Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
Кафедра Комп'ютерних інтелектуальних технологій та систем

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Розпізнавання лицьових масок на обличчі
з використанням відеокамери

Виконав:
студент 2 курсу, групи КІТм-20-1
Совєцький М.О.

Спеціальність 123 Комп'ютерна інженерія
Тип програми освітньо-професійна
Освітня програма Комп'ютерні
інтелектуальні технології
Керівник проф. Корабльов М.М.

Допускається до захисту _____
(підпис)

Зав. кафедри _____ проф. Руденко О.Г.
(підпис)

2021 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління

Кафедра Комп’ютерних інтелектуальних технологій та систем

Рівень вищої освіти другий (магістерський)

Спеціальність (напрям) 123 – Комп’ютерна інженерія
(код і назва)

Освітня програма Комп’ютерні інтелектуальні технології
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
“____” _____ 20__ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Совєцькому Михайлу Олеговичу
(прізвище, ім’я, по батькові)

1. Тема роботи Розпізнавання лицьових масок на обличчі з використанням відеокамери

затверджена наказом по університету від “ 08 ” листопада 2021 р. № 1666Ст

2. Термін подання студентом роботи до екзаменаційної комісії 10 грудня 2021 р.

3. Вхідні дані до роботи

1) навчальна множина зображень обличь людей з лицьовими масками та без них;
2) побудова штучної нейронної мережі;
3) середа розробки PyCharm;
4) мова програмування – Python.

4. Перелік питань, що потрібно опрацювати в роботі

1) огляд предметної області;
2) аналіз предмету дослідження;
3) дослідження методів використання нейронних мереж;
4) розробка штучної нейронної мережі;
5) експериментальні дослідження
6) аналіз якості згорткової нейронної мережі;
7) висновки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) Демонстраційні матеріали. Плакати - 18 арк. ф. А4

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Видача та узгодження теми проекту	08.11.2021	
2	Огляд стану проблеми та постановка задачі	09.11.2021 – 10.11.2021	
3	Аналіз літератури за напрямком магістерської	10.11.2021 – 14.11.2021	
	роботи		
4	Вибір методів рішення для реалізації та їх	16.11.2021 – 20.11.2021	
	обґрунтування		
5	Експериментальні дослідження та	21.11.2021 – 28.11.2021	
	аналіз якості нейронної мережі		
6	Оформлення пояснювальної записки	29.11.2021 – 05.12.2021	
7	Підготовка графічного матеріалу	06.12.2021 – 10.12.2021	
8	Перевірка виконаного проекту курівником	10.12.2021	
9	Захист проекту	16.12.2021 – 17.12.2021	

Дата видачі завдання 08 листопада 2021 р.

Студент _____
(підпис)

Керівник роботи _____ проф. Корабльов М.М.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 80 с., 25 рис., 20 джерел.

ЗГОРТКОВІ НЕЙРОННІ МЕРЕЖІ, ВИЯВЛЕННЯ ОБ’ЄКТІВ, ГЛИБИННІ НЕЙРОННІ МЕРЕЖІ, МАШИННЕ НАВЧАННЯ, ЛИЦЬОВІ МАСКИ, ФУНКЦІЯ ВТРАТ, АЛГОРИТМ ОПТИМІЗАЦІЇ

Метою магістерської кваліфікаційної роботи є розв’язання задачі розпізнавання лицьових масок на обличчі з використанням відеокамери в режимі реального часу.

Предметом дослідження є алгоритми виявлення, відслідковування та розпізнавання об’єктів на основі глибокого навчання.

Методами дослідження є моделі згорткових нейронних мереж, програмні бібліотеки для роботи з зображеннями.

У ході виконання кваліфікаційної роботи проводено аналіз підходів та методів виявлення і відстеження об'єктів на відео. Проаналізовано архітектури і типи нейронних мереж та обрана згорткова нейронна мережа для розпізнавання об’єкту на відео.

Проведені експерементальні дослідження, що представлені у вигляді побудованих графіків та розробленого додатку, які показали ефективність застосування згорткових нейронних мереж для розпізнавання лицьових масок на обличчі з використанням відеокамери.

ABSTRACT

Explanatory note of the attestation work: 80 pages, 25 figures, 20 sources.

CONVOLUTION NEURAL NETWORKS, OBJECT DETECTION, DEEP NEURAL NETWORKS, MACHINE LEARNING, FACE MASKS, LOSS FUNCTION, OPTIMIZATION ALGORITHM.

The purpose of the master's certification is to solve the problem of recognizing facial masks on the face using a real-time video camera.

The subject of research is algorithms for detecting, tracking and recognizing objects based on in-depth learning.

Research methods are models of convolutional neural networks, software libraries for working with images.

In the course of attestation work the analysis of approaches and methods of detection and tracking of objects on video is carried out. Neural network architectures and types are analyzed, and a convoluted neural network for video object recognition is selected.

Experimental studies were presented in the form of graphs and an developed application, which showed the effectiveness of the use of convolutional neural networks for the recognition of facial masks on the face using a video camera.

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерної інженерії та управління

Кафедра Комп'ютерних інтелектуальних технологій та систем

АНОТАЦІЯ

КВАЛІФІКАЦІЙНОЇ РОБОТИ

рівень вищої освіти другий (магістерський)

Розпізнавання лицьових масок на обличчі з використанням відеокамери

Виконав:
студент 2 курсу, групи КІТм-20-1
Совєцький М.О.
Спеціальність 123 Комп'ютерна інженерія
Тип програми освітньо-професійна
Освітня програма Комп'ютерні інтелектуальні технології
Керівник проф. Корабльов М.М.

2021 р.

Актуальність теми дослідження. Маска для обличчя стала широко використовуватися в даний час як частина стандартного захисту від поширення вірусів, особливо під час спалаху вірусу Covid-19 в 2020 році. У деяких випадках багатьом людям або організаціям необхідно вміти визначати ситуацію, чи мають люди маски для обличчя в даному місці або в даний час. Ці дані повинні бути автоматизовані та отримані в режимі реального часу. Існує потреба в обчислювальному процесі для аналізу і розпізнавання особи з маскою або без неї.

В магістерській кваліфікаційній роботі досліджено науково-прикладну проблему створення додатку розпізнавання лицьових масок на обличчі за допомогою відеокамери в режимі реального часу, що може дозволити підвищити безпеку людей у місцях соціального скопичення.

Об'єктом дослідження є процеси побудови систем вявлення об'єктів у режимі реального часу.

Предметом дослідження є підходи виявлення і відстеження людей на відео, найпоширеніші додатки машинного навчання, нейронні мережі для розпізнавання об'єктів.

Дослідження грунтується на аналізі розвитку комп'ютерного зору, його недоліках та сильним сторонам ранніх підходів, та методів його застосуванню в минулому та у наш час. Для вирішення поставлених завдань використано: методи локалізаціїї та класифікації, база даних для тренування глибинної нейронної мережі, алгоритм аптомизації, методі об'єктно-орієнтованого програмування.

Метою даної роботи є розробка програмної системи для розпізнавання лицьових масок на обличчі за допомогою відеокамери у режимі реального часу, яка буде здатна використана до практичного застосування у подальшому розвитку.

Вимогами до системи є:

- повинна мати алгоритм машинного навчання, який буде добре підходити під вирішення задачі;

- розроблений додаток, який в кінцевому вигляді повинен будти представлений у виді вікна, в якому буде транслюватись зображення з відеокамери;

- додаток повинен виявляти людей, та класифікувати їх на дві групи (за лицьовою маскою, або без неї) у режимі реального часу.

У першому розділі були розглянуті підходи для виявлення і відстеження людей на відео. Виявлення об'єктів стало свідком швидких революційних змін у сфері комп'ютерного зору. Його участь у комбінації класифікації об'єктів, а також локалізації об'єктів робить його однією з найскладніших тем у галузі комп'ютерного зору. Простіше кажучи, ціль цієї техніки виявлення - визначити, де знаходяться об'єкти на даному зображенні, і до якого класу належить кожен об'єкт. Також були розглянуті процеси виявлення об’єктів (класифікація та локалізація), найпоширеніші застосування, та проблеми у цих процесів. Область виявлення об'єктів не така нова, як може здатися. Фактично виявлення об'єктів розвивалося протягом останніх 20 років. Прогрес в області виявлення об'єктів зазвичай поділяють на два окремі історичні періоди (до і після впровадження Deep Learning):

- період виявлення класичними методами (до 2014 року);
- період виявлення на основі глибинного навчання (після 2014 року)

Були проведені порівняння між класичними методами виявлення об’єктів, та підходами які використовують глибинне навчання.

Недоліки ранніх підходів:

- пропуски об’єктів. В класичних підходах є проблеми з виявленням осіб у різних позах, якщо буде використовуватись певна модель для кожної пози. Загально доступні попередньо навчені моделі призначені для виявлення прямостоячої людини. Вони добре виявляють осіб з лицьової та фронтальної сторони. Проте виявлення осіб з фронтальних сторін в цілому є слабким;

- помилкові виявлення та дуплікати. Об'єкти інших класів часто виявляються як людина. Змінюючи значення параметрів алгоритму можна досягти компромісу між пропущеними та помилковими виявленнями. Визначивши значення мінімального розміру об’єкту виявлення, можна

уникнути від деяких помилкових виявлень;

- обмеження. Хоча гістограма орієнтованих градієнтів (HOG) була досить революційною на початкових етапах виявлення об'єктів, цей метод має багато проблем. Він вимагає багато часу для складних обчислень пікселів на зображеннях та неефективний у деяких сценаріях виявлення об'єктів у вузьких просторах.

Сильні сторони ранніх підходів:

- використання меншої обчислювальної потужності порівняно з сучасними підходами на основі глибинного навчання (не мають потреби в роботі GPU в режимі реального часу). Ці алогоритми є легкими для застосування завдяки реалізації в бібліотеках комп'ютерного зору, таких як OpenCV;

- можливість комбінування з машинами опорних векторів (SVM) для досягнення високої точності виявлення об'єктів;

- створення ефекту ковзного вікна для обчислення кожної позиції;

- створення дескриптора ознак корисного виявлення об'єктів;

Були розглянуті метрики оцінки якості виявлення об’єктів, так як для вирішення поставленого завдання треба розраховувати метрику для двох або більше классів, проведений аналіз показав, що метрика середньої точності (mAP) добре для цього підходить.

Крім того, було визначено, що нейронніх мережі добре піходять для вирішення поставленних задач. Були розглянуті функції активації нейронних мереж, та зроблено постановку задачі.

У другому розділі було розглянуто тему глибинних нейронних мереж, як вони будуються, та чим відрізняються від звичайних нейронних мереж. Одна з ключових відмінностей – алгоритми глибинне навчання масштабуються зі зростанням даних, тоді як неглибинне навчання сходиться. Неглибинне навчання відноситься до методів машинного навчання, які досягають певного рівня продуктивності, коли до мережі додаються більше прикладів та навчальних даних.

Ключовою перевагою мереж глибинного навчання є те, що вони часто

продовжують покращуватися зі збільшенням обсягу даних. У машинному навчанні вручну вибираються ознаки та класифікатор для сортування зображень. У глибинному навчанні витяг ознак і етапи моделювання виконуються автоматично.

Були розглянуті три популярних типи глибинних мереж. Проведений аналіз показав, що багатошарова нейронна мережа та рекурентна нейронна мережа не зможуть показати ефективних показників при вирішенні проблеми розпізнавання об’єкту на зображенні. Але як було зазначено, для вирішення задачі виявлення об'єкта певного класу на фото або відео підходить саме згорткова нейронна мережа, бо її архітектура дозволить зробити модель детектування, яка самостійно знаходить в зображенні високрівневі ознаки, використовуючи лише зображення у якості вхідних даних.

У третьому розділі була побудована архітектура системи, яка складається з декількох основних частин або модулів, кожен з яких виконує свою унікальну функцію:

- модуль машинного навчання, який відповідає за цикли навчання нейронної мережі шляхом прийняття на вхідні дані зображення людей в масках і без них. Потім вихідні ваги між нейронами записуються в окремий файл;

- модуль обработки зображення, який масштабує, обрізає і нормалізує зображення PIL (це безкоштовна додаткова бібліотека для мови програмування Python, яка додає підтримку для маніпулювання безлічі різних форматів файлів зображень) для моделі PyTorch, для передбачення на записі з веб-камери;

- модуль включення відео-камери комп’ютеру, та розпізнавання лицьової маски.

Було обрано набір даних з 1 376 зображень. 690 зображень показують людей з масками на обличчі, а 686 зображень показують людей без масок для тренування згортковоъ нейронноъ мережі. Також обрані додаткові бібліотеки для розробки програмного додатку, наприклад модель виявлення лицьової маски на обличчі буде будуватися за допомогою Sequential API бібліотеки Keras, що дозволяє створювати нові шари для моделі крок за кроком. При розгляданні алгоритмів оптимізації був обраний Адам, який поєднує в собі ідеї

алгоритму середньоквадратичного поширення кореня та алгоритму градієнтного спуску з імпульсом. Оскільки в проекті будуть присутні лише два класи, то втрата бінарної перехресної ентропії добре підходить під функцію втрат у моделі ЗНМ.

Четвертий розділ демонструє програмну реалізацію додатку, який розпізнає лицьові маски на обличчі. Були описані вхідні данні та як вони розділяються на навчальну та тестову множину. Розглядається побудова шарів згорткової нейронної мережі крок за кроком. Після побудови та тренування моделі були показані графіки тренування та втрат, на яких можна побачити ефективність навчення моделі. Наприкінці можна побачити результат роботи програмного додатку, але під час усіх тестувань було виявлено багато збоїв. Є припущення, що набір даних для навчання не забезпечує достатньої кількості даних для усіх можливих сценаріїв, що призводить до неточності навченої моделі та прогнозу. Якщо підвищити кількість даних у декілька разів, та у якості даних брати не лише зображення, а й відео, то модель може потребувати біль ітерацій на навчання, що збільшить витраченого часу на її навчання (при тренуванні з наявними даними та 30 ітераціями знадобилось 20-30 хвилин).

Ключові слова: згоркові нейронні мережі, виявлення об’єктів, глибинні нейронні мережі, мішинне навчання, лицьові маски, функція втрат, алгоритм оптимізації.

Совєцький М.О. Використання згорткових нейронних мереж для виявлення лицьових масок на зображенні і відео // XXV Міжнародний молодіжний форум "РАДІОЕЛЕКТРОНІКА І МОЛОДЬ У XXI СТОЛІТТІ", 21-23 квітня 2021 р., м. Харків

Совєцький М.О. Щодо окремих недоліків нейромереж при створенні анімацій // II Всеукраїнська конференція молодих вчених "ПРАКТИКА ІННОВАЦІЙНОГО ПОШУКУ", 17 грудня 2020 р., м. Дніпро

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

DL – deep learning – глибинне навчання;

IoT – internet of things – інтернет речей;

IoU – intersection over Union – коефіцієнт Жаккара;

IR − інформаційний пошук − information retrieval;

ML − machine learning − машинне навчання;

MLP – multilayer perceptron − багатошаровий перцептрон;

БНМ (НМ) – багатошарова нейронна мережа;

ЗНМ – згорткова НМ;

ОРС – оптичне розпізнавання символів;

РНМ – рекурентна НМ;

ШНМ – штучна нейронна мережа;

ШН – штучний нейрон.

ВСТУП

Маска для обличчя стала широко використовуватися в даний час як частина стандартного захисту від поширення вірусів, особливо під час спалаху вірусу Covid-19 в 2020 році. У деяких випадках багатьом людям або організаціям необхідно вміти визначати ситуацію, чи мають люди маски для обличчя в даному місці або в даний час. Ці дані повинні бути отримані в режимі реального часу і автоматизовані. Існує потреба в обчислювальному процесі (тобто машинному навчанні) для аналізу і розпізнавання особи з маскою або без неї.

Для того щоб досягти машинного навчання, воно починається з нейронної мережі в комп’ютерній науці. Ця техніка побудови комп'ютерної програми, яка навчається на основі даних. Вона дуже слабо заснована на тому, як, на нашу думку працює людський мозок. Спочатку створюється набір програмних "нейронів" і з'єднуються разом, що дозволяє їм посилати повідомлення один одному. Потім мережа просить вирішити задачу, яку вона намагається вирішити знову і знову, кожного разу зміцнюючи зв'язки, що ведуть до успіху і зменшуючи ті, які ведуть до невдачі [1].

Глибинне навчання зробило революцію в розпізнаванні образів і машинному навчанні. Йдеться про нарахування балів в адаптивних системах з довгими ланцюжками потенційно причинно-наслідкових зв'язків між вивченням, дослідженням, і впровадженням.

Давній термін "глибинне навчання" був вперше введений в машинному навчанні Ріною Дехтерем у 1986 році, а в штучних нейронних мережах (ШНМ) - Айзенбергом та іншими (2000). Згодом він став особливо популярним в контексті глибинних НМ, найбільш успішних Deep Learners, які, однак, набагато старше - їх вік налічує півстоліття.

Зображення та відео стали усюдисущими в Інтернеті, що стимулювало розробку алгоритмів, які можуть аналізувати їх семантичний зміст для різних

додатків, включаючи пошук і узагальнення. Нещодавно були продемонстровані згорткові нейронні мережі (ЗНМ) як ефективний клас моделей для розуміння змісту зображень, що дає найсучасніші результати в розпізнаванні, сегментації, виявленні та вилученні зображень і відео. Передбачається, що архітектури ЗНМ здатні до навчання потужним характеристикам на основі слабо маркованих даних, які значно перевершують по продуктивності методи, засновані на характеристиках, і що ці переваги дивно стійкі до деталей зв'язку архітектури в часі. Якісне дослідження виходів мережі і матриць змішування виявляє інтерпретовані помилки.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

Виявлення осіб є життєво важливим і основним кроком для виявлення масок, розпізнавання облич, розпізнавання облич із масками, визначення пози голови та безлічі інших додатків, пов'язаних з особами. Тому зазвичай алгоритми виявлення масок поділяються на дві задачі:

- виявлення осіб на даному зображенні;
- класифікація зображення як особи з маскою або без маски, що є завданням бінарної класифікації.

Визначається додатковий третій клас для відображення невизначеності або, коли маска надіта неправильно. Ця робота є формою класифікації зображень та вилучення ознак. Для таких завдань глибинне навчання виявилося надзвичайно потужним інструментом. Алгоритми глибинного навчання допомагають вилучати з зображень найбільш значущі ознаки завдання класифікації зображень. Велика кількість досліджень була проведена в області виявлення та розпізнавання облич, виявлення об'єктів.

1.1 Огляд підходів виявлення і відстеження людей на відео

Виявлення об'єктів стало свідком швидких революційних змін у сфері комп'ютерного зору. Його участь у комбінації класифікації об'єктів, а також локалізації об'єктів робить його однією з найскладніших тем у галузі комп'ютерного зору. Простіше кажучи, ціль цієї техніки виявлення - визначити, де знаходяться об'єкти на даному зображенні, і до якого класу належить кожен об'єкт.

Одним з перших завдань комп'ютерного зору – це виявлення людей, що почало застосовувати глибинне навчання.

Процес виявлення об’єктів складається з двох етапів:

- визначення наявності об’єкта на зображенні (класифікація);

- визначення розташування цього об’єкта (локалізація).

Найпоширеніші застосування виявлення об'єктів:

- розпізнавання номерних знаків – використання технології виявлення об'єктів та оптичного розпізнавання символів (OPC) для розпізнавання буквено-цифрових символів на автомобілі. Ви можете використовувати об'єкти для захоплення зображень та виявлення транспортних засобів на конкретному зображенні. Як тільки модель виявляє номерний знак, технологія OPC працює над перетворенням двовимірних даних на закодований машиною текст;

- виявлення та розпізнавання осіб - як було зазначено вище, одним з основних застосувань виявлення об'єктів є виявлення та розпізнавання осіб. За допомогою сучасних алгоритмів можна виявити людські особи на зображенні чи відео. Завдяки методам одномоментного навчання стало можливим розпізнавати особи лише по одному навченому зображенню;

- відстеження об'єктів – під час перегляду гри в футбол або баскетбол, м'яч може потрапити далеко. У таких ситуаціях корисно відстежувати рух м'яча та відстань, яку він долає. Для цього відстеження об'єктів може забезпечити безперервну інформацію про напрямок руху м'яча;

- самокеровані автомобілі – для автономних автомобілів дуже важливо вивчати різні елементи навколо автомобіля під час руху. Модель виявлення об'єктів, навчена кількох класах для розпізнавання різних об'єктів, стає життєво важливою роботою автономних автомобілів;

- робототехніка - багато завдань, такі як підйом важких вантажів, операції з підбору та розміщення вантажів та інші виконуються роботами в режимі реального часу. Виявлення об'єктів необхідне роботам для виявлення предметів і автоматизації завдань.

Але є деякі проблеми для вирішення задачі виявлення об'єктів:

- пошук об’єктів невеликого розміру;
- погане освітлення області зору;
- накладання об’єктів, велика концентрація об’єктів;

- наявність некоректних виявлень.

Огляд цих підходів показав, що тема комп’ютерного зору набула широкого росповсюдження у процеі виявлення об’єктів, але треба більш детально розглянути методи в яких це використовується.

Область виявлення об'єктів не така нова, як може здатися. Фактично виявлення об'єктів розвивалося протягом останніх 20 років. Прогрес в області виявлення об'єктів зазвичай поділяють на два окремі історичні періоди (до і після впровадження Deep Learning):

- період виявлення класичними методами (до 2014 року);
- період виявлення на основі глибинного навчання (після 2014 року).

На рис. 1.1 відображені основні етапи розвитку методів виявлення об’єктів:

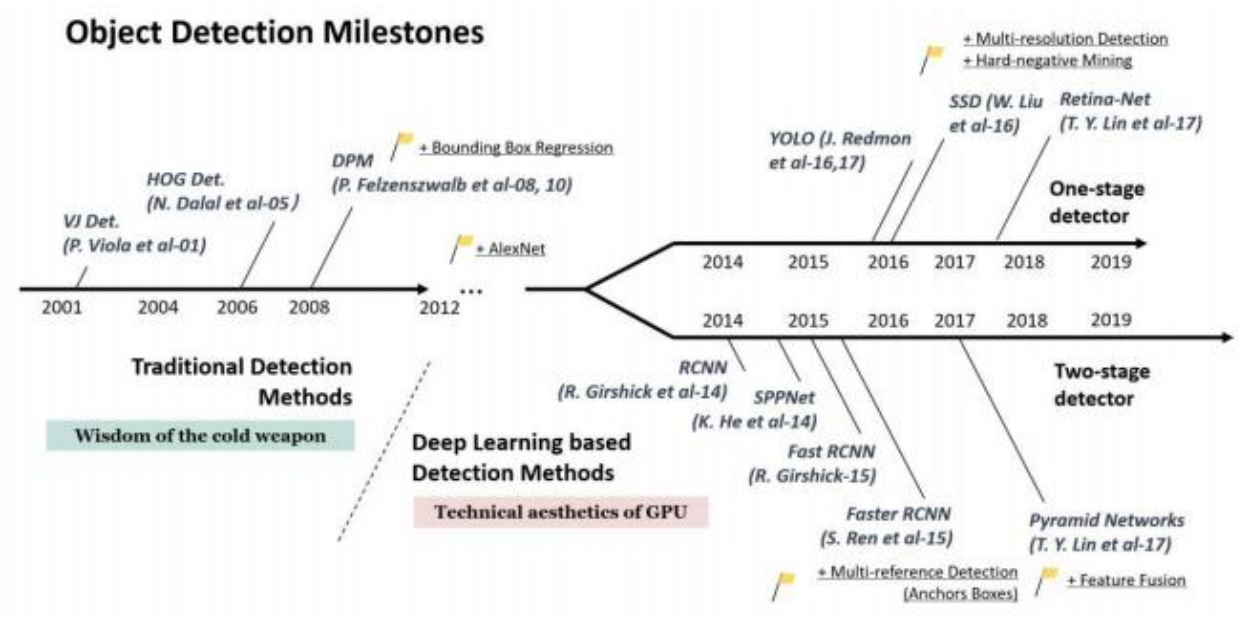

Рисунок 1.1 – Основні етапи розвитку методів виявлення об’єктів [2].

До класичних методів можна віднести гістограму орієнтованих градієнтів – один із найстаріших методів виявлення об'єктів. Вперше він був представлений у 1986 році. Незважаючи на деякі розробки в наступне десятиліття, підхід не набув великої популярності до 2005 року, коли його почали використовувати у багатьох завданнях, пов'язаних із комп'ютерним

зором. HOG використовує дескриптор ознак для ідентифікації об'єктів зображення.

Дескриптор ознак, що використовується в HOG - це уявлення частини зображення, з якого вилучається лише необхідна інформація, ігноруючи все інше. Функція дескриптора ознак полягає у перетворенні загального розміру зображення на форму масиву або вектора ознак. У HOG використовується процедура орієнтації градієнта для локалізації найважливіших частин зображення [3].

Недоліки ранніх підходів:

- пропуски об’єктів. В класичних підходах є проблеми з виявленням осіб у різних позах, якщо буде використовуватись певна модель для кожної пози. Загально доступні попередньо навчені моделі призначені для виявлення прямостоячої людини. Вони добре виявляють осіб з лицьової та фронтальної сторони. Проте виявлення осіб з фронтальних сторін в цілому є слабким;

- помилкові виявлення та дуплікати. Об'єкти інших класів часто виявляються як людина. Змінюючи значення параметрів алгоритму можна досягти компромісу між пропущеними та помилковими виявленнями. Визначивши значення мінімального розміру об’єкту виявлення, можна уникнути від деяких помилкових виявлень;

- обмеження. Хоча гістограма орієнтованих градієнтів (HOG) була досить революційною на початкових етапах виявлення об'єктів, цей метод має багато проблем. Він вимагає багато часу для складних обчислень пікселів на зображеннях та неефективний у деяких сценаріях виявлення об'єктів у вузьких просторах.

Сильні сторони ранніх підходів:

- використання меншої обчислювальної потужності порівняно з сучасними підходами на основі глибинного навчання (не мають потреби в роботі GPU в режимі реального часу). Ці алогоритми є легкими для застосування завдяки реалізації в бібліотеках комп'ютерного зору, таких як OpenCV;

- можливість комбінування з машинами опорних векторів (SVM) для досягнення високої точності виявлення об'єктів;
- створення ефекту ковзного вікна для обчислення кожної позиції;
- створення дескриптора ознак корисного виявлення об'єктів.

Підходи, які використовують глибинне навчання:

- регіональні згорткові нейронні мережі є удосконаленням процедури виявлення об'єктів у порівнянні з попередніми методами HOG. У моделях R-ZHM отримаються найістотніші ознаки (зазвичай близько 2000 ознак), використовуючи селективні ознаки. Процес вибору найбільш значущих витягів може бути обчислений за допомогою алгоритму селективного пошуку, який може досягти цих найважливіших регіональних пропозицій;
- You only look once (YOLO) – одна з найпопулярніших модельних архітектур та алгоритмів для виявлення об'єктів. Зазвичай першою концепцією, яку можна знайти у Google під час пошуку алгоритмів виявлення об'єктів, є архітектура YOLO. Модель YOLO використовує один з найкращих архетипів нейронних мереж для отримання високої точності та загальної швидкості обробки. Ця швидкість та точність є основною причиною її популярності [3].

1.2 Додатки машинного навчання

Існує два основних додатки машинного навчання, які аналізують зображення, що містять особи: виявлення осіб і порівняння осіб.

Система виявлення осіб призначена для відповіді на питання: чи є особа особа на цій фотографії? Система виявлення обличчя розпізнає наявність, місце розташування, масштаб і (можливо) орієнтацію будь-якої особи, присутнього на нерухомому зображенні або відеокадрі. Ця система призначена для виявлення присутність осіб незалежно від таких ознак, як стать, вік і волосся на обличчі.

Об'єкт зазвичай виявляється шляхом сегментації руху у

відеозображенні. Найпростішими підходами виявлення об'єктів є віднімання фону, оптичний потік і метод просторово-часової фільтрації. Вони описані у наступних підрозділах.

Система порівняння осіб призначена для відповіді на питання: чи співпадає особа на зображенні з особою на іншому зображенні? Система порівняння осіб бере зображення особи і робить прогноз збігається ця особа з іншими особами в наданій базі даних. Системи порівняння осіб призначені для порівняння і прогнозування потенційних збігів осіб, незалежно від їх вираження, зачіски на обличчі, і віку.

Як системи виявлення осіб, так і системи порівняння осіб можуть надавати оцінку рівня достовірності передбачення у вигляді ймовірності або довірчої оцінки. Наприклад, система виявлення особи може передбачити, що область зображення є особою з довірчою ймовірністю 90%, а інша область зображення є особою з довірчою ймовірністю 60%. Область з більш високим показником впевненості більш імовірніше має відповідне обличчя.

Якщо система виявлення осіб не виявляє особу належним чином або дає менш достовірне пророкування фактичної особи, це називається пропущеним виявленням або помилково негативним результататом. Якщо система виявлення особи невірно пророкує наявність особи при високому рівні достовірності, це є помилковою тривогою або хибним спрацьовуванням. Аналогічно, система порівняння осіб може не зіставити дві особи, які належать одній людині (пропущене виявлення / помилково негативні результати), або невірно передбачити, що дві особи різних людей - це одна і та ж особа (помилкова тривога / хибним спрацьовуванням).

Показники впевненості є важливим компонентом систем виявлення і порівняння осіб. Ці системи роблять прогнози про наявність особи на зображенні або його збігу з особою на іншому зображенні з відповідним рівнем впевненості в прогнозі. Користувачі цих систем повинні враховувати поріг впевненості / подоби, що надається системою, при розробці свого застосування і прийнятті рішень на основі результатів роботи системи.

Наприклад, у додатку який використовує зображення як віхдні данні, використовуваному для визначення схожих членів сім'ї, якщо поріг впевненості встановлений на 80%, то додаток буде повертати збіги, коли передбачення досягнуто 80% рівня впевненості, але не повертатиме збіги нижче цього рівня. Цей поріг може бути прийнятним, оскільки ризик пропущених виявлень або помилкових тривог для даного типу сценаріїв використання невеликий. Однак в тих випадках, коли ризик пропущеного виявлення або помилкової тривоги вище, система повинна використовувати більш високий рівень довіри. У сценаріях, де важлива висока точність збігу осіб, слід використовувати поріг впевненості / подоби 99%.

1.3 Метрика оцінки якості виявлення об’єктів

Оцінка якості детектування має одночасно враховувати і класифікацію, і локалізацію. Метрикою, яка одночасно може оцінити ці задачі, є Average Precision (AP) [4],

$$AP = \frac{1}{11}\sum Recall_i \tag{1.1}$$

яка пов’язана із визначенням помилок першого і другого роду, де Precision – точність, це здатність класифікатора ідентифікувати лише релевантні об'єкти. Це доля справді позитивних виявлень, а Recall – повнота, з іншого боку, вимірює здатність моделі знаходити всі релевантні випадки (тобто всі істини) – частка справжніх позитивних результатів, виявлених серед усіх істин.

$$Precision = \frac{TP}{TP+FP} \tag{1.2}$$

$$Recall = \frac{TP}{TP+FN} \tag{1.3}$$

Для них важливими є такі компоненти:

TP (True Positive) – кількість об'єктів, які належать до позитивного класу і класифіковані як позитивні.

FP (False Positive) – кількість об'єктів, які належать до негативного класу і класифіковані як позитивні.

FN (False Negative) – кількість об'єктів, які належать до позитивного класу і класифіковані як негативні.

$$Recall_i = [0,0.1,0.2,\dots,1.0]$$

$\mathrm{Pr}ecision(\mathrm{Re}call_i)$ - значення для Precision на кривій ROC-AUC (рис. 1.2) у точках, які приймають значення із $\mathrm{Re}call_i$

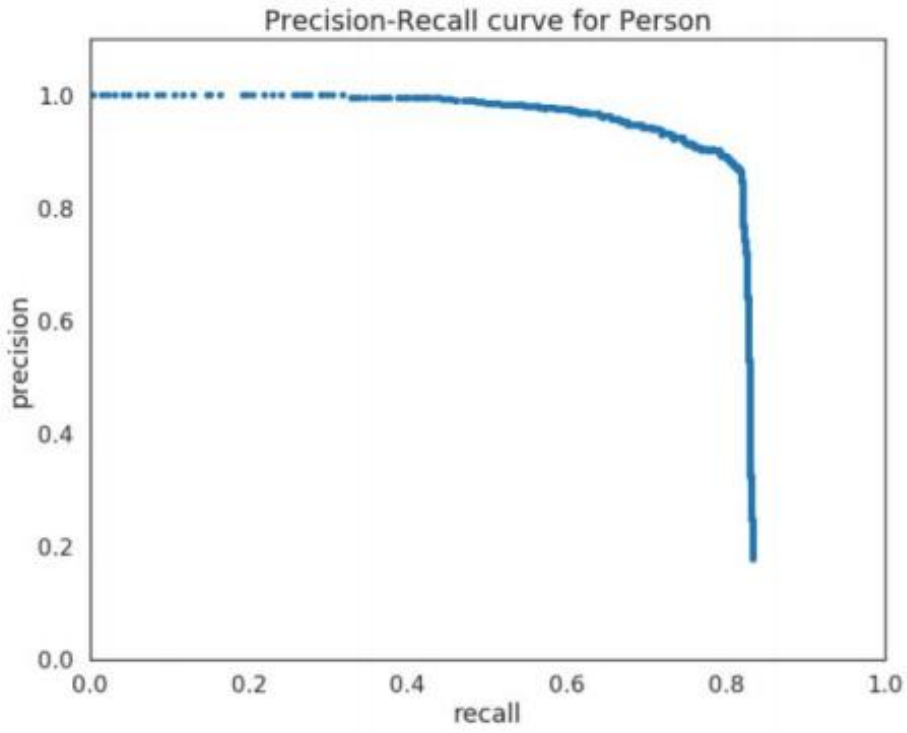

Рисунок 1.2 ─ Приклад побудови кривої ROC-AUC [5]

Для оцінки локалізації використовується метрика Intersection over Union (рис 1.3) (на основі коефіцієнта Жаккара):

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \quad (1.4)$$

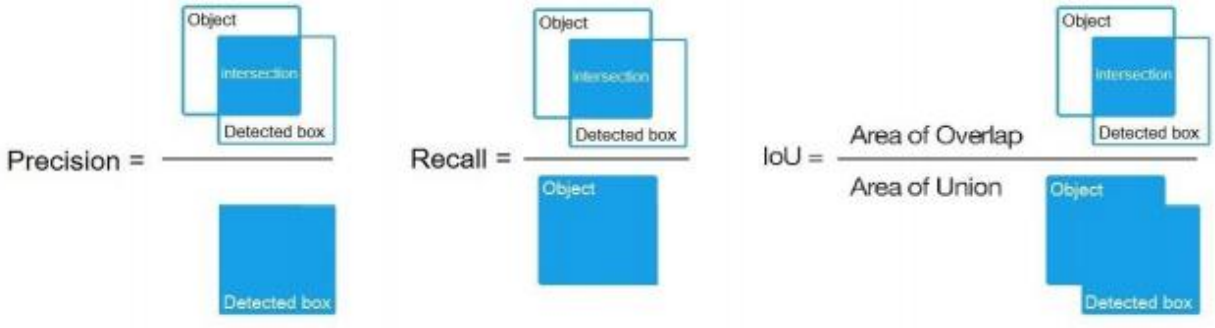

Рисунок 1.3 – Візуалізація знаходження коефіцієнту IoU [6]

Ці дві метрики поєднуються завдяки введенням таких означень для TP, FP, FN:

TP – кількість детекцій з $IoU > 0.5$.

FP – кількість детекцій з $IoU \leq 0.5$ або знайдені більше одного разу.

FN – кількість об'єктів, які не були знайдені, або були знайдені з IoU $\leq$ 0.5.

Average Precision розраховується індивідуально для кожного класу. Це означає, що існує стільки значень AP, скільки класів. Ці значення AP усереднюють отримання метрики: середня середня точність (mAP). Точніше, середня середня точність (mAP) – це середнє значення AP за всіма класами:

$$mAP = \frac{\sum_{q-1}^{Q} AP(q)}{Q}, \quad (1.5)$$

де Q – це кількість наявних у тренувальному наборі класів. Чим вищий показник mAP, тим точніші виявлені рамки об'єктів.

Так як для вирішення поставленого завдання треба розраховувати метрику для двох або більше классів, проведений аналіз показав что метрика середньої точності добре для цього підходить.

1.4 Нейронні мережі для розпізнавання об’єктів

Людський мозок складається приблизно з 1011 обчислювальних одиниць "нейронів", що працюють паралельно та обмінюються інформацією через свої сполучні елементи "синапси"; ці нейрони підсумовують всю інформацію, що надходить до них, і якщо результат вище заданого потенціалу, званого потенціалом дії, вони посилають імпульс по аксону на наступний етап.

Таким чином, штучна нейронна мережа складається з простих обчислювальних одиниць "штучних нейронів", і кожна одиниця з'єднана з іншими одиницями через вагові конектори. Потім ці одиниці обчислюють виважену суму вхідних входів і знаходять вихід за допомогою функції зминання або активації. На рис. 1.4 показано схему штучного нейрона.

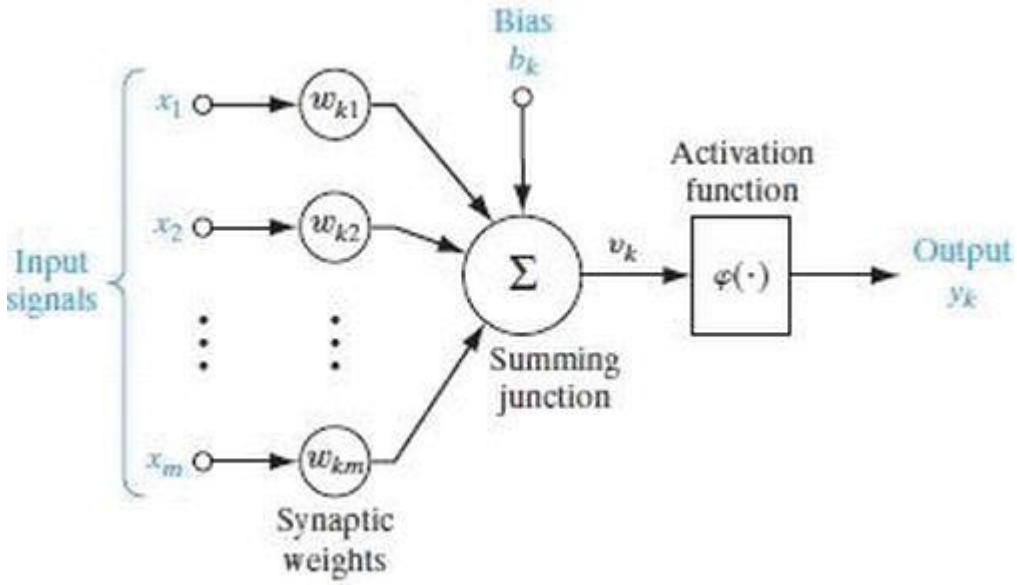

Рисунок 1.4 - Схема штучного нейрона

Виходячи з схеми та функції нейронної мережі, можна виділити три основні елементи нейронної моделі:

- синапси, або сполучні ланки, мають вагу або силу, де вхідний сигнал x_i, підключений до нейрона k, множиться на синаптичну вагу w_{ik};
- суматор для підсумовування завислих входів;

- функція активації отримання вихідного сигналу нейрона. Її також називають функцією, що змінює, оскільки вона змінює (обмежує) амплітудний діапазон вихідного сигналу до кінцевого значення.

Зміщення b_k має ефект збільшення чи зменшення чистого входу функції активації, залежно від цього, позитивний він чи негативний, відповідно.

Нейронна мережа містить шари взаємозалежних вузлів. Кожен вузол є перцептрон і схожий на множинну лінійну регресію. Перцептрон подає сигнал, отриманий внаслідок множинної лінійної регресії, на функцію активації, яка може бути нелінійною.

У багатошаровому перцептроні (MLP) перцептрони розташовані як взаємопов'язані шарів. Вхідний шар збирає вхідні шаблони. Вихідний шар має класифікацію або вихідні сигнали, яким можуть відповідати вхідні шаблони.

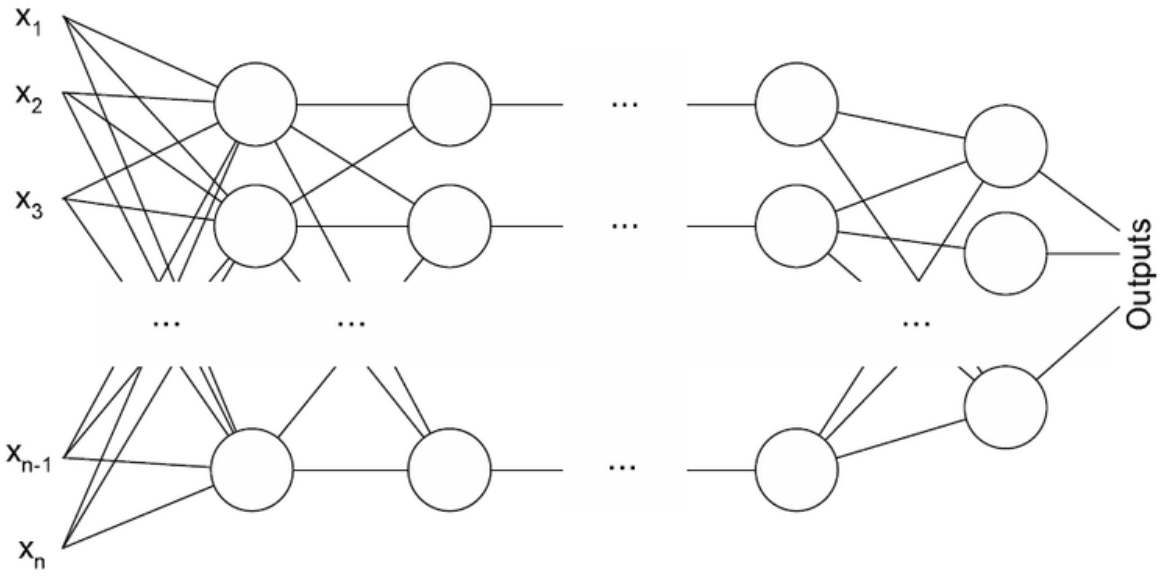

Рисунок 1.5 ─ Архітектура багатошарового перцептрона

Приховані шари точно налаштовують вагові коефіцієнти на вході, поки похибка нейронної мережі стане мінімальною.

Усі завдання класифікації залежить від набору мічених даних. Тобто людина повинна перенести свої знання на набір даних, щоб нейронна мережа

змогла вивчити кореляцію між мітками та даними. Це відомо як навчання під наглядом.

Завдання, які вирішуються за домогою НМ:

- розпізнавання облич, ідентифікація людей на зображеннях, розпізнавання виразів обличчя (сердитий, радісний);
- розпізнавання об'єктів на зображеннях (знаки зупинки, пішоходи, дорожні знаки);
- розпізнавання жестів на відео;
- визначення голосу, ідентифікація дикторів, транскрипція мови у текст, розпізнавання настрою голосі;
- класифікувати текст як спам (в електронних листах) чи шахрайство (у страхових заявах).

Будь-які мітки, які можуть генерувати люди, будь-які результати, які вам важливі і які корелюють із даними, можуть бути використані для навчання нейронної мережі.

Глибинне навчання не потребує позначок для виявлення подібності. Навчання без міток називається навчанням без нагляду. Дані без міток становлять більшість даних у світі. Один із законів машинного навчання говорить, що точність навчання алгоритму залежить від більшої кількості даних. Тому навчання без позначки потенційно здатне створювати високоточні моделі.

1.5 Функції активації нейронів НМ

Функції активації $f(S)$ вводять додатковий крок на кожному шарі під час прямого розповсюдження, але його обчислення того варте.

Якщо нейронна мережа буде працювати без функцій активації, у цьому випадку кожен нейрон виконуватиме лише лінійне перетворення входів за допомогою ваг та зсувів. Тому що не має значення скільки прихованих шарів підключено до нейронної мережі. Всі шари будуть поводитися однаково,

оскільки композиція двох лінійних функцій сама є лінійною функцією. Хоча нейронна мережа стає простішою, навчання будь-якого складного завдання неможливе, і модель буде просто моделлю лінійної регресії.

Необхідно підібрати функцію активації для вихідного шару в залежності від типу задачі прогнозування, що вирішується - зокрема, від типу прогнозованої змінної.

Існує 3 основних типи активаційних функцій:

- бінарна ступінчаста функція;
- лінійна функція активації;
- нелінійні функції активації.

1. Бінарна ступінчаста функція (рис. 1.7) залежить від порогового значення, яке вирішує, чи має бути активований нейрон чи ні.

Вхід, який подається на функцію активації, порівнюється з певним порогом; якщо вхід більший за нього, то нейрон активується, інакше він деактивується, тобто його вихід не передається на наступний прихований шар.

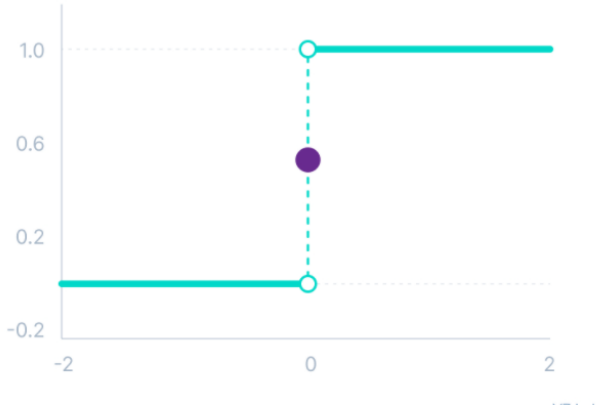

Рисунок 1.6 – Бінарна ступінчаста функція

Математично це можна уявити як:

$$f(S) = \begin{cases} 1, \text{якщо } S \geq 0 \\ 0, \text{якщо } S < 0 \end{cases} \quad (1.6)$$

Ось деякі обмеження двійкової ступінчастої функції:

- вона може забезпечити багатозначні виходи - наприклад, її не можна використовувати на вирішення завдань багатокласової класифікації;
- градієнт ступінчастої функції дорівнює нулю, що заважає процесу зворотного розповсюдження.

2. Лінійна функція активації (рис. 1.8) також відома як функція тотожності, коли активація пропорційна входу.

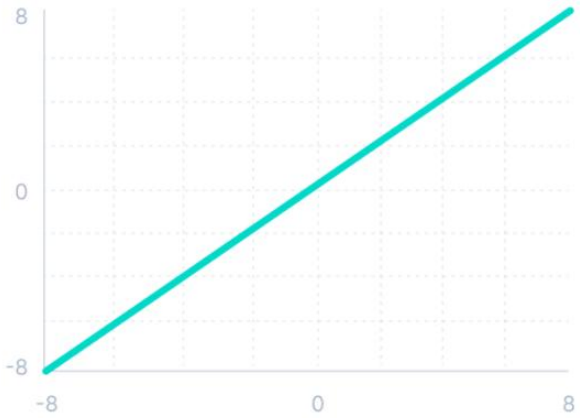

Рисунок 1.7 – Лінійна функція активації

Математично це можна уявити як:

$$f(S) = S \tag{1.7}$$

Проте лінійна функція активації має дві основні проблеми:

- неможливо використовувати зворотне поширення, оскільки похідна функції є константою і не має відношення до входу x;
- при використанні лінійної функції активації всі шари нейронної мережі згортаються в один. Незалежно від кількості шарів у нейронній мережі, останній шар все одно буде лінійною функцією першого шару. Таким чином, по суті, лінійна функція активації перетворює нейронну мережу лише на один шар.

3. Нелінійні функції активації – це та ж сама лінійна функція

активації, але через свою обмежену потужність вона не дозволяє моделі створювати складні відображення між входами та виходами мережі.

Нелінійні функції активації вирішують такі обмеження лінійних функцій активації:

- використовують зворотне поширення, оскільки тепер похідна функції буде пов'язана з входом, і можна повернутися назад і зрозуміти, які ваги у вхідних нейронах можуть забезпечити найкраще передбачення;

- укладають кілька шарів нейронів, оскільки тепер вихід буде нелінійною комбінацією вхідних даних, що пройшли кілька шарів. Будь-який вихід може бути представлений як функціональне обчислення нейронної мережі.

Як приклад можна узяти нейронну мережу для класифікації, де швидше за все, вирішується або бінарна або багатокласова задача класифікації. В останньому випадку дуже ймовірно, що функція активації для останнього шару – це так звана функція активації Softmax, яка призводить до багатокласового розподілу ймовірностей для цільових класів.

Функція Softmax дозволяє нам виразити вхідні дані як дискретного розподілу ймовірності. Математично це визначається так:

$$Softmax(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \tag{1.8}$$

Для кожного значення (тобто входу) у вхідному векторі значення Softmax – це експонента окремого входу, поділена на суму експонент всіх входів. Це гарантує, що станеться кілька речей:

- негативні входи будуть перетворені на невід'ємні значення завдяки експоненційній функції;

- кожен вхід перебуватиме в інтервалі (0,1);

- оскільки знаменник у кожному обчисленні Softmax той самий, значення стають пропорційними одне одному, що гарантує, що у сумі вони

дорівнюють 1.

1.6 Вимоги до системи та постановка задачі дослідження

Метою магістерської кваліфікаційної роботи є вирішення проблеми розпізнавання лицьових масок на обличчі з використанням відеокамери.

Для досягнення мети необхідно вирішити наступні завдання:

- провести аналіз методів виявлення об'єктів на зображенні;
- розглянути типи нейронних мереж;
- проаналізувати архітектури нейронних мереж;
- розробити та навчити модель нейронної мережі для виявлення обличчя та маски на відео;
- кінцевий вигляд програми повинен бути представлений у виді вікна, в якому ведеться запис з відеокамери, де буде розпізнаватись та вказувамитесь наявність макси на обличчі людини в режимі реального часу;
- побудувати графіки, та оцінити ефективність розробленної моделі.

Розвязання цих завдань дозволить створити додаток відстеження наявності лицьової маски на оличчі в режимі реального часу, який можно використовувати для рішення різних практичних задач.

2 ВИБІР АРХІТЕКТУРИ НЕЙРОННОЇ МЕРЕЖІ ДЛЯ РОЗПІЗНАВАННЯ ЛИЦЬОВИХ МАСОК НА ОБЛИЧЧІ

2.1 Глибинні нейронні мережі

Глибинне навчання – це спеціалізована форма машинного навчання. Робочий процес машинного навчання починається з ручного вилучення відповідних характеристик із зображень. Потім ці ознаки використовуються при створення моделі, яка класифікує об'єкти на зображенні. У процесі глибинного навчання відповідні характеристики витягуються із зображень автоматично. Крім того, глибинне навчання здійснює "наскрізне навчання" – коли мережі надаються вихідні дані та завдання, яке необхідно виконати, наприклад, класифікація, і вона автоматично вчиться, як це зробити.

Ще одна ключова відмінність – алгоритми глибинне навчання масштабуються зі зростанням даних, тоді як неглибинне навчання сходиться. Неглибинне навчання відноситься до методів машинного навчання, які досягають певного рівня продуктивності, коли до мережі додаються більше прикладів та навчальних даних.

Ключовою перевагою мереж глибинного навчання є те, що вони часто продовжують покращуватися зі збільшенням обсягу даних. У машинному навчанні вручну вибираються ознаки та класифікатор для сортування зображень. У глибинному навчанні витяг ознак і етапи моделювання виконуються автоматично.

Для навчання глибинної мережі можна використовувати декілька типів навчання, як з учителем (supervised learning), тобто з наявною множиною тренувальними анотованими данними, так і навчання без вчителя (unsupervised learning). Існує три типи шарів нейронів в нейронної мережі: вхідний шар, прихований шар (шари) і вихідний шар. Зв'язки між нейронами мають вагу, яка визначається важливістю елемента вхідних даних.

Глибинною нейронною мережею називається мережа, яка має більше двох прихованих шарів. Для навчання глибинної нейронної мережі необхідно мати великий набір даних.

Нейрони групуються в три різні типи шарів:

- вхідний шар - отримує вхідні дані. Вхідний шар передає вхідні дані першому прихованому шару;
- прихований шар (шари) – виконують математичні обчислення вхідних даних. Одна з труднощів при створенні нейронних мереж полягає у визначенні кількості прихованих шарів та кількості нейронів для кожного шару;
- вихідний шар – повертає вихідні дані.

Термін "глибинний" у глибинному навчанні означає наявність більш ніж одного прихованого шару.

Кожен зв'язок між нейронами асоціюється із вагою. Ця вага визначає важливість вхідного значення. Початкова вага встановлюється випадковим чином.

Кожен нейрон має функцію активації. Ці функції важко зрозуміти без математичних міркувань, простіше кажучи, одна з його цілей – "нормалізувати" вихід нейрона.

Після того, як набір вхідних даних пройшов через усі шари нейронної мережі, вона відправляє вихідні дані назад через вихідний шар.

При контрольованому навчанні машина може навчитися виконувати певне завдання, вивчаючи приклади (дані) завдання, яке вона має виконати.

Для досягнення прийнятного рівня точності програм глибинного навчання потрібен доступ до величезних обсягів навчальних даних та обчислювальної потужності. Оскільки програми глибинного навчання можуть створювати складні статистичні моделі безпосередньо на основі власних ітераційних результатів, вони здатні створювати точні прогностичні моделі на основі великих обсягів немаркованих, неструктурованих даних. Це важливо, оскільки Інтернет речей (IoT) продовжує ставати все більш

поширеним, оскільки більшість даних, створюваних людьми та машинами, є неструктурованими та немаркованими.

Ітераційно порівнюючи вихідні результати з включеними в набір даними, можна обчислити функцію втрат, що вказує, наскільки сильно помиляється алгоритм. Після кожної ітерації (епохи) ваги між нейронами перерозподіляються за допомогою методу градієнтного спуску для мінімізації функції втрат.

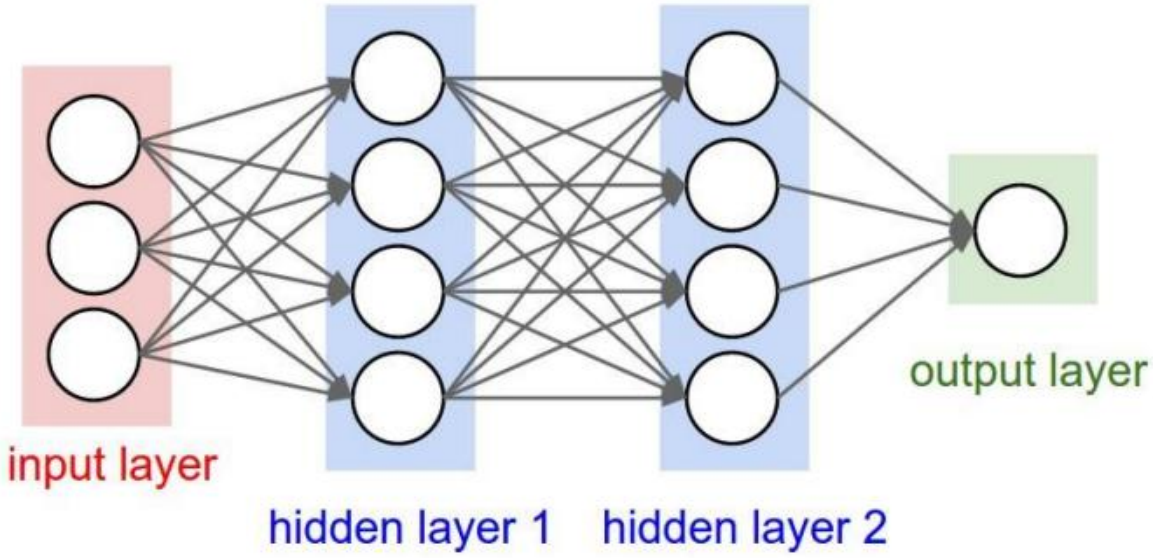

Рисунок 2.1 ─ Схематичне зображення нейронної мережі [7]

Використання нейронних мереж можливе у декількох варіантах:

- тренування штучної нейронної мережі на власних даних;
- використання обраної архітектури штучної нейронної мережі з готовими вагами (pretrained model), натренованої на певному наборі даних для певної кількості класів;
- дотреновування штучної нейронної мережі своїми даними (fine-tuning, transfer learning).

Для тренування нейронної мережі на власних даних вони мають бути анотовані, тобто мати значення правильних міток (ground truth values), які має передбачати нейронна мережа. Отримання таких анотацій зазвичай

потребує значних ресурсів. Модель, натренована на невеликому наборі даних, матиме тенденцію до перенавчання, тобто показувати слабкі результати для примірників, які не входили в тренувальні дані, і матиме слабку генералізацію на зміну умов отримання зображення. Тому набір даних для тренування має мати якомога більше різноманітних варіацій (освітлення, кут повороту, розмір, колір, форма тощо). Самостійне тренування моделі вимагає значних обчислювальних потужностей, проте дозволяє обмежити класи лише до необхідних для задачі. Найоптимальнішим варіантом є дотреновування готової моделі з використанням своїх даних.

Глибинні нейронні мережі навчаються з нуля за допомогою величезних наборів даних, які містять мільйони зображень і зазвичай добре узагальнюються для великої кількості класів. Хоча з усіх цих класів для задач переважно використовується лише декілька, багато зображень об’єктів різних класів мають спільні риси, а функції одного детектора об'єктів одного класу добре працюють під час пошуку об'єкта іншого класу, що дає змогу перетренувати кілька останніх шарів, приймаючи решту як вже побудований екстрактор ознак.

2.2 Типи глибинних нейронних мереж

Сьогодні популярні три наступні типи глибинних нейронних мереж:

- багатошарові нейронні мережі (БНМ);
- рекурентні нейронні мережі (РНМ);
- згорткові нейронні мережі (ЗНМ).

2.2.1 Багатошарові нейронні мережі

Багатошарова нейронна мережа - є групою з кількох перцептронів або нейронів кожному шарі (рис. 2.2). БНМ також відома як нейронна мережа з прямою передачею даних, оскільки вхідні дані обробляються тільки в

прямому напрямку [8].

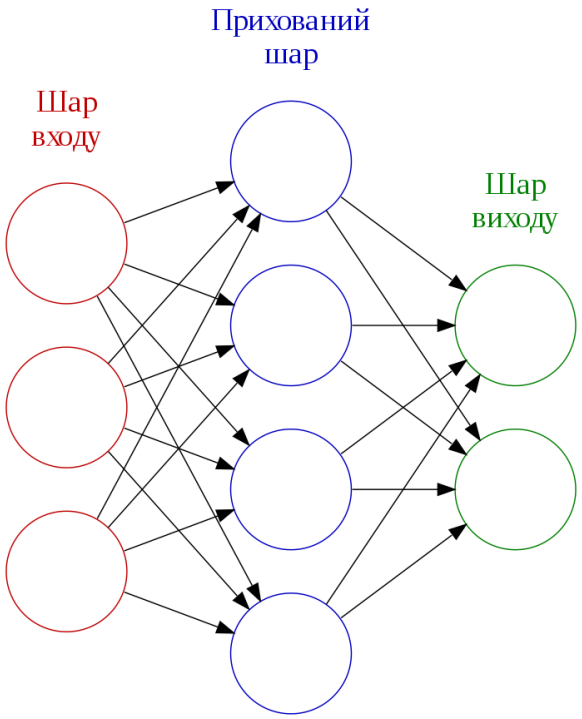

Рисунок 2.2 – Схема простої БНМ [8]

Багатошарова нейронна мережа здатна навчати будь-яку нелінійну функцію. Тому ці мережі відомі як універсальні апроксиматори функцій. БНМ мають здатність до навчання ваг, які відображають будь-який вхід на вихід.

При розв'язанні задачі класифікації зображень за допомогою БНМ першим кроком є перетворення двовимірного зображення на одномірний вектор перед навчанням моделі. Це має два недоліки:

- кількість навчальних параметрів різко зростає зі збільшенням розміру зображення;

- БНМ втрачає просторові особливості зображення. Просторові особливості відносяться до розташування пікселів у зображенні.

Одна загальна проблема всіх цих нейронних мереж – градієнт, що зникає і розривається. Ця проблема пов'язана з алгоритмом зворотнього розповсюдження. Вага нейронної мережі оновлюється за допомогою

алгоритму зворотного розповсюдження шляхом знаходження градієнтів.

БНМ можна використовувати для вирішення проблем, пов'язаних із:

- табличних даних;
- образотворчих даних;
- текстових даних.

Виходячи з проведеного аналізу можна зазначити, що використання БНМ для розпізнавання для розпізнавання лицьових масок на обличчі не являється доцільним.

2.2.2 Рекурентні нейронні мережі

Рекурентні нейронні мережі (рис. 2.3) (РНМ) – це ще один клас штучних нейронних мереж, які використовують послідовну подачу даних. РНМ були розроблені для вирішення проблеми тимчасових рядів послідовних вхідних даних.

Вхід РНМ складається з поточного входу та попередніх зразків. Тому зв'язки між вузлами утворюють спрямований граф вздовж тимчасової послідовності. Крім того, кожен нейрон РНМ має внутрішню пам'ять, яка зберігає інформацію про обчислення з попередніх зразків.

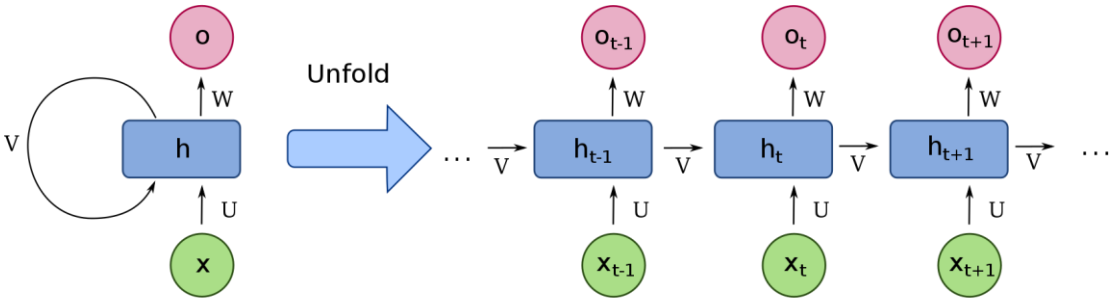

Рисунок 2.3 ─ Розгорнута базова РНМ [9]

Моделі РНМ широко використовуються в обробці природної мови (ОПМ) завдяки перевагі в обробці даних із вхідною довжиною, яка не є

фіксованою. Завдання штучного інтелекту тут - побудувати систему, здатну розуміти природну мову, якою говорить людина, наприклад, моделювання природної мови, вбудовування слів і машинний переклад.

Проведений аналіз показав, що використання РНМ для розпізнавання для розпізнавання лицьових масок на обличчі також не являється доцільним.

2.2.3 Згорткові нейронні мережі

Згорткові нейронні мережі (ЗНМ) - це нейронні мережі, найбільш часто використовувані для аналізу зображень. ЗНМ отримує на вхід зображення у вигляді тривимірної матриці. Перші два вимірювання відповідають ширині і висоті зображення в пікселях, а третє - значенням RGB кожного пікселя.

ЗНМ складаються з наступних послідовних модулів (кожен може містити більше одного шару):

- згортка;
- функція активації ReLu;
- об'єднання;
- повністю підключенні шари;
- вихідний шар.

Операція згортки – це операція множення матриці за елементами. Згорткові шари беруть тривимірну вхідну матрицю, про яку було сказано раніше, і пропускають фільтр (також відомий як згорткове ядро) через зображення, застосовуючи його до невеликого вікна пікселів за раз (тобто 3x3 пікселя) і переміщаючи це вікно, поки не буде просканувати всі зображення. Операція згортки обчислює точковий добуток значень пікселів в поточному вікні фільтра разом з вагами, визначеними в фільтрі. Результатом цієї операції є кінцеве згорнуте зображення.

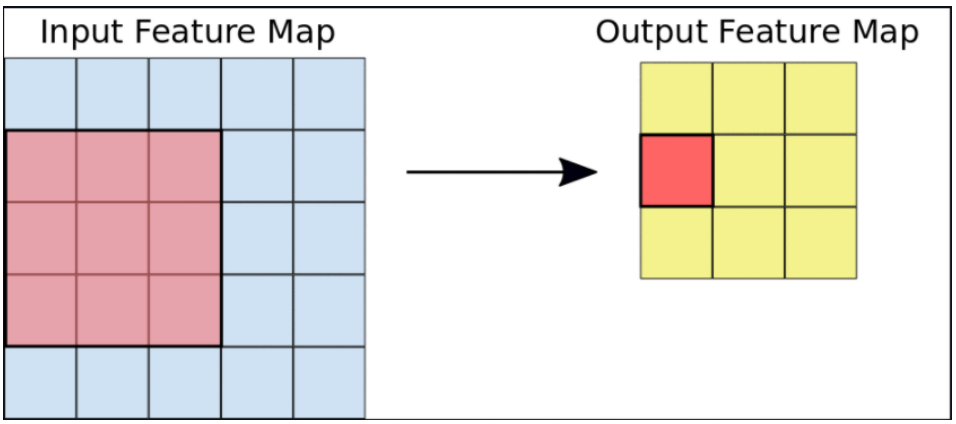

Рисунок 2.4 – Згортка 3x3 глибиною 1, виконана над вхідними картою ознак 5x5, також глибиною 1[10]

Суть ЗНМ для класифікації зображень полягає в тому, що в міру навчання моделі вона дізнається значення матриць фільтрів, які дозволяють їй отримувати важливі особливості (форми, текстури, кольорові області і т.д.) з зображення. Кожен згортковий шар застосовує один новий фільтр до згорнутому зображенню попереднього шару, який може отримати ще одну особливість. Таким чином, чим більше фільтрів, тим більше особливостей ЗНМ може витягти з зображення.

Після кожної операції згортки ЗНМ застосовує на виході функцію Rectified Linear Unit (ReLu) до згорнутого зображення. Вона вносить нелінійність в модель, що допомагає краще узагальнюватися і уникати переоцінки.

Об'єднання – це процес, при якому ЗНМ знижує вибірку згорнутого зображення, зменшуючи число вимірювань карти ознак. Це робиться для скорочення часу обробки та обчислювальної потужності. Під час цього процесу зберігається найбільш важлива інформація про ознаки. Існує кілька методів, які можуть бути використані для об'єднання. Найбільш поширені з них - максимальне об'єднання і середнє об'єднання. Проекту курсової роботи більш підходить максимальне об'єднання, так як воно найбільш ефективно в більшості випадків. Максимальна об'єднання дуже схоже на процес згортки. Вікно ковзає по карті характеристик і витягує плитки певного розміру. Для

кожної плитки max pooling вибирає максимальне значення і додає його в нову карту характеристик (рис. 2.5).

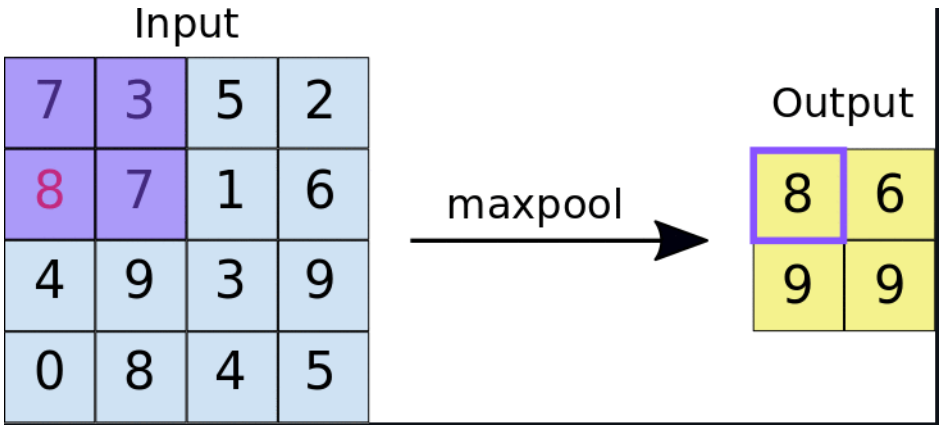

Рисунок 2.5 – Максимальна об'єднання, виконане над картою характеристик 4x4 з фільтром 2x2 і кроком 2[10]

Після об'єднання завжди залишається один або кілька повністю пов'язаних шарів. Ці шари виконують класифікацію на основі ознак, витягнутих з зображення за допомогою раніше згаданих процесів згортки. Останній повністю підключений шар – вихідний шар, який застосовує функцію softmax до виходу попереднього повністю підключеного шару і повертає ймовірність для кожного класу.

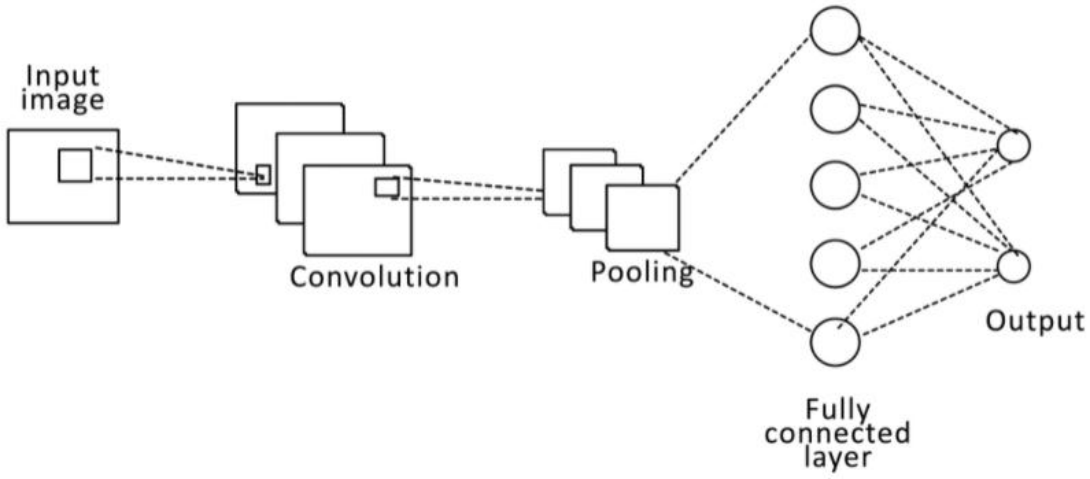

Рисунок 2.6 – Загальна форма ЗНМ для класифікації зображень

Як можна побачити, для вирішення задачі виявлення об'єкта певного класу на фото або відео підходить саме згорткова нейронна мережа, бо її архітектура дозволить зробити модель детектування, яка самостійно знаходить в зображенні високрівневі ознаки, використовуючи лише зображення у якості вхідних даних.

Визначення того, чи містить зображення людини в масці чи ні, є простим завданням класифікації. Треба класифікувати зображення між двома дискретними класами: ті, які містять маску на обличчі, і ті, які не містять. Для цього потрібно мати базу даних фотографій людей з масками на обличчі, та без масок.

3 СТВОРЕННЯ СИСТЕМИ ДЛЯ РОЗПІЗНАВАННЯ ЛИЦЬОВИХ МАСОК НА ОБЛИЧЧІ

Для розробки обрано ОС Windows 10. Ця операційна система добре поширена, має сильну підтримку з боку виробника.

Була обрана мова програмування Python, тому що вона добре підходить для розробки алгоритмів машинного навчання, та має декілько необхідніх додаткових бібліотек.

Для мови програмування Python добре підходить середа розробки PyCharm.

3.1 Вибір бази даних для навчання нейронної мережі

Існує безліч наборів даних осіб, які можуть бути використані для оцінки алгоритмів глибинного навчання. Є кілька популярних наборів даних для адаптації розпізнавання осіб: Labelled Faces in the Wild: 13 000 зображень облич, зібраних з Інтернету, позначених ім'ям людини, Toronto Face Dataset, Olivetti: кілька зображень декількох різних людей, Multi-Pie: CMU Multi- PIE База даних осіб, Face-in-Action, JACFEE: японські і кавказькі особи, FERET: база даних технологій розпізнавання осіб, mmifacedb: MMI Facial Expression Database, IndianFaceDatabase, і багато інших джерел.

Для цілей даного проєкту, щоб впоратися з завданням розпізнавання осіб з маскою, існує три типи наборів даних осіб з маскою, включаючи Masked Face Detection Dataset (MFDD), Real-world Masked Face Recognition Dataset (RMFRD) і Simulated Masked Face Recognition Dataset (SMFRD).

Існують також загальнодоступні набори даних, наприклад, VGGFace2 - великомасштабний набір даних для розпізнавання осіб. Зображення завантажуються з Google Image Search і мають великі відмінності в позі, віці, освітленні, етнічної приналежності і професії. Набір даних містить 3,31

мільйона зображень 9131 випробуваного, в середньому 362,6 зображень для кожного випробуваного [11]. Набір даних для розпізнавання осіб з маскою повинен включати кілька зображень одного і того ж суб'єкта з маскою і без маски.

Для простоти було обрано набір даних з 1 376 зображень. 690 зображень показують людей з масками на обличчі, а 686 зображень показують людей без масок.

3.2 Вибір бібліотек для класифікації зображення

Було обрано бібліотеки Tensorflow, Keras і OpenCV для створення ЗНМ, які класифікують зображення як з маскою і без маски.

TensorFlow – інтерфейс для вираження алгоритмів машинного навчання, використовується для впровадження систем ML в виробництві у багатьох областях комп'ютерних наук, включаючи аналіз настроїв, розпізнавання голосу, витяг географічної інформації, комп'ютерний зір, резюмування тексту, інформаційний пошук [12]. У моделі вся архітектура ЗНМ (що складається з декількох шарів) використовує TensorFlow на бекенд. Він також використовується для зміни форми даних (зображення) при обробці даних.

Keras надає фундаментальні відображення і будівельні блоки для створення і транспортування ML-систем з високою швидкістю ітерацій. Він використовує всі переваги масштабованості і крос-платформних можливостей TensorFlow. Основними структурами даних Keras є шари і моделі [13]. Всі верстви, які використовуються в моделі ЗНМ, реалізовані за допомогою Keras. Поряд з перетворенням вектора класів в бінарну матрицю класів при обробці даних, він допомагає скласти загальну модель.

OpenCV (Open Source Computer Vision Library) – програмна бібліотека комп'ютерного зору і ML з відкритим вихідним кодом, використовується для розрізнення і розпізнавання осіб, розпізнавання об'єктів, угруповання рухів в

записах, відстеження прогресивних модулів, відстеження жестів очей, відстеження дій камери, виключення червоних очей зі знімків, зроблених з використанням спалаху, пошуку порівняльних знімків з бази даних зображень, сприйняття ландшафту і установки маркерів для накладення на нього збільшеною реальності і так далі [14]. Модель використовує ці можливості OpenCV при зміні розміру і перетворенні кольору зображень даних.

3.3 Створення архітектури системи

Система складається з декількох основних частин або модулів, кожен з яких виконує свою унікальну функцію:

- модуль машинного навчання, який відповідає за цикли навчання нейронної мережі шляхом прийняття на вхідні дані зображення людей в масках і без них. Потім вихідні ваги між нейронами записуються в окремий файл;

- модуль обработки зображення, який масштабує, обрізає і нормалізує зображення PIL (це безкоштовна додаткова бібліотека для мови програмування Python, яка додає підтримку для маніпулювання безлічі різних форматів файлів зображень) для моделі PyTorch, для передбачення на записі з веб-камери;

- модуль включення відео-камери комп’ютеру, та розпізнавання лицьової маски.

Модель виявлення лицьової маски на обличчі буде будуватися за допомогою Sequential API бібліотеки Keras. Це дозволяє створювати нові шари для моделі крок за кроком.

Для виявлення обличчя буде використовуватись каскадні класифікатори з урахуванням ознак Хаара. Ознака Хаара складається із суміжних прямокутних областей. Вони позиціонуються на зображенні, далі підсумовуються інтенсивності пікселів в областях, після чого обчислюється

різниця між сумами. Ця різниця і буде значенням певної ознаки, певного розміру, певним чином позиціонованого на зображенні [15]. Це алгоритм машинного навчання для виявлення об'єктів, що використовується для ідентифікації об'єктів на зображенні або відео і заснований на концепції ознак, запропонованої Полом Віолою та Майклом Джонсом. Спочатку алгоритму потрібно безліч позитивних (зображення з обличчям) та негативних (зображення без облич) зображень для навчання класифікатора. Потім етапі виявлення у методі вікно встановленого розміру рухається зображення, і кожної області зображення, над якою проходить вікно, розраховується ознака Хаара. Наявність або відсутність предмета у вікні визначається різницею між значенням ознаки та порогом, що навчається [15]. І тому використовуються ознаки Хаара, показані на рис. 3.1. Кожна ознака - це одне значення, отримане шляхом віднімання суми пікселів під білим прямокутником із суми пікселів під чорним прямокутником.

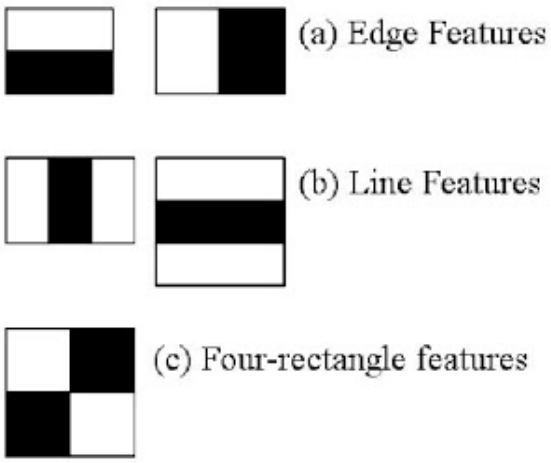

Рисунок 3.1 – Ознаки Хаара [16]

Каскадний класифікатор, який буде використовуватись – це Face Detection Cascade Classifier. Цей каскадний класифікатор розроблений OpenCV. У ньому попередньо навчена модель з передніми рисами обличчя буде використана для виявлення осіб в режимі реального часу.

Нарешті, модель ЗНМ разом із каскадним класифікатором навчається

протягом 30 циклів із двома класами, одне із яких позначає клас зображень із лицьовими масками на обличчі, а інший – без лицьових масок.

3.4 Вибір алгоритму оптимізації навчання нейронної мережі

Функція втрат – це міра того, наскільки добре модель прогнозування передбачає очікуваний результат (або значення).

У процесі навчання нейронної мережі треба мінімізувати втрати функції та оновлювати параметри підвищення точності. Параметри нейронної мережі це зазвичай ваги зв'язків.

Таким чином, оптимізатор – це метод досягнення найкращих результатів, допомога у прискоренні навчання. Іншими словами, це алгоритм, який використовується для незначної зміни параметрів, таких як ваги та швидкість навчання, щоб модель працювала правильно та швидко [17].

Вибір алгоритму оптимізації для моделі глибинного навчання може означати різницю між хорошими результатами за хвилини, години та дні.

3.4.1 Градієнтний спуск

Градієнтний спуск (рис. 3.2), ймовірно, є найпопулярнішим і найширше використовується з усіх оптимізаторів. Це простий та ефективний метод пошуку оптимальних значень для нейронної мережі. Метою всіх оптимізаторів є досягнення глобального мінімуму, за якого функція вартості досягає найменшого можливого значення.

Щоразу, коли знаходиться градієнт і оновлюється значення ваги та зміщення, відбувається наближення до оптимального значення. До початку навчання нейронної мережі витрати будуть високими. На кожній ітерації навчання нейронної мережі (пошук градієнтів та оновлення ваг та зсувів) вартість зменшується та наближається до глобального мінімального значення

Але не завжди функція втрат є гладкою. У багатьох випадках ці функції можуть бути невипуклими. Проблема з опуклою функцією полягає в тому, що існує ймовірність того, що вона може застрягти в локальному мінімумі, і втрати ніколи не зійдуться до глобального мінімального значення.

3.4.2 Градієнтний спуск з імпульсом

У більшості випадків градієнтний спуск з імпульсом сходиться швидше, ніж стандартний алгоритм градієнтного спуску. У стандартному алгоритмі градієнтного спуску робляться більші кроки в одному напрямку та менші кроки в іншому напрямку, що уповільнює алгоритм. На зображенні нижче видно, що стандартний градієнтний спуск робить великі кроки у напрямку "y" менші кроки у напрямку "x".

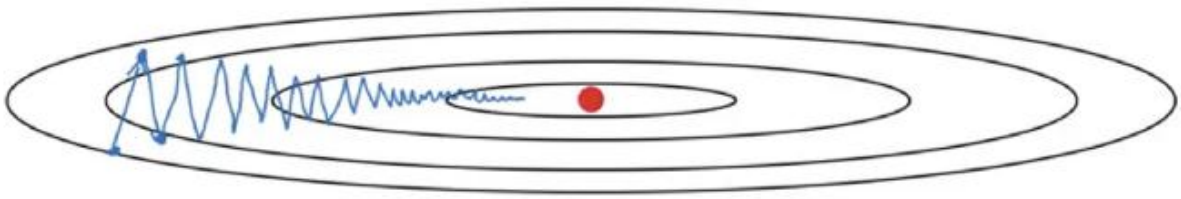

Рисунок 3.2 – Стандартний градієнтний спуск

3.4.2 Середньоквадратичне поширення

Середньоквадратичне поширення кореня (RMSprop) - це експоненційно загасаюче середнє значення. Оптимізатор RMSprop схожий на алгоритм градієнтного спуску з імпульсом. Оптимізатор RMSprop обмежує коливання у вертикальному напрямку. Таким чином, ми можемо збільшити швидкість навчання, і наш алгоритм може робити великі кроки у горизонтальному напрямку, сходячись швидше. Різниця між RMSprop та градієнтним спуском полягає в тому, як обчислюються градієнти [18].

3.4.3 Алгоритм оптимізації Адам

Алгоритм оптимізації Адам - один із найефективніших алгоритмів оптимізації у навчанні нейронних мереж. Він поєднує в собі ідеї алгоритму середньоквадратичного поширення кореня та алгоритму градієнтного спуску з імпульсом. Замість адаптувати швидкість навчання параметрів на основі середнього першого моменту (середнього значення), як у RMSProp, Адам також використовує середнє значення других моментів градієнтів. Зокрема, алгоритм обчислює експоненційне ковзне середнє градієнта та квадратичний градієнт [18].

3.5 Вибір функції втрат

У рамках алгоритму оптимізації помилка для поточного стану моделі має оцінюватися багаторазово. Це вимагає вибору функції помилки, умовно званої функції втрат, яка може бути використана для оцінки втрат моделі, щоб ваги можна було оновити для зменшення втрат при наступній оцінці.

Вибір функції втрат повинен відповідати рамкам конкретного завдання прогностичного моделювання, наприклад класифікації або регресії. Крім того, конфігурація вихідного шару також має відповідати вибраній функції втрат.

3.5.1 Функція втрат для класифікації

Проблеми класифікації пов'язані з прогнозуванням дискретного висновку класу. Це передбачає поділ набору даних на різні та унікальні класи на основі різних параметрів таким чином, щоб новий та невідомий запис міг бути віднесений до одного з класів.

Лист можна класифікувати як спам або не спам, а дієтичні переваги людини можна віднести до однієї з трьох категорій - вегетаріанці,

невегетаріанці і вегани.

3.5.2 Втрата бінарної перехресної ентропії

Це найбільш поширена функція втрат, яка використовується для класифікаційних завдань, в яких є два класи. Слово "ентропія", що здається недоречним, має статистичну інтерпретацію.

Ентропія – це міра випадковості в оброблюваної інформації, а крос-ентропія – міра різниці випадковості між двома випадковими величинами.

Якщо розбіжність передбачуваної ймовірності з фактичною міткою збільшується, зростає втрата перехресної ентропії. Виходячи з цього, прогноз ймовірності .011 при фактичної мітці спостереження 1 призведе до великих втрат. В ідеальній ситуації "досконала" модель мала б логарифмічну втрату 0. Якщо подивитися на функцію втрат, все стане ще зрозуміліше.

$$J = -\sum_{i-1}^{N} y_i \log\big(h_\theta(x_i)\big) + (1 - y_i)\log(1 - h_\theta(x_i)), \qquad (3.1)$$

де y_i – істинна мітка, а $h_\theta(x_i)$ – передбачене значення після гіпотези.

Оскільки бінарна класифікація означає, що класи набувають значення 0 або 1, якщо y_i = 0, цей термін перестає існувати, а якщо y_i = 1, то $(1 - y_i)$ термін стає рівним 0 [19].

Оскільки в проекті будуть присутні лише два класи, то втрата бінарної перехресної ентропіїї добре підходить під функцію втрат у моделі ЗНМ.

4 ПРОГРАМНА РЕАЛІЗАЦІЇ СИСТЕМИ

4.1 Опис вхідних даних

Вхідними даними є 690 зображень з обличчями людей без лицьової маски, та 686 зображень з обличчями людей з лицьовою маскою, які були взяті з публічного проекту на сайті GitHub [20] (рис. 4.1).

Для застосування алгоритмів комп’ютерного зору, в даних мають бути присутні проблеми:

- наявність зображень однієї і тієї самої людини за накладанням на нього додаткового шуму або розмиття;
- наявність зображень, які відзеркаленні по одній с ось;
- кожне зображення може мати різну роздільну здатність;
- наявність однакових зображень, які мають різні обертання.

Рисунок 4.1 – Приклади особливостей вхідних даних

4.2 Розділення даних для тренування

На цьому етапі йде розділення даних на навчальну множину, що містить зображення, на яких навчатиметься модель ЗНМ, і тестова множина із зображеннями, на яких тестуватиметься наша модель.

При цьому розроблена функція train_test_split буде приймати значення

split_size 0.8, що означає, що 80% всіх зображень потраплять в навчальну множину, а 20% зображень, що залишилися - в тестову множину (рис.4.2).

```
The number of images with facemask: 690
The number of images without facemask: 686
train images with mask: 552
test images without mask: 138
train images with mask: 548
test images without mask: 138
```

Рисунок 4.2 – Відображення розділення даних на множину навчання та тестування

Після розбиття ми бачимо, що необхідний відсоток зображень був розподілений як на навчальну, так і на безліч тестів, як згадувалося вище.

4.3 Побудова моделі

Будується модь послідовної ЗНМ із різними шарами.

Перший шар - це шар Conv2D (цей шар створює ядро згортки, яке згортається із входом шару для отримання виходів) зі 100 фільтрами та розміром фільтра або ядра 3X3. На цьому першому етапі використовується функція активації "ReLu". Ця функція ReLu означає лінійну одиницю, яка виводить вхідні дані якщо вони позитивні, інакше виводиться нуль. Розмір вхідних даних також ініціалізується як 150X150X3 для всіх зображень, які будуть навчені та протестовані за допомогою цієї моделі.

Потім використовується MaxPooling2D, максимальне об'єднання для зменшення просторових розмірів вихідного обсягу.

Наступний шар - знову шар Conv2D з ще 100 фільтрами того ж розміру 3X3, а як функція активації використовується "ReLu". За цим шаром Conv2D слід шар MaxPooling3=2D з розміром блоку 2X2.

На наступному етапі використовується шар Flatten(), щоб перетворити всі шари в один шар 1D.

Після шару Flatten використовується шар Dropout (0.5), щоб запобігти переоцінки моделі.

Наступний буде використовуватись щільний шар (це шар нейронної мережі, де кожен нейрон щільного шару отримує вхід від усіх нейронів попереднього шару, та генерує щільний шар на вихідний сигнал, який являє собою вектор) із 50 одиницями та функцією активації "ReLu".

Останній шар моделі буде ще одним щільним шаром, з двома одиницями, а функція активації, що використовується, буде функцією "Softmax". Функція Softmax виводить вектор, який є розподілом ймовірностей кожної з вхідних одиниць. Тут використовуються дві вхідні одиниці. Функція Softmax виводить вектор із двома значеннями розподілу ймовірностей (рис. 4.3).

Після побудови моделі (рис. 4.3), вона компілюється та визначається функція втрат та оптимізатор. У цій моделі був обраний оптимізатор Адам та бінарна перехресна ентропія як функцію втрат для навчання.

```
model = tf.keras.models.Sequential([
    Conv2D(100, (3,3), activation='relu', input_shape=(150, 150, 3)),
    MaxPooling2D(2,2),
    Conv2D(100, (3,3), activation='relu'),
    MaxPooling2D(2,2),
    Flatten(),
    Dropout(0.5),
    Dense(50, activation='relu'),
    Dense(2, activation='softmax')   # dense layer has a shape of 2 as we
have only 2 classes
])
model.compile(optimizer='adam',              loss='binary_crossentropy',
metrics=['accuracy'])
```

Рисунок 4.3 – Формування ЗНМ моделі

Побудовану модель ЗНМ можна графічно побачити на рис. 4.4.

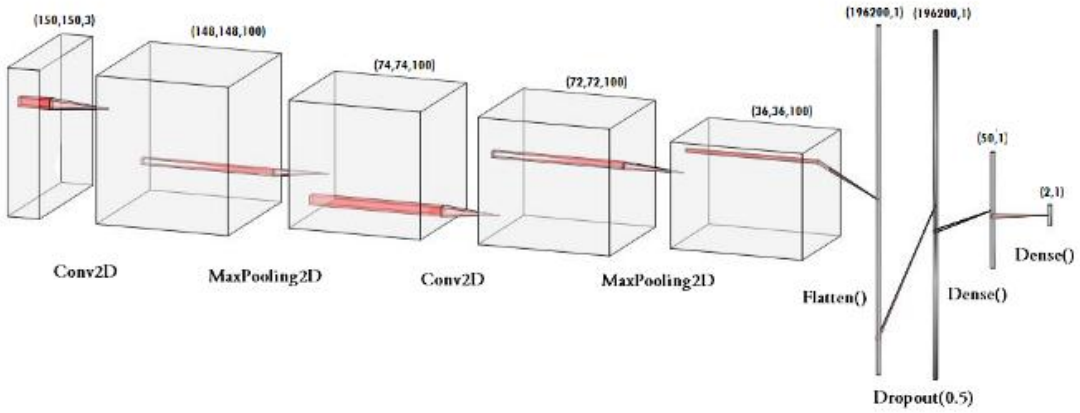

Рисунок 4.4 – Графічний приклад моделі ЗНМ для лицьових масок

4.4 Тренування моделі

Навчяння моделі за допомогою наступних функцій. Спочатку відкриваються 2 навчальні потоки з двох каталогів навчальних та тестових (валідаційних) зображень. Потім використовується функція ImageDataGenerator з бібліотеки Keras (рис. 4.5), що дозволяє доповнювати зображення в режимі реального часу, поки модель ще вчиться. Ця функція застосовує будь-які довільні перетворення до кожного навчального зображення, коли воно передається в модель.

Доповнення зображення – це техніка застосування різних перетворень до вихідного зображення, у результаті виходить кілька перетворених копій однієї й тієї зображення. Кожна копія, однак, відрізняється від іншої у певних аспектах залежно від застосовуваних технік збільшення, таких як зсув, поворот, перевертання тощо.

Застосування цих невеликих варіацій до вихідного зображення не змінює його цільовий клас, лише дозволяє по-новому поглянути на об'єкт у реальному житті.

Ці методи доповнення зображень не тільки розширюють розмір набору даних, але й включають рівень варіативності, що дозволяє моделі краще

узагальнювати дані, що не зустрічаються в полі зору. Крім того, модель стає більш стійкою під час навчання на нових, трохи змінених зображеннях.

Поворот зображення є однією з технік, що широко використовуються, доповнення і дозволяє моделі стати інваріантною до орієнтації об'єкта. В тренуванні зображення буде довільно обертатися в діапазоні від 0 до 40 градусів.

При повороті зображення деякі пікселі виходять за межі і залишають порожню область, яку необхідно заповнити. Її можна заповнити різними способами, наприклад постійним значенням, найближчими значеннями пікселів і т. д. Це вказується в аргументі fill_mode. Було обрано використовувати значення "nearest", яке просто замінює порожню область найближчими значеннями пікселів.

Може статися так, що об'єкт не завжди знаходиться у центрі зображення. Щоб подолати цю проблему, можна посунути пікселі зображення по горизонталі або вертикалі, це робиться шляхом додавання певного постійного значення всіх пікселів.

Клас ImageDataGenerator має аргумент height_shift_range для вертикального зсуву зображення і width_shift_range для горизонтального зсуву зображення. Якщо значення є плаваючим числом, це вказує на відсоток ширини чи висоти зображення, який потрібно зрушити. В іншому випадку, якщо це ціле значення, то просто ширина або висота зсуваються на кількість пікселів.

Перевертання зображень також є чудовою технікою доповнення, і має сенс використовувати її з великою кількістю різних об'єктів.

Параметр horizontal_flip і vertical_flip для перевертання вертикальної або горизонтальної осі. Однак, ця техніка повинна відповідати об'єкту на зображенні. Наприклад, вертикальне перевертання автомобіля не буде розумним у порівнянні з симетричним об'єктом, таким як футбол чи щось інше. Проте є намір перевернути зображення тільки в горизонтальному напрямку.

Збільшення масштабу або довільно наближає зображення або віддаляє його.

Було надано число з плаваючою точкою в аргумент zoom_range, що означає масштабування зображення в діапазоні 1-zoom_range, 1+zoom_range.

Зрушення об’єкту означає, що зображення буде спотворено вздовж осі, переважно для створення або виправлення кутів сприйняття. Зазвичай це використовується для доповнення зображень, щоб комп'ютери могли побачити, як людина бачить речі під різними кутами.

```
train_datagen = ImageDataGenerator(rescale=1.0/255,
                                   rotation_range=40,
                                   width_shift_range=0.2,
                                   height_shift_range=0.2,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   horizontal_flip=True,
                                   fill_mode='nearest')
```

Рисунок 4.5 – Метод доповнення зображення перед тренуванням

Останнім кроком є підстановка зображень з навчальної та тестової множини до послідовної моделі, яка була побудувана за допомогою бібліотеки keras. Навчання моделі йде протягом 30 епох (ітерацій). Однак можна навчити модель на більшу кількість епох для досягнення вищої точності.

Під час навчання зберігаються контрольні точки в окремих каталогах кожної контрольної точки, щоб зберегти модель з найкращою точністю та найменшими втратами. Нарешті, викликається функція fit_generator (рис. 4.6) моделі і навчання починається. Під час процесу відстежується точність навчання та перевірки, а також втрати (ці значення використовуватимесь пізніше для побудови кривих навчання).

Контрольні точки фіксують точне значення всіх параметрів (об'єктів tf.Variable), що використовуються моделлю. Контрольні точки не містять жодного опису обчислень, визначених моделлю, і тому зазвичай корисні

лише тоді, коли доступний вихідний код, який використовуватиме збережені значення параметрів.

Формат SavedModel, з іншого боку, включає серіалізоване опис обчислень, визначених моделлю, крім значень параметрів (контрольна точка). Моделі у цьому форматі не залежать від вихідного коду, який створив модель. Таким чином, вони підходять для розгортання через TensorFlow Serving, TensorFlow Lite, TensorFlow.js або програми іншими мовами програмування (API TensorFlow C, C++, Java, Go, Rust, C# і т.д.).

```
history = model.fit_generator(train_generator,
                              epochs=epochs,
                              validation_data=validation_generator,
                              callbacks=[checkpoint])
```

Рисунок 4.6 – Метод навчання моделі

Після побудови моделі ми позначаємо дві можливості для результатів, 0 як "без маски" і 1 як "з маскою". Також встановлено колір граничного прямокутника, використовуючи значення RGB червоний для "без маски" та зелений для "з маскою".

Після цього це буде використатися для визначення того, чи надіта на людині маска, використовуючи веб-камеру комп'ютера. Для цього спочатку необхідно продати виявлення особи. Для цього буде використовуватись каскадні класифікатори з урахуванням ознак Хаара для виявлення рис обличчя.

Цей каскадний класифікатор розроблений бібліотекою OpenCV для виявлення передньої особи шляхом навчання тисяч зображень. Для цього потрібно завантажити файл .xml і використовувати його для виявлення обличчя.

На останньому етапі використовується бібліотека OpenCV для запуску нескінченного циклу для використання веб-камери за комп’ютеру, в якій

визначається особа за допомогою каскадного класифікатора.

Модель передбачить можливість кожного з двох класів (без маски, з маскою). На основі того, яка ймовірність вище, буде вибрано позначку, яка відображатиметься навколо осіб.

4.5 Результати тестування навченої моделі

Показано кілька фотографій та результати тестування (рис 4.7). Як можно побачити, виявлено два типи масок на обліччі – лицьова і фронтальна. Але під час усіх тестувань було виявлено багато збоїв. При поганому освітленні кімнати, модель могла видавати помилкові показання. Якщо особа вдягнена в одяг, який закриває область шиї, модель у 50% випадків видає некоректні результати. Також на відстані особи від камери більш ніж на полтора метри модель зовсім не видає результатів.

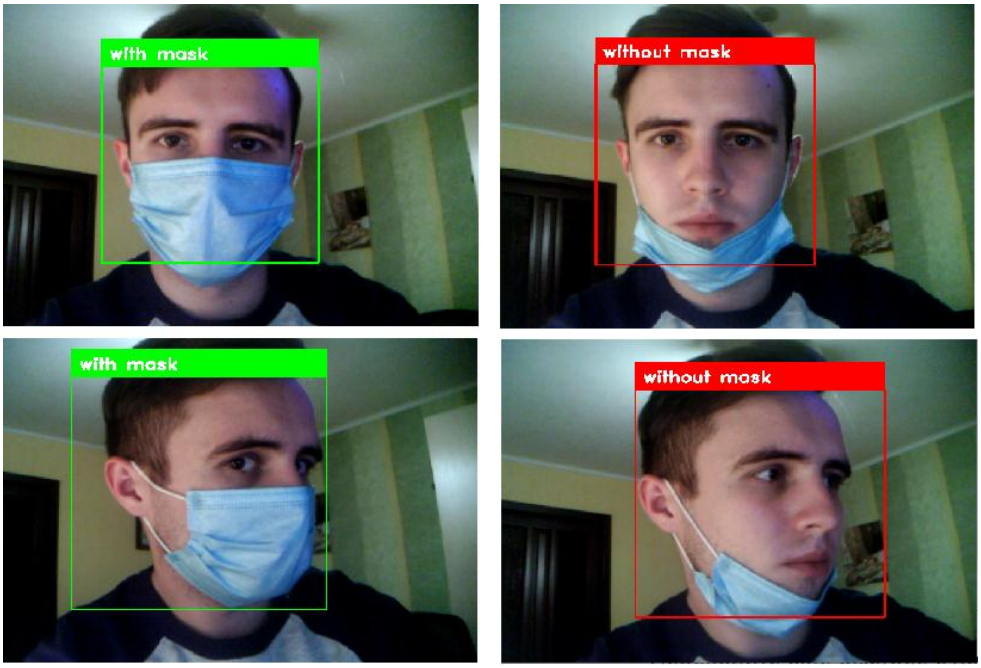

Рисунок 4.7 – Результати визначення лицьової маски на обличчя через веб-камеру ком’ютера

Є припущення, що набір даних для навчання не забезпечує достатньої

кількості даних для усіх можливих сценаріїв, що призводить до неточності навченої моделі та прогнозу. Якщо підвищити кількість даних у декілька разів, та у якості даних брати не лише зображення, а й відео, то модель може потребувати біль ітерацій на навчання, що збільшить витраченого часу на її навчання (при тренуванні з наявними даними та 30 ітераціями знадобилось 20-30 хвилин).

4.6 Побудова графіків та аналіз отриманих результатів

На рис. 4.8 можна побачити, що після 30-ї ітерації наша модель має точність 93,97% на навчальній множині та точність 99,29% на тестовій множині. Це означає, що модель добре навчена.

```
Epoch 30/30
============================] - 88s 584ms/step - loss: 0.1783 - accuracy: 0.9397
- val_loss: 0.0397 - val_accuracy: 0.9929
```

Рисунок 4.8 – Результат останньої ітерації в останньому кроці

За допомогою бібліотеки візуалізації даних двовимірною графікою Matplotlib, можна візуалізувати результати виконання навчання у виді графіків.

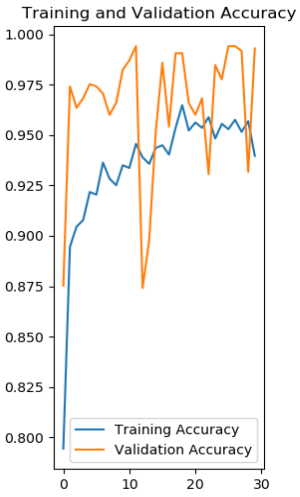

Рисунок 4.9 – Графік точності на навчальній та тестовій множині

На рис. 4.9 показано, як змінювалась точність під час тренування моделі. Точність навчальної множини набула максимального значення ± 96,46 % на 18 ітерації, а точність тестової множини набула максимального значення ± 99,41 % на 11 ітерації.

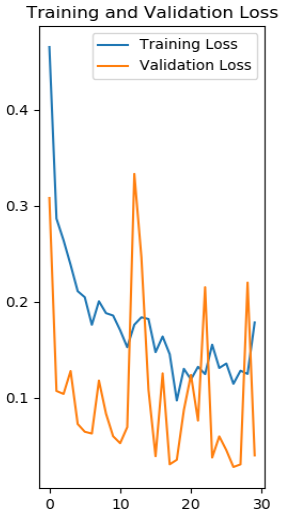

Рисунок 4.10 - Графік втрат на навчальній та тестовій множині

На рис. 4.10 показано, що втрати під час навчання зменшуються у міру проходження епохи навчання. Зі збільшенням числа епох очікується, що втрати в навчанні зменшаться.

Втрата навчальної множини набула мінімального значення ± 9,77 % на 18 ітерації. А втрата тестової множини набула максимального значення ± 33,41 % на 12 ітерації, та мінімального значення ± 2,8 % на 26 ітерації.

Таким чином, можна побачити що тренування моделі пройшло успішно, але під час тестування програмного додатку були знайдені недоліки у роспізнанні лицьової маски. Сбільшення бази даних для тренування ЗНМ може вирішити цю проблему.

ВИСНОВКИ

Метою магістерської кваліфікаційної роботи є вирішення задачі розпізнавання лицьових масок на обличчі з використанням відеокамери.

В магістерській кваліфікаційной роботі проводився огляд підходів виявлення і відстеження людей на відео, аналіз додатків машинного навчання, а також розглянуті деякі архітектури нейронних мереж. На основі усіх досліджень була розглянута продуктивність згорткових нейронних мереж у класифікації зображень. Було виявлено, що архітектури ЗНМ здатні навчати потужні ознаки з невеликого набору даних та мітками, які значно перевершують за продуктивністю методи, що ґрунтуються на ознаках, і що ці переваги стійкі до деталей апаратного забезпечення архітектур у часі. На основі усіх досліджень було вирішено використовувати згорткову нейронну мережу, бо її архітектура дозволить зробити модель детектування, яка самостійно знаходить в зображенні високрівневі ознаки, використовуючи лише зображення у якості вхідних даних.

Були розглянуті та обрані алгоритми оптимізаціх та функції втрат для досягнення найкращих результатів.

Для навчання нейронної мережі було знайденно декілька ресурсів з великими обсягами даних осіб людей в масках та без, а також обрано бібліотеки для подальшої реалізації алгоритму.

Був проведений аналіз отриманих результатів, який показав максимальну точність тестування 99,41%, а мінімальну втрату тестової множини 2,8%, це означає що навіть за невеликим обсягом вхідних даних модель ЗНМ добре навчається.

У кінцевому результаті був створений додаток, який в режимі реального часу розпізнає лицьові маскі на обличчі людей.

Але під час подальшого збору результатів було виявлено, що набір даних для навчання не забезпечує достатньої кількості даних для усіх

можливих сценаріїв, що призводить до неточності навченої моделі та прогнозу. Якщо підвищити кількість даних у декілька разів, та у якості даних брати не лише зображення, а й відео, то модель може потребувати більш ітерацій на навчання, що збільшить витраченого часу на її навчання (при тренуванні з наявними даними та 30 ітераціями знадобилось 20-30 хвилин).

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЬ

1. Нильсен М.А. Нейронні мережі та глибоке навчання. Determination Press, 2017.

2. Object Detection in 20 Years: A Survey. URL: https://arxiv.org/pdf/1905.05055.pdf (дата звернення: 14.11.2021).

3. Object Detection Algorithms and Libraries. URL: https://neptune.ai/blog/object-detection-algorithms-and-libraries (дата звернення: 14.11.2021).

4. The PASCAL Visual Object Classes (VOC) Challenge. URL: https://homepages.inf.ed.ac.uk/ckiw/postscript/ijcv_voc09.pdf (дата звернення: 16.11.2021).

5. Understanding the mAP Evaluation Metric for Object Detection. URL: https://medium.com/@timothycarlen/understanding-the-map-evaluation-metricfor-object-detection-a07fe6962cf3 (дата звернення: 18.11.2021).

6. Cloud Computed Machine Learning Based Real-Time Litter Detection using Micro-UAV Surveillance. URL: https://soe.rutgers.edu/sites/default/files/imce/pdfs/gset-2018/Cloud%20Computed%20Machine%20Learning%20Based%20Real-Time%20Litter%20Detection%20using%20Micro-UAV%20Surveillance.pdf (дата звернення: 18.11.2021).

7. CS231n: Convolutional Neural Networks for Visual Recognition. URL: http://cs231n.github.io/neural-networks-1/ (дата звернення: 21.11.2021).

8. Штучна нейронна мережа URL: https://uk.wikipedia.org/wiki/Штучна_нейронна_мережа (дата звернунння: 27.11.2021).

9. Recurrent neural networ. URL: https://en.wikipedia.org/wiki/-Recurrent_neural_network (дата звернення: 27.11.2021).

10. ML Practicum: Image Classification. URL:

https://developers.google.com/machine-learning/practica/image-classification/convolutional-neural-networks (дата звернення: 28.11.2021).

11. Q. Cao, L. Shen, W. Xie, O. M. Parkhi, A. Zisserman. VGGFace2: A dataset for recognising face across pose and age. International Conference on Automatic Face and Gesture Recognition. URL: http://www.robots.ox.ac.uk/~vgg/publications/2018/Cao18/cao18.pdf (дата звернення 28.11.2021).

12. Citing TensorFlow. URL: https://www.tensorflow.org/about/bib – (дата звернення: 28.11.2021).

13. About Keras. URL: https://keras.io/about/ (дата звернення: 28.11.2021).

14. OpenCV About. URL: https://opencv.org/about/ (дата звернення: 29.11.2021).

15. Признаки Хаара. URL: https://ru.wikipedia.org/wiki/Признаки_Хаара (дата звернення: 01.12.2021).

16. Cascade Classifier. URL: https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html (дата звернення: 01.12.2021).

17. Реализуем и сравниваем оптимизаторы моделей в глубоком обучении. URL: https://habr.com/ru/company/skillfactory/blog/525214/ (дата звернення: 02.12.2021).

18. Стохастический градиентный спуск. URL: https://ru.wikipedia.org/wiki/Стохастический_градиентный_спуск#RMSProp (дата звернення: 02.12.2021).

19. Understanding Loss Functions in Machine Learning. URL: https://www.section.io/engineering-education/understanding-loss-functions-in-machine-learning/ (дата звернення: 03.12.2021).

20. Prajnasb. Delete data_generator. URL: https://github.com/prajnasb/observations/tree/master/experiements/data (дата звернення 04.12.2021).