікації двох класів використовуються хіндж, квадратичний хіндж або бінарна крос-ентропія. Важливо помітити, що бінарна крос-ентропія працює з вихідними значеннями сигмоїди, а категоріальна — з softmax (версія функції max, що залишає інформацію про немаксимальні значення) [8].

$$E = \frac{1}{N}\sum_{i=1}^{N} \max(1 - d_i y_i, 0)\,, y_i \in \{-1,1\}$$

$$E = \frac{1}{N}\sum_{i=1}^{N} (\max(1 - d_i y_i, 0))^2\,, y_i \in \{-1,1\}$$

$$E = -\frac{1}{N}\sum_{i=1}^{N} [d_i \log y_i + (1 - d_i) \log(1 - y_i)]$$

де E — помилка нейронної мережі;

N — кількість нейронів у вихідному шарі;

y_i — вихідне значення на i-му нейроні;

d_i —очікуване значення мережі на i-му нейроні.

При класифікації більше двох класів у якості функції помилки нейронної мережі використовується категоріальна крос-ентропія:

У завданнях регресії використовуються середній квадрат помилок, середня сума модулів помилок, середній квадрат логарифмічних помилок (див. формулу, середній абсолютний відсоток помилок.

$$E = \frac{1}{N}\sum_{i=1}^{N} |d_i - y_i|$$

$$E = \frac{1}{N}\sum_{i=1}^{N} (\log(d_i + 1) - \log(y_i + 1))^2$$

$$E = \frac{1}{N}\sum_{i=1}^{N} \frac{|d_i - y_i|}{\max(|y_i|, \varepsilon)}$$

Проблемою середнього квадрата помилок є наявність безлічі локальних мінімумів, через що при навчанні більша ймовірність, що градієнтний спуск застрягне в одному з них, що призводить до гіршої якості розпізнавання. Також при використанні даного критерію якості помилка різко зростає, якщо існують значні, але рідкісні помилки, тому його варто використовувати, якщо такі помилки важливі.

При використанні середнього модуля помилок помилка зростає вже не так різко, якщо існують значні, але рідкісні помилки, а середній квадрат логарифмічних помилок ще більше зменшує вплив рідкісних значущих помилок. Середній абсолютний відсоток помилок дозволяє відійти від абсолютних значень до відсотків.

2.1.2 Опис архітектури згорткової нейронної мережі

Згорткові нейронні мережі складаються з шарів згортки і шарів пулінгу (pooling), які також називають шарами субдискретизації (рис. 1.3). Вхідними даними для нейронної мережі, в разі зображення, є 2D-тензор [9] яскравості його пікселів. У разі, якщо зображення кольорове, воно розбивається на канали RGB і подається вже три зображення [10].

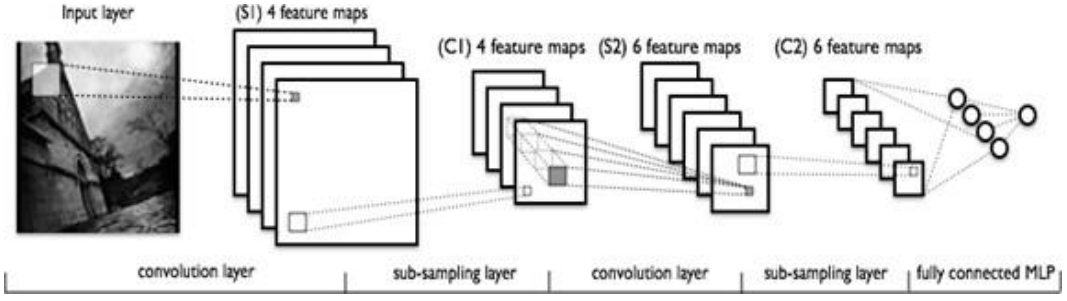


Рисунок 2.3 — Структура згорткової нейронної мережі

Потім для кожного каналу виконується операція згортки за допомогою фільтра згортки (рис. 2.4). Фільтр виділяє певні ознаки на зображенні,

формуючи нове зображення, яке називається картою ознак. Наприклад, виділяються горизонтальні або вертикальні лінії, вони стають більш яскравими, в порівнянні з іншими елементами на зображенні. При цьому, фільтрів може бути багато, кожен з яких виділяє на зображенні щось своє і формує свою карту ознак.

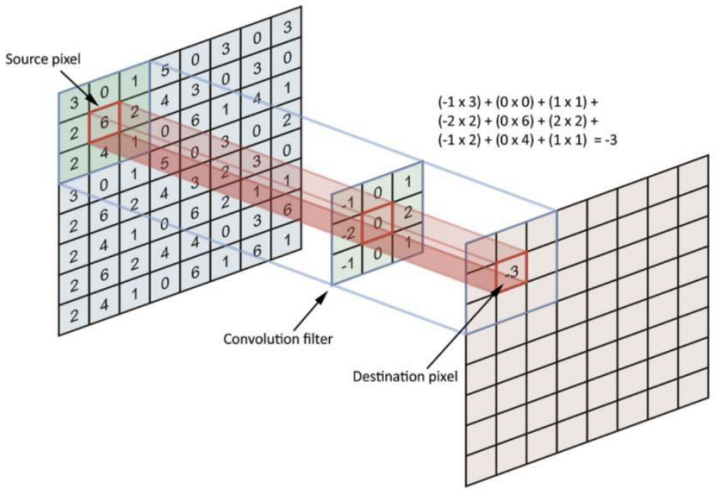


Рисунок 2.4 — Візуалізація процесу згортки

У разі двовимірного тензора фільтр є матрицею невеликого розміру, яка містить вагові коефіцієнти. Під час виконання операції згортки, фільтр за допомогою методу ковзаючого вікна проходить по зображенню, перемножує елементи зображення на відповідні елементи фільтра, підсумовує їх, а також додає ще один ваговий коефіцієнт w_0 у якості зсуву. Це значення також пропускається через функцію активації і утворює нове значення пікселя на зображенні. Формальний вид розрахунку нового значення v пікселя за допомогою фільтра розміром 3x3 наведено у формулі

$$v_{k,m} = f\left(\sum_{i=1}^{3}\sum_{j=1}^{3} x_{i+k,j+m} w_{ij} + w_0\right)$$

де k, m — індекси пікселів на зображенні;

f — функція активації;

x — вихідне значення пікселю;

w_{ij} — відповідна вага у фільтрі згортки;

w_0 — зміщення на +1.

Розмір фільтра і крок, з яким він проходить по зображенню, є гіперпараметрами мережі, тобто параметрами, які вибираються заздалегідь як частина конфігурації і потім не змінюються. Кількість таких фільтрів не обмежена і також становить частину конфігурації мережі. Кожен фільтр формує свою карту ознак (канал), обчислюючи її з карт ознак попереднього шару. При чому, так як каналів на попередньому шарі може бути кілька, як у випадку з кольоровими зображеннями, для формування нової карти ознак цей фільтр проходить по кожному з вхідних каналів, а результат підсумовується.

Зазвичай для фільтра згортки вибирається крок, що дорівнює одиниці. Якщо вибирається крок більшого розміру, зображення ставатиме меншого розміру в залежності від кроку. Також важливо помітити, що після операції згортки в залежності від розміру фільтру, розмір зображення стає менше по краях, так як фільтр не може вийти за межі зображення. Якщо такий ефект потрібно прибрати, перед згорткою до зображення по краях додається відступ (padding) з нульових значень.

Після виконання згортки, отримані канали проходять через шар пулінгу. Він також виконується за допомогою ковзного вікна, проходячи по зображенню і з усіх елементів в області вікна обчислює єдине значення. Існує три типи пулінгу – max pooling, min pooling і average pooling. При max пулінгу з усіх елементів у вікні вибирається найбільший елемент, при min пулінгу – найменший, а при average – середній. Зазвичай для шарів пулінгу вибирається крок, рівний розміру вікна пулінгу, щоб зображення стискалося, залишаючи лише найбільш значущі елементи перед тим як перейти до обробки в наступних шарах мережі. Найпоширенішим типом пулінгу є max pooling, візуалізація процесу якого наведено на рисунку 2.5, а приклади зміни зображення при його використанні — на рисунку 2.6.

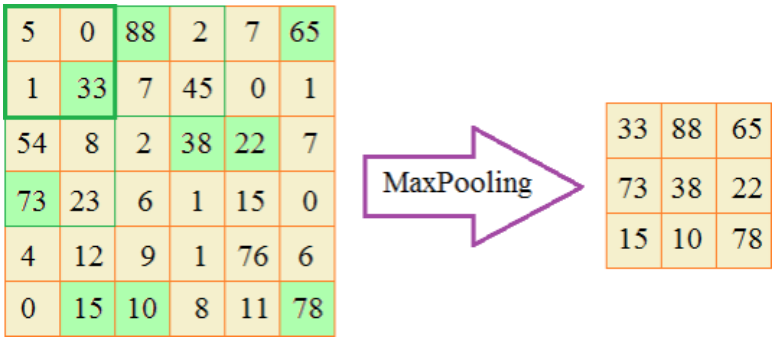


Рисунок 1.5 — Візуалізація процесу max pooling

Рисунок 1.6 — Приклади зображення при використанні max pooling

У згортковій мережі шари згортки і пулінгу зазвичай чергуються. Кожен наступний шар згортки виділяє все більш складні ознаки. Спочатку виділяються прості елементи зображення, а наступні шари виділяють все більш складні. В [14] проводилося дослідження, присвячене візуалізації роботи ЗНМ, виділяючи частини зображення, на яких мережа показує більшу чутливість. Приклади таких областей на зображенні в залежності від шару представлено на рисунку 2.7.

Рисунок 2.7 — Приклади областей, к котрим згорткова нейронна мережа більш

чутлива в залежності від шару

Як можна помітити, на першому шарі вона виділяє лінії, градієнти, елементи, пов’язані з кольором і відтінку. На другому шарі виділяються прості форми і текстури. На наступних шарах виділяються частини об’єктів, наприклад, колеса машин. На одному з останніх шарів мережа бачить вже цілі об’єкти.

Після того як зображення пройшло через шари згортки і пулінгу, отримані карти ознак виділяють найбільш значущі частини зображення в компактному вигляді. Далі вони можуть подаватися на вхід іншим алгоритмам, наприклад, для вирішення задачі класифікації або регресії. Зазвичай в ЗНМ після шарів згортки додається повнозв’язна нейронна мережа, яка і вирішує поставлене завдання.

Принцип навчання згорткових шарів нейронної мережі такий же, як і в персептроні. Але в даному випадку за допомогою методу градієнтного спуску оптимізуються вагові коефіцієнти в ядрах згортки.

2.1.3 Відомі проблеми глибокого навчання та методи їх подолання

При розробці глибоких нейронних мереж, що мають велику кількість шарів існує проблема загасання і вибуху градієнта. Для того, щоб пояснити звідки беруться такі ефекти, припустимо, що у нас є мережа з кількістю шарів, наприклад, більше ста, а також для спрощення будемо вважати, що функція активації – лінійна, а вектор зсувів дорівнює нулю. В такому випадку вихідний вектор мережі матиме загальний вигляд, наведений у формулі.

$$y = w^{[l]}w^{[l-1]}w^{[l-2]} \dots w^{[3]}w^{[2]}w^{[1]}x$$

де x — вхідний вектор мережі;

w — матриця ваг на шарі *l*.

Тоді, якщо значення елементів всіх матриць ваг трохи більше, ніж у одиничної матриці, значення будуть рости з кожним наступним шаром і в підсумку значення вихідного вектора *y* будуть дуже великими, викликаючи вибух градієнта. А якщо значення матриць ваг будуть трохи менше одиниці, тоді *y* буде дуже маленькими, через що градієнт буде затухати. Оскільки навчання мережі залежить від градієнта, при таких ефектах мережа просто не зможе навчатися.

Тому, крім стандартних повнозв'язних шарів, а також шарів згортки, пулінгу, в глибоких нейронних мережах можуть використовуватися додаткові, комплексні види шарів, що комбінують в себе інші шари. У таких випадках такі шари можуть складаютися в цілі модулі. Одним з таких видів шарів є residual шари, що в перекладі — залишкові, які зазвичай і використовуються для вирішення проблеми, пов'язаної зі загасанням і вибухом градієнта.

Значення на нейронах residual шарів формуються так, як і на звичайному шарі, але з додаванням значень нейронів якогось попереднього шару перед застосуванням функції активації.

$$a^{[l+2]} = f(w^{[l+2]}a^{[l+1]} + b^{[l+2]} + a^{[l]})$$

де *a* — вектор значень на нейронах шару *l*;

w — матриця ваг на шарі *l;*

b — вектор зміщень на шарі *l;*

f – функція активації.

Таким чином збільшується вплив результату з попередніх шарів на наступні. Було виявлено, що це дозволяє робити нейронні мережі більш глибокими без втрати точності, наприклад, створювати мережі, що мають в собі

більше ста або навіть тисячі шарів. При використанні residual шарів в мережі, така мережа називається ResNet, а групи шарів, що реалізує схему, наведену у формулі вище, називають residual модулем [15].

Також, у згорткових мережах іноді буває складно вибрати які параметри фільтра згортки використовувати, яку їх кількість вибрати, або може замість згортки взагалі використовувати шар пулінгу. Цю проблему вирішує ще один вид комплексних шарів — inception [16]. Їх ідея в тому, щоб з набору карт ознак,

отриманого після попередньої операції, генерувати нові карти ознак за різними параметрами і складати їх усіх в одну стопку. Дану схему ілюструє рисунок 2.8.

Таким чином з попереднього шару можуть виходити декілька відгалужень зі своєю чередою шарів зі своїми параметрами, а потім конкатенуватись в результуючий шар.

В такому випадку мережа під час навчання сама для себе вибирає які параметри і типи шарів краще використовувати. Для непотрібних або менш значних шарів мережа обнулить матриці ваг або зробить їх невеликими так, що вони будуть мало впливати на подальший результат. Головне в даному випадку, щоб розміри результуючих карт признаків збігалися.

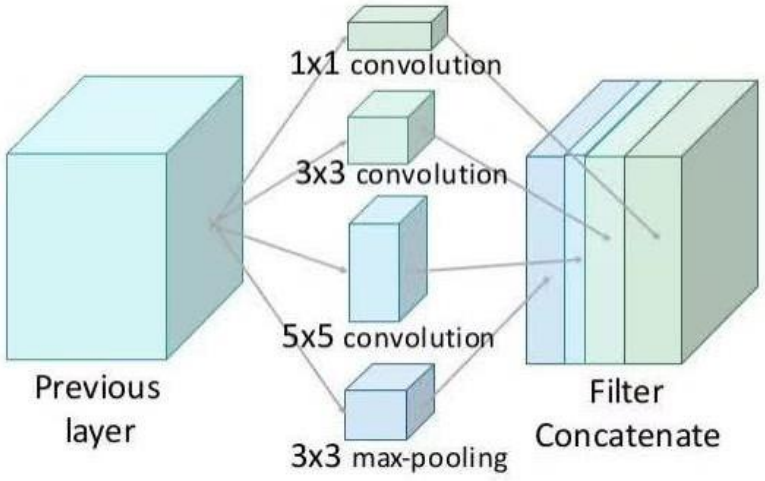


Рисунок 2.8 — Ілюстрація схеми inception модулю

При навчанні у методах ШІ також існує проблема зміщення та дисперсії (bias and variance). У глибокому навчанні вона асоціюється з недонавчанням та перенавчанням мереж. Високе зміщення свідчить, що мережа неспроможна добре підлаштуватися під навчальну вибірку, викликаючи недонавчання, а висока дисперсія показує, що мережа надто підлаштувалася під навчальну вибірку, викликаючи перенавчання. Хорошим балансом є поєднання маленьких зміщень та дисперсії. Візуалізацію цих ефектів наведено на рисунку 2.9.

Рисунок 2.9 — Візуалізація проблеми зміщення та дисперсії

Точки кожному графіку представляють якісь навчальні дані, де X — вхідні дані для мережі, а Y — очікувані вихідні значення. При високому зміщенні видно, що багато значень X на лінії, яка утворюється пророкуваннями мережі на тренувальних даних в результаті навчання мають дуже велику відстань до істинно правильних Y. При високому значенні дисперсії лінія, що утворюється після навчання, проходить через всі навчальні приклади, тому на тестовому датасеті та при подальшій експлуатації мережі, де вхідні приклади матимуть трохи інші значення, сумарна помилка мережі матиме велике значення.

На третьому графіку видно, що поєднання низького зміщення та дисперсії дають досить хороший результат — мережа бачить саму закономірність між вхідними та вихідними значеннями, не підлаштовуючись під тренувальні дані, тим самим не перенавчаючись.

Зазвичай проблема зміщення та дисперсії вирішується за допомогою кращого підбору структури мережі та використання великих датасетів. Але

можливі ситуації, коли з якихось причин немає можливості покращити структуру мережі або використовувати більший датасет. Тому для вирішення цієї проблеми використовується метод, званий регуляризацією. Існує два види регуляризації: L2 та L1 [19].

L2-регуляризація реалізується за допомогою додавання до функції помилки додаткового доданку таким чином, що утворюється помилка мережі, наведена у формулі

$$E = \frac{1}{m}\sum_{i=1}^{m} L\left(d_i, y_i\right) + \frac{\lambda}{2m}\sum_{l=1}^{n}\left\|w^{[l]}\right\|_2^2$$

де m — кількість навчальних прикладів;

L — використовувана функція помилки;

λ — гіперпараметр L2-регуляризації;

n — кількість шарів мережі, виключаючи вхідний шар;

w — матриця ваг шару l.

L1-регуляризація дозволяє «стискати» матрицю ваг, додаючи до неї більше нулів. Оскільки нулі займають менше місця у пам’яті, ваги також займають менше місця. Однак, також практика показує, що це дозволяє зменшити займане місце зовсім небагато, тому набагато частіше використовується L2-регуляризація.

Параметр λ називається параметром регуляризації та підбирається експериментальним чином у межах від 0 до нескінченності. Цей параметр налаштовує чутливість до вхідних даних. Чим більше це значення, тим менш чутливими до вхідних значень мережі стають її прогнози. Зазвичай оптимальним значенням цього параметра є значення, що дорівнює одиниці.

Беручи на увагу вищеописані методи і комбінуючи їх можна робити нейронні мережі дуже глибокими та більш ефективними за якістю

розпізнавання, ніж звичайні мережі, які подібні методи не використовують.

2.2 Опис архітектури системи розпізнавання облич

Система розпізнавання облич складається з декількох етапів: детекція обличь на зображенні, виділення ознак кожного обличчя і вже їх безпосереднє розпізнавання [20].

Детекція обличчя, як і детекція будь-якого іншого об’єкта на зображенні, полягає в знаходженні координат його області, що його обмежує (bounding box) в пікселях. За допомогою нейронної мережі, ця задача може вирішуватися як завдання регресії, в якій потрібно знайти ці координати.

На перший погляд, можна навчити мережу, яка на вхід приймає зображення, а на виході видає чотири числа – координати центру або одного з кутів знайденої області, а також її висоту і ширину. Це найбільш очевидний спосіб і його недоліком є те, що він в кращому випадку зможе знаходити тільки один об’єкт. На різних зображеннях може бути різна кількість осіб, а архітектура мережі задається спочатку і не змінюється, тому потрібен інший підхід.

Таким підходом, що в тому чи іншому вигляді використовується у всіх мережах детекції, є реалізація детекції через класифікацію з використанням ковзаючого вікна [21]. Воно проходить по зображенню і класифікує те, що знаходиться в області вікна. Крок і розмір ковзаючого вікна може вибиратися будь-яким, часто прохід по зображенню виконується по кілька разів, з різним розміром вікна або зображення.

Така ідея працює добре в принципі, однак вона може містити величезну кількість областей, кожну з яких необхідно класифікувати, а тому це працює дуже повільно. Для вирішення такої проблеми виділяють два основних типи методів: двоетапні і одноетапні [22].

Двоетапні методи на першому етапі за допомогою будь-якого допоміжного алгоритму знаходять регіони інтересу, де можуть знаходитись

об’єкти, а потім на другому етапі вони перевіряються класифікатором з локалізацією. Локалізація уточнює координати об’єкта в області. Існує безліч алгоритмів знаходження регіонів інтересу [23], порівняння деяких з яких наведено на рисунку 2.10.

Method	Approach	Outputs Segments	Outputs Score	Control #proposals	Time (sec.)	Repea-tability	Recall Results	Detection Results
Bing	Window scoring		✓	✓	0.2	★★★	★	·
CPMC	Grouping	✓	✓	✓	250	-	★★	★
EdgeBoxes	Window scoring		✓	✓	0.3	★★	★★★	★★★
Endres	Grouping	✓	✓	✓	100	-	★★★	★★
Geodesic	Grouping	✓		✓	1	★	★★★	★★
MCG	Grouping	✓	✓	✓	30	★	★★★	★★★
Objectness	Window scoring		✓	✓	3	·	★	·
Rahtu	Window scoring		✓	✓	3	·	·	★
RandomizedPrim’s	Grouping	✓		✓	1	★	★	★★
Rantalankila	Grouping	✓		✓	10	★★	·	★★
Rigor	Grouping	✓		✓	10	★	★★	★★
SelectiveSearch	Grouping	✓	✓	✓	10	★★	★★★	★★★

Рисунок 2.10 — Методи пропозиції регіонів інтересу

Зазвичай, використовується метод селективного пошуку Selective Search або метод Edge Boxes, оскільки вони працюють досить швидко і дають досить добрі результати детекції. Таким чином зменшується кількість областей для перевірки і прискорюється загальна швидкість роботи системи.

Одноетапні методи ґрунтуються на використанні згорткової реалізації ковзаючих вікон за допомогою згорткових нейронних мереж [24]. Її суть в тому, що після проходу зображення по шарах згортки, кожен піксель на результуючої карті ознак відповідає своїй області пікселів на оригінальному зображенні (рис. 2.11).

Згорткова нейронна мережа виконує всі ті ж лінійні перетворення, а значить можна навчити її виконувати завдання регресії таким чином, щоб кожна з останніх карт ознак відповідала певному значенню, як вихідні нейрони в повнозв’язній мережі. Так, перша і друга карти ознак можуть визначати координату центру області, а третя і четверта – її ширину і висоту. Додатково до координат області в такому методі додається ще значення, що визначає ймовірність або впевненість мережі в тому, що в даній області є шуканий

об’єкт. Так як згортка сама по собі реалізується як ковзаючі вікна, то такий механізм уже закладений в згорткові мережі, що дає великий приріст в швидкості.

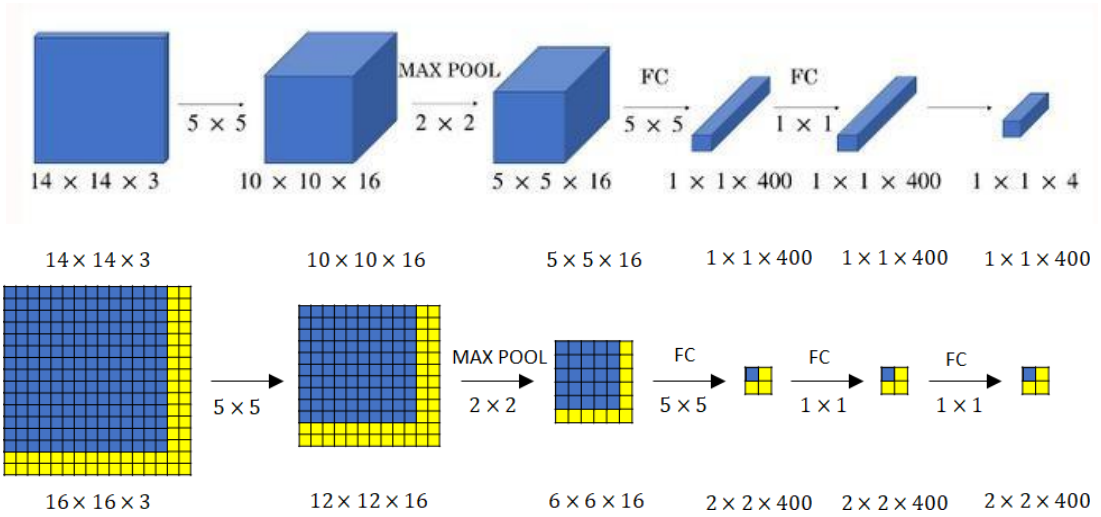


Рисунок 2.11 — Приклад та візуалізація конволюційного підходу детекції об’єктів за допомогою згорткових нейронних мереж

На практиці після виконання класифікації всіх необхідних областей на зображенні нейронна мережа може виділити велику кількість таких прямокутних областей, які можуть перетинатися одна з одною і вказувати на одинакові об’єкти. Але зрештою потрібно визначити лише одну область на одне обличчя. Тому після визначення областей, які містять обличчя людей, застосовується метод придушення областей, з упевненістю присутності обличчя, що дорівнює не максимальній серед усіх знайдених областей. Цей метод має назву Non Maximum Suppression (NMS) [25]. Його алгоритм у тому, що з усіх детектованих областей вибирається лише та, що має максимальний коефіцієнт впевненості, виданий мережею. Але потрібно також враховувати, що на зображенні можуть бути розташовані відразу кілька об’єктів, що детектуються, при чому відстань між ними буде невеликою, а тоді потрібно

розрізняти знайдені області по тому, до кого вони відносяться. Для цього перед вибором області з максимальною впевненістю кожна область порівнюється з усіма іншими і для кожних двох областей обчислюється значення метрики Intersection Over Union (IOU). Ця метрика використовується для порівняння, наскільки області перетинаються одна з одною. Вона розраховується як область перетину, поділена на об’єднану область. Більше значення показує більшу міру перетину, причому значення, рівне одиниці, — області повністю відповідають одне одному, а значення, рівне нулю, — області взагалі перетинаються.

Потім у порівнянні ступеня перетину всіх областей вони кластеризуються з допомогою введення порогового значення. Воно є гіперпараметром і підбирається вручну таким чином, щоб загальна точність детекції відповідала найкращим показникам. Якщо ступінь перетину між двома областями більше за поріг, вважається, що вони вказують на той самий об’єкт, тому групуються разом у свій кластер, звідки вже потім буде обрана область з найбільшою впевненістю.

Крім цього методу також існують інші, які є його продовженням з певними модифікаціями для досягнення кращих показників точності детекції, такі як Soft NMS, Softer NMS, IOU-Guided NMS, Adaptive NMS, DIOU-NMS, кожен з яких має свої переваги та недоліки [26].

Після детекції обличь на зображенні для кожного обличчя виділяються ознаки, що унікально характеризують його в стислому вигляді. Такі ознаки можуть представлятися у вигляді вектора чисел. При використанні глибокого навчання, ознаки виділяються самими нейронними мережами. ЗНМ виділяють ознаки у вигляді результуючих карт ознак зображення, а якщо в якості другої частини згорткової мережі додається персептрон, то вектором ознак є вихідний вектор мережі.

Безпосереднє розпізнавання обличь в свою чергу, ділиться на дві категорії — ідентифікація і верифікація. Ідентифікацією обличчя є процес порівняння одного обличчя з усіма розміченими зображеннями обличь, які містяться в базі

даних (БД), з метою знайти найбільш схоже або прийняти рішення про те, що такого обличчя в базі даних немає. Під розміченими зображеннями мається на увазі, що разом із зображенням обличчя в БД зберігається мітка, що ідентифікує людину, наприклад, його ім’я. Верифікація виконується тільки між двома зображеннями обличь з метою прийняти рішення – одна і та ж це людина, чи ні.

Для вирішення завдання ідентифікації і верифікації, вектори ознак, отримані з двох зображень, порівнюються між собою. Це часто виконується за допомогою сіамських нейронних мереж [27]. Сіамські нейронні мережі складаються з двох однакових нейронних мереж, як за структурою, так і за навченими вагами, тобто, по суті, це два екземпляри однієї і тієї ж мережі. В контексті задачі розпізнавання зображень, на вхід кожної з мереж подається будь-яке зображення, а на виході кожна мережа видає вектор чисел, який в стислому вигляді представляє це зображення. При чому для кожної мережі зображення задається своє, і завдання всієї структури сіамської нейронної мережі — будь-яким чином порівняти вектори на виходах двох її мереж і зробити висновок про те, наскільки вони схожі. Для безпосереднього прийняття рішення про схожість використовується додатковий пороговий гіперпараметр.

Якщо різниця між векторами менше заданого порогу, значить можна зробити висновок про те, що на зображенні представлений один і той же об’єкт або об’єкт одного і того ж класу. Для ідентифікації і верифікації обличь в якості гіперпараметра встановлюється граничне значення схожості, при якому буде вважатися, що на зображеннях знаходиться одна і та ж людина.

Саму мережу, що кодує зображення в вектори ознак, можна навчати двома способами – навчати безпосередньо метрику схожості векторів або виділяти ознаки через навчання класифікатора.

Під навчанням метрики схожості мається на увазі, що мережу навчають таким чином, щоб вектори ознак для зображень обличь однієї і тієї ж людини були близькі один до одного по відстані або куту, а для різних людей – досить різнилися одна від одної так, щоб можна було з упевненістю прийняти рішення

про те, що ці вектори не схожі.

Суть методу виділення векторів ознак із зображень обличь людей через навчання класифікатора полягає в тому, що навчають нейронну мережу, що класифікує зображення обличчя між фіксованим набором людей, після чого просто видаляють останній шар, що класифікує. Вся нейронна мережа, що залишилася, вже вміє виділяти ознаки [28].

При такому підході головним завданням стає правильний вибір функції помилки. Справа в тому, що зазвичай, для класифікації використовується помилка Softmax, але в контексті розпізнавання осіб вона має великий недолік – дана функція помилки не дає чіткого поділу об’єктів, що класифікуються, між їх класами.

Тому, якщо об’єкт одного класу має велику схожість з об’єктами іншого класу, якщо візуалізувати цю задачу як задачу кластеризації, то він буде знаходиться десь на краю кластера, в результаті чого мережа може легко віднести об’єкт, що класифікується, до невірного класу.

3 РОЗРОБКА МОДИФІКОВАНОЇ МОДЕЛІ РОЗПІЗНАВАННЯ ОБЛИЧ

4.4 Сучасні моделі згорткових нейронних мереж, їх застосування для різних задачі розпізнавання

Моделі сучасних нейронних мереж по задачі застосування діляться на мережі для детекції та мережі для ідентифікації/верифікації. Спочатку розглянемо мережі для детекції об’єктів. З них можна виділити мережі сім’ї R-CNN (R-CNN, Fast R-CNN, Faster R-CNN), мережі YOLO і MTCNN.

Мережі сім’ї R-CNN відносяться до двохетапних методів. Перша модель R-CNN [29] виконує детекцію через класифікацію областей на зображенні, але для оптимізації швидкості своєї роботи, вона спочатку знаходить тільки ті області для подальшої перевірки, в яких найбільш ймовірно знаходиться об’єкт. Алгоритм роботи цієї моделі наведено на рисунку 3.1.

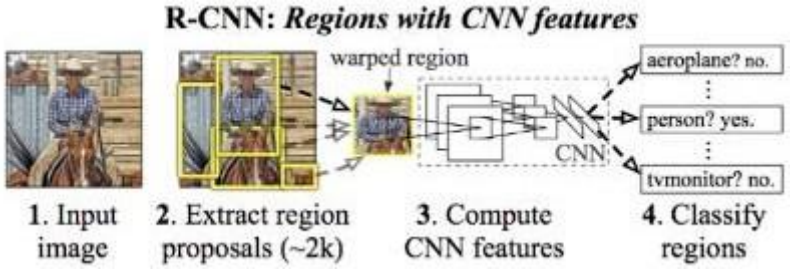


Рисунок 3.1 — Алгоритм роботи R-CNN

Регіони інтересу в цій мережі знаходяться за допомогою методу селективного пошуку. Його суть полягає в тому, щоб виділяти потенційні регіони на зображенні, пов’язаних один з одним за кольорами, відтінками, текстурами, формами. Запропоновані регіони інтересу потім подаються згортковій мережі, яка виділяє ознаки на зображенні, після чого виконується їх класифікація за допомогою методу опорних векторів SVM, а також регресія координат області, уточнюючи місце розташування об’єкта.

Проблемою даної моделі є те, що в ній доводиться виділяти ознаки,

класифікувати і виконувати регресію для кожного регіону інтересу окремо, що уповільнює мережу. Також, класифікація і регресія виконується різними моделями, які потрібно навчати окремо. Це збільшує час навчання і розмірність необхідного сховища даних для параметрів кожного з методів.

Для вирішення даної проблеми, було запропоновано мережу Fast R- CNN, яка є подальшим розвитком R-CNN [22]. Вона працює швидше за рахунок того, що тепер замість того, щоб пропускати кожен регіон інтересу через згорткову мережу окремо, мережі задається все зображення цілком, а запропоновані області просто накладаються на результуючі карти ознак. Також в цій моделі методи класифікації і регресії об’єднані в одну повнозв’язну мережу з двома виходами — для класифікації та регресії. Архітектура мережі Fast R-CNN зображена на рисунку 3.2.

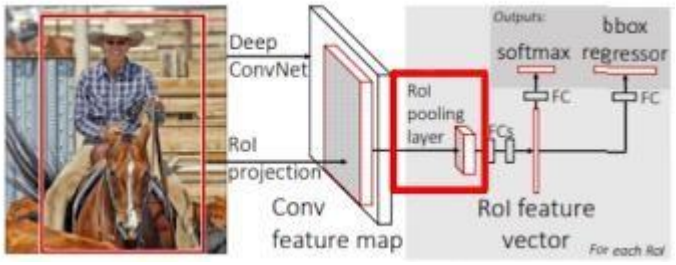


Рисунок 3.2 — Архітектура мережі Fast R-CNN

Надалі мережа Fast R-CNN розвивається в Faster R-CNN [33]. Її поліпшення полягає в тому, що вона повністю відмовляється від методу селективного пошуку для знаходження регіонів інтересу і замість неї використовує згорткову версію ковзного вікна. При чому, для пошуку регіонів інтересу і класифікації об’єктів використовуються окремі згорткові мережі.

Мережа YOLO [24] відноситься до одноетапних методів. Вона дуже схожа на Faster R-CNN тим, що також використовує згорткову версію для детекції, але YOLO знаходить і класифікує всі об’єкти на зображенні за один

прохід по єдиної нейронної мережі.

При згортковому підході ковзного вікна, зображення розбивається на квадратні області, в межах яких мережа повинна знайти центр об’єктів, що детектуються. Але, оскільки, об’єкти можуть перекриватися один одним, центри відразу декількох об’єктів можуть знаходиться в одній і тій же області, що перевіряється, тому мережі, які використовують даний підхід при регресії використовують кілька обмежуючих рамок об’єктів. Вони називаються якорями, оскільки кожна область відповідає за свій конкретний клас об’єкта, що детектується. Такий якірний підхід також реалізований в мережах Faster R- CNN і YOLO.

Ще одним з підходів для детекції об’єктів є каскадний підхід. При такому підході мережі шикуються в каскад, при якому вихід однієї мережі є входом для наступного. При чому, кожна наступна мережа уточнює результат попередньої, оскільки навчається на даних, яка не змогла правильно розпізнати попередня. Прикладом каскадного підходу є мережа MTCNN [25] для детекції обличь. Вона складається з трьох послідовно пов’язаних мереж – P-Net (Proposal Network), R-Net (Refine Network) і O-Net (Output Network). Архітектуру цих мереж представлено на рисунку 3.3.

В роботі MTCNN, зображення спочатку змінюється в розмірах, створюючи так звану піраміду зображень (image pyramid). Потім кожне з цих зображень подається на вхід першій мережі P-Net, з якого вона виконує регресію координат обмежуючих рамок обличь. Знайдені області передаються в мережу R-Net, яка відкидає ті, де обличь точно немає, і передає свої результати мережі O-Net, яка залишає лише одну обмежуючу рамку, а також додатково знаходить п’ять точок на обличчі. Це корисно для вирівнювання обличь за цими точкам, щоб потім здавати їх подальшим більш комплексним системам розпізнавання.

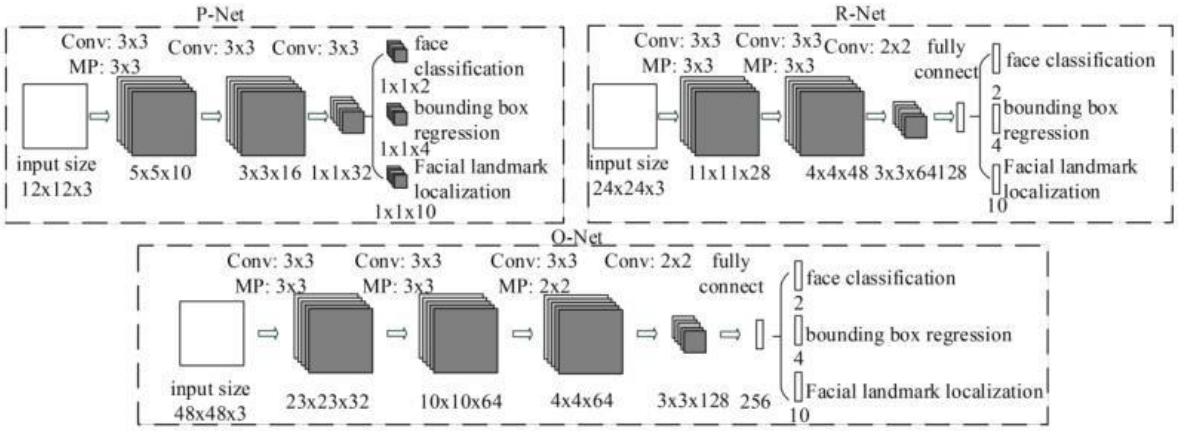


Рисунок 3.3 — Архітектура мережі MTCNN

З огляду на все вищесказане з приводу мереж для детекції, також варто відзначити, що існує залежність між швидкістю роботи нейронної мережі і точністю її розпізнавання. В [26] виконувалося дослідження для того, щоб порівняти одноетапні і двоетапні методи детекції на прикладі деякого набору існуючих мереж для детекції, що використовують різні методи. Все це проводилося на наборі даних COCO [27], що містить 330 тисяч зображень з півтора мільйона об’єктами, кожен з яких відноситься до однієї з 80 категорій. Цей набір був розроблений спеціально для детекції та сегментації об’єктів, які входять до складу представлених в наборі категорій.

Результати дослідження можна побачити на графіку, наведеному на рисунку 3.4. Вісь X позначає час, за яке мережа виконує детекцію в мілісекундах, а вісь Y — середню точністю детекції в процентах. У контексті даного питання детекції за допомогою одноетапних і двоетапних методів, на малюнку головне значення має те, що мережі, які використовують одноетапні методи, працюють швидше, але точність у них гірше, ніж у двоетапних методів, і навпаки.

Мережа RetinaNet, яка була розроблена в результаті даного дослідження, використовує нову функцію помилки, яка називається Focal Loss. Це дало мережі можливість використовувати одноетапні методи, щоб бути такою ж швидкою, але при цьому точність мережі не гірше, ніж у двоетапних мереж,

таких як Faster R-CNN.

$$FL(p,y) = -(1-p_t)^{\gamma}\log(p_t), p_t = \begin{cases} p, y = 1 \\ 1-p, \text{інакше} \end{cases}$$

де p — передбачений мережею результат;

y — очікуваний результат;

γ — гіперпараметр згладжування, чим більше значення, тим більш чутливою буде помилка до прикладів, складних для розпізнавання.

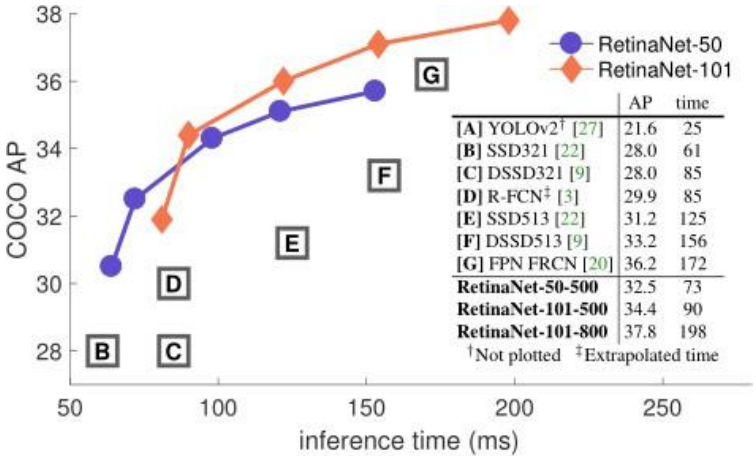


Рисунок 3.4 — Залежність точності детекції від швидкості

Далі розглянемо відому модель мережі для ідентифікації та верифікації осіб — FaceNet. Ця мережа є сіамською мережею, яка кодує два вхідних зображення обличь в вектор ознак, а потім порівнює їх між собою по відстані, щоб зрозуміти наскільки схожі ці два обличчя.

В даному випадку головне завдання — навчити мережу таким чином, щоб для однієї і тієї ж людини нейронна мережа видавала найбільш схожий вектор ознак, а для різних людей — різні вектори. Для цього в даній мережі була розроблена спеціальна функція втрат, звана Triplet Loss. Вона оперує трьома

зображеннями обличь. На двох із зображень знаходиться одна і та ж людина, а на третьому — інша. Одне з двох зображень тієї ж людини називається якорем (anchor), друге береться за позитивний (positive) приклад, а зображення з іншою людиною — за негативний (negative). Формальний опис даної функції втрат наведено нижче.

$$||f(x^a) - f(x^p)||^2 + \alpha < ||f(x^a) - f(x^n)||^2$$

де $f(x^a)$ — вектор ознак, отриманий з якорного зображення;

$f(x^p)$ — вектор ознак, отриманий з позитивного прикладу;

$f(x^n)$ — вектор ознак, отриманий з негативного прикладу;

α — константа, що задає відступ між зображеннями різних людей.

Суть використання цієї функції втрат в тому, щоб навчити нейронну мережу для тієї ж людини видавати близькі по відстані один до одного вектори ознак, а для різних — вектори, що відрізняються на задану константу, яка називається відступом (рис. 3.5). Відступ потрібен, щоб нейронна мережа не намагалася знаходити найбільш тривіальне рішення, роблячи всі вектори однаковими.

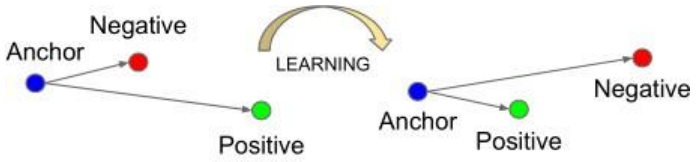


Рисунок 3.5 — Процес навчання при використанні Triplet Loss

При цьому важливо відзначити, що одним з важливих аспектів використання Triplet Loss є те, що в якості негативних прикладів варто підбирати середні по складності, тобто зображення людей повинні бути не дуже

схожі, але і не дуже різнитися. Виходячи з проведених досліджень було виявлено, що таким чином мережа навчається найкраще і не застряє в екстремумах при навчанні методом градієнтного спуску.

Подальшим розвитком мережі FaceNet стала мережа OpenFace [24]. Її розробка була акцентована на тому, щоб дослідити і довести можливість навчання мережі для розпізнавання облич на відкритому наборі даних, на відміну від її попередників, які зазвичай використовували приватні датасета на мільйони зображень. Також ця мережа передбачалася бути використаною для розпізнавання облич на мобільних пристроях, потужність яких не така велика, і при цьому бути все ще досить швидкою, щоб можна було використовувати її в режимі реального часу. Тому структура OpenFace є модифікованою структурою мережі FaceNet зі зменшеною кількістю параметрів для більш швидкого навчання і роботи мережі.

Як уже згадувалося раніше, при вирішенні завдання ідентифікації і верифікації, також використовують метод кодування зображень обличь в вектори ознак через навчання класифікатору. При цьому функція помилки Softmax не дає строгого поділу класів між собою на їх границях. В якості вирішення проблеми пропонувалися досконаліші функції помилки для класифікації в контексті виділення векторів ознак із зображень осіб для їх подальшого розпізнавання. Однією з таких функцій помилок є функція ArcFace, запропонована і випробувана в однойменній нейронній мережі [21].

$$L = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s*\cos\left(\theta_{y_i}\right)-m}}{e^{s*\cos\left(\theta_{y_i}\right)-m}+\sum_{j=1,j\neq y_i}^{C}e^{s*\cos(\theta_j)}}$$

Візуалізація прикладу класифікації при використанні помилки ArcFace в порівнянні з Softmax представлено на рисунку 3.6. Кожен колір і лінія на малюнку зображує свій клас. При цьому можна помітити, що у Softmax границі класів мають дуже розмиті переходи з одного класу в інший, на відміну від

функції ArcFace, де між кожним класом існує досить відстані, при якій важко віднести об’єкт, що класифікується, до невірного класу.

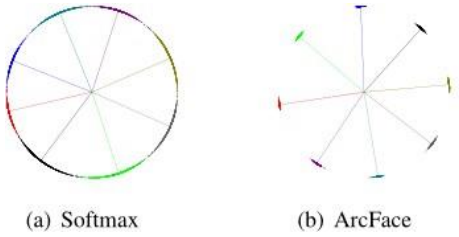


Рисунок 3.6 – Візуалізація класифікації зображень при використанні функцій помилок Softmax і ArcFace

4.5 Модификація системи розпізнавання облич

Під час виконання аналізу існуючих мереж було розглянуто множину різних структур мереж.

У реалізації деяких з них основними пропозиціями виступали не стільки сама архітектура мережі, скільки підхід до її навчання. Наприклад, для мереж, що кодують зображення особи в векторне подання, було виділено два основні методи.

Перший метод полягає у використанні спеціальної функції помилки Triplet Loss, що дозволяє скоротити відстань векторів ознак між зображеннями одного і того ж людини і збільшити відстань між векторами зображень різних людей.

Другий метод пропонує навчити класифікатор з подальшим видаленням останнього шару, що класифікує, в результаті чого виходить мережа, яка вже сама по собі вміє правильно виділяти ознаки.

Відомо, що для кращого навчання мережі при використанні функції помилки Triplet Loss потрібно підбирати середні по складності триплети. Однак, це також є складністю, оскільки їх підбір потрібно робити вручну, і якщо датасета зображень дуже великий, а саме такі датасети потрібні для

кращого навчання мереж, то створення такого датасету стає довгим рутинним процесом.

Як було виявлено, мережа OpenFace, яка є продовженням мережі FaceNet, використовує в якості функації помилки Triplet Loss, і є однією з сучасних мереж для розпізнавання облич.

Тому в якості модифікації мережі для розпізнавання осіб пропонується взяти мережу OpenFace, і навчити її як класифікатор, додавши в кінець цієї мережі додатковий шар, що класифікує.

Структура мережі OpenFace, як і у її попередника FaceNet, має inception архітектуру. Шари inception утворюються з розгалуження шару, коли до нього додаються шари не всі послідовно, а паралельними групами, які вже можуть складатися з послідовних шарів. Після цього inception шар конкатенує всі розгалуження в один цілісний шар. Тому в таблиці 3.1 наведена загальна структура мережі, а на рисунку 3.7 — опис окремих відгалужень, з яких утворюються результуючі inception шари.

Також, крім іншого методу навчання, запропоновано реалізувати модифіковану структуру мережі, а саме — спростити її, але при цьому навчити на досить великому датасеті. Для цього в модифікованої структурі мережі вирішено видалити 4 і 5 групи inception шарів. За аналітичними оцінками таким чином можна домогтися більш швидкого навчання мережі, не погіршуючи якість розпізнавання. На рисунку 3.7 виділено видалені групи шарів в модифікованій мережі.

Запропонована архітектура OpenFace приведена в таблиці 3.1.

Таблиця 3.1 — Загальна структура мережі

№	Тип шару	Опис
1	вхідний шар	кольорове зображення розміром 96x96 пікселів
2	згортка	96 фільтрів розміром 7x7 пікселів з кроком 2x2

3	макс-пулінг	розмір 3x3 пікселів з кроком 2x2
4	згортка	64 фільтрів розміром 1x1 пікселів з кроком 1x1
5	згортка	192 фільтрів розміром 3x3 пікселів з кроком 1x1
6	макс-пулінг	розмір 3x3 пікселів з кроком 2x2
7	inception (3a)	конкатенує 4 відгалуження шарів з попереднього
8	inception (3b)	конкатенує 4 відгалуження шарів з попереднього
9	inception (3c)	конкатенує 3 відгалуження шарів з попереднього
10	inception (4a)	конкатенує 4 відгалуження шарів з попереднього
11	inception (4e)	конкатенує 3 відгалуження шарів з попереднього
12	inception (5a)	конкатенує 3 відгалуження шарів з попереднього
13	inception (5b)	конкатенує 3 відгалуження шарів з попереднього
14	average-пулинг	розмір 3x3 пікселя з кроком 1x1
15	повнозв’язний шар	128 нейронів в шарі
20	L2-регуляризація	

При цьому також важливо відзначити, що в наведених таблицях та рисунку шар згортки у всіх випадках складається з трьох шарів – безпосередньо згортка, батч-нормалізація, а потім активація за допомогою функції активації ReLU.

При додаванні класифікуючого шару в кінець цієї мережі додається новий повнозв’язний шар, кількість нейронів в якому дорівнює кількості класів при навчанні мережі. Кожен клас повинен представляти окрему людину і мережа буде навчатися співвідносити різні зображення облич людей до певного з класів. У свою чергу, кількість класів буде варіюватися відповідно до навчального набору даних, тому немає сенсу ставити заздалегідь визначену кількість класів.

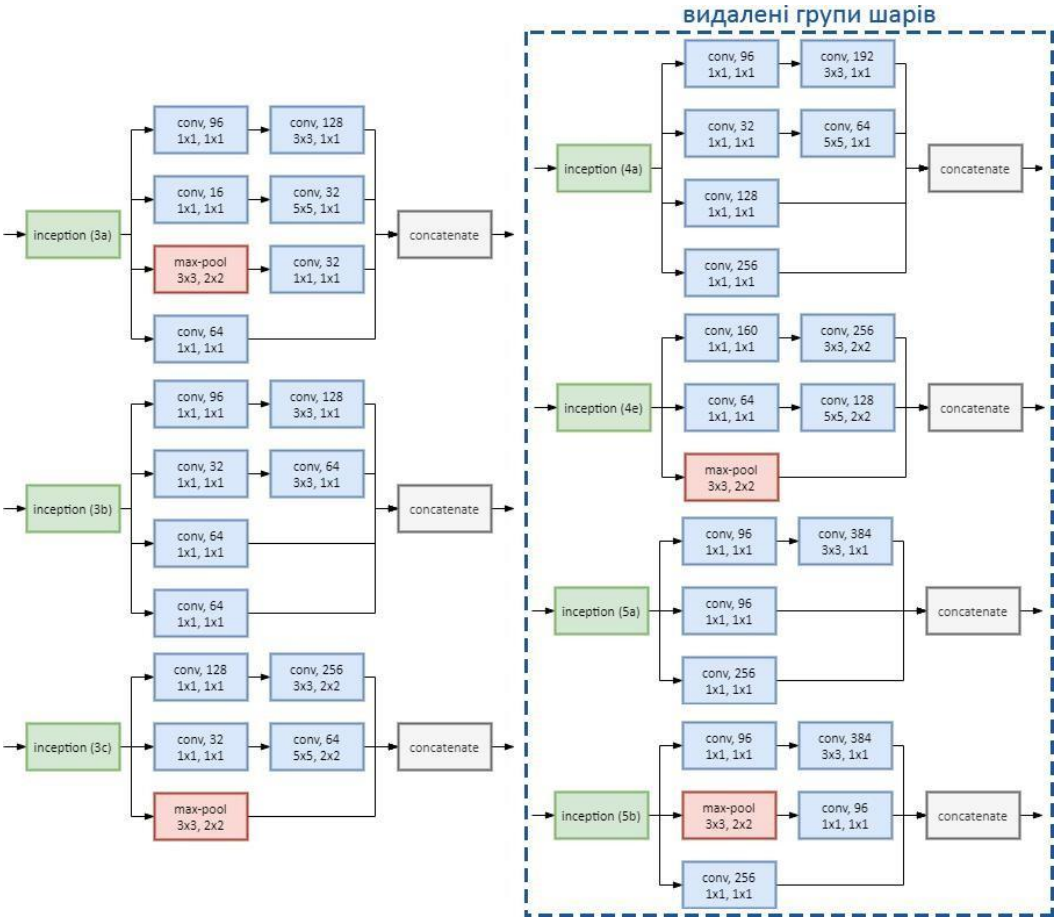


Рисунок 3.7 — Структура inception модулів оригінальної мережі OpenFace та видалена частина шарів в модифікованій версії

4 РЕАЛІЗКЦІЯ МОДИФІКОВАНОЇ МОДЕЛ РОЗПІЗНАВАННЯ ОБЛИЧ

4.1 Підготовка середовища для розробки

Навчання мережі було вибрано виконувати на базі лабораторії ПМІ «ML: Нейронні мережі та машинне навчання». Специфікація робочої станції приведена в таблиці 4.1.

Таблиця 4.1 – Специфікація робочої станції:

Тип	Назва	Специфікація
Назва станції	Комплект науково–дослідного обладнання для лабораторії «Leader-Pro»	Ін. № 1014600338
Процесор	AMD Ryzen Threadripper 2990WX	32 core, 64 treads, 4 channel, 3.0GHz
Відеокарта	MSI GeForce RTX 2080 Ti Gaming Trio	11 GB, GDDR6 1755 MHz / 1350 MHz 4352 cores Підтримка CUDA
Материнська плата	ASUS ROG ZENITH II EXTREME ALPHA	AMD sTRX4, 8 x DDR4, Intel XMP, AMP
Операційна система	Windows 10 Pro	x64

Для машинного навчання часто використовується мова програмування Python. Вона є кросплатформною високорівневою мовою, що спрощує роботу

з ним, а також для нього існує велика кількість готових бібліотек, включаючи бібліотеки, що спрощують виконання математичних розрахунків, бібліотеки роботи з зображеннями, а також бібліотеки для створення і навчання складних архітектур нейронних мереж. З огляду на все це, для реалізації власного модуля штучної нейронної мережі було вирішено використовувати саме цю мову.

При навчанні нейронних мереж необхідно робити дуже багато обчислень і центральні процесори (CPU) справляються з цим набагато повільніше, ніж графічні (GPU). Тому в машинному навчанні зазвичай використовуються GPU і бібліотека TensorFlow [22], що дозволяє безпосередньо взаємодіяти з графічним процесором для виконання обчислень.

Також реалізовувати повністю з нуля комплексні архітектури нейронних мереж є дуже трудомістким завданням, тому для глибокого машинного навчання також існує безліч готових бібліотек для спрощення створення нейронних мереж. Для експериментів було обрано бібліотеку Keras [23]. Вона має простий інтерфейс, який дозволяє зібрати мережу будь-якої складності з готових компонентів – шарів різного типу, безлічі функцій активації, помилок, різних оптимізацій градієнтного спуску, що використовується при навчанні, і безліч інших корисних можливостей. Крім побудови архітектури нейронної мережі, вона також дозволяє навчати її і оцінювати якість навчання по її підсумковій помилці. Сама бібліотека Keras включається в стандартну поставку TensorFlow з версії 2.0.

На додаток до основних бібліотеках для машинного навчання TensorFlow і Keras було вибрано використовувати бібліотеки Numpy, Matplotlib, OpenCV. Numpy є популярною бібліотекою для математичних розрахунків, Matplotlib дозволяє малювати графіки, що корисно при виведенні результатів навчання і їх аналізі, а бібліотека OpenCV корисна при роботі з зображеннями, оскільки вона надає можливість завантажувати зображення будь-якого популярного формату, виводити їх, а також малювати на них , що може знадобитися, наприклад, для малювання обмежувальних рамок осіб на зображеннях.

Для проведення експериментів з параметрами нейронної мережі корисний інтерактивний пакет розробки Jupyter Notebook [44]. Він дає можливість писати код Python в спеціальних блокнотах, де можна запускати його блоками з висновком результатів після кожного з них. При чому все результати зберігаються, щоб потім їх можна було відкрити і подивитися знову, щоб, наприклад, проаналізувати або згадати в якому форматі повертає результат та чи інша функція. Корисною можливістю є ще те, що в ньому можна відразу виводити всі зображення і графіки без необхідності створення цілих віконних додатків для цього. Також блоки можуть використовуватися не тільки для коду, але і для описів тексту в форматі Markdown, який додає можливості повноцінного редактора тексту, дозволяючи використовувати різний розмір шрифту тексту, використовувати різні виділення тексту, такі як жирність або курсив, різний колір тексту, так і в цілому Markdown дозволяє також використовувати всю мову розмітки HTML, додаючи ще більше можливостей форматування. Це корисно, коли потрібно, наприклад, задокументувати, описати, виділити завдання, процес або результати проведених експериментів прямо в ході програмування. Таким чином, блокноти широко використовуються в світі Python-розробки і спрощують життя дослідникам вивчати особливості якихось бібліотек і проводити власні дослідження зі збереженням і описом результатів без необхідності створювати цілі додатки, концентруючи увагу тільки на завданнях досліджень.

В області програмування на Python також широко відоме поняття віртуального середовища. Воно дозволяє створювати для додатків, що розробляються, своє власне, незалежне від інших проектів середовище з виконуваними файлами і бібліотеками. Туди входять сам інтерпретатор Python заданої версії та всі встановлювані через менеджер пакетів Pip бібліотеки. Це все дозволяє легко переносити проекти з однієї машини на іншу, а також спрощує встановлення залежностей, оскільки всі версії встановлених бібліотек залишатимуться тими самими в межах одного середовища, гарантуючи

сумісність версій бібліотек, інших залежностей та інтерпретатора Python між собою.

Таким чином, було виконано процес налаштування віртуального середовища та встановлення всіх необхідних бібліотек Python для подальшої роботи, команди якого наведено на рисунку 3.1. Для роботи було використано версію Python 3.7.2.

```
# Перехід у директорію проекту
cd D:/Projects/face-recognition-cnn

# Створення та активація віртуального оточення у поточній директорії
python -m venv .
scripts/activate

# Встановлення залежностей через Pip
pip install tensorflow==2.4.1
pip install Keras==2.4.3
pip install numpy==1.19.5
pip install matplotlib==3.4.0
pip install jupyterlab==3.0.12
pip install opencv-python==4.5.1.48
```

Рисунок 4.1 — Команди процесу налаштування середовища розробки

Бібліотека TensorFlow для роботи з GPU вимагає бібліотеку CUDA, яка використовується в роботі з графічними відеокартами NVIDIA, тому для роботи з цією бібліотекою було вирішено використовувати комп’ютери, оснащені саме цими відеокартами. Для того, щоб TensorFlow міг використовувати GPU за допомогою CUDA, також було необхідно встановити відповідні бібліотеки – CUDA Toolkit та cuDNN, інакше під час виконання програми, яка використовує Tensorflow, будуть виводитися попередження та помилки про те, що він не зміг знайти відповідні DLL бібліотеки та замість GPU використовуватиме CPU.

Інсталятор CUDA Toolkit надається безкоштовно і його можна завантажити з офіційного сайту NVIDIA [45]. На сторінці завантаження необхідно вибрати операційну систему (ОС), версію ОС та тип інсталятора. У

цьому випадку було вибрано ОС Windows версії 10. NVIDIA підтримує лише x64 архітектуру процесора і оскільки робоча станція відповідає цій вимогі, було обрано саме цю архітектуру. Тип інсталятора надає вибір між локальним інсталятором і через мережу Інтернет. У першому випадку CUDA Toolkit містить всі необхідні компоненти для установки, а в другому – вони будуть завантажуватися вже самим установником в процесі його роботи. Як тип інсталятора було вибрано локальний тип, щоб у разі розриву Інтернет-з'єднання можна було спокійно продовжити інсталяцію. Процес установки CUDA наведено рисунку 4.2.

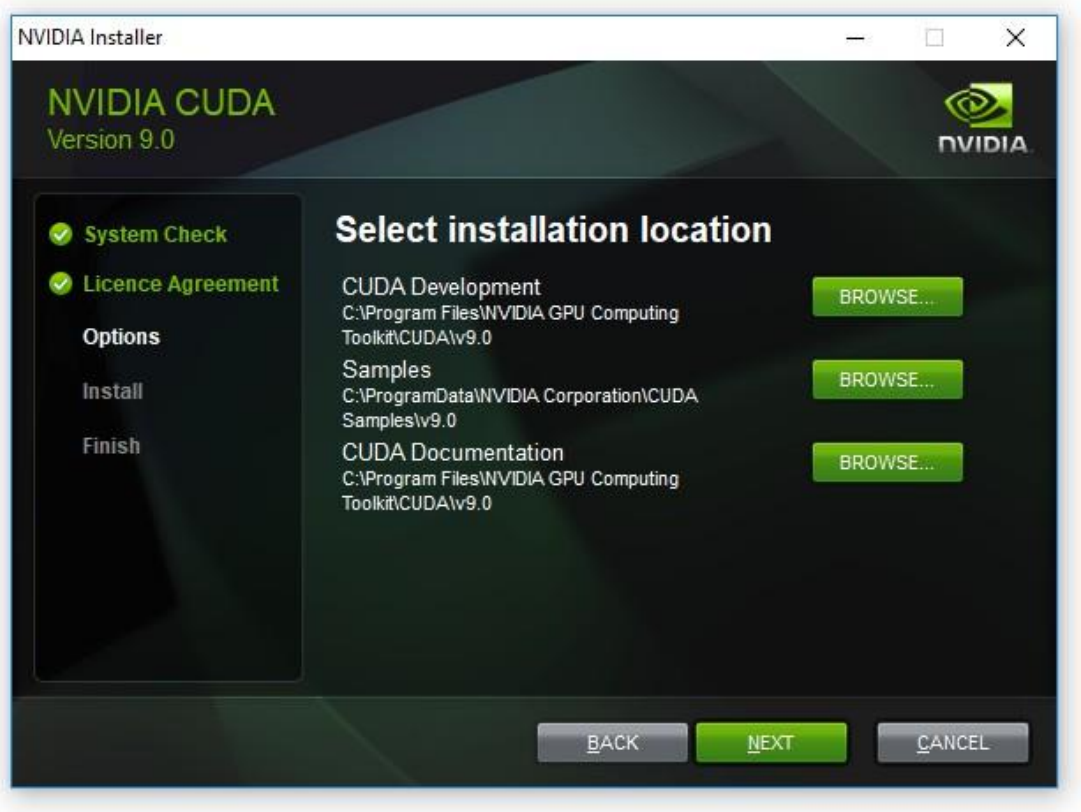


Рисунок 4.2 — Процес установки CUDA

Бібліотека cuDNN працює безпосередньо разом із CUDA, але її необхідно встановлювати окремо від неї. Її також було завантажено з офіційного сайту CUDA у вигляді архіву з файлами C++ та DLL, які потрібно було вручну перенести у відповідні підкаталоги у директорії із встановленим CUDA. Перед

завантаженням та вибором версії cuDNN також було виконано перевірку сумісності версій за допомогою команди наведеної на рисунку 4.3.

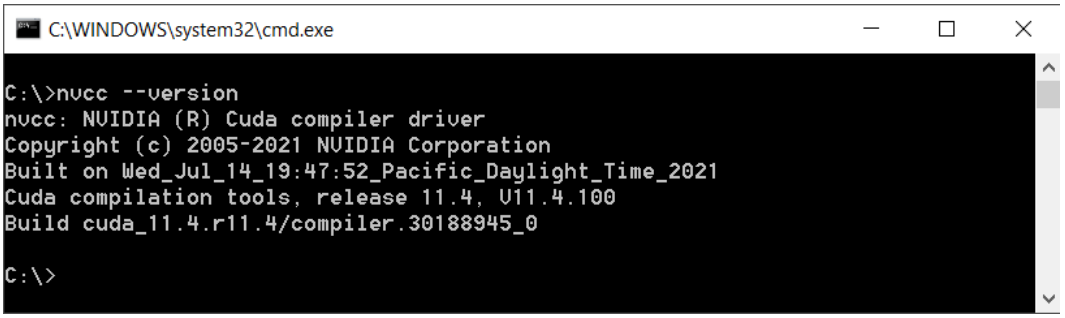


Рисунок 4.3 — Перевірка версії встановленого CUDA

Крім того, TensorFlow працює з поняттям тензорів. Вони визначаються розмірністю і зазвичай використовуються розмірності від 0 до 3. Наприклад, 0D-тензор – це число, 1D – вектор чисел, 2D – матриця [26]. Саме розрахунки з цими тензорами і виконуються на процесорі. Для прискорення і поліпшення точності при роботі з тензорами Google розробила власні процесори, звані тензорними (TPU), і дозволяє експериментувати з ними на своєму сервісі Google Colab, де також можна використовувати і GPU.

Сервіс Google Colab [27] надає дослідникам безкоштовні ресурси для обчислень з попередньо встановленими TensorFlow, Keras і всіма іншими бібліотеками, які часто застосовуються для машинного навчання і аналізу даних. Цей сервіс представлений у вигляді Jupyter Notebook в браузері, який дозволяє виконувати власний код Python. Самі екземпляри Jupyter Notebook запускаються у віртуальних машинах з системою Linux і крім коду в блоках Notebook можна вводити більшість доступних в Bash команд, наприклад, для переходу по каталогам системи або, щоб при необхідності доустановити будь-які додаткові бібліотеки через менеджер пакетів, які не встановлені спочатку. Лімітами Colab є кількість використовуваної пам’яті, час життя процесу Python, але Google не публікує чіткий перелік лімітів, так як він заявляє, що вони динамічні і весь час змінюються в залежності від потреб користувачів і

навантаження на сервер. Тим не менш, це відмінний інструмент, щоб експериментувати і навчати власну мережу на продуктивних серверах з сучасними процесорами, розробленими спеціально для машинного навчання. Вигляд інтерфейсу Google Colab із прикладами процесу роботи наведено на рисунку 4.4.

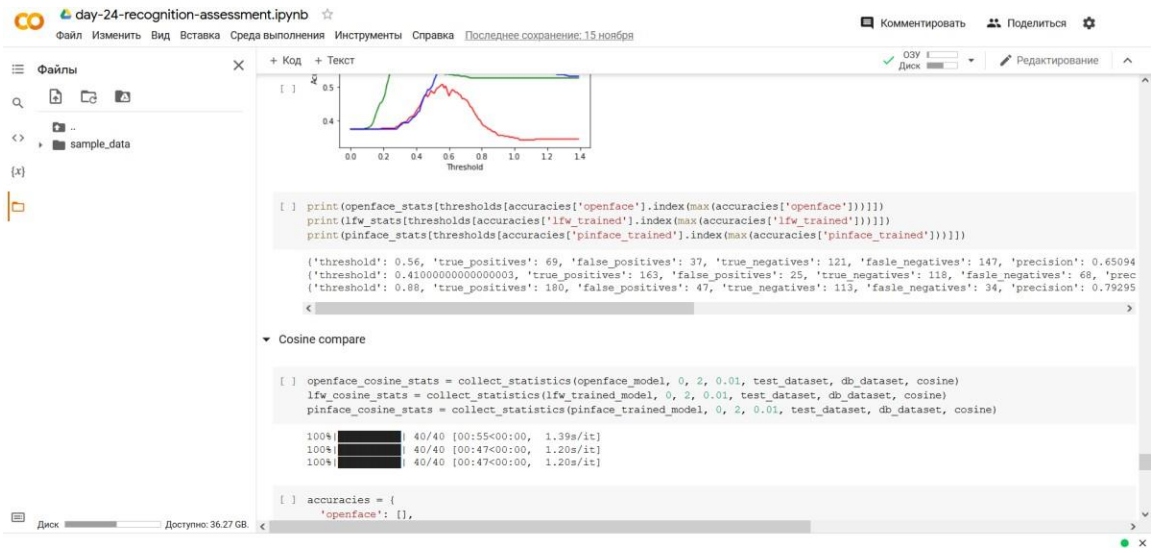


Рисунок 4.4 — Вигляд інтерфейсу Google Colab

Для написання кінцевого коду для використання нейронної мережі як середовища розробки було вибрано безкоштовний, легкий, але розширюваний редактор вихідного коду від Miscrosoft – Visual Studio Code [48]. Він підтримує величезну кількість мов, в тому числі і Python у вигляді спеціального плагіна, який і був встановлений для подальшої роботи. Це дозволяє використовувати редактор як повноцінний IDE, який підтримує автодоповнення коду і налагодження.

4.2 Визначення набору навчальних та тестових даних

Для навчання нейронних мереж для розпізнавання обличь потрібні великі набори зображень. При цьому вибірку розбивають на навчальну і тестову. На навчальній вибірці нейронна мережа навчається, а приклади з тестового набору є для мережі новими, тобто тими, які вона ще не бачила, тому її

використовують для тестування якості роботи мережі. З навчальної вибірки часто також виділяють певну частину прикладів для валідації під час навчання. Безпосередньо на вибірці валідації мережу не навчають, але після кожної епохи навчання на ній перевіряють помилку мережі. Якщо помилка на навчальній вибірці продовжує падати, а на вибірці валідації в якийсь момент помилка починає рости, це є ознакою того, що пора закінчувати навчання, інакше мережа може перенавчитися.

Можна створювати свій власний набір даних, але це дуже довго і витратно, тим більше, що для того, щоб дійсно добре навчити нейронну мережу, набір для навчання мережі повинен бути дуже великим, різноманітним, а також збалансованим за всіма характеристиками прикладів. Тому зазвичай використовують вже готові набори, на яких навчають, тестують і оцінюють моделі нейронних мереж. Всі ці набори знаходяться в публічному доступі, деякі з них є умовними стандартами, на яких часто порівнюють існуючі моделі мереж один з одним для отримання більш правильного показника ефективності, оскільки якість розпізнавання сильно залежить від навчального набору даних і не зовсім правильно порівнювати мережі, навчені на різних наборах даних.

Також перевагою використання готових наборів є те, що деякі з них спеціально створювалися для того, щоб ускладнити розпізнавання осіб за допомогою нейронних мереж, додаючи різного роду спотворення і перекриття особи. Якщо навчати нейронну мережу на таких наборах, вона повинна навчитися краще розпізнавати обличчя, звертаючи увагу лише на їх дійсно значущі елементи, що відрізняють одну особу від іншої.

Існує величезна кількість наборів даних для різних цілей, а також набори, створені спеціально для тестування моделей розпізнавання осіб. При розробці власної системи розпізнаванн було розглянуто та використано набори описані нижче.

Набір Labeled Faces in the Wild (LFW) [49] складається з 13233 зображень 5749 осіб. Він був отриманий з фотографій людей, доступних в мережі

Інтернет, за допомогою детектора осіб, заснованому на OpenCV реалізації методу Віоли-Джонса. Всі зображення є кольоровими, мають розмір 255x255 пікселів і містять область навколо голови людини. На кожну людину припадає різна кількість зображень, від 1 до 530. Файли зображень мають формат JPG і розподілені по 5749 директоріям, кожна з яких визначає конкретну людину і має назву відповідну імені та прізвищу цієї людини розділених між собою через знак підкреслення. Усередині директорій, ім'я кожного файлу відповідає імені директорії з доповненням номера зображення в цій директорії, доданий через знак підкреслення. Приклад фотографій облич із датасету LFW наведено рисунку 4.5.

Рисунок 4.5 — Приклади зображень із датасету LFW

Набір PinsFace [21] складається із зображень осіб знаменитостей, зібраний з мережі Інтернет. Всього в датасеті 17534 зображень, які можна класифікувати між 105 людьми. Всі зображення також кольорові і містять особи в різних кутах, освітлення, емоціях, макіяжі або без нього. У датасеті всього 105 директорій, кожна з яких містить більше сотні файлів із зображеннями облич людей, відповідним директоріям. Назви директорій відповідають іменам людей, а кожен файл всередині директорій має назву схожу з директорією, в якій знаходиться, з додаванням номера зображення в цій

директорії. Приклади зображень із датасету PinsFace наведено на рисунку 4.6.

Рисунок 4.6 — Приклади зображень із датасету PinsFace

Набір ORL Faces [22] містить набір зображень облич, зроблених в лабораторних умовах. У ньому є десять різних зображень кожного з 40 різних людей. Для деяких людей зображення були зроблені в різний час, варіюючи освітлення, вираз обличчя (відкриті/закриті очі, посміхається/не посміхається) і деталі особи (окуляри/без окулярів). Всі зображення були зроблені на темному однорідному фоні, коли випробовувані перебували у вертикальному фронтальному положенні з допуском на деякий бічний рух. Файли мають формат PGM і розмір кожного зображення в наборі дорівнює 92x112 пікселям, при цьому всі зображення в датасеті чорно-білі. Зображення організовані в 40 каталогів по одному для кожної людини, які мають імена в формі sX, де X вказує номер людини від 1 до 40. У кожному з цих каталогів є десять різних зображень цього об’єкта, які мають імена в формі Y.pgm, де Y — номер зображення для цього об’єкта від 1 до 10. Приклади зображень з датасету ORL Faces наведено на рисунку 4.7.

Рисунок 4.7 — Приклади зображень із датасету ORL Faces

Вищенаведені набори зображень можна використовувати для навчання і тестування власної системи розпізнавання обличь. Для збільшення розміру навчального набору зображень також існує метод, званий аугментація даних (data augmentation) [24]. Його суть полягає в додаванні нових зображень на основі вже наявних за рахунок різних трансформацій, додаванні різного роду ефектів. Наприклад, можна відобразити зображення по вертикалі, збільшити або зменшити його в розмірах, додати розмиття, шуми. Також можна змінити яскравість, контрастність, колірний баланс зображення. Останнє може добре допомогти з розпізнаванням осіб при висвітленні будь-яким кольором, відмінного від денного. Якщо застосовувати зрушення і, наприклад, перемістити зображення в сторону на якусь частину його розміру, можливо, вдасться домогтися розпізнавання по половині обличчя.

Додавання зображень з використанням аугментації даних може виконуватися прямо перед навчанням в програмі за допомогою будь-яких бібліотек, тому цей спосіб є дуже ефективним з точки зору витрат за часом, оскільки не потрібно пробляти це все вручну.

4.3 Визначення алгоритмів оцінки мережі для розпізнавання облич

Для оцінки систем розпізнавання існує набір метрик, які ґрунтуються на матриці заплутаності (confusion matrix) [20]. Вона має вигляд, наведений в таблиці 4.2. Колонки в таблиці означають істинні значення, а рядки – передбачені мережею.

Таблиця .2 — Матриця заплутаності

	Позитивні	Негативні
Позитивні	Істинно позитивні (True Positive, TP)	Помилково позитивні (False Positive, FP)
Негативні	Помилково негативні (False Negative, FN)	Істинно негативні (True Negative, TN)

В контексті розпізнавання осіб з певною базою даних із зображеннями облич набору людей, ця матриця буде інтерпретуватися наступним чином:

— істинно позитивні приклади позначають зображення, які були правильно співвіднесені з людьми, які знаходяться в БД, тобто співвіднесення вірно;

— помилково позитивні приклади позначають зображення, які були співвіднесені з будь-якими людьми з БД, але насправді ці зображення до них не належать, тобто співвіднесення невірно;

— істинно негативні приклади позначають зображення, які не були співвіднесені з будь-якими людьми з БД, тому що в дійсності вони до них не належать, тобто співвіднесення вірно;

— помилково негативні приклади позначають зображення, які не були співвіднесені з будь-якими людьми з БД, але насправді вони до них відносяться, тобто співвіднесення невірно.

Перед оцінюванням якості розпізнавання мережі тестовий датасет розбивається на дві частини: одна частина зображень буде відходити до бази даних людей, до яких потрібно буде співведнести або неспівведнести

зображення, що перевіряються, а друга частина буде безпосередньо складати ці зображення, які будуть перевірятися на відповідність будь-якого з людей в базі даних або невідповідність нікому з них.

Потім, в процесі оцінювання виконується підрахунок кількості зображень, які потрапили в одну з груп матриці заплутаності. З цієї кількості в подальшому розраховуються метрики, які дають більш високорівневе уявлення про якість розпізнавання. Одними із самих основних і найбільш часто використовуваних метрик є метрики точності, чуйності, випадання і акуратності.

Точність (precision) — це частка виявлених зображень, квадратна міра якої відповідає потребам користувача. Це додатково називається надійністю або повторюваністю і полягає в тому, що вимірювання, які повторюються в незмінних умовах, показують еквівалентні результати. У бінарній класифікації точність також називається позитивним прогностичним значенням. Розраховується точність за формулою:

$$Precision = \frac{TP}{TP+FP}$$

.

Чуйність (recall) — це відсоток позитивних випадків, які були правильно ідентифіковані. Це частка успішно виявлених релевантних

Після визначення метрик, за якими буде виконуватися оцінка навчених моделей і їх порівняння, необхідно вибрати тестовий датасет для цього процесу. Так як потрібно порівнювати кожне зображення всіх людей з кожним зображенням інших людей, доведеться виконувати дуже багато операцій. Тому було вирішено вибрати невеликий датасет і виконувати тестування і оцінку саме на ньому. Таким датасетом став набір ORL Faces. Він містить всього сорок різних людей, у кожного по десять зображень. Для різних людей фотографії були зроблені в різних умовах – при різному освітленні, часу зйомки, повороті голови, виразу обличчя, а також при наявності або відсутності додаткових аксесуарів таких як окуляри. Оскільки розмір цього датасета досить малий, на ньому не варто навчати мережу для розпізнавання осіб, якщо потрібно досягти хороших показників, але завдяки його невеликому розміру, він відмінно підійде для тестування мережі.

Таким чином, цей датасет було вирішено випадковим чином розбити на дві частини: базу даних і тестові зображення з ймовірністю попадання людини в базу даних, що дорівнює 50%. При чому, в базу потрапляє тільки одне зображення людини, всі інші автоматично потрапляють в тестовий датасет. Тому в базі даних з обличчями людей присутнє тільки по одному зображеннюна людину і мережа повинна співвіднести або неспіввіднести кожне тестове зображення з відповідним зображенням в базі даних.

Псевдокод алгоритму генерації тестового датасету та датасету бази даних представлено на рисунку 4.8.

Оскільки після видалення останнього шару, що класифікує, мережа видає вектор ознак, такі вектори, отримані з різних зображень потрібно порівняти. Існує два основних методи для порівняння векторів ознак: обчислення Евклідової відстані або порівняння по косинусу кута між векторами.

```
Початок
  Дано шлях до директорії датасету;
  Дано вірогідність попадання в базу даних;
  список піддиректорій = завантажити із шляху до директорії датасету
  тестовий датасет = пустий ассоціативний масив;
  датасет БД = пустий ассоціативний масив;
  Для кожного імені людини из списка піддиректорій:
    тестовий датасет[ім'я людини] = пустий масив;
    випадкове число = згенерувати випадкове число в межах [0; 1);
    Якщо випадкове число < вірогідності попадання в базу даних:
      додати в базу даних = True;
      датасет БД[ім'я людини] = пустий масив;
    Інакше:
      додати в базу даних = False;
    Для кожного зображення в піддиректорії імені людини:
      Якщо додати в базу даних:
        додати зображення в масив датасет БД[ім'я людини];
        додати в базу даних = False;
      Інакше:
        додати зображення в масив тестовий датасет[ім'я людини];
Кінець
```

Рисунок 4.8 — Алгоритм генерації датасетів для оцінки мереж

Псевдокод изагального алгоритму оцінки мережі наведено далі, на рисунку 4.9.

```
Початок
  Дано тестовий датасет, датасет бази даних, поріг;
  True Positives = 0, False Positives = 0, True Negatives = 0, False Negatives = 0;
  Для кожного імені людини testPerson із тестового датасету:
    Для кожного зображення testImage із тестового датасету[testPerson]:
      відстані = пустий масив;
      testVector = розрахувати вектор признаків з testImage;
      Для кожного імені людини dbPerson із датасету БД:
        Для кожного зображення dbImage із датасету БД[dbPerson]:
          dbVector = розрахувати вектор признаків из dbImage;
          відстань = Евклідова відстань між testVector і dbVector;
          додати кортеж (відстань, ім'я dbPerson) в масив відстаней;
      відсортувати масив відстаней за зростанням;
      мінімальна відстань = перший елемент масива відстаней;
      ім'я людини dbPerson = перший елемент масива відстаней;
      Якщо мінімальна відстань < поріга:
        Якщо testPerson и dbPerson це одна й та сама людина:
          True Positives = True Positves + 1;
        Інакше:
          False Positives = False Positives + 1;
      Інакше:
        Якщо ім'я людини testPerson відсутня в датасеті бази даних:
          True Negatives = True Negatives + 1;
        Інакше:
          False Negatives = False Negatives + 1;
Кінець
```

Рисунок 4.9 — Алгоритм тестування розпізнавання згорткової мережі та підрахунку елементів матриці заплутаності

Після отримання значення метрик схожості між векторами потрібно прийняти рішення про те, чи є отримане значення досить маленьким, щоб

прийняти ці два вектори як такі, що описують одну і ту ж людину. Для цього вводиться додатковий гіперпараметр алгоритму порівняння – поріг. Для кожної моделі мережі розпізнавання його потрібно підбирати індивідуально експериментальним чином. Метою підбору даного параметра є підібрати його так, щоб отримані результати метрик давали найкращий результат для моделі. Порівняння в ньому виконується по Евклідовій відстані, але замість нього може бути використаний будь-який інший метод порівняння двох векторів.

4.4 Проєктування класів для навчання та тестування мережі

Структура програмної частини з розробки, навчання та оцінювання нейронної мережі для розпізнавання облич відповідає діаграмі класів, наведених на рисунках 4.10, 4.11. На першому рисунку представлено архітектуру базових класів, що задіяні в алгоритмі навчання, а на другому — безпосередньо архітектура тієї частини, що стосується алгоритмів навчання та оцінювання мережі.

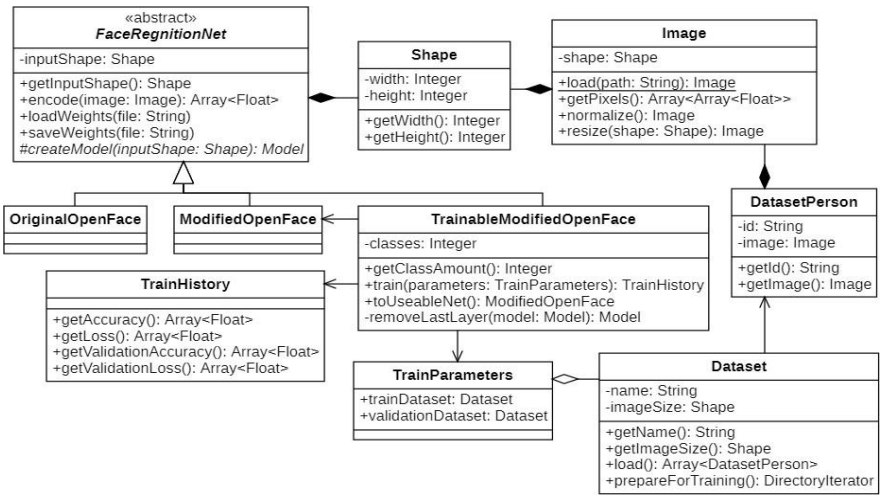


Рисунок 4.10 — Діаграма загальних класів та класів для реалізації навчання модифікованої мережі розпізнавання облич

Одним із головних класів є клас FaceRecognitionNet. Він є абстрактним класом оскільки його мета – надати інтерфейс нейронної мережі для розпізнавання осіб для класів, які будуть успадковані від нього, а також спростити їх, виділивши загальну реалізацію. Такими класами-спадкоємцями є

реалізація оригінальної мережі OpenFace та модифікованої мережі. Таким чином, за допомогою поліморфізму досягається можливість використовувати будь-яку реалізацію класу мережі для розпізнавання без зміни самого алгоритму.

Клас FaceRecognitionNet містить такі поля та методи:

— поле inputShape зберігає розмір зображення, яке має подаватися на вхід мережі і може бути встановлено фіксовано в спадкоємці класу або, наприклад делегуватися ззовні як параметр через конструктор або сеттер спадкоємця, тобто бути динамічним;

— метод getInputShape повертає значення розміру зображення, встановлене в полі inputShape;

— метод encode приймає на вхід зображення, що є фотографією обличчя будь-якої людини, яку необхідно розпізнати, і кодує це зображення у вектор ознак, який потім повертається з цього методу у вигляді масиву з числами речовинного типу;

— метод loadWeights приймає на вхід рядкове значення, що є шляхом до файлу, з якого завантажує ваги в модель нейронної мережі для їх подальшого використання, замінюючи поточні;

— метод saveWeights приймає на вхід рядкове значення, що є шляхом до файлу, в який зберігає поточні ваги моделі нейронної мережі;

— метод createModel є абстрактним методом, який має бути реалізовано у спадкоємцях і має наступний опис: приймає на вхід розмір зображень, який використовуватиметься в моделі нейронної мережі; конструює і повертає сам екземпляр цієї моделі, модель представлена у вигляді класу Model із бібліотеки Keras;цей метод буде використаний виключно всередині класу FaceRecognitionNet для внутрішнього зберігання та делегування необхідних операцій.

Для того, щоб отримувати інформацію про елементи датасету у вигляді, більш придатному для використання, наприклад, вже при розпізнаванні, є клас DatasetPerson.

Далі, на рисунку 4.11 зображено діаграму класів для алгоритмів розпізнавання та оцінювання роботи мереж. Частина цих класів вже було описано вище, тому нижче було описано інші класи.

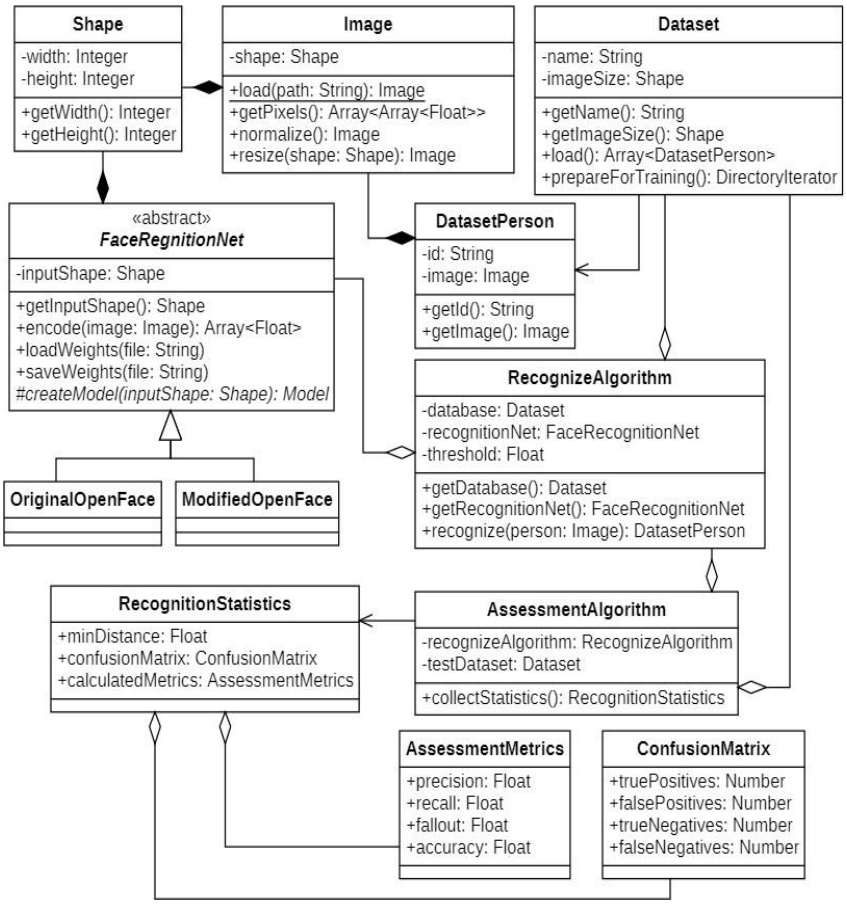


Рисунок 4.11 — Діаграма класів для реалізації алгоритмів розпізнавання облич та оцінювання нейронної мережі

4.5 Програмна реалізація та навчання системи розпізнавання облич

Під час програмування нейронної мережі для розпізнавання облич, основна розробка проводилась в Google Colab, оскільки даний сервіс вже має попередньо встановлені бібліотеки для машинного навчання. Після програмування реалізації та проведення невеликих експериментів, що дозволяють переконатися, що не виникає ніяких помилок під час виконання коду Python, подальше навчання проводилося на власному комп’ютері.

Навчання проводилося два рази на двох різних датасетах: PinFaces і LFW.

Всього процес навчання мережі зайняв два дні. Один день було витрачено на навчання на датасеті PinFaces, які мають всього 105 класів, а ще день – на навчання на датасеті LFW, який має 5749 класів. Незважаючи на таку різницю в кількості класів між цими двома датасетами, вони мають приблизно однакову загальну кількість зображень, тому навчання на кожному з них зайняло приблизно однаковий час. Додатково кожен датасет розширювався виконанням аугментації за допомогою Keras. Зображення зсувалися в сторони в межах 10%, поверталися в межах 40 градусів, масштабувати в межах 20%, а також дзеркально відображалися по вертикалі. Це все виконувалося автоматично випадковим чином спеціальним завантажувачем даних Keras. Він потрібен для того, щоб завантажувати зображення з диска поступово оскільки великі датасети можуть не вміститися в оперативній пам’яті, а також він виконує внутрішні оптимізації для прискорення навчання в TensorFlow.

При навчанні на кожному з датасетів, зображення випадковим чином поділялися на навчальну вибірку і вибірку валідації, де 80% зображень відходило до вибірки для навчання, а 20% – для валідації. За допомогою спеціальних функцій Keras було налаштовано автоматичне збереження ваг при досягненні кращого показника точності на вибірці валідації після завершення кожної епохи. Якщо результат погіршився або не змінився, ваги на черговий епосі навчання не зберігаються.

Тому сигналом про зупинку навчання під час цього процесу навчання послужив показник досягнення найкращої точності на вибірці валідації. Якщо при навчанні цей показник не поліпшується досить велику кількість епох поспіль, значить навчання зупинялося. При цьому за замовчуванням для навчання виставлялося 500 епох. Якщо було видно, що точність ще поліпшується, мережа навчалася ще 500 епох, до тих пір поки не буде видно, що навчання пора припиняти. Всього для навчання на датасеті LFW знадобилося 794 епохи, а на PinFaces – 906 епох, домігшись точності на вибірках валідації – 97% і 98% відповідно.

На рисунку 4.12 представлено динаміку зміни втрати на датасеті валідації

для набору LFW. Як можна побачити, помилка змінювалася досить рівно протягом усього навчання і при зупинці мережа досягла рівня помилки, близької до 0,25%.

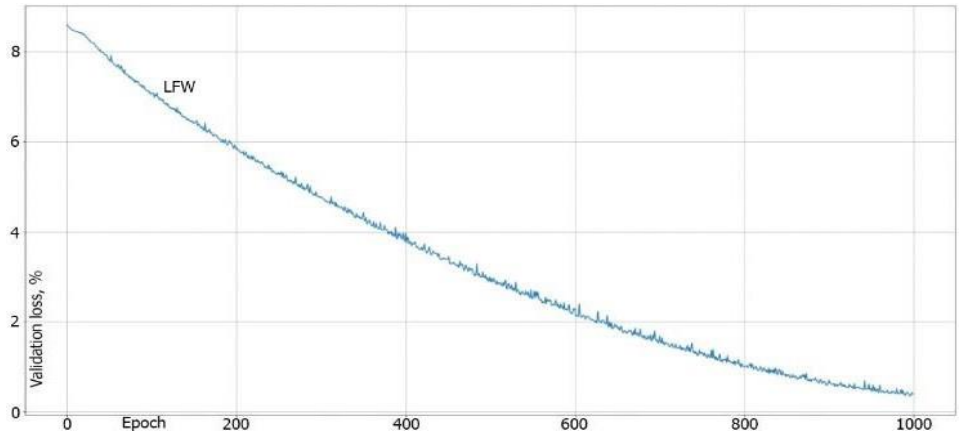


Рисунок 4.12 — Валідація на датасеті LFW

Динаміку зміни помилки на датасеті валідації для набору PinsFace представлено на рисунку 4.13. PinsFace має досить невелику кількість класів у порівнянні з LFW, тому на графіку можна помітити, що мережа досягла маленького відсотка помилки швидше – приблизно на рівні 300 епох і далі залишалася приблизно на одному рівні.

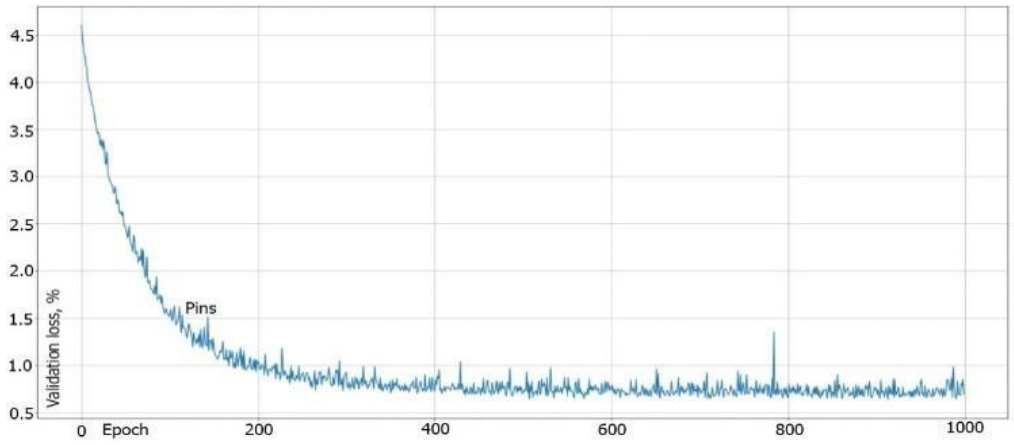


Рисунок 4.13 — Валідація на датасеті PinsFace

Такі ж висновки можна зробити і з графіків з динамікою зміни значення акуратності класифікації для використаних датасетів, представлених на рисунку 4.14. Мережа під час навчання на PinsFace досягає акуратності, близької до 100%, майже вдвічі швидше, ніж LFW. Звідси можна дійти до висновку, що для навчання на PinsFace необхідно близько 500 епох, а LFW достатнім буде 850-900 епох.

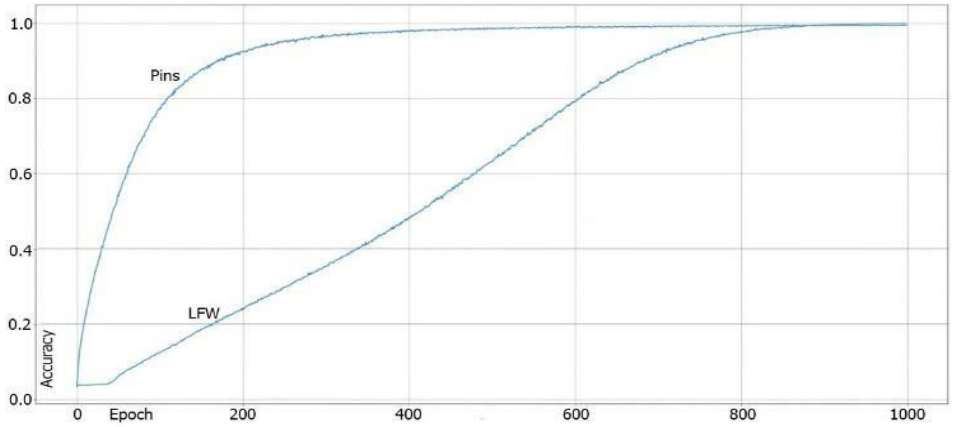


Рисунок 4.14 — Точність навчання на використаних датасетах

4.6 Оцінка розробленої системи розпізнавання облич

Під час оцінки крім двох навчених моделей також оцінювалася оригінальна заздалегідь навчена мережа OpenFace. Її вихідний код був знайдений на GitHub [25]. При чому варто також відзначити, що спочатку вона була написана за допомогою бібліотеки Pytorch [56], яка є аналогом Tensorflow і також використовується для машинного навчання. Pytorch безпосередньо несумісний з бібліотекою Keras, на якій виконувалася розробка та навчання власної мережі, тому використовувалася її конвертована в Keras версія з тими ж вагами.

Під час оцінки мереж підбирати поріг можна вручну, але це довго і можна випадково упустити потрібне значення. Тому його знаходження було вирішено реалізувати шляхом грубого перебору в циклі. Першими ручними експериментами було визначено можливі значення порога, тобто в якому приблизному діапазоні його шукати. З’ясувалося, що потрібне значення може

бути як менше нуля, так і більше одиниці, але не більше двох. Тому було обрано шукати найкращі граничні значення в діапазоні [0; 2) з кроком 0.01. Таким чином, провівши експерименти і домігшись найкращих параметрів алгоритму порівняння для кожної з моделей, було отримано результати, описані далі.

На рисунку 4.15 наведено графік залежності акуратності розпізнавання від порогового значення для кожної з мереж, отримані при порівнянні векторів ознак за допомогою Евклідової відстані. З отриманого графіка добре видно, що акуратність розпізнавання при найкращому порозі для модифікованої мережі вища, ніж оригінальна. Різниця між двома навченими мережами невелика, хоча можна помітити різницю у формі графіків. Графік мережі, яка навчалася на датасеті LFW вужчий, коли графік для мережі, навченої на PinsFace має ширшу форму. Це означає, що якість розпізнавання менше залежить від порогу для PinsFace, де кількість класів менше. Звідси можна дійти до висновку, що залежність якості розпізнавання від порога корелює з кількістю класів, тобто чим менше класів — тим менше залежність.

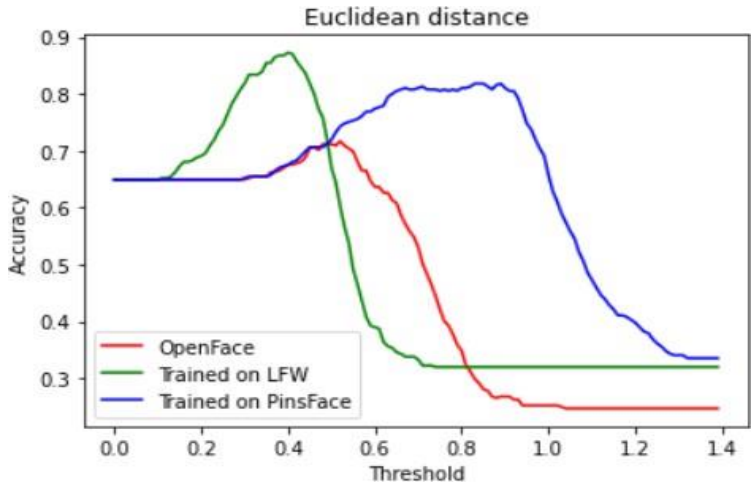


Рисунок 4.15 — Залежність акуратності розпізнавання від порога порівняння векторів по Евклідовій відстані

Далі для кожної мережі було розраховано весь набір метрик у точці, що відповідає максимальній акуратності. Отримані значення наведено в таблиці 4.3. Зліва результати оригінальної мережі OpenFace, посередині —

модифікованої мережі, навченої на датасеті LFW, а справа — модифікованої мережі, навченої на датасеті PinFaces.

Для кожної моделі на першому рядку представлений параметр порогу, при якому вектори повинні вважатися такими, що описують одну і ту ж людину. Для кожної моделі він є різним оскільки кожна модель навчалася за допомогою різних методів і/або на різних датасетах, тому підбирати його потрібно для кожної окремо. Далі перераховані значення категорій матриці заплутаності, після чого результати метрик точності, чуйності, випадання і акуратності відповідно.

Таблиця 4.3 — Порівняння найкращих результатів моделей при порівнянні векторів за допомогою Евклідової відстані

	Оригінальна OpenFace	Навчена на LFW	Навчена на PinFaces
Поріг	0.52	0.4	0.83
Число TP прикладів	34	104	102
Число FP прикладів	10	21	41
Число TN прикладів	242	232	213
Число FN прикладів	99	28	29
Точність	77.27%	83.2%	71.33%
Чуйність	25.56%	78.79%	77.86%
Випадання	96.03%	91.7%	83.86%
Акуратність	71.69%	87.27%	81.81%

Як можна побачити при навчанні власних моделей вдалося підвищити загальну акуратність розпізнавання приблизно на 10-16%. При чому мережа, яка навчалася на датасеті LFW дала на 6% кращий результат, ніж при навчанніна PinFaces. Можливо це пов’язано з тим, що LFW має більшу кількість класів, а яка навчалася на датасеті LFW дала на 6% кращий результат, ніж при навчанніна PinFaces. Можливо це пов’язано з тим, що LFW має більшу кількість класів, а саме 5749, в порівнянні з PinFaces, в якому їх всього 105, тому мережа могла

навчитися відрізняти один від одного більш складні приклади, тобто зображення облич людей, схожих між собою.

Відповідні висновки відносно впливу на точність розпізнавання було розглянуто в [27]. Це дослідження підтверджується порівнянням акуратності модифікованих моделей.

Що ще примітно, за кількістю нарахованих значень TP, FP, TN і FN видно, що оригінальна мережа при найкращому показнику своєї акуратності віддає більшу перевагу негативному розпізнаванню, тобто відносить зображення облич до категорій тих, кого немає в базі даних. При чому робить вона це як правильно, так і ні. У свою чергу, модифіковані мережі справляються з цим завданням краще – у них розподіл позитивно та негативно розпізнаних прикладів більш збалансований, що робить загальну акуратність цих мереж вище.

Також було виконано аналогічні дії зі збирання та аналізу статистики при порівнянні векторів ознак за допомогою кута косинуса між ними. Методом перебору в циклі було знайдено найкращі значення порогового значення та розраховано інші метрики у відповідних точках. Графік залежності акуратності від порога наведено рисунку 4.16.

Як можна побачити, результати майже ідентичні. Різницю можна помітити лише в дуже невеликих деталях, таких як форма кутів, що формуються лініями.

Звідси можна дійти до висновку, що з завдання розпізнавання осіб вибір методу порівняння векторів ознак немає значення – обидва методи порівняння векторів ознак придатні для використання при вирішенні даної задачі. Відповідний набір метрик для найкращих значень порога порівняно з куту косинуса наведено в таблиці 4.4.

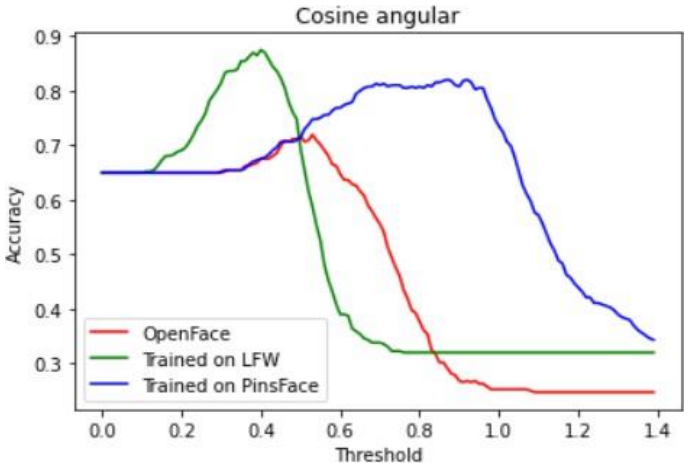


Рисунок 4.16 — Залежність акуратності розпізнавання від порога при порівнянні векторів по куту косинуса

Таблиця 4.4 — Порівняння найкращих результатів моделей при порівнянні векторів за допомогою косинуса кута

	Оригінальна OpenFace	Навчена на LFW	Навчена на PinFaces
Поріг	0.53	0.4	0.87
Число TP прикладів	35	103	106
Число FP прикладів	10	19	44
Число TN прикладів	242	234	210
Число FN прикладів	98	29	25
Точність	77.77%	84.43%	70.66%
Чуйність	26.32%	78.03%	80.92%
Випадання	96.03%	92.5%	82.68%
Акуратність	71.95%	87.53%	82.08%

Під час тестування також було виконано вимірювання швидкості розпізнавання для кожної мережі. Результати порівняння наведено на графіку, зображеному на рисунку 4.17. Наведені значення відображають середній час, що займає розпізнавання одного тестового зображення з усіма зображеннями в базі даних, тобто розпізнавання однієї людини. Як можна побачити, оригінальна мережа займає приблизно 1,5 секунд, коли в модифікованій

структурі мережі вдалося зменшити цей час приблизно на 0.25 секунд.

Рисунок 4.17 — Середній час розпізнавання на одну ітерацію

Додатково до оцінки точності розпізнавання мереж було виконано експеримент порівняння швидкості навчання на різних процесорах. За результатами навчання мереж на Tensorflow (рис. 4.18) видно, що час навчання на всіх епохах займає приблизно однаковий час, тому, щоб не перенавчати мережу цілком, було вирішено навчити модифіковану мережу на 100 епохах на CPU і GPU. При цьому навчання виконувалося на датасеті LFW, який показує найкращі характеристики якості розпізнавання. Таким чином час навчання 100 епох на CPU зайняв у сумі 23146 секунд, а на GPU — 4606 секунд, тобто приблизно 6,4 години і 1,3 відповідно.

Також було виконано спробу навчання мережі на сервісі Google Colab на процесорах TPU. Оскільки цей сервіс є безкоштовним і їм одночасно користується безліч людей, він був створений лише для інтерактивної роботи. Тому в процесі роботи він регулярно вимагає від користувача підтвердження присутності, видаючи вікно з відповідною кнопкою, через що повноцінне навчання мережі на ньому стає трудомістким.

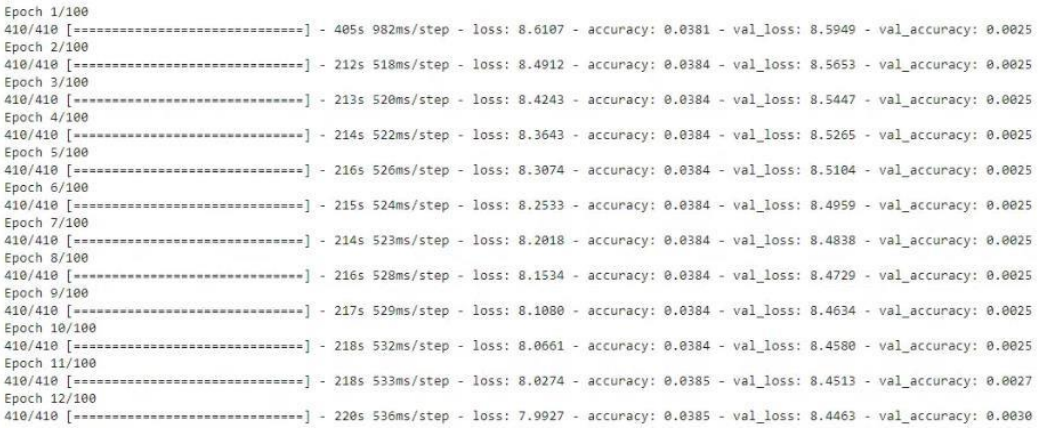


Рисунок 4.18 — Процес навчання мереж на Tensorflow з Keras

Враховуючи це, експерименти на ньому було виконано лише на 10-ти епохах із приблизним розрахунком часу, який би знадобився для навчання на 100 епохах. При цьому для розрахунку часу решти епох було взято лише час епох, наступних після першої, оскільки, як можна побачити на рисунку 3.18, при навчанні на Tensorflow з Keras перша епоха може займати найбільший час, коли наступні вимагають набагато менше часу завдяки кешуванню даних під час навчання на першій епосі. Таким чином, час навчання на Google Colab на CPU склав приблизно 46900 секунд, на GPU – 8285 секунди, а на TPU – 47778 секунди, тобто приблизно 13 годин, 2,3 години і 13,3 годин відповідно.

Сумарні результати досліджень швидкості навчання на CPU, GPU робочої станції кафедри ПМІ та CPU, GPU, TPU виділеної машини Google Colab зведено у таблиці 4.5.

Таблиця 4.5 — Приблизний час навчання на робочій станції ПМІ та виділеній машині Google Colab на CPU, GPU та TPU

	Робоча станція кафедри ПМІ	Виділена машина Google Colab
100 епох на CPU	6,4 годин	13 годин
100 епох на GPU	1,3 годин	2,3 годин
100 епох на TPU	–	13,3 годин

За результатами дослідження швидкості навчання мережі можна дійти до висновку, що у робочої станції кафедри ПМІ мережу можна навчити у кілька разів швидше, виконуючи навчання на GPU-процесорах. Також можна

помітити, що TPU на Google Colab працює дуже повільно в порівнянні з класичним GPU-процесором і навіть трохи повільніше за CPU. Тому, експериментуючи з глибоким навчанням на Google Colab, краще віддавати перевагу саме GPU, ніж CPU або TPU.

5 ЕКОНОМІЧНА ЧАСТИНА

5.1 Комерційний та технологічний аудит науково-технічної розробки

Метою даного розділу є проведення технологічного аудиту, в даному випадку системи реєстрації та розпізнавання напряму руху за допомогою комп’ютерного зору. Особливістю розробки є системи реєстрації та розпізнавання напряму руху за допомогою комп’ютерного зору.

Актуальність полягає у використанні систем комп'ютерного зору при розробці автоматично керованих засобів та автономних мобільних роботів у сфері складської логістики.

Аналогом може бути Робот Kiva компанії Amazon за ціною 30000$, який використовує аналогічне програмне забезпечення.

Для проведення комерційного та технологічного аудиту залучають не менше 3-х незалежних експертів. Оцінювання науково-технічного рівня розробки та її комерційного потенціалу рекомендується здійснювати із застосуванням п’ятибальної системи оцінювання за 12-ма критеріями, у відповідності із табл. 5.1.

Таблиця 5.1 — Рекомендовані критерії оцінювання комерційного потенціалу розробки та їх можлива бальна оцінка

Бали (за 5-ти бальною шкалою)					
Критерій	0	1	2	3	4
Технічна здійсненністьконцепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздат-ність продукту в реальних умовах
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку

Продовження табл. 5.1

Ринкові переваги					
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практик на здійсненність					
	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так із комерційної реалізації ідеї
	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
0	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві

Продовження табл. 5.1

1	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
2	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Усі дані по кожному параметру занесено в таблиці 5.2

Таблиця 5.2 — Результати оцінювання комерційного потенціалу розробки

Критерії оцінювання	ПІБ експертів		
	Експерт 1	Експерт 2	Експерт 3
	Бали		
Технічна здійсненність концепції	3	4	4
Наявність аналогів на ринку	3	3	4
Цінова політика	4	4	4
Технічні та споживчі властивості виробу	4	4	4
Експлуатаційні витрати	4	4	3
Ринок збуту	4	3	4
Конкурентоспроможність	3	4	3
Фахівці з технічної і комерційної реалізації	4	3	4
Фінансування	4	4	4
Матеріально-технічна база	3	3	3
Термін реалізації ідеї	4	4	4
Супровідна документація	4	3	4
Сума	44	43	45
Середньоарифметична сума балів	(44+43+45) / 3 = 44		

За даними таблиці 5.2 можна зробити висновок щодо рівня комерційного потенціалу даної розробки. Для цього доцільно скористатись рекомендаціями, наведеними в таблиці 5.3.

Таблиця 5.3 — Рівні комерційного потенціалу розробки

Середньоарифметична сума балів, розрахована на основі висновків експертів	Рівень комерційного потенціалу розробки
0-10	Низький
11-20	Нижче середнього
21-30	Середній
31-40	Вище середнього
41-48	Високий

Як видно з таблиці, рівень комерційного потенціалу розроблюваного нового програмного продукту є високим, що досягається за рахунок використання систем комп'ютерного зору при розробці автоматично керованих засобів та автономних мобільних роботів у сфері складської логістики.

5.2 Прогнозування витрат на виконання науково-дослідної (дослідно-конструкторської) роботи

5.2.1 Основна заробітна плата розробників, яка розраховується за формулою:

$$З_о = \frac{M}{T_р} \cdot t, \quad (5.1)$$

де М — місячний посадовий оклад конкретного розробника (дослідника), грн.;

$T_р$ — число робочих днів за місяць, 20 днів;

t — число днів роботи розробника (дослідника).

Результати розрахунків зведемо до таблиці 5.4.

Таблиця 5.4— Основна заробітна плата розробників

Найменування посади	Місячний посадовий оклад, грн.	Оплата за робочий день, грн.	Число днів роботи	Витрати на заробітну плату, грн.
Керівник проекту	33000	1650,00	45	74250,00
Програміст	26000	1300,00	45	58500,00
Всього				132750,00

Так як в даному випадку розробляється програмний продукт, то розробник виступає одночасно і основним робітником, і тестувальником розроблюваного програмного продукту.

5.2.2 Додаткова заробітна плата розробників, які брати участь в розробці обладнання/програмного продукту.

Додаткову заробітну плату прийнято розраховувати як 14% від основної заробітної плати розробників та робітників:

$$З_д = З_о \cdot 14\ \% / 100\% \qquad (5.2)$$

$$З_д = (132750{,}00 \cdot 14\% / 100\ \%) = 18585{,}00\ \text{(грн.)}$$

5.2.3 Нарахування на заробітну плату розробників.

Згідно діючого законодавства нарахування на заробітну плату складають 22% від суми основної та додаткової заробітної плати.

$$Н_з = (З_о + З_д) \cdot 22\% / 100\% \qquad (5.3)$$

$$Н_з = (132750{,}00 + 18585{,}00) \cdot 22\ \% / 100\% = 33293{,}70\text{(грн.)}$$

Оскільки для розроблювального пристрою не потрібно витрачати матеріали та комплектуючи, то витрати на матеріали і комплектуючі дорівнюють нулю.

5.2.4 Амортизація обладнання, яке використовувалось для проведення розробки в спрощеному вигляді розраховується за формулою:

$$A = \frac{Ц}{Тв} \cdot \frac{t_{вик}}{12} \text{ [грн.].} \quad (5.4)$$

де Ц — балансова вартість обладнання, грн.;

Т — термін корисного використання обладнання згідно податкового законодавства, років;

$t_{вик}$ — термін використання під час розробки, місяців.

Розрахуємо, для прикладу, амортизаційні витрати на комп'ютер балансова вартість якого становить 20000 грн., термін його корисного використання згідно податкового законодавства — 2 роки, а термін його фактичного використання —2,25 міс.

$$A_{обл} = \frac{20000}{2} \times \frac{2,25}{12} = 1875 \; грн.$$

Аналогічно визначаємо амортизаційні витрати на інше обладнання та приміщення. Розрахунки заносимо до таблиці 5.5. Так як вартість ліцензійної ОС та спеціалізованих ліцензійних нематеріальних ресурсів менше 20000 грн, то даний нематеріальний актив не амортизується, а його вартість включається у вартість розробки повністю, $В_{нем.ак.}$ =11000 грн.

Таблиця 5.5—Амортизаційні відрахування на матеріальні та нематеріальні ресурсидля розробників

Найменування обладнання	Балансова вартість, грн.	Строк корисного використання, років	Термін використання обладнання, місяців	Амортиза-ційні відрахуван-ня, грн.
Комп'ютер та комп'ютерна периферія	20000	2	2,25	1875,000
Офісне обладнання (меблі)	21500	4	2,25	1007,813
Приміщення	850000	20	2,25	7968,750
Всього				10851,56

Тарифи на електроенергію для непобутових споживачів (промислових підприємств) відрізняються від тарифів на електроенергію для населення. При цьому тарифи на розподіл електроенергії у різних постачальників (енергорозподільних компаній), будуть різними. Крім того, розмір тарифу залежить від класу напруги (1-й або 2-й клас). Тарифи на розподіл електроенергії для всіх енергорозподільних компаній встановлює Національна комісія з регулювання енергетики і комунальних послуг (НКРЕКП). Витрати на силову електроенергію розраховуються за формулою:

$$В_е = В \cdot П \cdot Ф \cdot К_п, \qquad (5.5)$$

де В — вартість 1 кВт-години електроенергії для 1 класу підприємства, В = 6,2 грн./кВт;

П — встановлена потужність обладнання, кВт. П = 0,5кВт;

Ф — фактична кількість годин роботи обладнання, годин.

$К_п$ — коефіцієнт використання потужності, $К_п$ = 0,9.

$$В_е = 0,9 \cdot 0,5 \cdot 8 \cdot 45 \cdot 6,2 = 1004,4 \text{ (грн.)}$$

5.2.5 Інші витрати та загальновиробничі витрати.

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками. Витрати за статтею «Інші витрати» розраховуються як 50…100% від суми основної заробітної плати дослідників:

$$I_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \quad (5.6)$$

де $Н_{ів}$ — норма нарахування за статтею «Інші витрати».

$$I_в = 132750{,}00 * 80\% / 100\% = 106200 \text{ (грн.)}$$

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов'язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов'язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов'язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін. Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуються як 100…150% від суми основної заробітної плати дослідників:

$$Н_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (5.7)$$

де $Н_{нзв}$ — норма нарахування за статтею «Накладні (загальновиробничі) витрати».

$$Н_{нзв} = 132750{,}00 * 133\% / 100\ \% = 176558 \text{ (грн.)}$$

5.2.6 Витрати на проведення науково-дослідної роботи.

Сума всіх попередніх статей витрат дає загальні витрати на проведення науково-дослідної роботи:

$$В_{заг}=132750,00+18585,00+33293,70+10851,56+11000+1004,40+106200+$$
$$+176558 = 490242,16грн.$$

5.2.7 Розрахунок загальних витрат на науково-дослідну (науково-технічну) роботу та оформлення її результатів.Загальні витрати на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховуютьсяза формулою:

$$ЗВ = \frac{В_{заг}}{\eta} \text{ (грн)}, \qquad (5.8)$$

де η — коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи.

Так, якщо науково-технічна розробка знаходиться на стадії: науково-дослідних робіт, то η=0,1; технічного проектування, то η=0,2; розробки конструкторської документації, то η=0,3; розробки технологій, то η=0,4; розробки дослідного зразка, то η=0,5; розробки промислового зразка, то η=0,7; впровадження, то η=0,9.Оберемо η = 0,5, так як розробка, на даний момент, знаходиться на стадії дослідного зразка:

$$ЗВ = 490242,16 / 0,5 = 980484 грн.$$

5.3 Розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором

В ринкових умовах узагальнювальним позитивним результатом, що його може отримати потенційний інвестор від можливого впровадження результатів тієї чи іншої науково-технічної розробки, є збільшення у потенційного інвестора величини чистого прибутку. Саме зростання чистого прибутку забезпечить потенційному інвестору надходження додаткових коштів, дозволить покращити фінансові результати його діяльності, підвищить конкурентоспроможність та може позитивно вплинути на ухвалення рішення щодо комерціалізації цієї розробки.

Для того, щоб розрахувати можливе зростання чистого прибутку у потенційного інвестора від можливого впровадження науково-технічної розробки необхідно:

— вказати, з якого часу можуть бути впроваджені результати науково-технічної розробки;

— зазначити, протягом скількох років після впровадження цієї науково-технічної розробки очікуються основні позитивні результати для потенційного інвестора (наприклад, протягом 3-х років після її впровадження);

— кількісно оцінити величину існуючого та майбутнього попиту на цю або аналогічні чи подібні науково-технічні розробки та назвати основних суб’єктів (зацікавлених осіб) цього попиту;

— визначити ціну реалізації на ринку науково-технічних розробок з аналогічними чи подібними функціями.

При розрахунку економічної ефективності потрібно обов’язково враховувати зміну вартості грошей у часі, оскільки від вкладення інвестицій до отримання прибутку минає чимало часу. При оцінюванні ефективності інноваційних проектів передбачається розрахунок таких важливих показників:

— абсолютного економічного ефекту (чистого дисконтованого доходу);

— внутрішньої економічної дохідності (внутрішньої норми дохідності);

— терміну окупності (дисконтованого терміну окупності).

Аналізуючи напрямки проведення науково-технічних розробок, розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором можна об’єднати, враховуючи визначені ситуації з відповідними умовами.

5.3.1 Розробка чи суттєве вдосконалення програмного засобу (програмного забезпечення, програмного продукту) для використання масовим споживачем.

В цьому випадку майбутній економічний ефект буде формуватися на основі таких даних:

$$\Delta П_i = (\pm \Delta Ц_0 \cdot N + Ц_0 \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \quad (5.9)$$

де $\pm\Delta Ц_о$ — зміна вартості програмного продукту (зростання чи зниження) від впровадження результатів науково-технічної розробки в аналізовані періоди часу;

N — кількість споживачів які використовували аналогічний продукт у році до впровадження результатів нової науково-технічної розробки;

$Ц_о$ — основний оціночний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки, $Ц_о = Ц_б \pm \Delta Ц_о$;

Цб — вартість програмного продукту у році до впровадження результатів розробки;

ΔN— збільшення кількості споживачів продукту, в аналізовані періоди часу, від покращення його певних характеристик;

Λ — коефіцієнт, який враховує сплату податку на додану вартість. Ставка податку на додану вартість дорівнює 20%, а коефіцієнт λ = 0,8333.

Р — коефіцієнт, який враховує рентабельність продукту;

Θ — ставка податку на прибуток, у 2025 році ϑ =18%.

Припустимо, що при прогнозованій ціні 11200 грн. за одиницю виробу, термін збільшення прибутку складе 3 роки. Після завершення розробки і її вдосконалення, можна буде підняти її ціну на 400 грн. Кількість одиниць реалізованої продукції також збільшиться: протягом першого року — на 1500 шт., протягом другого року – на 1800 шт., протягом третього року на 2000 шт. До моменту впровадження результатів наукової розробки реалізації продукту не було:

$$\Delta П_1 = (0*400 + (11200 + 400)*1500)* 0,8333* 0,33) * (1 - 0,18) = 3788399,848 \text{ грн.}$$

$$\Delta П_2 = (0*400 + (11200 + 400)*(1500+1800)* 0,8333* 0,33) * (1 - 0,18) = 8632139,655 \text{ грн.}$$

$$\Delta П_3 = (0*400 + (11200 + 400)*(1500+1800+2000)* 0,8333* 0,33) * (1 - 0,18) = 13863739,445 \text{грн.}$$

Отже, комерційний ефект від реалізації результатів розробки за три роки складе 26284278,95 грн.

5.3.2 Розрахунок ефективності вкладених інвестицій та періоду їх окупності.Розраховуємо приведену вартість збільшення всіх чистих прибутків ПП, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \qquad (5.10)$$

де $\Delta П_i$ — збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої науково-дослідної (науково-технічної) роботи, грн;

Т — період часу, протягом якою виявляються результати впровадженої науково-дослідної (науково-технічної) роботи, роки;

τ — ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ =0,05…0,15;

t — період часу (в роках).

Збільшення прибутку ми отримаємо, починаючи з першого року:

$$ПП = (3788399,848/(1+0,1)^1)+(8632139,655/(1+0,1)^2)+(13863739,445/$$
$$/(1+0,1)^3)= 3443999,86 + 7133999,715 + 10416032,64 = 20994032,22 \text{ грн.}$$

Далі розраховують величину початкових інвестицій PV, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки. Для цього можна використати формулу:

$$PV = k_{інв} * ЗВ, \quad (5.11)$$

де $k_{інв}$ — коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію, це можуть бути витрати на підготовку приміщень, розробку технологій, навчання персоналу, маркетингові заходи тощо; зазвичай $k_{інв}$=2...5, але може бути і більшим;

ЗВ — загальні витрати на проведення науково-технічної розробки та оформлення її результатів, грн.

$$PV = 2 * 980484 = 1960968,65 \text{ грн.}$$

Тоді абсолютний економічний ефект $Е_{абс}$або чистий приведений дохід (NPV, Net Present Value) для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$Е_{абс} = ПП – PV, \quad (5.12)$$

$$Е_{абс} =20994032,22 – 1960968,65 = 19033063,57 \text{ грн.}$$

Оскільки $E_{абс} > 0$ то вкладання коштів на виконання та впровадження результатів даної науково-дослідної (науково-технічної) роботиможе бути доцільним.

Для остаточного прийняття рішення з цього питання необхідно розрахувати внутрішню економічну дохідність або показник внутрішньої но-рми дохідності (IRR, InternalRateofReturn)вкладених інвестицій та порівняти її з так званою бар'єрною ставкою дисконтування, яка визначає ту мінімальну внутрішню економічну дохідність, нижче якої інвестиції в будь-яку науково-технічну розробку вкладати буде економічно недоцільно.

Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $E_в$. Для цього використаємо формулу:

$$E_в = \sqrt[T_ж]{1 + \frac{E_{абс}}{PV}} - 1, \tag{5.13}$$

$T_ж$ – життєвий цикл наукової розробки, роки.

$$E_в = \sqrt[3]{1 + 19033063{,}57/1960968{,}65} - 1 = 1{,}204$$

Визначимо мінімальну ставку дисконтування, яка у загальному вигляді визначається за формулою:

$$\tau = d + f, \tag{5.14}$$

де d — середньозважена ставка за депозитними операціями в комерційних банках; в 2023 році в Україні d = (0,09…0,14);

f — показник, що характеризує ризикованість вкладень; зазвичай, величина f = (0,05...0,5).

$$\tau_{\min} = 0{,}14 + 0{,}05 = 0{,}19.$$

Так як $E_в > \tau_{min}$, то інвестор може бути зацікавлений у фінансуванні даної наукової розробки.

Розрахуємо термін окупності вкладених у реалізацію наукового проекту інвестицій за формулою:

$$T_{ок} = \frac{1}{E_в}, \quad (5.15)$$

$$T_{ок} = 1 / 1{,}204 = 0{,}83р.$$

Оскільки $T_{ок}$ < 3-х років,а саме термін окупності рівний 0,83роки, то фінансування даної наукової розробки є доцільним.

Висновки до розділу: економічна частина даної роботи містить розрахунок витрат на розробку нового програмного продукту, сума яких складає 980484 гривень. Було спрогнозовано орієнтовану величину витрат по кожній з статей витрат. Також розраховано чистий прибуток, який може отримати виробник від реалізації нового технічного рішення, розрахованоперіод окупності витрат для інвестора та економічний ефект при використанні даної розробки. В результаті аналізу розрахунків можна зробити висновок, що розроблений програмний продукт за ціною дешевший за аналог і є висококонкурентоспроможним. Період окупності складе близько 0,83роки.

ВИСНОВКИ

Під час виконання магістерської кваліфікаційної роботи, насамперед, був проведений аналіз предметної області. Після цього було визначені основні проблеми, що виникають при виявленні та розпізнаванні облич.

Проведений аналіз методів для розпізнавання обличь показав що розпізнавання за допомогою нейронних мереж є точнішим і гнучкішим методом, порівняно з іншими.

Для вибору оптимальної структури нейронної мережі були розглянуті основні типи штучних нейронних мереж, на основі цього вибрана згорткова нейронна мережа. Спочатку було вивчено теоретичну інформацію про предметну область, достатню для розробки власної ЗНМ, а також для того, щоб визначитися, що саме можна покращити. Це включило наступне:

— визначено математичну постановку задачі розпізнавання облич;

— вивчено принципи роботи нейронних мереж в цілому, структуру і алгоритм конкретно згорткової мережі, що також включає в себе повнозв’язну мережу, а також вивчено використовувані функції активації, функції помилки і додаткові архітектури та методи в нейронних мережах, які можуть допомогти в розробці глибоких нейронних мереж і дати краще розуміння їх роботи;

— вивчено архітектуру системи для розпізнавання облич, її базові компоненти, їх алгоритм роботи і як вони взаємодіють між собою;

— досліджено теоретичну інформацію про вже існуючі сучасні моделі нейронних мереж, які використовуються в сфері розпізнавання облич.

На основі дослідження існуючих нейронних мереж було вибрано модифікацію для власної мережі розпізнавання облич, а також підхід до її навчання. Після цього було реалізувано мережу з модифікацією та протестовано її.

. В кінці проведено економічний розрахунок системи, що розробляється, який показав економічну доцільність її створення.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Навчальний посібник з дисципліни Системи візуалізації та розпізнавання образів [навчальний посібник] / Смолій В.В., Савицька Я.А., Місюра М.Д., Шкарупило В.В. // - К.: ФОП Ямчинський О.В., 2020.- 200 с.

2. Вовк С.М., Гнатушенко В.В., Бондаренко М.В. Методи обробки зображень та комп'ютерний зір. Навчальний посібник. – Д.:«ЛІРА», 2016. – 148 с.

3. Бодянський Є. В. Аналіз та обробка потоків даних засобами обчислювального інтелекту: Монографія / Є. В. Бодянський, Д. Д. Пелешко, О. А. Винокурова, С. В.Машталір, Ю. С. Іванов. Львів : Видавництво Львівської політехніки, 2016. 236 с.

4. Довбиш А.С. Основи теорії розпізнавання образів: навч. посіб. : у 2 ч. / А.С. Довбиш, І.В. Шелехов. Суми: Сумський державний університет, 2015. Ч.1. 109 с.

5. Застосування систем штучного інтелекту [Електронний ресурс] – режим доступа: https://sites.google.com/site/eksperntisistemi/zastosuvanna-sistem-stucnogointelektu.

6. Розпізнавання облич [Електронний ресурс] – режим доступа : http://wiki.tntu.edu.ua/Розпізнавання_обличь:_від_теорії_ до_практики.

7. Viola P. Rapid object detection using a boosted cascade of simple features / P. Viola, M. Jones // Proc. of CVPR. – 2001. – Vol.1. – P. 511-518.

8. Turk M. Eigenfaces for recognition / M. Turk, A. Pentland // Journal of Cognitive Neuroscience. – 1991. – Vol. 13, No. 1. – P. 71–86.

9. Belhumeur P.N. Eigenfaces vs. Fisherfaces: recognition using class specific linear projection / P.N. Belhumeur, J.P. Hespanha, D.J. Kriegman // IEEE Trans. On PAMI. – 1997. – Vol. 19, No. 7. – P. 711–720.

10. Багатошаровий перцептрон [Електронний ресурс] – режим доступа: http://dn.khnu.km.ua/dn/k_default.aspx?M=k1113&T=07&lng=1&st=0.

11. Bartlett M.S. Face recognition by independent component analysis / M.S. Bartlett, J.R. Movellan, T.J. Sejnowski // IEEE Trans. Neural Netw. – 2002. –

Vol.13, No. 6. – P. 1450–1464.

12. Shen L. A review on Gabor wavelets for face recognition / L. Shen, L. Bai // Journal of Pattern Analysis and Applications. – 2006. – Vol.

13. No. 2-3. – P. 273- 292. 9. Vaswani N. Principal components null space analysis for image and video classification / N. Vaswani, R. Chellappa // IEEE Trans. Image Process. – 2006. – Vol. 15, No. 7. – P. 1816–1830.

14. Руденко О.В. Штучні нейронні мережі: Навчальний посібник / О.В.Руденко, Є.В.Бодянський. - Харків : ТОВ «Компанія СМІТ», 2006. — 404 с. - ISBN 966-8630-73-X.

15. Рашкевич Ю.М. Нейроподібні методи, алгоритми та структури обробки сигналів і зображень у реальному часі: монографія. / Ю.М. Рашкевич, Р.О. Ткаченко, Цмоць І.Г., Д.Д. Пелешко. Львів: Видавництво Львівської політехніки, 2017. 256 с. Допоміжна

16. Duda R. O. Pattern Classification, second ed. / R. O. Duda, P. E. Hart, D. G. Stork. – John Wiley & Sons, New York, 2001. 738 p.

17. Зайченко Ю. П. Основи проектування інтелектуальних систем: навчальний посібник / Ю. П. Зайченко. – К. :106 Видавничий Дім «Слово», 2004. – 352 с.

18. Hastie T. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. – 2nd ed. / T. Hastie, R. Tibshirani, J. Friedman. – Springer-Verlag, 2009. – 746 p. Інформаційні ресурси в Інтернет

19. Ramakrishnan S. Introductory Chapter Face Recognition - Overview, Dimensionality Reduction, and Evaluation Methods / S. Ramakrishnan // Face Recognition - Semisupervised Classification, Subspace Projection and Evaluation Method: IntechOpen, 2013. P. 137-144.

20. What Are Tensor Cores? Mixed-Precision Computing. [Електронний ресурс]. – Режим доступу до ресурсу: https://www.techspot.com/article/ 2049-what-are-tensor-cores/.

21. Masters T. Deep Belief Nets in C++ and CUDA C Volume 3: Convolutional Nets / T. Masters. – Apress Media, 2018. – 184 p.

22. Zeiler M. D. Visualizing and understanding convolutional networks / M. D. Zeiler, R. Fergus // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): arXiv, 2014. P. 1-11.

23. He K. Deep residual learning for image recognition / K. He, X. Zhang, S. Ren, J. Sun // Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition: arXiv, 2016. P. 770–778.

24. Szegedy C. Going deeper with convolutions / C. Szegedy // Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition: arXiv, 2015. P. 1–9.

25. Understanding the Bias-Variance Tradeoff [Електронний ресурс]. – Режим доступу до ресурсу: https://towardsdatascience.com/understanding-the-bias-variance-tradeoff-165e6942b229.

26. Kukačka J. Regularization for Deep Learning: A Taxonomy / J. Kukačka, V. Golkov, D. Cremers // Technical University of Munich: arXiv. 2017. P. 1–2

27. Sundaram M. Face Recognition: Demystification of Multifarious Aspect in Evaluation Metrics / M. Sundaram, A. Mani // Face Recognition - Semisupervised Classification, Subspace Projection and Evaluation Method: IntechOpen, 2013. P. 137–144.

28. Spatial Localization and Detection [Електронний ресурс]. – Режим доступу до ресурсу: http://cs231n.stanford.edu/slides/2016/ winter1516_lecture8.pdf.

29. Задача нахождения объектов на изображении [Електронний ресурс]. – Режим доступу до ресурсу: https://neerc.ifmo.ru/wiki/index.php?title= Задача_нахождения_объектов_на_изображении.

30. Hosang J. What Makes for Effective Detection Proposals? / J. Hosang, R. Benenson, P. Dollar, B. Schiele // IEEE Transactions on Pattern Analysis and Machine Intelligence: arXiv, 2016. P. 814–830.

31. Convolutional Neural Networks [Електронний ресурс]. – Режим доступу до ресурсу: https://www.coursera.org/learn/convolutional-neural- networks.

32. Hosang J. Learning non-maximum suppression / J. Hosang, R. Benenson,

B. Schiele // Proceedings - 30th IEEE Conference on Computer Vision Pattern Recognition: arXiv, 2017. P. 6469–6477.

33. Gong M. A review of non-maximum suppression algorithms for deep learning target detection / M. Gong, D. Wang, X. Zhao, H. Guo, D. Luo, M. Song // Seventh Symposium on Novel Photoelectronic Detection Technologyand Application 2020, Kunming, China: SPIE, 2020. P. 133-141.

34. One Shot learning, Siamese networks and Triplet Loss with Keras [Електронний ресурс]. – Режим доступу до ресурсу: https://medium.com/@crimy/one-shot-learning-siamese-networks-and-triplet-loss-with-keras-2885ed022352.

35. Machine vision. Article from Wikipedia, the free encyclopedia [Електронний ресурс]: – Режим доступу: https://en.wikipedia.org/wiki/Machine_vision

36. Introducing Myriad X: Unleashing AI at the Edge. [Електронний ресурс]: – Режим доступу: https://newsroom.intel.com/editorials/introducing-myriad-xunleashingai-at-the-edge/#gs.owkeej

37. Computer Vision: Algorithms and Applications by Richard Szeliski. Доступно безкоштовно онлайн. http://szeliski.org/Book/

38. Computer Vision: A Modern Approach (Second Edition) by David Forsyth and Jean Ponce. Доступно безкоштовно онлайн. http://luthuli.cs.uiuc.edu/~daf/CV2E-site/cv2eindex.html14.

39. CodeBlocks – середовище програмування мовою C/C++ [Електронний ресурс]. URL: https://progtips.ru/instrumenty-programmista/codeblocks.html(Дата звернення: 16.04.2023).

ДОДАТОК А

Технічне завдання

Міністерство освіти та науки України

Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра обчислювальної техніки

ЗАТВЕРДЖУЮ

Завідувач кафедри ОТ

_____проф., д.т.н. О. Д. Азаров

"_29_" _____02___2024 р.

ТЕХНІЧНЕ ЗАВДАННЯ

на виконання магістерської кваліфікаційної роботи

«Система розпізнавання обличчя на фотографії з використанням нейронної мережі»

08-54.МКР.006.00.000 ПЗ

Науковий керівник

к.т.н., доц. каф. ОТ

_____Муращенко О.Г.

виконав:

магістрант 2 курсу,

Чміхаленко В.Є.

Вінниця 2024

1. Підстава виконання магістерської кваліфікаційної роботи

1.1 Сучасний етап розвитку інформаційних технологій характеризується розробкою, створенням та широким впровадженням відеоінформаційних технологій, що засновані на обробці та використанні зображень. Розпізнавання облич — один з підрозділів більш широкої категорії розпізнавання образів.

1.2 Наказ про затвердження теми МКР

2 Мета і призначенням МКР

2.1 Метою роботи є підвищення достовірності роботи програмного модуля для розпізнавання облич з веб-камери.

2.2 Призначення розробки — виконання магістерської кваліфікаційної роботи.

3 Вихідні дані для виконання МКР

Вихідні дані для виконання МКР: розпізнавання проводиться з веб-камери, навчання проводиться з графічних файлів типу jpg розмірністю 250x250 пікселів.

4 Вимоги до виконання МКР

МКР повинна задовольняти такі вимоги:

— забезпечити навчання нейронної мережі для розпізнавання облич з відеопотоку;

— провести тестування створеної системи;

5 Етапи МКР та очікувані результати

Етапи роботи та очікувані результати приведено в табл. А.1.

6 Матеріали, що подаються до захисту МКР

До захисту МКР подаються: пояснювальна записка МКР, ілюстративні та графічні матеріали, протокол попереднього захисту МКР на кафедрі, відзив наукового керівника, відзив опонента, протоколи складання державних екзаменів, анотації до МКР українською та іноземною мовами, довідка про відповідність оформлення МКР діючим вимогам.

Таблиця А.1 — Етапи МКР

№ з/	Назва етапів виконання магістерської роботи	Строк виконання етапів роботи	Прим.
1	Постановка мети та задач роботи		
2	Аналіз предметної області		
3	Обгрунтування методу розв’язання задачі		
4	Проектування інтелектуального модуля розпізнавання облич на фотографії		
5	Програмна реалізація модуля розпізнавання облич на фотографії		
6	Аналіз результатів тестування модуля розпізнавання облич на фотографі		
7	Розробка інструкції користувача		
8	Розрахунок економічної частини роботи		
9	Оформлення пояснювальної записки та ілюстративного матеріалу		
10	Аналіз виконання роботи, висновки, додатки		
11	Перевірка якості виконання магістерської роботи та усунення недоліків		

7 Порядок контролю виконання та захисту МКР

Виконання етапів розрахункової та графічної документації МКР контролюється науковим керівником згідно зі встановленими термінами. Захист МКР відбувається на засіданні Державної екзаменаційної комісії, затвердженою наказом ректора.

8 Вимоги до оформлення МКР

8.1 При оформлюванні МКР використовуються:

— ДСТУ 3008: 2015 «Звіти в сфері науки і техніки. Структура та правила оформлювання»;

— ДСТУ 8302: 2015 «Бібліографічні посилання. Загальні положення та правила складання»;

— Методичні вказівкидо виконання магістерських кваліфікаційних робіт зі спеціальності 123 — «Комп'ютерна інженерія». Кафедра обчислювальної техніки ВНТУ 2022.

8.2 Порядок виконання МКР викладено в «Положення про кваліфікаційні роботи на другому (магістерському) рівні вищої освіти СУЯ ВНТУ–03.02.02 П.001.01:21.

ДОДАТОК Б

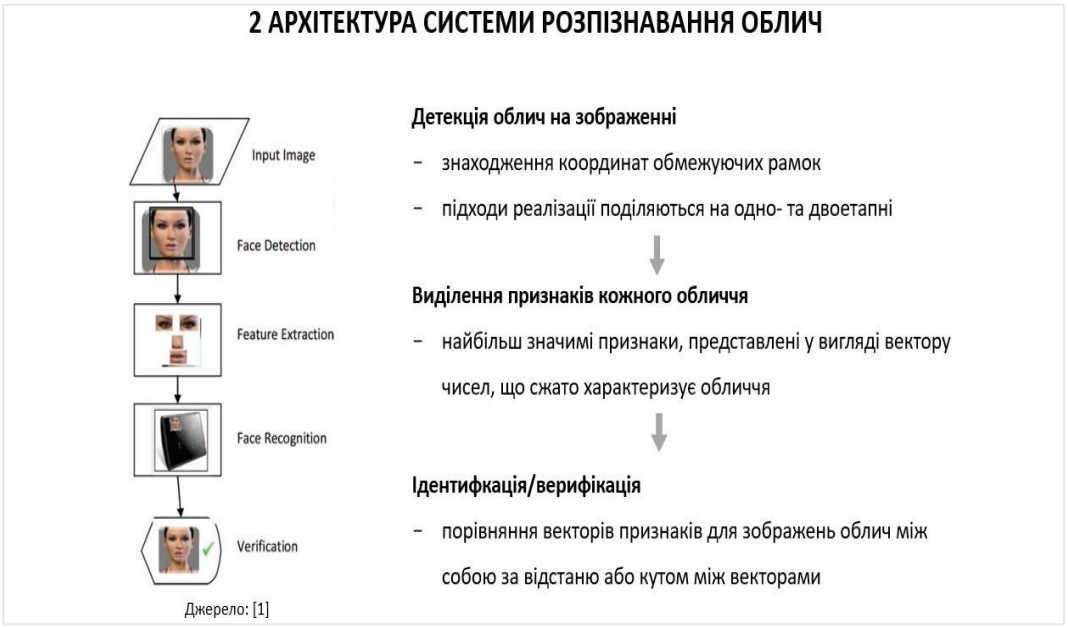


Рисунок Б.1 — Архітектура розпізнавання облич

ДОДАТОК В

3 ВИДІЛЕННЯ ПРИЗНАКІВ

Головна задача: навчити нейронну мережу для зображень однієї людини видавати схожі вектори признаків, а для різних людей – різні вектори.

Існуючі підходи

1. Використання функції помилки Triplet Loss

(використовується у мережі FaceNet и OpenFace)

$$\mathcal{L}(A,P,N) = \max\left(\|\mathrm{f}(A) - \mathrm{f}(P)\|^2 - \|\mathrm{f}(A) - \mathrm{f}(N)\|^2 + \alpha, 0\right)$$

Джерело: [3]

2. Навчання класифікатору

- навчити мережу класифікувати фіксований набір людей
- після навчання видалити останній (класифікуючий) шар
- мережа, що залишилась, вже вміє правильно виділяти признаки

Рисунок В.1 — Виділення при знаків

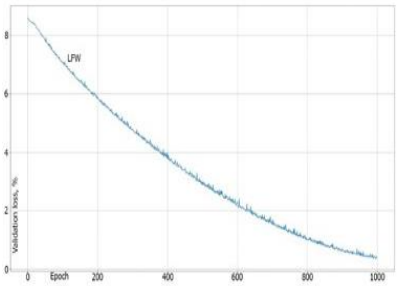
ДОДАТОК Г

Джерело: Авторський рисунок на основі [4]

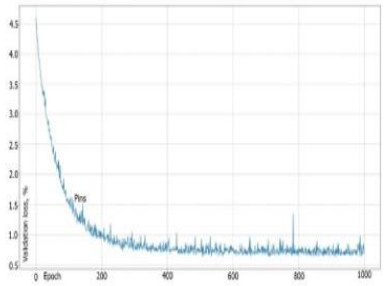
Рисунок Г.1 — Модифікація мережі

ДОДАТОК Д

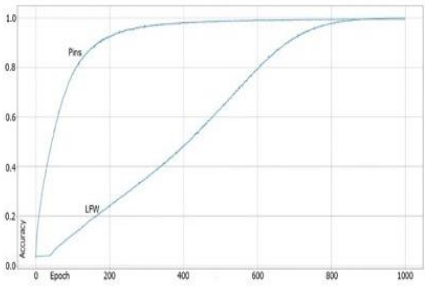
РОЗРОБКА ТА НАВЧАННЯ МОДЕЛЕЙ



Помилка на датасеті валідації LFW



Помилка на датасеті валідації PinsFace



Точність навчання на використаних датасетах

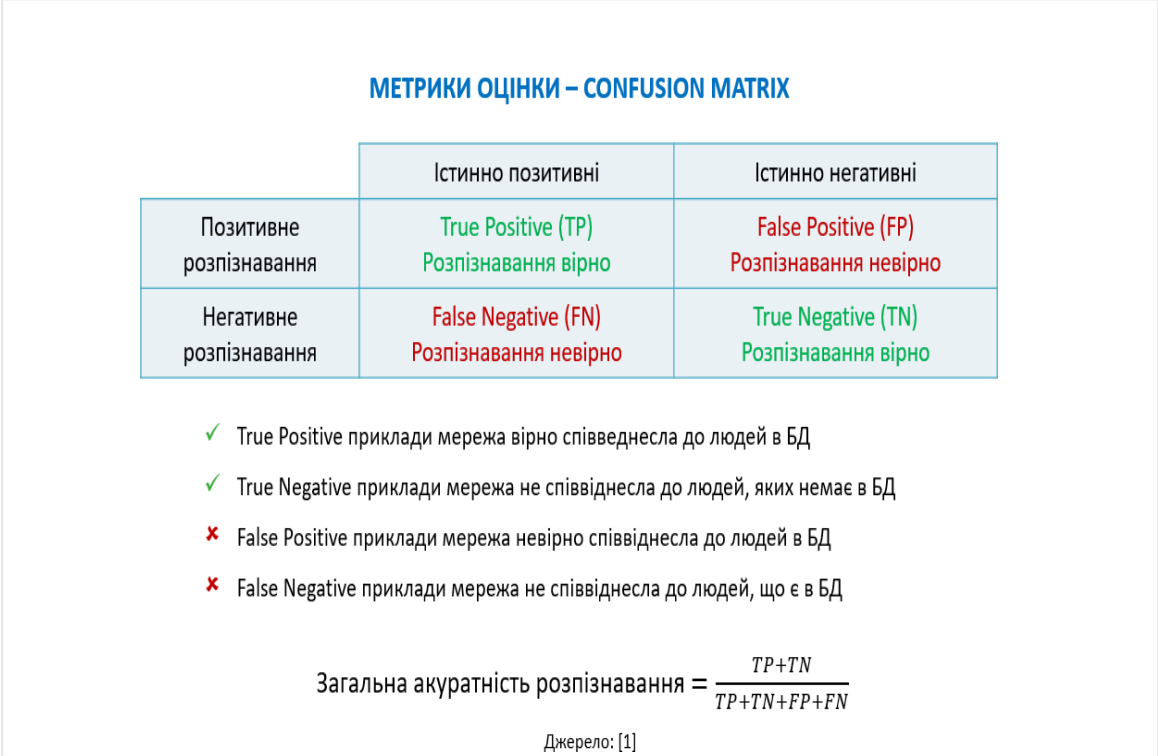
Результати навчання:

- всього було навчено дві моделі: на LFW та PinsFace
- навчання кожної з моделей тривало 1000 епох (приблизно день)
- **умова зупинки**: припинення покращення точності протягом великої кількості епох навчання
- на датасеті LFW достатнім може бути 850-900 епох
- на датасеті PinsFace достатнім може бути близько 500 епох

Рисунок Д1 — Розробка та навчання моделей

ДОДАТОК Е

ТЕСТУВАННЯ МОДЕЛЕЙ



МЕТРИКИ ОЦІНКИ – CONFUSION MATRIX

	Істинно позитивні	Істинно негативні
Позитивне розпізнавання	True Positive (TP) Розпізнавання вірно	False Positive (FP) Розпізнавання невірно
Негативне розпізнавання	False Negative (FN) Розпізнавання невірно	True Negative (TN) Розпізнавання вірно

- ✓ True Positive приклади мережа вірно співведнесла до людей в БД
- ✓ True Negative приклади мережа не співвіднесла до людей, яких немає в БД
- ✘ False Positive приклади мережа невірно співвіднесла до людей в БД
- ✘ False Negative приклади мережа не співвіднесла до людей, що є в БД

$$\text{Загальна акуратність розпізнавання} = \frac{TP+TN}{TP+TN+FP+FN}$$

Джерело: [1]

Рисунок Е.1 — Тестування моделей

ДОДАТОК Ж

ПРОТОКОЛ
ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НАНАЯВНІСТЬТЕКСТОВИХЗАПОЗИЧЕНЬ

Назвароботи: Система розпізнавання обличчя на фотографії з використанням нейронної мережі

Типроботи: магістерська кваліфікаційна робота
(БДР,МКР)

Підрозділ кафедра обчислювальної техніки
(кафедра,факультет)

ПоказникизвітуподібностіUnicheck

Оригінальність 92% Схожість 8%

Аналіз звіту подібності (відмітити потрібне):

✓ Запозичення, виявлення у роботі, оформлені коректно і не містять ознак плагіату.

□ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

□ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа,відповідальназаперевірку _____ Захарченко С.М.
(підпис) (прізвище,ініціали)

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Авторроботи _____ Чміхаленко В.Є.
(підпис) (прізвище,ініціали)

Керівникроботи _____ Муращенко О.Г.
(підпис) (прізвище,ініціали)