Форма № 24

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіоелектроніки,
Факультет комп'ютерних наук і технологій
(повне найменування інституту, назва факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)
бакалавр
(ступінь вищої освіти)

на тему РОЗРОБЛЕННЯ ЗАСТОСУНКУДЛЯ ВИЗНАЧЕННЯ ПАЦІЄНТІВ ЛІКУВАЛЬНИХ ЗАКЛАДІВ НА ВІДЕО.
APPLICATION DEVELOPMENT FOR HEALTHCARE PATIENTS RECOGNITION IN VIDEO

Виконав: студент(ка) 4 курсу, групи КНТ-217м
Спеціальності 122 Комп'ютерні науки
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Інженерія програмного забезпечення

Безугла А.М.
(прізвище та ініціали)
Керівник Табунщик Г.В.
(прізвище та ініціали)
Рецензент Шитикова О.В.
(прізвище та ініціали)

2021

Форма № 25

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

Інститут, факультет НУЗП», ФКНТ_____
Кафедра програмних засобів_____
Ступінь вищої освіти бакалавр_____
Спеціальність 122 Комп'ютерні науки_____
(код і найменування)
Освітня програма (спеціалізація) Комп'ютерні науки_____
(назва освітньої програми (спеціалізації)

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
_____С.О. Субботін
"____" _____20___року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТА(КИ)

_____Безуглої Анни Максимівни_____ (прізвище, ім'я, по батькові)
1. Тема проекту (роботи) Розроблення застосунку для визначення пацієнтів лікувальних закладів на відео. Application Development for Healthcare Patients Recognition in Video
керівник проекту (роботи) Табунщик Галина Володимирівна,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від "30" березень 2021року № 103
2. Строк подання студентом проекту (роботи) 17 травня 2021 року
3. Вихідні дані до проекту (роботи) рекомендована література, технічне завдання
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Розробка архітектури програми. 3. Основні рішення щодо реалізації компонентів системи. 4. Експлуатація, тестування та експериментальне дослідження програми. 5. Технічне завдання. 6. Текст програми.
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Слайди презентації

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	Табунщик Г.В., професор		
Нормоконтроль	Камінська Ж.К., асистент		

7. Дата видачі завдання__29 лютого 2021 р._____ _____

КАЛЕНДАРНИЙ ПЛАН

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

Студент (ка) _____ **Безугла А.М.**
(підпис) (прізвище та ініціали)

Керівник проекту (роботи) _____ **Табунщик Г.В.**
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 47 с., 26 рис., 2 дод., 20 джерел.

ЗГОРТКОВІ НЕЙРОНІ МЕРЕЖІ, РОЗПІЗНАВАННЯ ЗОБРАЖЕНЬ, ПАЦІЄНТ, МЄДИЧНИЙ ПЕРСОНАЛ, PYTHON.

Об’єкт дослідження – процес роспізнавання людей у відеопотоці.

Предмет дослідження – методи та засоби розпізнавання людей у відеопотоці.

Мета роботи – створення застосунку для контролю відвідуваності пацієнтів, медичного персоналу та охорони безпеки у лікувальних закладах.

Результати. Розроблено та реалізовано модель з використанням згорткових нейроних мереж для розпізнавання та класифікації людей отриманих з відеопотоку.

Матеріали, методи та технічні засоби: методи класифікації, штучні нейроні мережі, мова програмування python, середа розробки PyCharm, хмарний сервіс Google Colab, веб-застосунок Jupyter Notebook, бібліотеки numpy, opencv-python, tensflow, matplotlib, image ai, keras, pandas.

Висновки. Розроблена модель розпізнавання фігури людей, що основана на згорткових нейроних мережах, використовує відео отримані зі архіву, що дозволить отримувати данні відвідуваності.

Галузь використання – медичні заклади.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 47 pages, 26 figures, 2 appendixes, 20 sources.

CONVOLVED NEURAL NETWORKS, IMAGE RECOGNITION, PATIENT, MEDICAL STAFF, PYTHON.

The object of study - the process of recognizing people in the video stream.

The subject of research - methods and means of recognizing people in the video stream.

The purpose of the work is to create an application for controlling the probability of people, medical staff and security in medical institutions.

Results. A model was developed and implemented using smoothed neural networks for recognition and classification of people obtained from the video stream.

Materials, methods and technical means: classification methods, artificial neural networks, python programming language, PyCharm development environment, web application Jupyter Notebook , Google Colab cloud service, numpy libraries, opencv-python, tensflow, matplotlib, image ai, keras.

Conclusions. The developed model of human figure recognition, based on convolutional neural networks, uses video obtained from the archive, which allows to obtain probability data.

Field of use - medical institutions.

ЗМІСТ

С.

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

IDE	інтегрована середа розробки
Python	високорівнева мова програмування
БД	база даних
ЗНМ	згорткова нейронна мережа

ВСТУП

Останні роки демонструють швидкий ріст обсягів цифрової інформації та поширення систем, які її зчитують, обробляють та використовують. Частина інформації збирається за допомогою систем відеоспостереження. Громадський транспорт, магазини та заклади, житлові будинки, ресторани оснащені сучасними системами відеоспостереження, за допомогою яких можна покращити життя людини, зробити більщ безпечним та комфортним: виявити злочинців, знайти загублених людей.

У цей самий час такі системи використовуються для роспізнавання та контролю співробітників у великих закладах або для статистики відвідуваності клієнтів.

Наприклад, у випадку великих офісів можна відстежувати час приходу співробітників на роботу і відходу, а також загальний час перебування на роботі. Для маленьких або середніх офісів, що не можуть дозволити собі охороника цілодобово можуть скористатися опцією відкриття дверей при розпізнаванні співробітників. Тривожний вихід з камери підключається до контролерів дверного замка або турнікета - після розпізнавання співробітника двері відкриються автоматично та прийде повідомлення до особистого кабінету.

Технології, що використовуються у більших частинах різних галузей починають розвиваютися дуже стрімко, з’являється можливість вдосконалити їх. Так перспективнимим у задачу роспізнавання об’єктів є згорткові нейронні мережі. Завдяки багатошаровій архітеркутрі знаходить потрібні особливості серед великих обсягів даних.

Метою дипломної роботи є розробка застосунку для визначення пацієнтів та медичного персоналу лікувальних закладів на відео за допомогою існуючих алгоритмів. Отримані результати зберігаються у базу даних для можливості подальшого аналізу.

Головними задачами дослідження є:

– вивчення та аналіз сучасних алгоритмів відстеження та класифікації об’єктів;

– порівняльний аналіз конкурентних систем;

– реалізувати програмне забезпечення для розпізнавання та класифікації об’єктів;

Результати роботи пропонуються компаніям, які займаються технічною підтримкою медичних закладів.

Впровадження запропонованої системи підвищить ефективність роботи та захист установ.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАННЯ ДОСЛІДЖЕННЯ

За день у лікарнях проходить дуже великий потік пацієнтів, медичного персоналу та інших людей. Медицина є сфера де треба виконувати моніторінг людей з боку адміністрації. Хто з пацієнтів прийшов, або пішов не зважаючі на постільний режим, чи виконує медичний персонал свої обов’язки. Сучасні алгоритми можуть дати можливість не слідкувати цілодобово, що може серйозно спростити роботу адміністрації та зняти частку навантаження на них.. Крім того, такі дані можуть допомогти при вирішенні конфліктних ситуацій та охорону закладу.

Одним із пріоритетних шляхів підвищення якості роботи та забезпеченння безпеки стає впровадження систем ідентифікації.

Для того щоб відрізнити пацієнта у відеопотоці, потрібно створити новий клас медичного персоналу. Так ми зможемо усіх людей поділити на медичних працівників та пацієнтів. Однією з проблем індентифікаціє медичного персоналу постає різність спец одягу. Є стандарти для лікарів, медсестер та хирургів, але кажний костюм індивідуальний, має свій колір та навіть різний дизайн. Це залежить від закладу.

У час пандемії є розповсюдженим захисний одяг для перебування у інфекційних відділеннях. Завдяки цьому одягу неможливо встановити посаду персоналу. Також адміністрація та інший персонал може не вдягати спец костюм та бути схожим на пацієнтів. Через не дуже вискоку якість зображень на бюджетних камерах відеоспостереження дуже важко впровадити систему індетифікації персони. Тому було вирішено лікарів, медсестер, хірургів віднести до класу «doctor», а всіх інших людей до «person».

Також важно зазначити, що інколи виникають проблеми з визначенням об’єкту на зображенні. Це може виникати через такі фактори:

– зміна позиції, об'єкт що рухається змінює свій вигляд на площині зображення, наприклад якщо крутиться;

– зміна освітлення, напрямок, інтенсивність, колір;

– шум. Отримання зображення пов'язана з певною кількістю шуму, що може залежити від якості матриці камери;

– перекриття. Потрібно спостерігати за ціллю, коли вона частково або повністю закрита іншим об’єктом на зображенні;

Тому було запропоновано зробити застосунок для індентифікації пацієнтів та медичного персоналу.

Для цього необхідно:

– провести дослідження сучасних методів знаходження та ідентифікації людини;

– провести порівняльний аналіз існуючих інструментів;

– обрати набір бібліотек для роботи з відео та обробки зображень;

– створити застосунок з визначеним вище функціоналом для користувача.

2 ОГЛЯД ІСНУЮЧИХ ПРОГРАМНИХ РІШЕНЬ ДЛЯ ВИРІШЕННЯ ЗАДАЧІ

Оскільки системи виявлення об’єктів користуються великим попитом в автономних транспортних засобах та системах безпеки, програмне забезпечення, яке зосереджується на відстеженні людей, є досить поширеним явищем. Прикладами таких послуг є SOLOSHOT3, Vision4ce та ART.

2.1 SOLOSHOT3

Продукт SOLOSHOT3 – камера, яка дозволяє робити відео у високій якості та має вбудовані алгоритми відстеження об’єктів. Повний комплект який надає виробник зображений на рисунку 2.1.

Однією з його переваг є якісне збільшення (збільшення зображення в 65 разів без втрати якості) та автоматичне відстеження даного об’єкта у кадрі. Для отримання даних з камери використовується легкий інтуітивний інтерфейс, написаний мовою програмування Python. Проте повної системи роботи з даними камери та їх відображення не існує. Крім того, виробник не рекомендує використовувати цю камеру для зйомки в приміщенні та в місцях, де багато людей, що є значним обмеженням у можливих сферах використання. [1].

Рисунок 2.1 – Обладення SOLOSHOT3

2.2 Vision4ce

Vision4ce рекомендує свою камеру разом із програмним забезпеченням для відстеження об'єктів. Їх програмне забезпечення працює з операційною системою Windows або Linux і дозволяє одночасно розпізнавати багато об'єктів у кадрі та відстежувати їх розташування, а також класифікувати об'єкти за рухом та нерухомістю [2]. Приклад роспізнавання морських судів у відкритому морі можна побачити на рисунку 2.2.

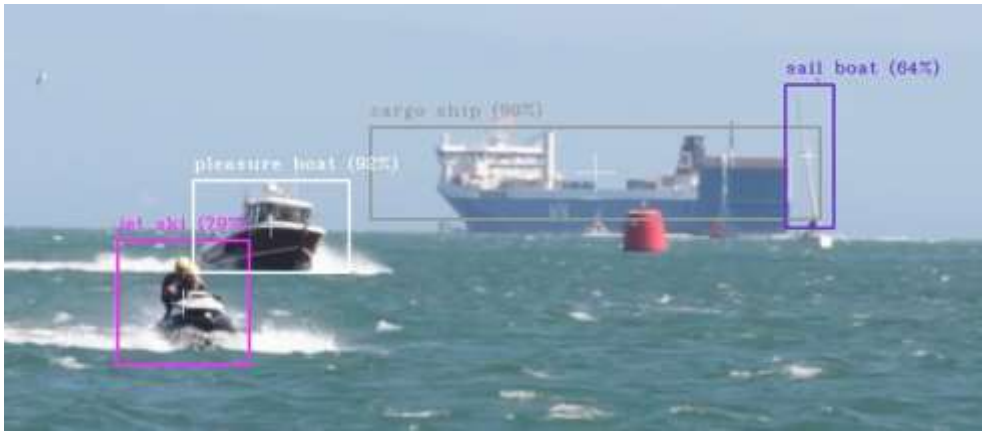


Рисунок 2.2 – Приклад розпізнавання об’єкту за допомогою Vision4ce

2.3 ART

Компанія ART запропоновує декілька систем: SMARTTRACK, TRACKPACK та ARTTRACK. Система SMARTTRACK більш висока продуктивність та надійність дозволяє створювати нові, більш вимогливі програми, такі як тренажери або використання в транспортних засобах, що рухаються, для таких програм, як перевірка систем спостереження водія. Вона підходить для відстеження голови або для використання пристрою взаємодії [3].

Для роботи в активних стереопрограмах із SMARTTRACK можна встановити зовнішню синхронізацію.

Система TRACKPACK розширює функціональність SMARTTRACK завдяки можливості знаходити дрібні предмети та класифікувати рухливий чи нерухомий об’єкт. Рекомендовано для розпізнавання обличчя та жестів.

Система ARTTRACK, як найбільш розроблена, має найкращу якість розпізнавання і стійка до швидко рухаються предметів та поганих умов освітлення в кадрі.

У цій роботі пропонується підхід для чіткого відстеження кількох людей у відео. Відправною точкою є модель, яка нагадує існуючі архітектури для оцінки однокадрової пози, але значно швидша. Досягається цього двома шляхами: спрощенням та розрідженням графіка взаємозв'язку тіло-частина та використанням останніх методів для швидшого виведення, та вивантаженням значної частки обчислень у зворотну архітектуру зворотного зв'язку, яка виявляти і асоціювати суглоби тіла однієї і тієї ж людини навіть у безладі [4]. Приклад роспізнавнаня людей під час спортивної гри наведен на рисунку 2.3.

Рисунок 2.3 – Приклад розпізнавання людей за допомогою ART

2.4 Висновки до розділу

Основним недоліком є висока вартість, деякі недоступні через політику корпорації, що робить їх недоступнішими. Тому було поставлене завдання

розроблення застосунку для роспізнавання людей у відеопотоці, що зберігається у архіві.

3 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМИ

Види штучних нейронних мереж, які використовуються для розпізнавання об’єктів:

- згорткова нейронна мережа (ЗНМ);
- HOG метод;
- рекурсивна нейронна мережа (РНМ);

3.1 HOG МЕТОД

Класичним алгоритмом знаходження об’єктів є Histogram of Oriented Gradients Detector (HOG). Найбільша його перевага полягає в тому, що він стійкий до поз, тобто здатність знаходити предмет не залежить від кута повороту об’єкта. До недоліків можна віднести той факт, що цим методом не вдається знайти дрібні предмети, не стійкий до освітленості, а якість розпізнавання гірша через те, що ящики знайденого об’єкта захоплюють область навколо об’єкта. Візуалізація алгоритму відображена на рисунку 3.1.

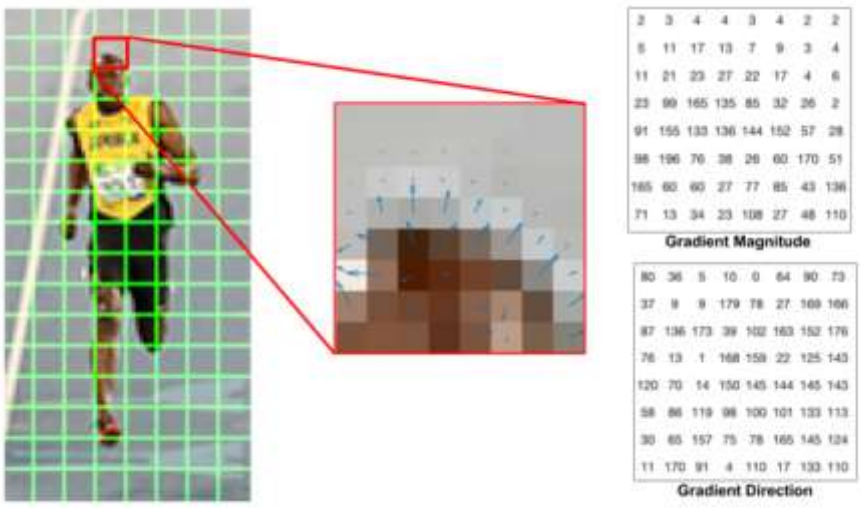


Рисунок 3.1 – Приклад метода HOG

3.2 Згорткові нейронні мережи

Один із найкращих результатів у розпізнаванні обличчя досягається завдяки використанню згорткових нейронних мереж (ЗНМ, англ. Convolutional neural network, CNN). Успіх зумовлений можливістю врахувати двовимірну топологію зображення, на відміну від багатошарового перцептрона.

Ця технологія зроблена за схожістю з принципами зорової кори, в якій були відкриті так звані прості клітини, що реагують на прямі лінії під різними кутами, і складні клітини, реакція яких пов'язана з активацією набору прості клітини [5].

ЗНМ шляхом спеціальної операції - фактичної згортки - допомогає водночас зменшити обсяг інформації, що зберігається в пам'яті інформації, яка краще справляється з зображеннями з більш високою роздільною здатністю, і виділити допоміжні функції зображення, такі як краї, контури або обличчя. На слідуючому етапі обробки з цих ребер і граней можна вилучити повторювані фрагменти текстур, які мають можливість скластися в фрагменти зображення [6].

Алгоритм за яким працює ЗНМ наведена на рисунку 3.2. Фактично кожен шар нейронної мережі застосовує своє перетворення. Коли на перших шарах мережа оперує поняттями "ребра", "грані" і т.п, то далі вживаються поняття "текстура", "частини об'єктів". В наслідок такого опрацювання маємо змогу більш правильно класифікувати зображення або відмітити на останньому етапі потрібний об'єкт на картинці.

Рисунок 3.2 – Загальний вигляд згорткової мережі [7]

ЗНН використовуються для:

– класифікації зображень за допомогою нейронних мереж;

– розпізнавання об'єктів;

– визначення моделі конкретної техніки за допомогою машинного навчання;

– розпізнавання облич і людей;

– тривимірна реконструкція облич і об'єктів по фотографії;

– розпізнавання мови й аналіз емоційної тональності тексту.

Однак він не підходить для роботи в режимі реального часу, оскільки обробка кадру займає занадто багато часу (близько 47 секунд для кожного зображення).

3.2.1 Архітектури згорткових нейроних мереж

Класичними реалізаціями ЗНМ є LeNet5 та AlexNet.

LeNet-5 -новаторська 7-шарова згорткова мережа, розроблена Яном ЛеКуномв 1995, яка класифікує рукописні цифри. Використовується банками для розпізнавання рукописних номерів на оцифрованих чеках. Він приймає зображення розміром 32x32x1 пікселів у півтонах сірого. Можливість обробки зображень з більш високою роздільною здатністю вимагає більше шарів згортки, тому ця техніка обмежена доступністю обчислювальних ресурсів. Архітектура зображена на рисунку 3.3.

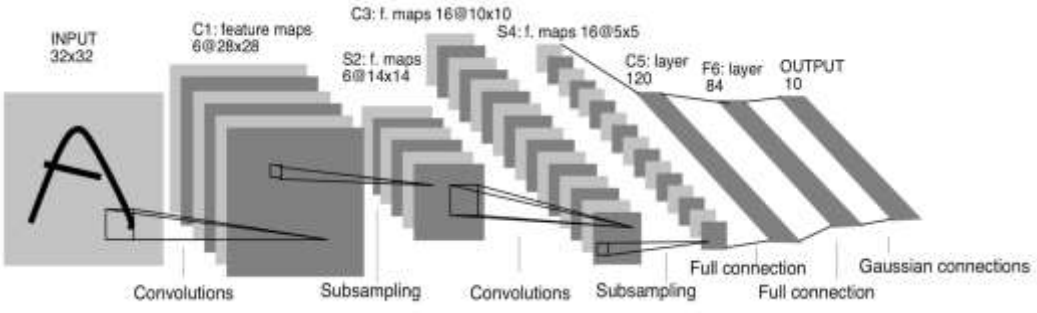


Рисунок 3.3 Архітектура LeNet-5 [8]

Стаття вважається однією з найвпливовіших у галузі дослідження конволюційних мереж. У статті розглядається нейронна мережа Alex Net, що створюється з п’яти згорткових шарів, а ще перемежається з під вибіркою та трьома повністю пов’язаними шарами. Мережа придатна систематизувати об’єкти поміж 100 різних категорій, а загальна чисельність параметрів у мережі становила 6,5 мільйона екземплярів [9]. Архітектура наведена на рис. 3.4.

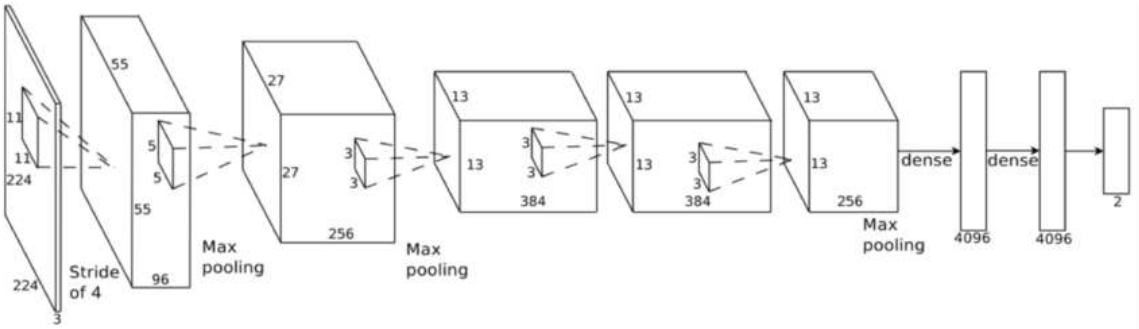


Рисунок 3.4 Архітектура LeNet-5 [10]

Отже, існує кілька архітектур нейронних мереж, створених для визначення об'єктів. Вони в основному поділяються на «дворівневі», такі як RCNN, fast RCNN і faster RCNN, і «однорівневі», такі як YOLO. «Дворівневі» нейронні мережі, перераховані вище, використовують так звані регіони на зображенні, щоб визначити, чи знаходиться в цьому регіоні певний об'єкт [11].

Як працює YOLO можна побачити на рисунку 3.5. Взагалі архітектура YOLO в перших блоках не сильно відрізняється від інших детекторів, тобто на вхід подається картинка, далі створюються feature maps за допомогою CNN (правда в YOLO використовується своя CNN під назвою Darknet-53), потім ці feature maps певним чином аналізуються (про це трохи пізніше), видаючи на виході позиції і розміри bounding boxes і класи, яким вони належать.

Sparse Prediction - це просто повторення того, як дворівневі алгоритми працюють: визначають окремо регіони і потім класифікують ці регіони.

Neck - це окремий блок, який створений для того, щоб агрегувати інформацію від окремих шарів з попередніх блоків для збільшення акуратності передбачення.

І, нарешті, те, що відрізняє YOLO від всіх інших архітектур - блок під назвою Dense Prediction.

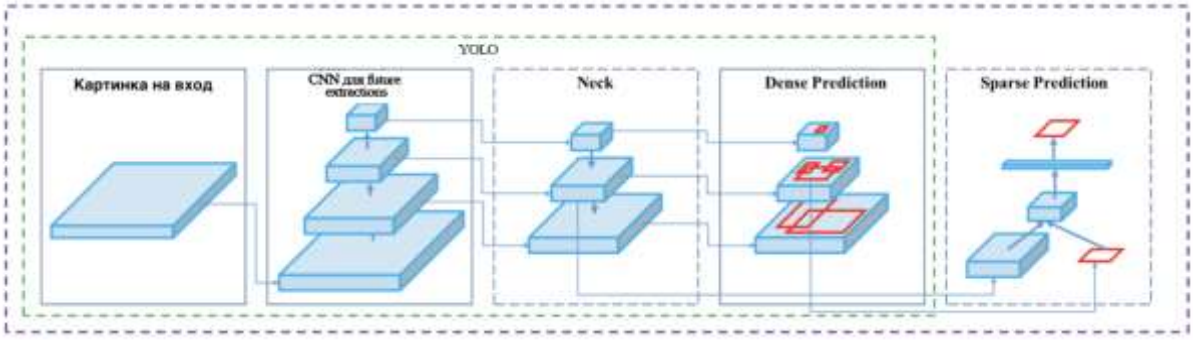


Рисунок 3.5 – Архітектура YOLO [11]

3.3 Висновки до розділу

Таким чином, у цьому розділі розглядаються основні алгоритми, що використовуються для пошуку та ідентифікації об’єктів, обґрунтовується вибір алгоритмів та розглядаються їх більш детально.

4 ЗАСОБИ РОЗРОБКИ

4.1 Засоби Python для реалізації штучних нейроних мереж

Python - мова програмування високого рівня. Стандартна бібліотека містить значну чисельність корисних функцій. Python підтримує декілька парадигм програмування, включно структурну, функціональну, об’єктно-орієнтовану, імперативну та аспектно-орієнтовану. Базові особливості мови містять в собі динамічне введення тексту, автоматична очистка пам’яті, зручні стандартні типи даних. Код на Python зазвичай компілюється у функції та класи, які потім можна об'єднати в модулі, які, у свою чергу, за бажанням об'єднуються у пакети.

Останнім часом Python стає більш популярним у галузі науки про Data Science. Це стало можливим завдяки появі бібліотек, здатних обробляти та візуалізувати великі дані на рівнях MATLAB та Mathematica.

Pandas - це пакет, зазначений задля легшої і зрозумілішої роботи з «поміченими» і «реляційними» даними. Також працює разом з NumPy, і окрім математичних обчислень забезпечує їх агрегацію і візуалізацію.

Python SciPy - це бібліотека Python з відкритим вихідним кодом, призначена для вирішення наукових і математичних проблем. Вона побудована на базі NumPy і дозволяє управляти даними, а також візуалізувати їх за допомогою різних високорівневих команд [12].

Самий основний пакет - NumPy. Він дає змогу виконувати операції над n-масивами і матрицями: додавання, віднімання, ділення, множення і т. д. Шляхом механізму векторизації, NumPy збільшує продуктивність і, відповідним чином, пришвидшує виконання операцій. Вона також пропонує ефективний багатовимірний контейнер загальних даних. З її допомогою можна визначати довільні типи даних.

Matplotlib - це основний інструмент для візуалізації даних у Python, що підтримується декількома платформами та середовищами розробки.

Якраз можливості Matplotlib дають змогу досліджувати Python як повноцінного конкурента MATLAB або Mathematica. Це гнучкий, легкий до налаштовуваний пакет, який постачається з NumPy, SciPy та IPython. Є можливість використовувати його для створення: графіків, діаграм розсіяння, гістограм та гістограм, кругових діаграм,діаграми «стовбур-листя», контурні графіки поля градієнтів, спектральні діаграми [13].

Бібліотека низкоуровнева, що означає великий обсяг коду для розширеної візуалізації. Але продуктивність і робота зі звичним мовою дозволяють закрити очі на цей недолік.

ImageAI - це бібліотека python, призначена для надання можливості розробникам, дослідникам та студентам створювати додатки та системи з окремими можливостями глибокого навчання та Computer Vision, використовуючи прості та кілька рядків коду. [14].

Pillow бібліотека Imaging Library, може быти використовована для роботи з зображенням достатньо легкому способі. У Pillow не було ніяких змін з 2009 [15].

OpenCV (Open Source Computer Vision Library) - це бібліотека з відкритим кодом, що містить близько 2500 алгоритмів для комп'ютерного зору, обробки зображень та числових алгоритмів. Він реалізований на C ++, але має інтерфейси на мові Python [16].

Бібліотека OpenCV має в собі 12 базових маленьких модулів, поділених за їх функціоналом:

- opencv_highgui – завантаження/зберігання відео;
- opencv_core – основні структури, обчислення;
- opencv_ml – методи та моделі машинного навчання;
- opencv_features2d – дескриптори;
- opencv_imgproc – обробка зображень;
- opencv_video – аналіз руху та відстеження об’єктів;
- opencv_calib3d – робота з камерою;

– opencv_objdetect – знаходження об’єктів на зображенні;

– модулі для старого коду задля підтримки роботи реалізацій та прискорення алгоритмів за допомогою завершення частини обчислень на відеокартах.

Keras - це API глибокого навчання, розроблений на Python, що роботає на платформі TensorFlow [17]. Він був розроблений з особистою увагою до швидких експериментів. Можливість швидкого переходу від ідеї до результату є базовим фактором гарного дослідження.

Фреймворк Keras поширюється за безкоштовною ліцензією і може бути використаний безкоштовно в комерційних проектах.

TensorFlow – один з популярних інструментів для машинного навчання нейронних мереж.

Принцип роботи з tensorflow доволі прості. Маємо скласти граф операцій, після цього додати дані на цей графік і надати команду для обчислення. Для того щоб tensorflow мав можливість навчати модель нам потрібно додати 2 речі: функцію втрат і сам алгоритм оптимізації. Повний процес наведен на рисунку 4.1.

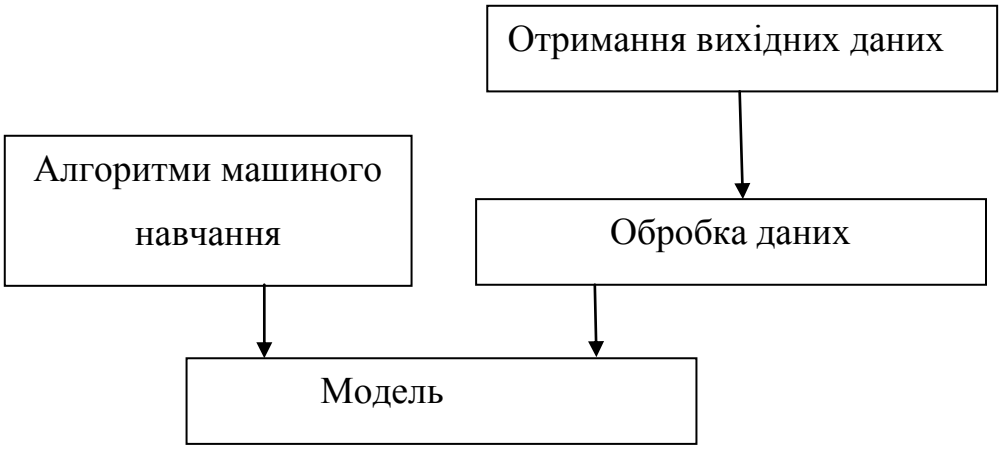


Рисунок 4.1 Процесс створення моделі

Деякі бібліотекі є несумістні з різними версіями інших бібліотек. Повний список бібліотек з коректними версіям:

- tensorflow 2.4.0;
- keras 2.4.3;
- numpy 1.19.3;
- pillow 7.0.0;
- scipy 1.4.1;
- h5py 2.10.0;
- matplotlib 3.3.2;
- opencv-python 0.2.0.

4.2 Хмарне середовище

Google Colab - це хмарний сервіс, що працює на базі Jupyter Notebook. Google Colab дає все необхідне задля машинного навчання одразу в браузері, дає безкоштовний доступ до GPU і TPU [18].

Безкоштовні потужні графічні процесори GPU і TPU, завдяки яким можна займатися не тільки базової аналітикою даних, але і більш складними дослідженнями в області машинного навчання. З тим, що CPU обчислює годинами, GPU або TPU справляються за хвилини або навіть секунди.

Перваги:

- як і Google Документи, він дає можливість працювати з Python-бібліотеками для аналізу даних онлайн;
- Colab надає потужні процесори для хмарних обчислень. У нього інтуїтивно зрозумілий інтерфейс, який дозволяє не перевантажувати комп'ютер і робити все обчислення швидко;
- в Google Colab зберігається доступ до аккаунту з будь-яких пристроїв. Правда, якщо з обережністю ставитеся до своєї конфіденційності, Jupyter залишиться кращим варіантом;

– серце Colab - це спільне використання. При роботі над проектом в команді Colab дає можливість вільно правити, коментувати і редагувати код з різних акаунтів.

Тому було вирішено віикористовувати Google Colab тому що, навіть для невеликих наборів даних, таких як MNIST і простий моделі CNN, можна отримати підвищення продуктивності в 3-7 разів та з’економити декілька годин.

5 РОЗРОБКА ПРОГРАМИ

5.1 Середовище розроблення

PyCharm - це інтелектуальна середовище розробки Python із повним набором інструментів для продуктивної розробки в Python. Випускається у двох версіях - безкоштовна версія PyCharm Community Edition і використовує більший набір функцій PyCharm Professional Edition. PyCharm виконує перевірку коду, автозаповнення, в тому числі на основі інформації, отриманої під час виконання коду, навігація кодом, забезпечує рефакторинг [19]. Інтерфейс середи наведен на рисунку 5.1.

Є можливість настроювати Крос-платформну IDE. Завантажується PyCharm на Windows, Mac OS і Linux використовуючи один ліцензійний ключ.

Comunity edition безкоштовна версія, що володіє усіченим набором можливостей. Поширюється під ліцензією Apache 2. Можливості:

- спрощена ide для розробки;
- безкоштовна, з відкритим кодом;
- розуміє контекст редактор, відладчик
- рефакторингом інспекції;
- навігація по проекту, підтримка тестування, що настроюється UI, гарячі клавіші Vim.

Professional Edition має кілька варіантів ліцензій, які відрізняються функціональністю, вартістю і умовами використання. Є безкоштовним для освітніх установ і проектів з відкритим вихідним кодом. Можливості:

- повнофункціональна середа розробки для роботи на Python, у тому числі для багатомовних веб-додатків з фреймворками;
- підтримка фреймворків Django, Flask, Google App Engine, Pyramid,;
- підтримка мов JavaScript, CoffeeScript, TypeScript, CSS, Cython і ін.;
- дистанційна розробка, Підтримка роботи з БД і мови SQL;
- виявлення дублюючого коду.

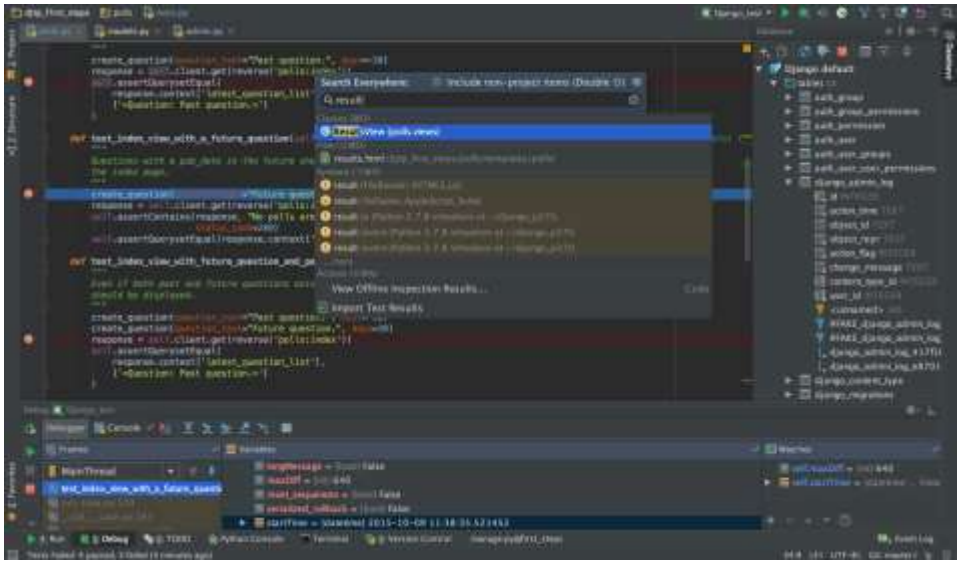


Рисунок 5.1 - Інтерфейс PyCharm [20]

Можливості:

– постійний аналіз коду, підсвічування синтаксису і помилок;

– проста навігація між проектів і сирцевого коду: відображення файлової структури проекту, стрімктй перехід між файлами, класами і методами;

– рефакторинг. Перейменування, вилучення методу, введення змінної, введення константи, підвищення та зниження методу тощо; - інструменти для веб-розробки з використанням фреймворку Django;

– інструменти для веб-розробки завдяки використанню фреймворку Django;

– вбудовані інструменти для юніт-тестування;

– зневаджувач для Python.

5.2 Структура програми

Розроблений застосунок утворюється з декількох основних етапів – попредене оброблення даних, навчання, тестування (рис. 5.2).

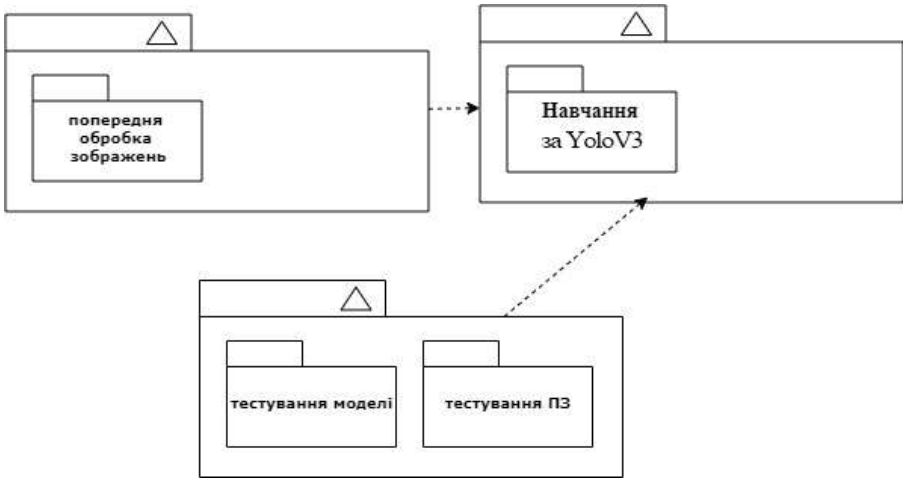


Рисунок 5.2 Структура пакетів застосуваня

Процес пошуку об’єкта полягає в наступному: по-перше, використовується алгоритм визначення тіла та обличчя людини, він повертає координати квадратів граней та тіл.

Потім координати блоків тіла передаються в алгоритмі для ідентифікації людини, яка кодує зображення у вектор чисел і знаходить відповідність у базі даних.

Далі, поки об’єкт не буде за межами кадру, алгоритм відстеження буде відстежувати його місцезнаходження.

Основний алгоритм визначення об’єктів наведено на рисунку 5.2

Рисунок 5.2 Алгоритм роспізнавання об’єктів

6 ЕКСПЛУАТАЦІЯ ПРОГРАМИ

6.1 Тренування моделі

Результат роботи нейронної мережі залежить від кількох факторів:

- типу архітектури нейронної мережі;
- якість і кількість освітніх даних;
- кількості параметрів обраної мережі.

Для повного розуміння конструювання моделі нейронної було наведена схема конструювання на рисунку 6.1.

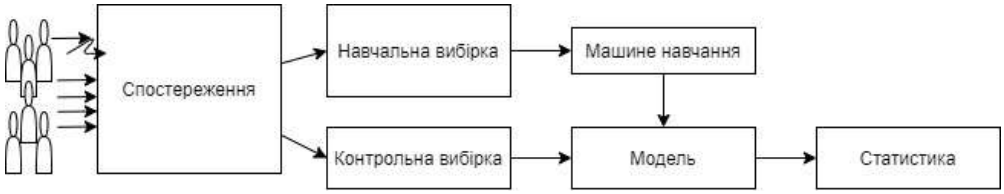


Рисунок 6.1 Процесс конструювання моделі

Для навчання використовувались зображення лікарів та медичного персоналу, що вільно розповсюженні. Було підготовлено 2 набор даних зображень - це папки, що містить 200 зображень медичного персоналу та окрему звичанйих людей.

Наявністьзображень недостатньо, але нам також потрібно вказати, де знаходиться спеціальний об'єкт на конкретному зображенні. Для цієї операції нам знадобиться зовнішнє програмне забезпечення: LabelImg. Вона допомогає розмітити об’єкти на зображенні да зберегти данні у потрібному форматі YOLO. Повний інтерфейс програми зображен на рисунку 6.2. Після завершення тереба архівувати папку з зображеннями.

Рисунок 6.2 Розмітка об’єктів у LabelImg

Для підготовки набору зображень використовувся безкоштовний сервер, запропонований Google colab.

Це безкоштовно, з єдиним недоліком: можливо користуватися ним 12 годин поспіль, після цього вас буде відключено, а файли буде видалено.

Вирішувалась ця проблема, підключивши Google colab до Google drive, тому не втарчалися файли у разі відключення.

Код автоматично встановить даркнет (фреймворк, який використовується для запуску та тренування YOLO), він зробить конфігурацію та запустить навчання. На рисунку 6.3 зображено результат тренування моделі у Google Colab.

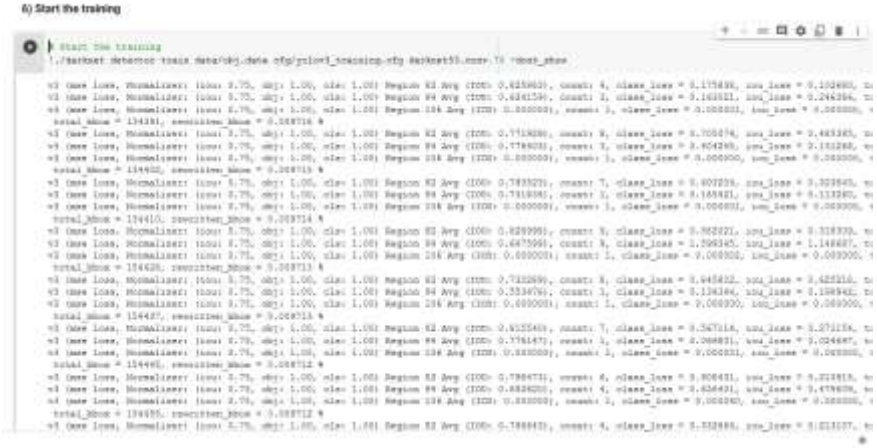


Рисунок 6.3 Процес тренування набору данних

Кожні 100 ітерацій детектор об’єктів буде оновлюватися та зберігатися на диску Google, всередині папки “yolov3”.

Отриманий файл «yolov3_training_last.weights», що містить новий клас, треба додати до папки проекту. Разом з ним до папки треба додати файли «yolo.h5» та «yolov3_testing.cfg», що можливо взята з GitHub.

6.2 Тестування моделі

Окрім написання коду, також потрібно провести тестування.

PyCharm вміє автоматично розпізнавати тести в unittest, pytest або Nose.

Щоб запустити нові тести, оберіть правою кнопкою миші контекстне меню з будь-якого файлу Python і виберіть «Запустити поточний файл тесту». Для нашого проекту потрібно вибрати поточну папку і шаблон * main.py.

Тепер можна запустити всі тести, клікнувши на Run Tests в рядку стану або з верхнього бару.

Крім того тести можна виконувати окремо, що економить багато часу, працюючи лише з невдалими методами.

Результати тестів відображаються у вкладці Output (розділ Python Test Log меню, що випадає). Відео зберігаються у ту саму папку, де були завантажені початкові відео, з назвою «video detected». Приклад результату тренування зроблено на малюнку 6.3.

Рисунок 6.4 Приклад отриманого зображення

ВИСНОВКИ

Під час роботи над проектом було проаналізовано проблему виявлення людей на відео. Порівняно алгоритми пошуку та ідентифікації об’єктів та обґрунтовано вибір кожного із використаних методів.

Як результат роботи було розроблено застосунок по ідентифікації об’єктів у відеопотоці на прикладі ідентифікації пацієнтів та медичного персоналу у медичних закладах.

Для оброблення зображень, та навчання нейроних мереж використовувались методи класифікації, штучні нейроні мережі, мова програмування python, середа розробки PyCharm, хмарний сервіс Google Colab, бібліотеки numpy, opencv-python, tensflow, matplotlib, image ai, keras, pandas.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. SoloShot3 [Електрон. ресурс] – Режим доступу: https://soloshot.com/collections/soloshot3

2. Vision4ce [Електрон. ресурс] – Режим доступу: https://www.vision4ce.com

3. ARTrackingSystem [Електрон. ресурс] – Режим доступу: https://ar-tracking.com/en/product-program/smarttrack3

4. ArtTrack: Articulated Multi-person Tracking in the Wild [Електрон. ресурс] – Режим доступу: https://pose.mpi-inf.mpg.de/art-track/

5. Заряєва Д.М., Пазюк Ю.М. Пошук об’єктів у відеопотоціза допомогою згорткових нейронних мереж [Електорн. ресурс]/ Д.М. Заряєва, Ю.М. Пазюк – Режим доступу: http: //www.zgia.zp.ua/gazeta/konffacuv16_71.pdf

6. Evergreen [Електрон. ресурс] – Режим доступу: https://evergreens.com.ua/ua/articles/cnn.html

7. Структурно-Параметричний СинтезЗгорткових Нейронних Мереж [Текст]/ Синєглазов В., Чумаченко О. // 2018, -С.228.

8. Shemanovskiy [Електрон. ресурс] – Режим доступу: https://congyuzhou.medium.com/lenet-5-своими -руками -b60ae3727cd3

9. LeCun Y. Gradient-Based learning, applied to documents recognition [Електорон. ресурс] /Y.LeCun, L. Botou, Y. Bengio, P. Haffnerhttp– Режим доступу: //yann.lecun.com/exdb/publis/pdf/lecun-98.pdf

10. ResearchGate [Електрон. ресурс] – Режим доступу: https://www.researchgate.net/figure/AlexNet-CNN-architecture-layers_fig1_318168077

11. Habr [Електрон. ресурс] – Режим доступу: https://habr.com/ru/post/514450/

12. GeekBrains [Електрон. ресурс] – Режим доступу: https://gb.ru/posts/python_data_science

13. Wikipedia [Електрон. ресурс] – Режим доступу:

https://uk.wikipedia.org/wiki/Matplotlib

14. Official English Documentation for ImageAI [Електрон. ресурс] – Режим доступу: https://imageai.readthedocs.io/en/latest/

15. PythonRu [Електрон. ресурс] / Режим доступа: https://pythonru.com/biblioteki/osnovnye-vozmozhnosti-biblioteki-python-imaging-library-pillow-pil

16. Wikipedia [Електрон. ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/OpenCV

17. Convolutional Neural Network (CNN) [Електрон. ресурс] / Режим доступа: https://www.tensorflow.org/tutorials/images/cnn

18. Хабр [Електрон. ресурс] / Режим доступа: https://habr.com/ru/company/avito/blog/488936/

19. School technologies [Електрон. ресурс] / Режим доступа: http://technologies-school.blogspot.com/2018/03/pycharm-python.html

20. Бізнес інформатика [Електрон. ресурс] / Режим доступа: https://is42-2018.susu.ru/blog/2019/03/01/byil-zadan-vopros-chto-takoe-pycharm/

ДОДАТОК А

Текст програми

Main.py

```
from imageai.Detection import VideoObjectDetection
import tensorflow as tf
import os
import glob
import cv2
import random

execution_path = os.getcwd()

camera = cv2.VideoCapture(0)

detector = VideoObjectDetection()
detector.setModelTypeAsYOLOv3()
detector.setModelPath(os.path.join(execution_path , "yolo.h5"))
net = cv2.dnn.readNet("yolov3_training_last.weights", "yolov3_testing.cfg")
detector.loadModel()

video_path = detector.detectObjectsFromVideo(camera_input=os.path.join(execution_path , "video1.mp4",
    output_file_path=os.path.join(execution_path, "camera_detected_video")
    , frames_per_second=20, log_progress=True, minimum_percentage_probability=30)

print(video_path)
```

yolo_object_detection.py

```
import cv2
```

```
import numpy as np
import glob
import random

net = cv2.dnn.readNet("yolov3_training_last.weights", "yolov3_testing.cfg")

classes = ["doctor"]

images_path = glob.glob(r"D:\Doctor\images) Train Yolo google cloud\dataset\*.jpg")

layer_names = net.getLayerNames()
output_layers = [layer_names[i[0] - 1] for i in net.getUnconnectedOutLayers()]
colors = np.random.uniform(0, 255, size=(len(classes), 3))

random.shuffle(D:\Doctor\images)
for img_path in images_path:
    img = cv2.imread(img_path)
    img = cv2.resize(img, None, fx=0.4, fy=0.4)
    height, width, channels = img.shape
    blob = cv2.dnn.blobFromImage(img, 0.00392, (416, 416), (0, 0, 0), True, crop=False)

    net.setInput(blob)
    outs = net.forward(output_layers)

    class_ids = []
    confidences = []
    boxes = []
```

```
for out in outs:
    for detection in out:
        scores = detection[5:]
        class_id = np.argmax(scores)
        confidence = scores[class_id]
        if confidence > 0.3:
            print(class_id)
            center_x = int(detection[0] * width)
            center_y = int(detection[1] * height)
            w = int(detection[2] * width)
            h = int(detection[3] * height)
            x = int(center_x - w / 2)
            y = int(center_y - h / 2)

            boxes.append([x, y, w, h])
            confidences.append(float(confidence))
            class_ids.append(class_id)

indexes = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.4)
print(indexes)
font = cv2.FONT_HERSHEY_PLAIN
for i in range(len(boxes)):
    if i in indexes:
        x, y, w, h = boxes[i]
        label = str(classes[class_ids[i]])
        color = colors[class_ids[i]]
        cv2.rectangle(img, (x, y), (x + w, y + h), color, 2)
        cv2.putText(img, label, (x, y + 30), font, 3, color, 2)
```

```
    cv2.imshow("Image", img)
    key = cv2.waitKey(0)

cv2.destroyAllWindows()
```

ДОДАТОК Б

Презентація

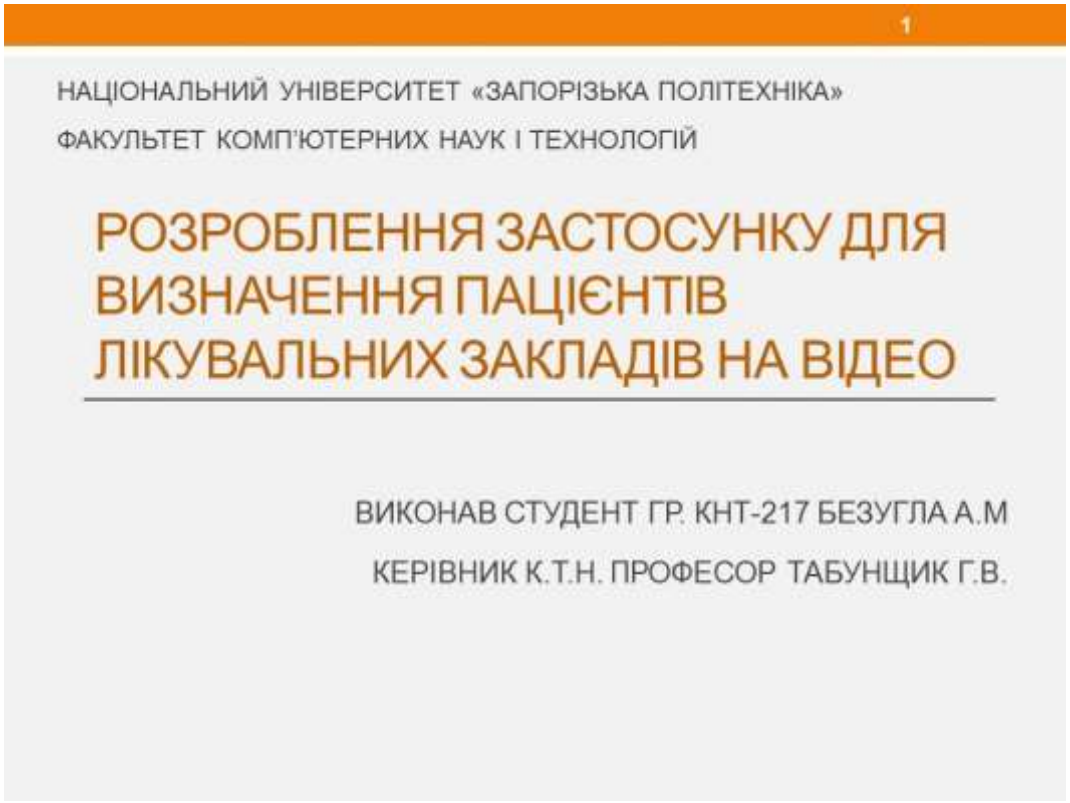


Рисунок Б.1 – Слайд 1

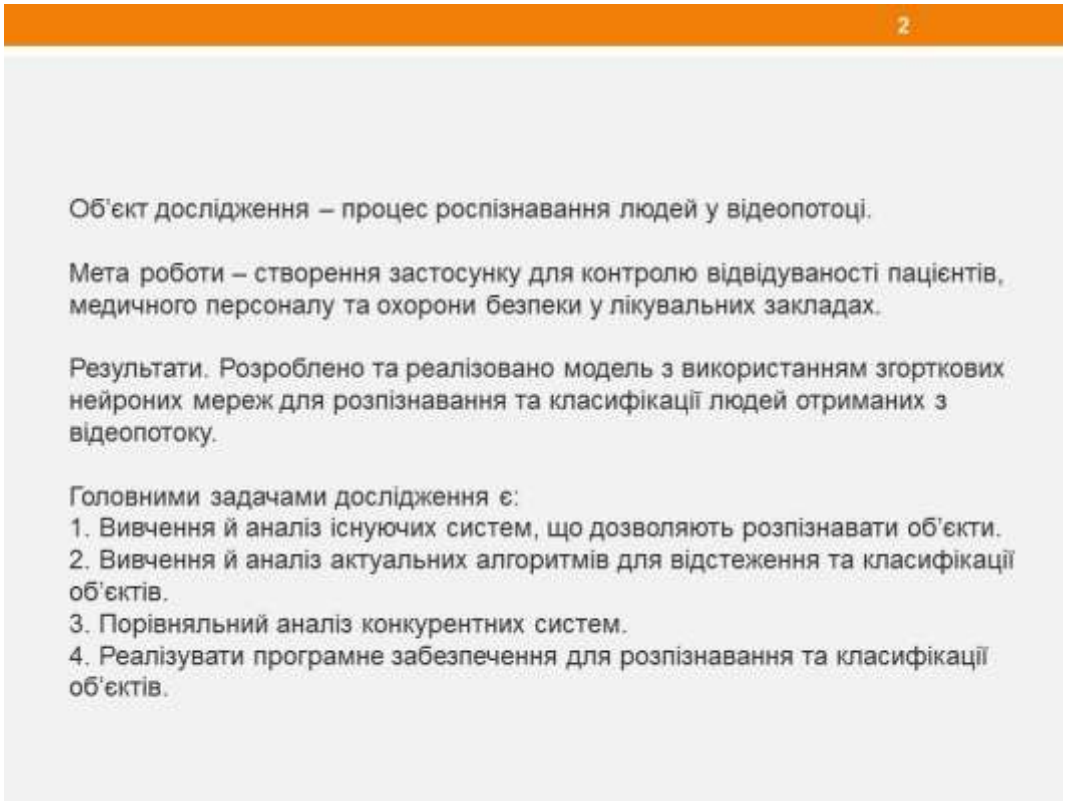


Рисунок Б.2 – Слайд 2

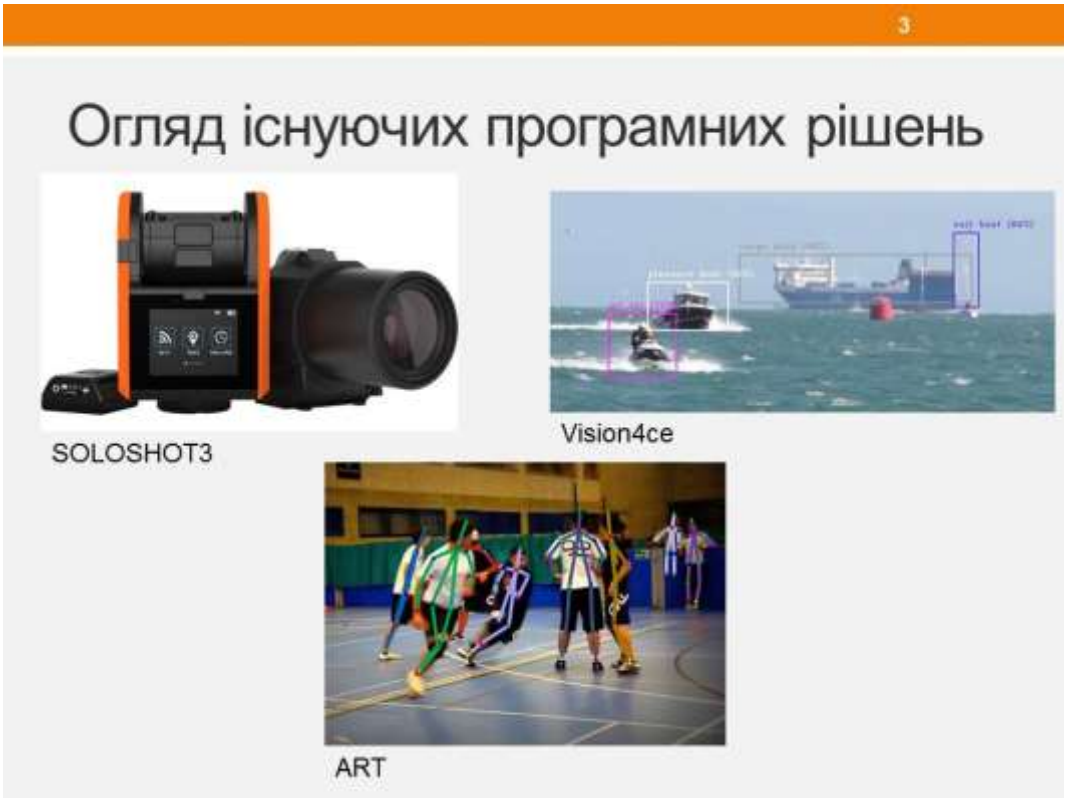


Рисунок Б.3 – Слайд 3

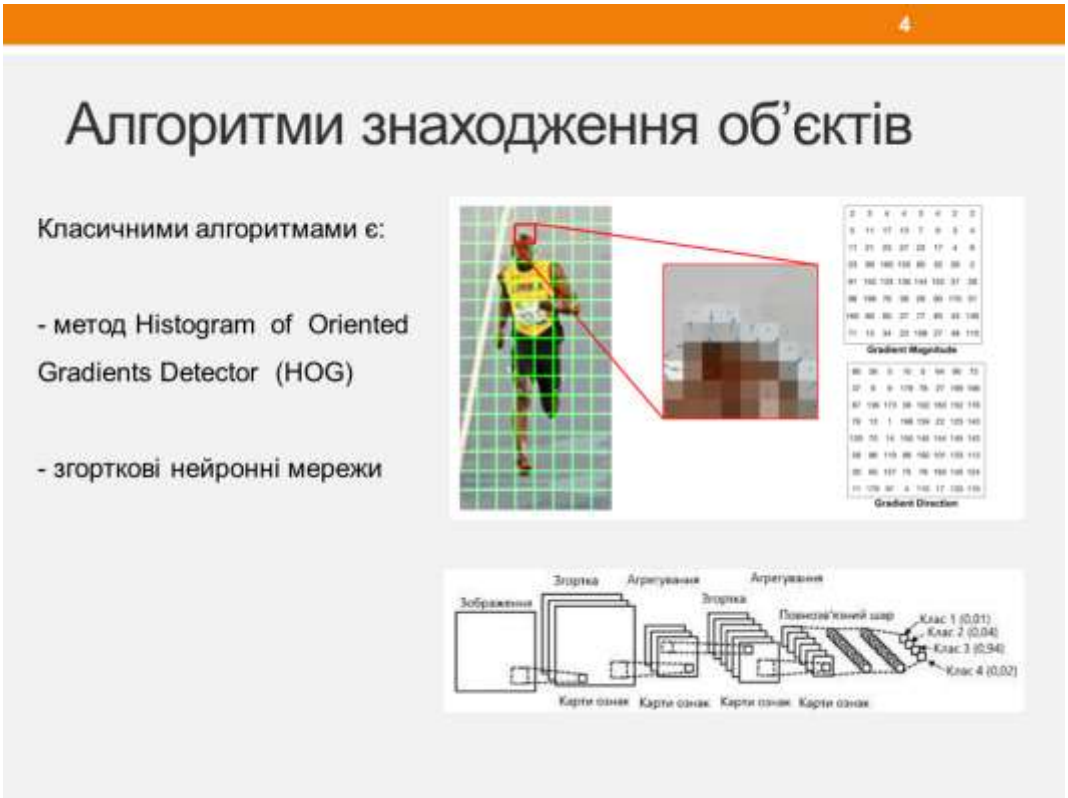
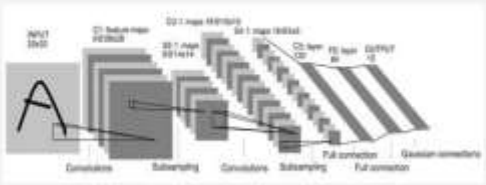


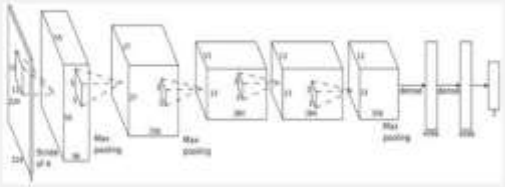
Рисунок Б.4 – Слайд 4

5

Архітектури згорткових нейроних мереж

Класичними реалізаціями ЗНМ є LeNet5 та AlexNet. Нова архітектура YOLO.





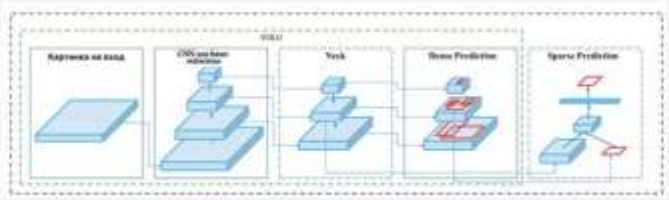


Рисунок Б.5 – Слайд 5

6

Засоби розробки

Повний список бібліотек з коректними версіям:

- tensorflow 2.4.0
- keras 2.4.3
- numpy 1.19.3
- pillow 7.0.0
- scipy 1.4.1
- h5py 2.10.0
- matplotlib 3.3.2
- opencv-python 0.2.0

Рисунок Б.6 – Слайд 6

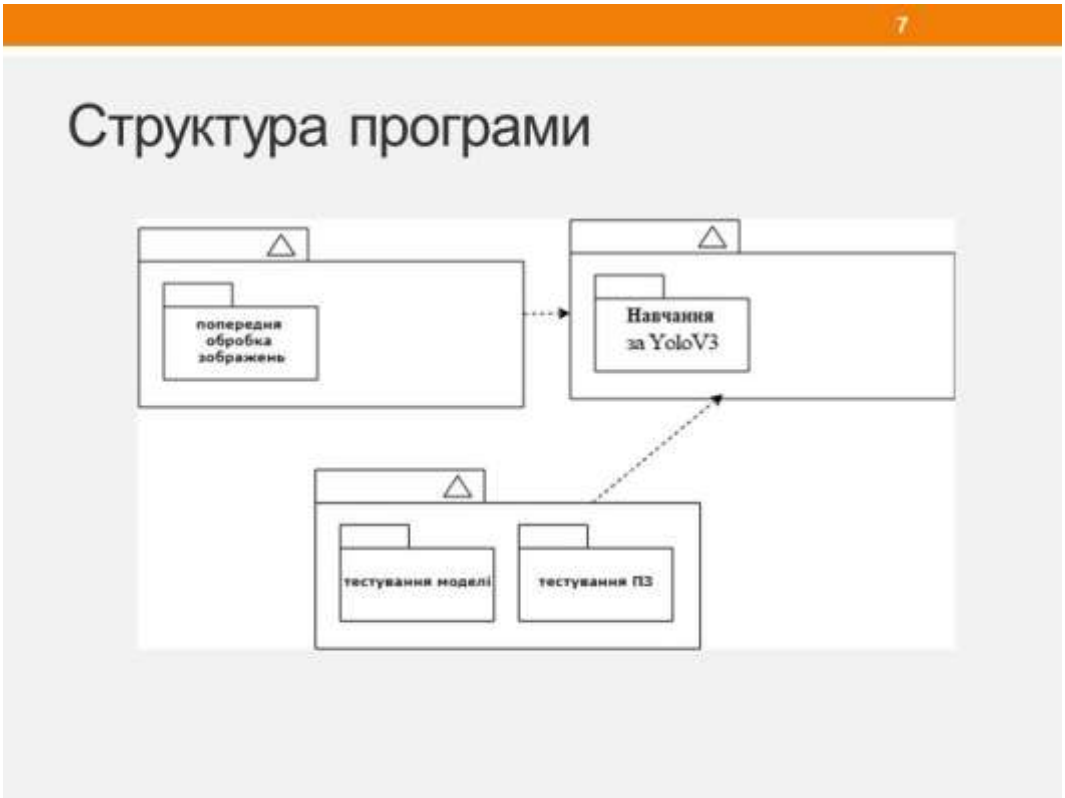


Рисунок Б.7 – Слайд 7

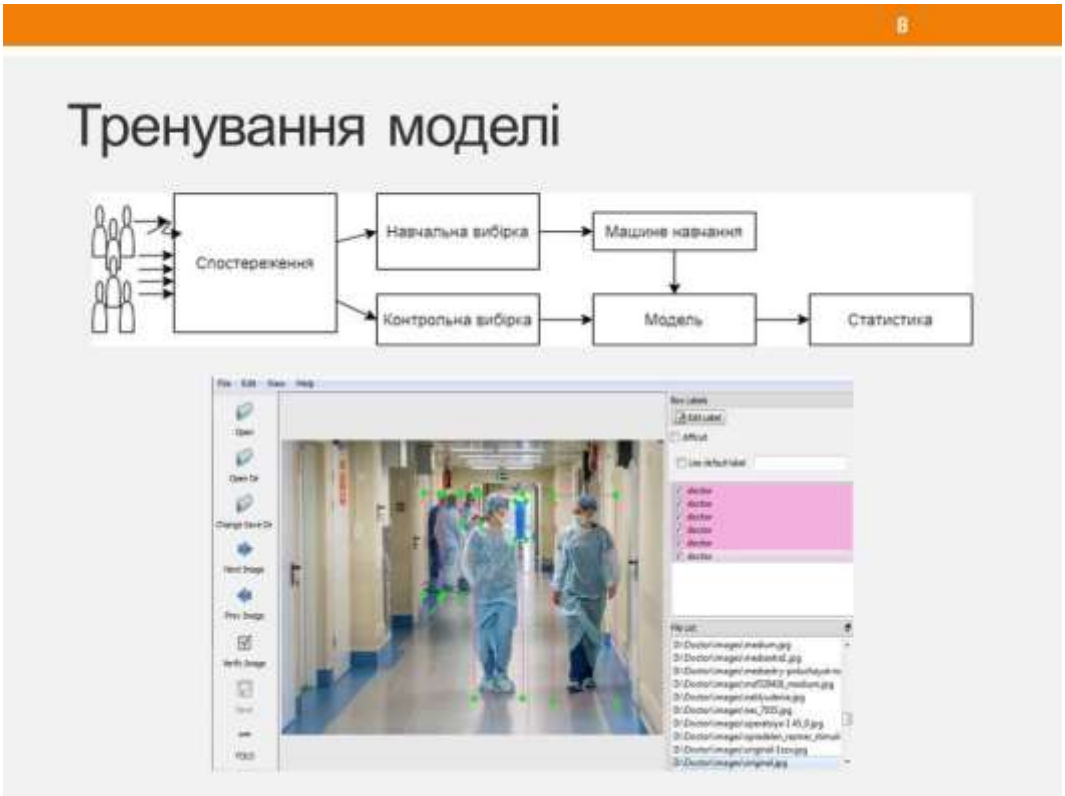


Рисунок Б.8 – Слайд 8

Рисунок Б.9 – Слайд 9

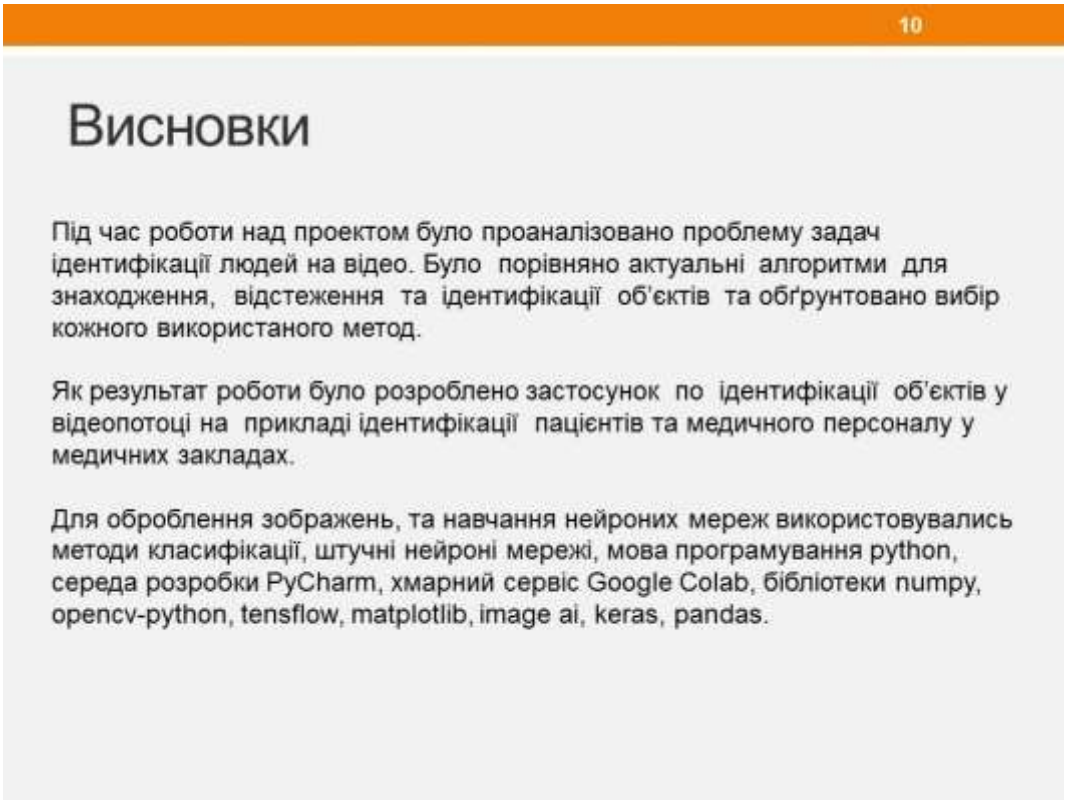


Рисунок Б.10 – Слайд 10