Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютерних технологій
(повна назва)

Кафедра Комп'ютерно-інтегрованих технологій, автоматизації та робототехніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Розробка інтелектуального модулю для ідентифікації об'єктів в ході сортування
(тема)

Виконав:
студент _4_ курсу, групи АКІТ-20-1
Кравченко К. К.
(прізвище, ініціали)

Спеціальність 151 Автоматизація та комп'ютерно-інтегровані технології
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Автоматизація та комп'ютерно-інтегровані технології
(повна назва освітньої програми)

Керівник доц. Жарікова І. В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Невлюдов І. Ш.
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет АКТ

Кафедра КІТАР

Рівень вищої освіти перший (бакалаврський)

Спеціальність 151 Автоматизація та комп'ютерно-інтегровані технології
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Автоматизація та комп'ютерно-інтегровані технології
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)

«____»_____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Студентові Кравченко Кирилу Костянтиновичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка інтелектуального модулю для ідентифікація об'єктів в ході сортування
затверджена наказом університету від 03 червня 2024 р. № 544 Ст
2. Термін подання студентом роботи до екзаменаційної комісії 25 червня 2024 р.
3. Вихідні дані до роботи
3.1 Призначення системи: ідентифікація об'єктів за допомогою комп'ютерного зору;
3.2 Функції: ідентифікація об'єктів у реальному часу з зображення, отриманого через локальну бездротову мережу з плати із зйомною камерою ESP-32CAM;
3.3 Мова програмування: Python.
4. Перелік питань, що потрібно опрацювати в роботі:
4.1 Вступ;
4.2 Аналіз сучасних методів ідентифікації;
4.3 Аналіз архітектури нейронних мереж;
4.4 Методи реалізації нейронних мереж;
4.5 Реалізації нейронної мережі;
4.6 Використання нейронної мережі;
4.7 Заходи з охорони праці;
4.8 Висновки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Графічний демонстраційний матеріал в форматі PowerPoint(*.ppt) формату А4 – 13 сторінок.

6. Консультанти розділів роботи (п.6 включається до завдання 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	Терміни виконання етапів роботи	Примітка
1	Аналіз літератури за темою роботи	29.04.2024	виконано
2	Аналіз технічного завдання	06.05.2024	виконано
3	Аналіз існуючих систем ідентифікації об'єктів	13.05.2024	виконано
4	Розробка нейронної мережі для ідентифікації об'єктів	20.05.2024	виконано
5	Розробка ПЗ для використання нейронної мережі для ідентифікації об'єктів у реальному часі	27.05.2024	виконано
6	Оформлення пояснювальної записки	13.06.2024	виконано
7	Подання роботи на нормоконтроль	20.06.2024	виконано
8	Перевірка роботи у системі Unicheck	21.06.2024	виконано
9	Отримання відгуку та рецензії	23.06.2024	виконано
10	Подання роботи до ЕК	25.06.2024	виконано
11	Представлення до захисту	25.06.2024	

Дата видачі завдання 22 квітня 2024 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Жарікова І. В.
(підпис) (посада, прізвище, ініціали)

Я, як студент ХНУРЕ, розумію і підтримую політику закладу із академічної доброчесності. Я не надавав та не одержував недозволену допомогу під час підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

«13» червня 2024 Кравченко К. К.

РЕФЕРАТ

Пояснювальна записка: 64 с., 5 табл., 42 рис., 2 дод., 22 джерел.

ІДЕНТИФІКАЦІЯ, МЕТОДИ ІДЕНТИФІКАЦІЇ, АВТОМАТИЗАЦІЯ, КОМП’ЮТЕРНИЙ ЗІР, НЕЙРОННІ МЕРЕЖІ, ПРОГРАМУВАННЯ.

Об’єкт розробки – процес ідентифікації об’єктів за допомогою комп’ютерного зору.

Предмет розробки – інтелектуальний модуль ідентифікації яблук у реальному часі за допомогою комп’ютерного зору з використанням нейронної мережі.

Мета роботи – розробка програмного забезпечення для ідентифікації об’єктів у реальному часі за допомогою комп’ютерного зору з використанням нейронної мережі.

У кваліфікаційній роботі розглянуто актуальність теми та існуючі методи автоматичної ідентифікації об’єктів. Проаналізовано способи застосування комп’ютерного зору для ідентифікації об’єктів. Створено та натреновано нейронну мережу для ідентифікації яблук та реалізовано ідентифікацію об’єктів у реальному часі. Проведено випробування та аналіз точності ідентифікації об’єктів.

Розроблене програмне забезпечення можна використовувати як основу для створення автоматизованої системи сортування об’єктів. Ця робота може використовуватись у навчальному процесі для допомоги студентами у дослідженні систем ідентифікації, комп’ютерного зору та основ програмування.

ABSTRACT

Explanatory message: 64 p., 5 tab., 42 fig., 2 app., 22 ref.

IDENTIFICATION, METHODS of IDENTIFICATION, AUTOMATION, COMPUTER SIGHT, NEURAL NETWORKS, PROGRAMMING.

Object of development – object identification process using computer vision.

Subject of development – real-time apple identification intelligent module using computer vision with the application of a neural network.

Purpose of work – development of software for real-time object identification using computer vision with the application of a neural network.

This qualification work examines the relevance of the topic and existing methods of automatic object identification. The existing methods of using computer vision for object identification are analyzed. A neural network for apple identification was created and trained, real-time object identification was implemented. Testing and analysis of object identification accuracy were conducted.

The developed software can be used as a basis for creating an automated object sorting system. This work can be used in the educational process to assist students in studying identification systems, computer vision, and programming fundamentals.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

КЗ – комп’ютерний зір;

ПЗ – програмне забезпечення;

CNN – convolutional neural network;

DCGAN – deep convolutional generative adversarial network;

RNN – recurrent neural network;

QR – quick response;

RFID – radio frequency identification;

ВСТУП

На сьогоднішній день, ідентифікація об’єктів є одним з вирішальних елементів систем автоматизації. Не маючи можливості розпізнати предмет з яким потрібно виконувати якісь дії неможливо буде побудувати автоматизовану систему. Тому питання ідентифікації завжди є актуальним при вирішенні великої кількості задач інженерів систем автоматизації.

Провідні наукові установи та організації, такі як Google, Facebook, OpenAI, університети MIT та Stanford, активно розробляють і вдосконалюють моделі глибокого навчання для різних завдань комп'ютерного зору, включаючи розпізнавання об’єктів у реальному часі. Дослідження у сфері комп’ютерного зору орієнтовані на підвищення точності та швидкості роботи моделей, а також на їх інтеграцію у різноманітні галузі, такі як автоматизація виробництва, охорона здоров'я, автономні транспортні засоби та розумні міста.

Розвиток технологій комп’ютерного зору та штучного інтелекту дозволяє реалізувати ідентифікацію об’єктів гнучкішою та зручнішою. Завдяки чому можливо впровадити автоматизацію процесів у тих галузях, в яких раніше це було дуже затратно або неможливо. Таким чином можливості нейромереж можна впровадити в процесах, які потребують складної ідентифікації об’єктів, класифікації та їх сортування.

Об’єкт розробки – процес ідентифікації об’єктів за допомогою комп’ютерного зору.

Предмет розробки – інтелектуальний модуль ідентифікації яблук у реальному часі за допомогою комп’ютерного зору з використанням нейронної мережі.

Мета роботи – розробка програмного забезпечення для ідентифікації об’єктів у реальному часі за допомогою комп’ютерного зору з використанням нейронної мережі.

Для досягнення поставленої мети необхідно вирішити такі завдання:

– проаналізувати сучасні методи ідентифікації об’єктів;

– проаналізувати архітектури нейронних мереж;

– обрати метод реалізації нейронної мережі;

– реалізувати нейронну мережу;

– реалізувати використання нейронної мережі для ідентифікації об’єктів у реальному часі.

Кваліфікаційна робота виконана згідно ДСТУ 3008:2015 [1] та керуючись методичними вказівками [2].

1 АНАЛІЗ СУЧАСНИХ МЕТОДІВ ІДЕНТИФІКАЦІЇ ОБ'ЄКТІВ

Ідентифікація об'єктів відіграє важливу роль в багатьох галузях промисловості. В логістиці, на виробництві або в торгівлі потрібно відслідковувати переміщення товарів, пакунків, партій від складу до місця призначення та рух об'єктів в автоматизованих розумних складах.

Найкращий метод ідентифікації той, який буде більш гнучким та швидким. Гнучкість методу залежить від того наскільки потрібне обладнання не буде залежати від власних особливостей об'єкта ідентифікації. Такими методами є радіочастотна ідентифікація (RFID), штрих- та QR-коди або комп'ютерний зір (КЗ).

1.1 Радіочастотна ідентифікація

Радіочастотна ідентифікація – визначення яке описує будь яку систему ідентифікації в якій електронний пристрій, прикріплений до предмету, використовує радіочастоти або магнітні поля для зв'язку. Зазвичай така система (рис. 1.1) має наступні компоненти: RFID-мітка, що прикріплена до предмету ідентифікації, зчитувач та допоміжна RFID-програма, яка керує обліком предметів.

Рисунок 1.1 – RFID система [3]

Системи радіочастотної ідентифікації дозволили автоматизувати велику кількість процесів та впровадили не тільки високу швидкість ідентифікації, але й безконтактність визначення об’єктів, що є важливим в умовах, коли прямий контакт до об’єктів є неможливим або небезпечним.

Систему ідентифікації з використанням RFID методу можна знайти в торгових центрах та магазинах одягу. В мітках, що звичайно знімаються на касі, знаходяться RFID-теги, які активують сенсори на вході до магазину та сенсори, в свою чергу, передають сигнал ПЗ, яке активує сигналізацію (рис. 1.2).

а – RFID-мітки в корпусах для закріплення на товарі; б – сигнальні рамки

Рисунок 1.2 – Елементи системи ідентифікації RFID методу

Маючи велику кількість переваг та відносно просту архітектуру системи радіочастотна ідентифікація також має і ряд недоліків. Найпростішим для виявлення недоліком є необхідність розгортання обладнання системи та прикріплення міток до кожного відстежуваного об’єкту. Також недоліками є нюанси роботи з радіохвилями: інтерференція сигналів та безпека даних. У деяких випадках, сигнали можуть бути перервані або змінені радіочастотними

пристроями або металевими об’єктами, що може призвести до помилок ідентифікації. Можливі випадки що інформація, яка передається через системи RFID, може бути піддатливою до несанкціонованого доступу або перехоплення, що створює потенційні загрози для приватності та безпеки даних [3].

1.2 Ідентифікація за штрих- та QR-кодами

Штрих-код – це придатне для зчитування комп'ютером представлення чисел і символів у вигляді паралельних чорно-білих смуг різної ширини та функціональних символів, які зчитує сканер штрих-коду.

Штрих-коди допомагають автоматизувати процес збору даних і зменшують кількість людських помилок при відстежуванні об’єктів.

Штрих-коди мають границю що є порожнім полем, розташованим з обох кінців штрих-коду, відстань якої становить 2,5 мм. У випадку якщо ширина зони недостатня, штрих-коди буде важко зчитати сканером. Також обов’язковими є початковий символ, вказуючий на тип або групу товарів, номери виробника й об’єкту та відокремлюючі штрихи (рис. 1.3).

Рисунок 1.3 – Приклад стандартного штрих-коду [4]

Об’єкти відмічені штрих-кодами можна швидко та ефективно сканувати статичним обладнанням, яке сканує усі проїжджаючі по конвеєру або іншим способом предмети, або завдяки людині оператору.

Прикладом використання штрих-кодів є комбінація сканеру та касового апарату. Касир підставляє штрих-код товару перед сканером, після чого ідентифікація та підрахунок вартості товарів виконується спеціальним ПЗ (рис. 1.4). Завдяки виконанню цих процесів автоматично обслуговування клієнтів зменшує вплив людської помилки й значно пришвидшує процес обслуговування клієнтів.

Рисунок 1.4 – Приклад каси з сканером штрих-коду

Використання штрих-кодів є зручним варіантом відстежування та ідентифікації об’єктів, але, як і у випадку з RFID-ідентифікацією, недоліками такого методу є необхідність виготовлення та нанесення штрих-кодів. Для деяких продуктів та виробників це не є суттєвою проблемою, наприклад, коли продукт може бути ефективно поміщений у контейнери або упаковку. Проте цей метод залишається недосконалим для процесів, в яких відбувається збір сировини, таких, як збір яблук з дерев.

Метод ідентифікації за QR-кодом подібний до методу, який використовує штрих-код, але можна сказати, що він є більш розвиненим та досконалим. QR-код (рис. 1.5) – це двомірний символ, розроблений в 1994 році компанією Denso, однією з найбільших компаній об’єднання Toyota, який в червні 2000 року було

затверджено як міжнародний стандарт ISO (ISO/IEC18004). Спочатку, він призначався для використання в контролі виробництва автомобільних деталей, але згодом набув поширення в інших галузях [5].

Рисунок 1.5 – QR-код посилання на сайт кафедри КІТАР

Сьогодні використання QR-кодів можна побачити в якості особистих ідентифікаторів, посилань для проведення оплати та фінансових транзакцій, посилань на сайти. Завдяки можливості камер сучасних смартфонів зчитувати інформацію з QR-кодів ми маємо можливість сплачувати за проїзд з телефону або подивитись меню у кафе. Це призводить до зменшення втрат з боку постачальника послуг та сприяє екологічному підходу, оскільки зменшується використання паперу та інших матеріалів, що сприяє збереженню ресурсів та зменшенню відходів.

У кутах розміщено характерний візерунок який потрібен для виявлення позиції та повороту коду. Між цими мітками орієнтування розташовується структура часу, яка використовується для синхронізації зчитування коду. Поруч із мітками орієнтування розташовують інформацію для визначення формату даних: рівень виправлення помилок, інформація про форматування даних, маска розбиття великих блоків ідентичних бітів для спрощення читання та один піксель значення якого завжди дорівнює 1. Всі інші пікселі використовуються для

представлення даних. Порядок, в якому впорядковується інформація, наведено на рисунку 1.6, де початковою позицією є правий нижній кут (рис. 1.6).

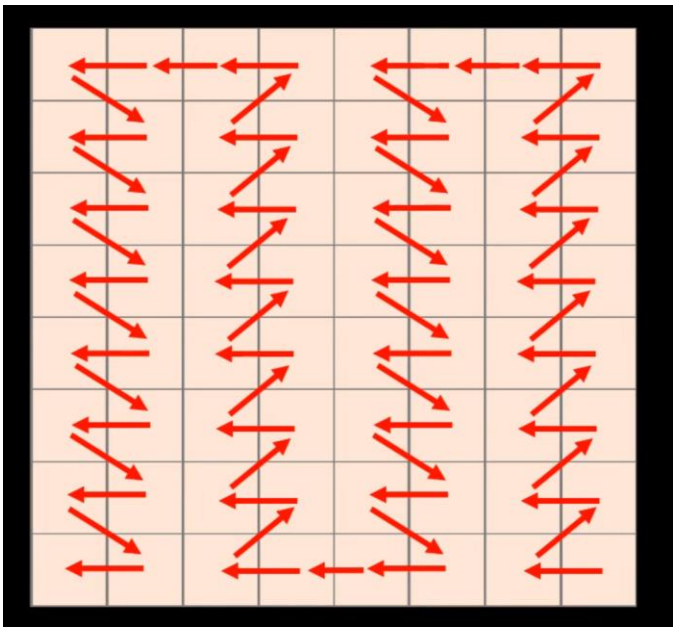


Рисунок 1.6 – Порядок впорядкування даних в QR-коді

Перевагами використання QR-кодів є швидкість та ефективність зчитування інформації, що дозволяє швидко ідентифікувати об’єкти та отримувати доступ до даних про них. Також можливість містити велику кількість інформації на порівняно невеликій площі робить їх універсальними в різних галузях життя. Частина недоліків QR-кодів збігається з недоліками використання штрих-кодів. Через потрібність фізичного існування мітки впливає навколишнє середовище, тому що зношеність матеріалу або пошкодження можуть призвести до втрати інформації або неможливості правильного зчитування коду. Також, QR-коди можуть стати об'єктом зловживань, оскільки доступ до інформації може бути необмеженим.

1.3 Ідентифікація за допомогою комп’ютерного зору

На сьогоднішній день використання комп’ютерного зору з метою ідентифікації об’єктів вже давно перестало бути темою наукової фантастики та набуло статусу важливого і перспективного напряму досліджень та розробок.

Перші алгоритми комп’ютерного зору були створені в 1970-х роках для виявлення основних шаблонів у зображеннях. Збільшення обчислювальних потужностей та розробка все більш ефективних алгоритмів оптимізації обробки інформації надало можливості використанню все складніших алгоритмів. Однак поява нейронних мереж та машинного навчання суттєво прискорили розвиток галузі комп’ютерного зору.

Інтеграція КЗ в системи автоматизації стає не лише перевагою перед конкурентами, а й необхідністю для бізнесу залишатися актуальним постачальником послуг або продуктів. Наприклад використання фермерами дистанційно керованих дронів, які оснащені приладами роботи КЗ, для виявлення шкідників та моніторингу здоров’я посівів дозволяє спрямувати матеріальні та людськи ресурси на інші задачі, що призведе до підвищення загальної ефективності.

При ідентифікації об’єктів за допомогою КЗ обробка інформації відбувається поетапно (рис. 1.7):

– захоплення зображення об’єкту ідентифікації за допомогою відеокамери чи іншого пристрою з можливістю зйомки;

– форматування зображення, що містить у собі фільтрацію шуму, підвищення чіткості, виявлення меж та інші операції необхідні для поліпшення процесу виділення об’єктів на зображенні;

– розпізнавання образів, під час якого відбувається порівняння отриманого зображення з шаблонами або моделями для ідентифікації об’єктів;

– прийняття рішення на основі результатів етапу розпізнавання образів.

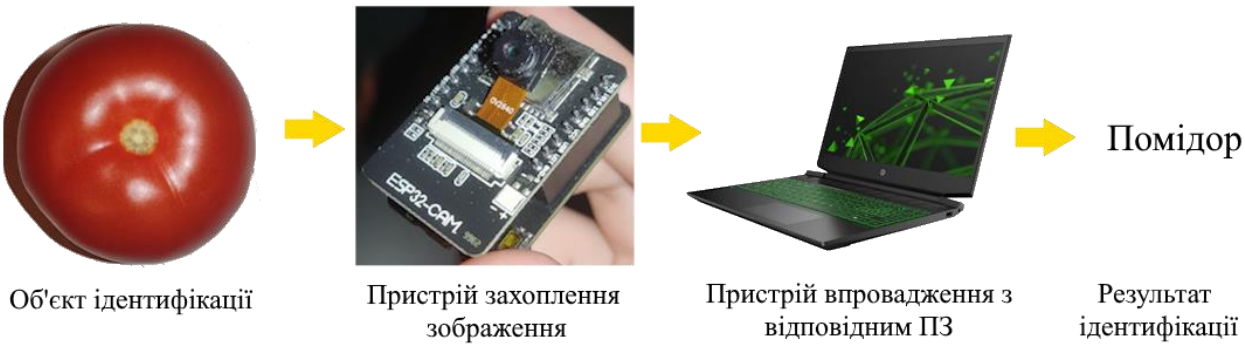


Рисунок 1.7 – Візуалізація роботи КЗ

Приклад використання КЗ є підтвердження особи за допомогою технології розпізнавання обличчя. Ця технологія стала однією з найпопулярніших та вже стала обов’язковою опцією для передових смартфонів після впровадження компанією «Apple» функції «Face ID» для свого нового, на той час, iPhone X. Завдяки цій функції власник смартфону отримав можливість розблоковувати свій мобільний пристрій, просто подивившись у фронтальну камеру телефону.

Щоб розглянути переваги та недоліки технології комп’ютерного зору потрібно глибше дослідити питання його реалізації, тобто проаналізувати які методи та алгоритми використовуються сьогодні для розробки нейронних мереж [6-7].

1.4 Аналіз архітектури нейронних мереж

Одним з головних аспектів успішної реалізації ідентифікації об’єктів за допомогою комп’ютерного зору є правильно обрана архітектура нейронної мережі.

Перед вибором архітектури нейронної мережі досліджується предмет ідентифікації, тип і розмір вхідних даних, вимоги до швидкодії системи, ресурси обчислювальної системи, рівень точності та оптимальний об’єм даних для навчання. Відповідно цих вимог, розробник має зробити вибір між простою

згортковою мережею для аналізу зображень або складною рекурентною мережею, натренованою для аналізу послідовностей даних.

У випадку з комп'ютерним зором нейронна мережа повинна відповідати наступним вимогам:

– навчання на великій кількості даних;

– ефективність роботи при обмежених обчислювальних ресурсах.

Архітектурами, які ефективно навчаються на великих об'ємах даних являються:

– рекурентні нейронні мережі (RNN) – добре підходять для обробки послідовних даних таких як текст. Використання пам'яті для збереження інформації попередніх станів робить їх ефективними для розуміння контексту;

– згорткові нейронні мережі (CNN) – їх перевага полягає в здатності автоматично визначати та виділяти важливі ознаки у вхідних даних. Завдяки цьому, така модель може виявляти складні залежності та шаблони даних [8];

– трансформери – архітектура, яка отримала визнання завдяки ефективності обробки послідовних даних. Такі мережі базуються на механізмі уваги та здатні ефективно моделювати залежності в довгих послідовностях, що робить їх корисними для різноманітних завдань в обробці природної мови, машинного перекладу та інших;

– глибокі згорткові генеративні моделі (DCGAN) – навчаються на великих обсягах даних і здатні генерувати нові зображення або інші типи даних дуже близько до тих, що були б зроблені людиною [9].

Серед наведених архітектур для вирішення задачі ідентифікації оптимальним варіантом являються згорткові нейронні мережі. Здатність виявляти складні моделі та шаблони даних отримується завдяки можливості навчання на великому обсягу даних. Внаслідок чого така модель є більш стійкою до змін у вхідних даних, таких як зміни освітлення, позицій об'єкта, шуми та інші, і має підвищену точність виявлення даних.

Наступна вимога якій повинна відповідати модель мережі є ефективність або оптимізованість для роботи у реальному часі, тобто швидко працювати на обмежених ресурсах обчислювального обладнання.

У згорткових мережах параметри фільтрів використовуються потворно на різних областях зображення. Внаслідок зменшується кількість параметрів мережі, що вимагаються для навчання, і обсяг пам’яті, необхідний для зберігання моделі.

Через фокусування на локальних областях зображення, а не на всьому одразу, зменшується обсяг обчислень, потрібний для аналізу даних. Також це дозволяє адаптувати систему до різних розмірів зображень та допомагає виявляти об’єкти у складних сценах.

Отже, проаналізувавши архітектури які відповідали висунутим вимогам було обрано модель згорткової нейронної мережі. Тепер потрібно дослідити які варіації згорткових моделей існують та визначити яка краще підходить для ідентифікації об’єктів у режимі реального часу.

Згорткові нейронні мережі мають таку назву через те, що процес виявлення контурів, з менших локальних ознак до більш загальних, математично описується операцію, що називається згорткою:

$$f[x,y] \cdot g[x,y] = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} f[n_1,n_2] \cdot g[x-n_1, y-n_2], \qquad (1.1)$$

де f –фільтр;

g – вхідні дані;

x, y – позиція на вихідному зображенні;

n_1, n_2 – позиція на фільтрі;

x-n_1, y-n_2 – позиція на вхідному зображенні.

Тобто, для кожної позиції n_1 та n_2 на фільтрі виконується множення значення фільтра f на відповідне значення вхідних даних g, після чого проводиться сумування всіх цих значень. Гарно ілюструє цей процес рисунок 1.8

де фільтр помножується на відповідну частину вхідного зображення і після чого ця сума утворює нове зображення.

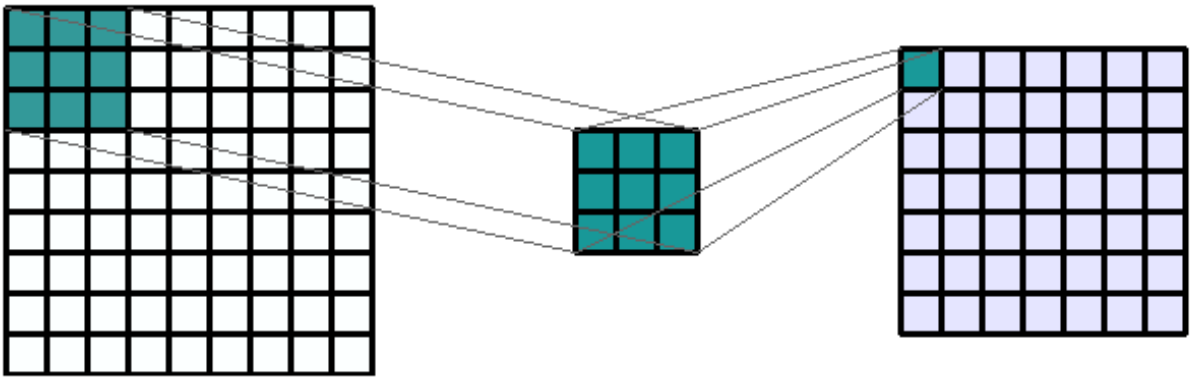


Рисунок 1.8 – Візуалізація процесу згортки [10]

Після операції згортки виконується максимізація для певної ділянки зображення. Отримані ознаки через фільтрацію певної ділянки зображення обробляються таким чином, що з певної ділянки попереднього результату знаходиться максимальне значення яке становиться вихідним результатом. Таку операцію ще називають пулінгом (від англ. Pooling – об’єднання) (рис. 1.9).

Рисунок 1.9 – Візуалізація процесу максимізації [11]

Фактично, ці два кроки є описом одного шару згорткової мережі. Якщо вихідні дані пропускати багато разів, то від простих ліній та точок поступово відбудеться перетворення до складних об’єктів, які бачить людське око.

Згорткові нейронні мережі поділяють за типом виміру на 1D, 2D та 3D.

1D CNN – модель, у який згортка виконується за одним виміром, тобто обробляє дані як одновимірну послідовність. Завдяки чому вона є гарним варіантом для аналізу часових рядів або одновимірних сигналів, таких як звукові сигнали.

Двовимірні згорткові мережі або 2D CNN використовують згортку для виявлення просторових властивостей, що робить їх ефективними у вирішанні багатьох задач, але окремо можна виділити класифікацію зображень і виявлення об’єктів.

Процес класифікації зображень складається з етапу виявлення ознак та етапу класифікації. Під час етапу виявлення ознак виконуються операції згортки, пулінгу та нелінійної активації на вхідному зображенні, з метою виявити шаблони. Після виявлення ознак дані пропускаються через повнозв’язні шари для класифікації зображень на основі виявлених шаблонів.

Якщо класифікація зображень визначає категорію, до якої належить то метою виявлення об’єктів або ідентифікація об’єктів є отримання координат об’єкту на зображенні.

Популярними архітектурами нейронних мереж які розв’язують такі задачі являються YOLO (You Only Look Once) та Faster R-CNN. Метод YOLO виконує ідентифікацію і класифікацію за один прохід по зображенню, що робить його швидким та ефективним. Цей алгоритм розділяє зображення на сітку з фіксованим розміром і для кожної комірки цієї сітки передбачає рамки обмеження об’єктів і ймовірності їх класифікації.

Така архітектура є ефективною у відношенні швидкості, оскільки алгоритм використає загальні згорткові операції для ідентифікації об’єктів на всьому зображенні одночасно (рис. 1.10).

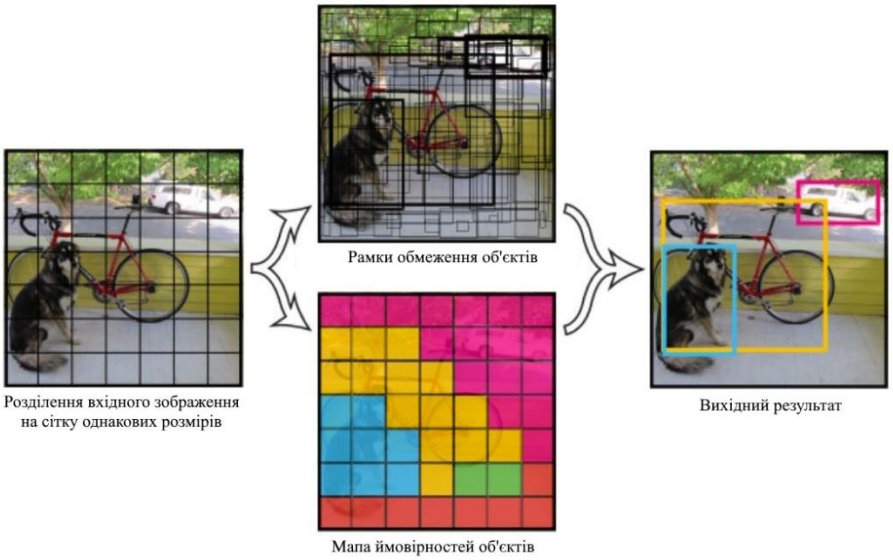


Рисунок 1.10 – Візуалізація роботи алгоритмів YOLO [11]

Сутність методу Faster R-CNN полягає в попередньому пропонуванні регіонів інтересу які можуть мати в собі об’єкт та подальшому уточненні наявності об’єктів в цих областях. Така структура є двошаровою тому, що використовуються дві нейронні мережі: регіональна згорткова, яка відповідає за генерації регіонів інтересу, та повнозв’язна мережа, що виконує класифікацію та уточнення цих областей (рис. 1.11).

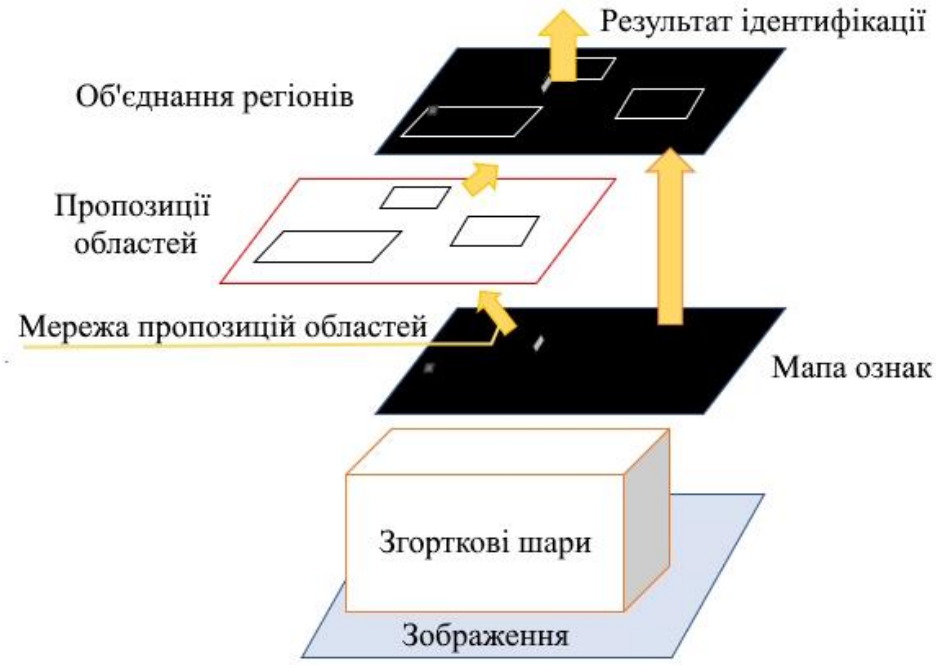


Рисунок 1.11 – Візуалізація роботи алгоритмів Faster R-CNN [12]

Хоча моделі YOLO і Faster R-CNN відомі своїми перевагами в області виявлення об'єктів на зображеннях, вони також мають свої недоліки, які варто враховувати при їх використанні. Одним з основних недоліків моделі YOLO є її обмежена точність при виявленні дрібних об'єктів або об'єктів з низькою контрастністю на зображенні. Крім того, вона може виявлятися менш ефективною в ситуаціях, де присутні перекривання об'єктів або коли об'єкти мають складну форму.

Щодо моделі Faster R-CNN, одним із значних недоліків є її повільна швидкість роботи порівняно з іншими архітектурами. Вона вимагає значно більше обчислювальних ресурсів через використання додаткових модулів, таких як регіональна згорткова нейронна мережа. Крім того, модель Faster R-CNN може бути складнішою у встановленні та налаштуванні порівняно з іншими архітектурами.

Остання нерозглянута модель – тривимірна згорткова мережа 3D CNN. Тривимірні згорткові мережі використовуються для аналізу тривимірних даних, що можуть включати в себе відеозаписи або об'єкти, які були просканові у тривимірному просторі. У порівнянні з двовимірними згортковими мережами, які працюють тільки у двох напрямках, тривимірні згорткові операції додають третє напрямок – глибину.

Принцип роботи тривимірних згорткових нейронних мереж (3D CNN) полягає в тому, що вони проходять за даними у всіх трьох напрямках, визначаючи ознаки в тривимірному об'ємі. Кожен фільтр (ядро) згортки просувається через тривимірний об'єм даних і виконує згорткову операцію для визначення ознак на різних рівнях глибини. Результатом є вихідний об'єм даних, який включає в себе визначені ознаки у всіх трьох просторових напрямках.

Такий підхід дозволяє аналізувати тривимірні дані, такі як відео або об'єкти в тривимірному просторі, з урахуванням їхньої просторової структури та глибини. Цей метод показує гарні результати у медичній діагностиці, розпізнаванні об’єктів та інших областях, де аналіз тривимірних даних є важливим, як показано на прикладі, на рисунку 1.12.

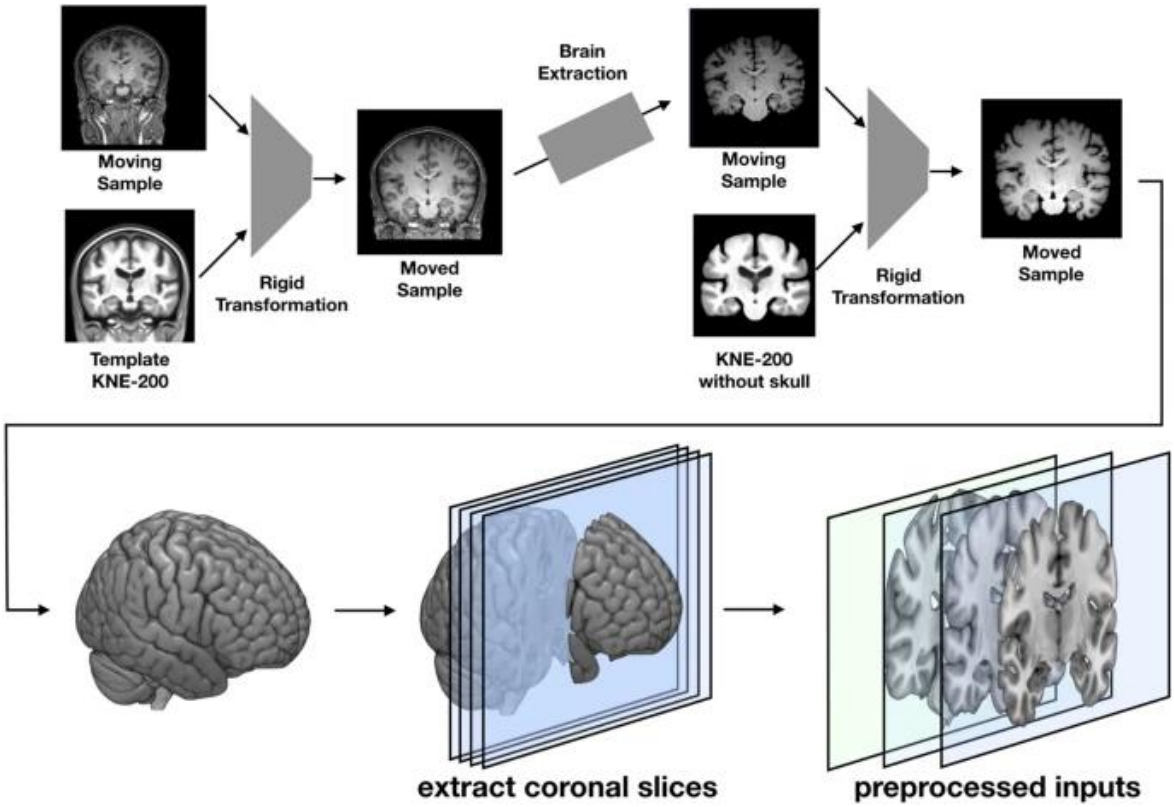


Рисунок 1.12 – Візуалізація процесу роботи 3D CNN для виявлення хвороби мозку [13]

Тривимірні згорткові мережі виявляються менш ефективними, ніж їх двовимірні аналоги, в контексті обробки тривимірних даних через збільшену складність та вимоги до обчислювальних ресурсів. Однією з найбільших проблем є збільшення об’єму даних при використанні тривимірних згорткових мереж, що призводить до збільшення обчислювального часу та вимог до пам’яті.

Такі мережі також можуть бути важкими у налаштуванні та підборі параметрів через більшу кількість варіантів налаштувань. Крім того, вони

можуть виявитися менш ефективними в розпізнаванні об’єктів, якщо ці об’єкти знаходяться на різних глибинах або мають складні тривимірні структури.

Порівнявши архітектури згорткових мереж можна зазначити, що вимогам, які висуває завдання ідентифікації об’єктів за допомогою комп’ютерного зору, відповідають двовимірні нейронні мережі 2D CNN. Така модель використовує менше обчислювальних ресурсів, має високу швидкість роботи та простіша в реалізації, у порівнянні з іншими варіантами моделей.

2 МЕТОДИ РЕАЛІЗАЦІЇ НЕЙРОННОЇ МЕРЕЖІ

2.1 Сучасні методи реалізації нейронних мереж

Набуття широкого застосування нейронних мереж призвело до бажання оптимізувати процес їх розробки, внаслідок чого були створені бібліотеки та інструменти, що спрощують та прискорюють цей процес. Такі бібліотеки, як TensorFlow, PyTorch, Keras дозволяють розробникам легко будувати та тренувати складні моделі нейронних мереж, забезпечуючи при цьому високий рівень абстракції та зручні інтерфейси для роботи. Використання цих інструментів не лише знижує поріг входу для новачків, але й надає потужні засоби для досвідчених дослідників і інженерів, що сприяє швидкому розвитку та впровадженню новітніх досягнень у сфері нейронних мереж.

Також було створено загальноприйняті типи датасетів для тренування цих мереж, що значно спростило процес порівняння та оцінки різних моделей. Стандартні датасети, такі як MNIST (Modified National Institute of Standards and Technology), CIFAR-10 (Canadian Institute for Advanced Research), ImageNet, COCO (Common Objects in Context), Open Image, широко використовуються для навчання та тестування нейронних мереж, забезпечуючи єдину базу для проведення експериментів. Використання цих датасетів дозволяє дослідникам зосередитись на розробці архітектур і алгоритмів, не витрачаючи час на збір та підготовку даних. Крім того, наявність добре відомих датасетів сприяє відтворюваності результатів і забезпечує об'єктивне порівняння продуктивності різних моделей. Це також стимулює спільноту до спільного використання ресурсів і знань, що прискорює прогрес у галузі нейронних мереж.

Існує технологія CUDA (Compute Unified Device Architecture) яка надає розробникам можливість використовувати обчислювальну потужність графічних процесорів для виконання складних математичних і наукових задач,

збільшуючи продуктивність обчислень порівняно з використанням лише центральних процесорів [14].

Окрім CUDA також існують різні сервіси та інструменти, що спрощують доступ до використання бібліотек для створення нейронних мереж. Два з найпопулярніших серед них – Google Colab та Anaconda. Вони дозволяють розробникам легко встановлювати необхідні пакети, налаштовувати середовище для розробки та швидко починати роботу з машинним навчанням.

Google Colab – це безкоштовний хмарний сервіс від Google, який дозволяє запускати Jupyter Notebook у веб-браузері без необхідності налаштування локального середовища. Основні переваги Google Colab включають доступ до потужних графічних процесорів і тензорних процесорів, що значно прискорює процес тренування моделей. Крім того, Google Colab забезпечує зручний доступ до Google Drive, що дозволяє легко зберігати та ділитися даними і моделями.

За допомогою Google Colab користувачі можуть швидко розпочати роботу з TensorFlow та іншими бібліотеками машинного навчання, не турбуючись про сумісність або встановлення пакетів. Це робить його ідеальним інструментом для навчання, досліджень та швидкого прототипування моделей. Крім того, Google Colab підтримує співпрацю, дозволяючи кільком користувачам одночасно працювати над одним документом, що сприяє командній роботі [15].

Anaconda – це популярна дистрибутивна платформа для наукових обчислень на Python та R, яка містить всі необхідні інструменти для роботи з машинним навчанням, аналізом даних та глибинним навчанням. Anaconda надає зручний інструмент управління пакетами та середовищами, що дозволяє користувачам легко встановлювати, оновлювати та керувати бібліотеками, такими як TensorFlow, PyTorch, Pandas та багатьма іншими.

Однією з ключових особливостей Anaconda є її інтеграція з Jupyter Notebook, що дозволяє розробникам писати та виконувати код у інтерактивному середовищі. Крім того, Anaconda забезпечує ізоляцію середовищ, що дозволяє уникнути конфліктів між пакетами та спрощує управління залежностями.

Завдяки цьому розробники можуть створювати і тестувати свої моделі в стабільному та контрольованому середовищі.

Anaconda також включає Anaconda Navigator – графічний інтерфейс для управління пакетами та середовищами, який робить процес налаштування та використання інструментів машинного навчання ще більш інтуїтивним. За допомогою Anaconda розробники можуть швидко розпочати роботу з TensorFlow та іншими бібліотеками, не витрачаючи багато часу на налаштування [16].

Використання Google Colab та Anaconda значно спрощує процес розробки нейронних мереж, надаючи потужні інструменти та зручне середовище для роботи з бібліотеками машинного навчання. Google Colab пропонує хмарне рішення з доступом до високопродуктивних обчислювальних ресурсів, тоді як Anaconda забезпечує зручне управління пакетами та середовищами для локальної розробки. Ці сервіси дозволяють розробникам швидко і ефективно створювати, тестувати та впроваджувати моделі машинного навчання, сприяючи прискоренню наукових досліджень та інновацій.

2.2 TensorFlow

TensorFlow – відкрита платформа для машинного навчання, розроблена компанією Google. Вона надає широкий набір інструментів та бібліотек, які дозволяють створювати та розгортати моделі машинного навчання для різних застосувань. TensorFlow підтримує численні алгоритми машинного навчання та глибинного навчання і забезпечує гнучкість у їх реалізації на різних пристроях, включаючи персональні комп’ютери (ПК), сервери, смартфони, планшети та інші мобільні пристрої завдяки TensorFlow Lite.

Однією з ключових переваг TensorFlow є її здатність обробляти великі обсяги даних та здійснювати високоефективні обчислення за допомогою графічних процесорів і тензорних процесорних блоків. Це дозволяє значно прискорити процес тренування моделей, що є критично важливим при роботі з

великими нейронними мережами. TensorFlow також підтримує розподілене обчислення, що забезпечує масштабованість і продуктивність навіть для найвимогливіших завдань.

TensorFlow надає потужний інструментарій для розробників, які бажають створювати складні моделі нейронних мереж. За допомогою TensorFlow, розробники можуть швидко прототипувати і тестувати нові ідеї, використовуючи високорівневий API(Application Programming Interface) Keras, який спрощує створення та навчання моделей. Keras забезпечує інтуїтивно зрозумілий інтерфейс, що дозволяє швидко будувати та налаштовувати нейронні мережі без необхідності занурюватися в деталі низькорівневого програмування.

TensorFlow також підтримує TensorBoard, інструмент для візуалізації, який допомагає розробникам аналізувати та налагоджувати моделі. З TensorBoard можна відслідковувати метрики навчання, візуалізувати архітектуру моделі та переглядати графіки обчислень, що дозволяє глибше розуміти процес навчання і швидко знаходити проблеми в моделях.

Одна з великих переваг TensorFlow полягає в його активній спільноті та розвиненій екосистемі. Відкритий код платформи дозволяє розробникам з усього світу вносити свій внесок у її розвиток, створюючи нові бібліотеки та інструменти, які доповнюють основний функціонал TensorFlow. У результаті, розробники мають доступ до безлічі додаткових ресурсів, включаючи попередньо навчені моделі, готові для використання у різних проєктах.

TensorFlow Hub, наприклад, є репозиторієм, де можна знайти багато попередньо навчених моделей, які можна легко інтегрувати у власні проєкти. Це значно прискорює процес розробки, оскільки розробники можуть використовувати вже наявні рішення для типових задач, таких як класифікація зображень, обробка тексту та багато іншого.

TensorFlow знаходить застосування у різних галузях, таких як охорона здоров’я, фінанси, роздрібна торгівля, автомобільна промисловість та багато інших. Наприклад, у медичних дослідженнях TensorFlow використовується для

аналізу медичних зображень та виявлення захворювань на ранніх стадіях. У фінансовому секторі TensorFlow допомагає виявляти шахрайські транзакції та прогнозувати ринкові тенденції. В автомобільній промисловості платформа використовується для розробки систем автономного водіння.

TensorFlow є потужним інструментом для розробки та впровадження моделей машинного навчання. Його гнучкість, продуктивність та підтримка розподілених обчислень роблять його одним з затребуваних інструментом для професіоналів, які прагнуть створювати високоефективні нейронні мережі. Завдяки розвиненій екосистемі та активній спільноті, TensorFlow продовжує залишається важливою частиною технологічного прогресу в галузі машинного навчання [17].

2.3 PyTorch

PyTorch – це відкрите програмне забезпечення для машинного навчання і штучного інтелекту, яке розробляється та підтримується Facebook. Він надає зручний і гнучкий інтерфейс для розробки інтелектуальних систем із застосуванням нейронних мереж та інших алгоритмів машинного навчання.

Однією з ключових особливостей PyTorch є його динамічний граф обчислень. Це означає, що ви можете визначати та змінювати структуру вашої моделі нейронної мережі під час її виконання, що спрощує експерименти з архітектурою моделі та дозволяє швидше розробляти нові ідеї.

PyTorch надає широкий спектр функцій для роботи з нейронними мережами. Він містить реалізації різних типів шарів, активаційних функцій, втрат та оптимізаторів. Крім того, PyTorch має вбудовану підтримку для обчислювальних операцій на GPU, що дозволяє прискорити тренування нейронних мереж на великих обсягах даних.

Одним з головних переваг PyTorch є його зручний інтерфейс. Він має простий синтаксис, що дозволяє швидко розробляти та експериментувати з

моделями нейронних мереж. Крім того, PyTorch надає багато корисних інструментів для візуалізації даних, аналізу результатів та налагодження моделей.

Популярність PyTorch постійно зростає у наукових та промислових колах через його зручний інтерфейс, швидкодію та потужність. Він використовується для різних завдань машинного навчання, включаючи класифікацію зображень, розпізнавання мови, обробку природної мови та багато іншого.

Крім того, PyTorch має активну спільноту користувачів та розробників, яка надає безкоштовну підтримку, документацію та навчальні матеріали для початківців і досвідчених користувачів. Завдяки чому його можна застосувати для вивчення та застосування інструментів машинного навчання [18].

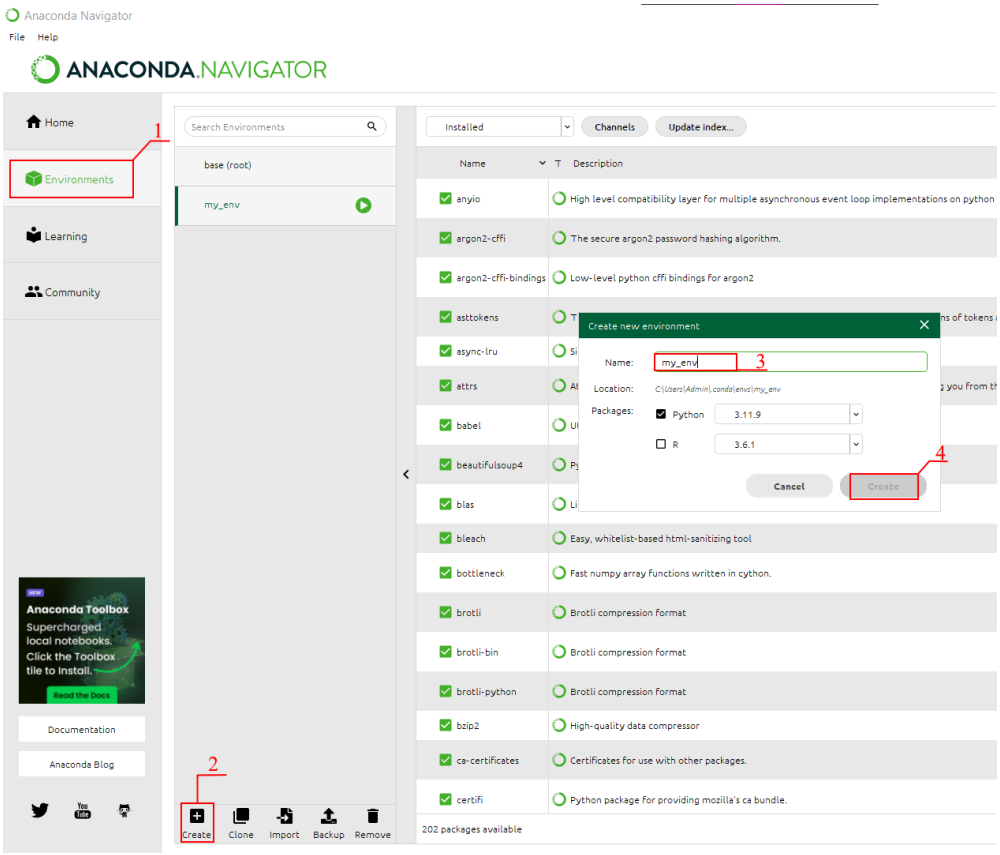
3 РЕАЛІЗАЦІЯ НЕЙРОННОЇ МЕРЕЖІ

У цій роботі нейронну мережу реалізовано на мові програмування Python як з використанням платформи Anaconda, так і Google Colab, використовуючи модель YOLOv8 від Ultralytics, натреновану на великому наборі даних OpenImage7. Використовуючи Anaconda було створено окреме локальне контрольоване середовище, в яке було завантажено та встановлено бібліотеки PyTorch, NumPy, Ultralytics YOLOv8 та OpenCV. Також було використано можливості хмарного середовища Google Colab. Це дозволило безкоштовно отримати доступ до потужних обчислювальних ресурсів, включаючи GPU, і використовувати заздалегідь встановлені середовища, такі як PyTorch, для ефективної роботи з нейронною мережею.

3.1 Реалізація нейронної мережі з використанням Anaconda

Перед використанням платформи Anaconda було встановлено Python 3.11, доступний для завантаження через Microsoft Store, що являє собою стандартний інтерпретатор Python без будь-яких додаткових змін або розширень, таких як IDLE (Integrated Development and Learning Environment). Цей інтерпретатор надає можливість розробки та виконання Python-коду безпосередньо на персональному комп’ютері.

Для встановлення Anaconda на персональний комп’ютер було завантажено виконавчий файл-інсталятор з офіційного веб-сайту та встановлено за стандартних параметрів. Після встановлення запущено програму Anaconda Navigator, через яку було інстальовано CMD Prompt та створено окреме середовище my_env, що розташовується в папці envs, яка знаходиться в корінній папці Anaconda (рис. 3.1).



1 – перейти до вкладки «Environments»; 2 – обрати «Create»; 3 – ввести назву середовища; 4 – створити середовище натиснувши «Create»

Рисунок 3.1 – Створення середовища за допомогою Anaconda Navigator

Після створення середовища треба запустити Anaconda Prompt та ввести команду переходу до корінної папки середовища та його запуску (рис. 3.2-3.3).

```
(my_env) C:\Users\Admin>cd %CONDA_PREFIX%
```

Рисунок 3.2 – Команда переходу до корінної папки

```
(my_env) C:\ProgramData\anaconda3\envs\my_env>conda initiate my_env
```

Рисунок 3.3 – Команда запуску середовища

Після цього послідовно вводяться команди завантаження бібліотек PyTorch, OpenCV, драйвери CUDA та інші (рис. 3.4).

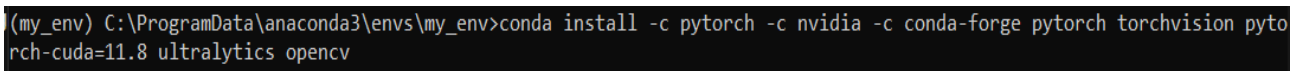


Рисунок 3.4 – Команда встановлення необхідних бібліотек

Після завершення завантаження вводиться команда тренування вже існуючої моделі за обраним датасетом (рис. 3.5). Датасет та існуючу модель можна завантажувати як за допомогою API ключа, так і завантажити його вручну. Датасет повинен відповідати формату .yaml, що містить в собі окремі папки з зображеннями та текстовими файлами, в котрих збережено дані координат об’єктів на зображенні. У випадку з використанням Anaconda для створення локального середовища датасет та існуючу модель YOLOv8 модифікації Small було завантажено вручну.

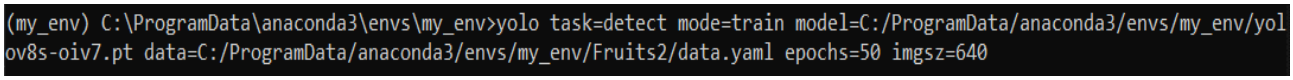


Рисунок 3.5 – Команда тренування на базі існуючої моделі

Після запуску цієї команди починається перевірка параметрів тренування, перевірка існування файлів необхідних для тренування та після цього починається сам процес тренування, створення нових ваг на базі вказаної існуючої моделі. Параметри, згідно яких було проведено процес тренування моделі, наведено у таблиці 3.1.

Таблиця 3.1 – Параметри тренування моделі

Параметр	Опис параметра
Кількість епох	50
Розмір батчу	16
Розмір зображення	640 · 640 пікселів
Швидкість навчання	0,01

На персональному комп’ютері з вказаним у таблиці 3.2 обладнанням цей процес зайняв приблизно 6 год 20 хв або, згідно даних, виведених в інтерфейс консолі після завершення, 6,283 год.

Таблиця 3.2 – Обладнання персонального комп’ютеру

Обладнання	Опис
Центральний процесор	Intel(R) Core(TM) i5-8300H, 2,30 ГГц
Оперативна пам’ять	20 ГБ DDR4
Жорсткий диск/SSD	SSD 512 ГБ
Графічний процессор	NVIDIA GeForce GTX 1050 2 ГБ

Проведемо оцінку моделі завдяки команді val (скорочено від англ.Validation – перевірка) на оціночному датасеті, відмінному від навчального та проаналізуємо графіки, згенеровані в результаті оцінки, та порівняємо декілька розмічених зображень з результатом визначення моделлю (рис. 3.6-3.11).

```
(my_env) C:\ProgramData\anaconda3\envs\my_env>yolo val model=C:/ProgramData/anaconda3/envs/my_env/runs/detect/train4/weights
/best.pt data=C:/ProgramData/anaconda3/envs/my_env/Fruits3/data.yaml imgsz=640 batch=16
```

Рисунок 3.6 – Команда оцінки натренованої моделі

Графік нормалізованої матриці помилок на рисунку 3.7 відображає, як модель класифікує об’єкти. Діагональні значення, рівні 1,00, показують, що всі приклади яблук та фону були правильно класифіковані. Це відмінний результат, що не супроводжується помилками класифікації. Проте такий результат скоріш є випадковим, ніж реальністю, оскільки навіть найточніші моделі можуть допускати помилки, хоча їх частка і становить менше 1 %.

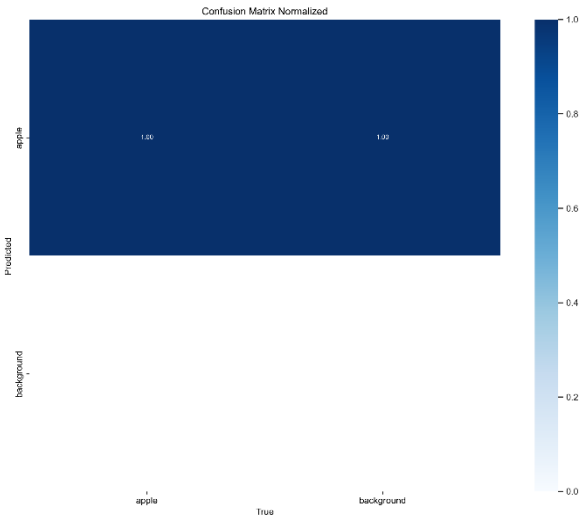


Рисунок 3.7 – Графік нормалізованої матриці помилок

Графік кривої F1-метрики по відношенню до впевненості на рисунку 3.8 демонструє баланс між точністю та повнотою за різних рівнів впевненості. Ідеальна крива була б прямокутником у верхньому лівому куті. Отримана крива наближається до ідеальної, що свідчить про високу продуктивність моделі.

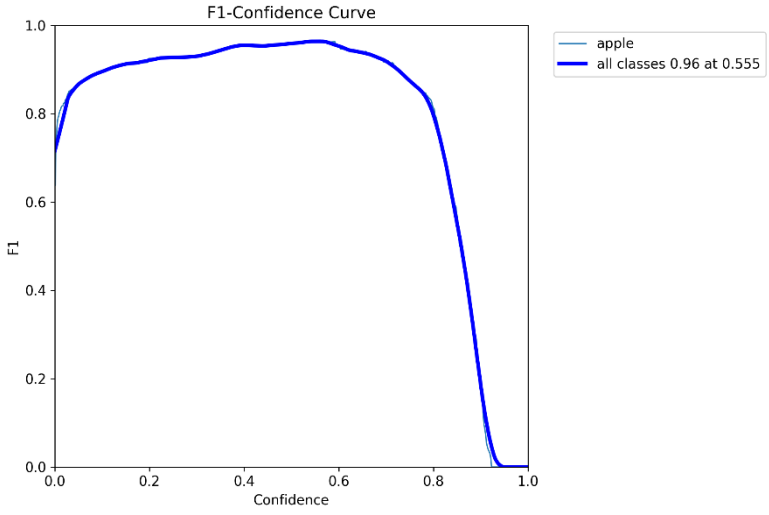


Рисунок 3.8 – Графік F1-метрики по відношенню до впевненості

Графік кривої Precision-Confidence на рисунку 3.9 показує співвідношення між точністю та впевненістю моделі. Ідеальна крива була б горизонтальною лінією на рівні 1,0 точності. Отримана крива наближається до ідеальної,

особливо за високих значень впевненості, а задовільна точність за низьких значень впевненості свідчить про здатність ідентифікувати об’єкти навіть за низьких рівнів впевненості, що корисно у практичних застосуваннях.

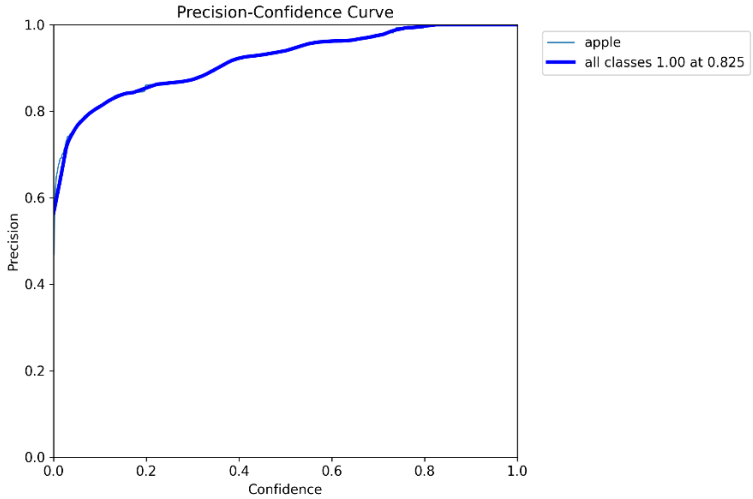


Рисунок 3.9 – Графік співвідношення між точністю та впевненістю

На графіку, зображеному на рисунку 3.10, показано криву точність-повнота, яка використовується для оцінки здатності моделі знаходити всі об’єкти (повнота) та правильно їх класифікувати (точність). Ідеальна крива мала б форму прямокутника у верхньому правому куті. Отримана крива дуже близька до ідеальної, що свідчить про високу якість виявлення об’єктів.

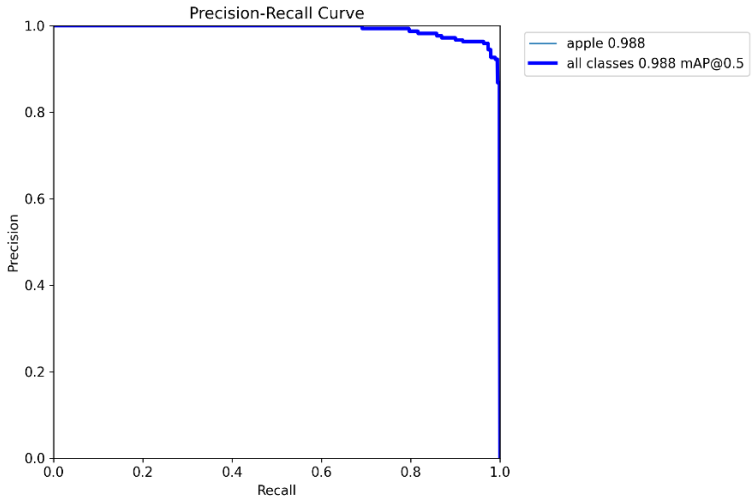


Рисунок 3.10 – Графік співвідношення між точністю та повнотою

Графік кривої на рисунку 3.11 демонструє співвідношення між повнотою та впевненістю моделі. Ідеальна крива мала б форму горизонтальної лінії на рівні повноти 1,0. Отримана крива краще задовільної та наближається до ідеальної, що вказує на здатність моделі виявляти більшість об’єктів навіть за низької впевненості.

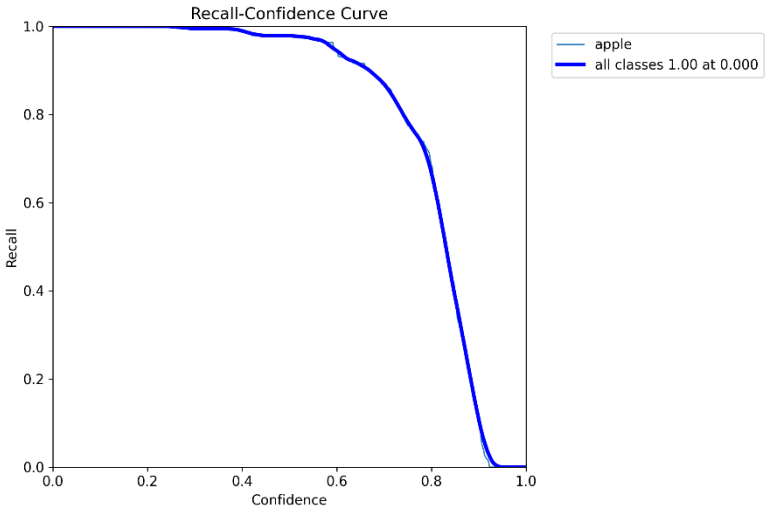


Рисунок 3.11 – Графік співвідношення між повнотою та впевненістю

Проаналізувавши оціночні графіки, можна зробити висновок, що результат задовільний, модель має високу здатність виявлення об’єктів навіть за низького рівня впевненості.

Застосуємо натреновану модель на декількох відео та проаналізуємо результати (рис. 3.12-3.13). Щоб застосувати модель на окремих файлах, треба призначити аргументу mode значення predict та вказати аргумент source, в якому буде прописаний шлях до цих файлів.

```
(my_env) C:\ProgramData\anaconda3\envs\my_env>yolo task=detect mode=predict model=C:/ProgramData/anaconda3/envs/my_env/runs/detect/
train4/weights/best.pt source=C:/ProgramData/anaconda3/envs/my_env/newdataset
```

Рисунок 3.12 – Команда застосування моделі на даних з папки newdataset

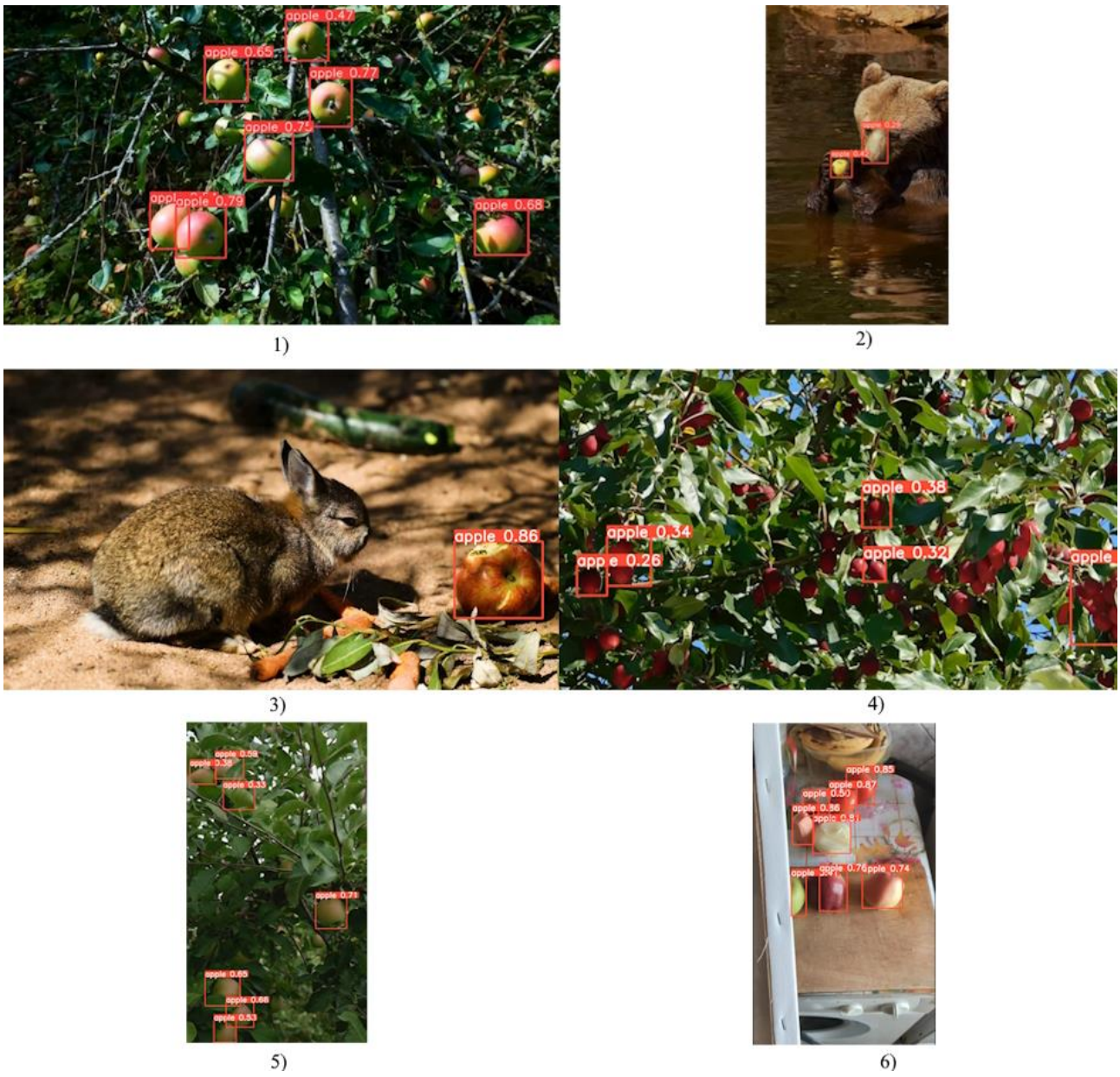


Рисунок 3.13 – Колаж результатів роботи моделі

Проаналізуємо представлені на рисунку 3.13 кадри.

На кадрі 1 зображено дерево з яблуками на його гілках. Модель виявила більшість плодів, які знаходилися відносно ближче до камери, з задовільним рівнем впевненості. Плоди, розташовані на більшій відстані від камери або ті, що мають розмите зображення, модель не ідентифікувала. Помилкових ідентифікацій не було.

Кадр 2 містить тварину, яка тримає яблуко в лапах. Система виявила яблуко, що тримає в лапах тварина, з невисоким рівнем впевненості. Також

частина тварини також була ідентифікована як яблуко з низьким рівнем впевненості.

Кадр 3 містить яблуко та тварину, в оригіналі відео обидва об’єкти не рухаються. Яблуко було ідентифіковано з високим рівнем впевненості. Помилкових ідентифікацій не було.

На кадрі 4 зображено дерево з великою кількістю яблук на його гілках з низьким рівнем їх освітленості. Обраний кадр результату роботи моделі демонструє ідентифікацію окремих груп яблук у різних зонах із низькою впевненістю. Помилкових виявлень не було.

На кадрі 5 зображено дерево з яблуками на його гілках, колір яких має схожий відтінок з оточенням. Було виявлено всі яблука, не сховані за гілками або листям, з рівнем впевненості, що варіюється від низького до задовільного. Один із листків помилково визначено як яблуко з низьким рівнем впевненості.

Кадр 6 містить три яблука та декілька овочів. Усі яблука були виявлені із задовільним рівнем впевненості. Об’єкти, які не є яблуками, але мають схожі риси, були помилково ідентифіковані з високим рівнем впевненості.

Результат роботи моделі демонструє, що модель задовільно працює для невеликої кількості об’єктів, але позитивно ідентифікує елементи, які не є яблуками. Причинами таких результатів є модифікація передтренованої моделі, кількість класів, які ідентифікувала передтренована модель, та недостатня кількість епох.

Модифікація YOLOv8 Small є моделлю, яка, згідно із заявами розробників, являє компроміс між точністю та ресурсоємністю. За словами розробників модель забезпечує високу точність виявлення об’єктів за помірного споживання обчислювальних ресурсів, що робить її ефективною для використання в умовах обмежених ресурсів, таких як мобільні пристрої або вбудовані системи.

Датасет OpenImageV7, на якому проводилося тренування передтренованої моделі, містить понад 9 мільйонів зображень, що охоплюють більше 600 різних класів об’єктів. Завдяки великій кількості класів модель отримує широкий

спектр інформації, що сприяє її загальній ефективності. Однак, через те, що модель натренована на великій кількості класів, за умови перетренування на меншу кількість класів можуть виникати проблеми, які було виявлено під час аналізу результатів роботи моделі.

Кількість епох тренування цієї моделі становила 50. Разом із відносно невеликим датасетом для тренування це призвело до того, що ваги передтренованої моделі не були перезаписані достатньо. Це може вказувати на те, що модель зберегла більшість своїх початкових параметрів, що були отримані під час тренування на великому датасеті OpenImageV7.

3.2 Реалізація нейронної мережі з використанням Google Colab

Для початку роботи в хмарному середовищі Google Colab необхідно мати обліковий запис Google. Після входу в систему можна створити новий зошит, вибравши відповідну опцію у меню Colab.

Моделі YOLO оптимізовані для роботи на графічних процесорах, які можна обрати в хмарному середовищі Google Colab. Використовуючи Google Colab, користувачі можуть легко вибрати середовище з підтримкою GPU, що дозволяє значно скоротити час тренування моделей YOLO. Тому в налаштуваннях середовища виберемо T4 GPU – графічний процесор, розроблений для широкого спектру обчислювальних задач, зокрема для обробки даних, штучного інтелекту та машинного навчання.

Завдяки більшому обсягу обчислювальних ресурсів ми маємо можливість використовувати модель YOLOv8 модифікації Medium, яка відрізняється підвищеною точністю та продуктивністю порівняно з меншими моделями. Ця модифікація забезпечує більш глибоке навчання та здатна виявляти об’єкти з високою точністю навіть у складних умовах, таких як зображення з низьким рівнем освітлення або багатошаровими об’єктами. Також застосуємо датасет з більшою кількістю даних.

Створимо новий зошит та введемо команду завантаження бібліотеки Ultralytics в особистому проєкті Google Colab (рис. 3.14).

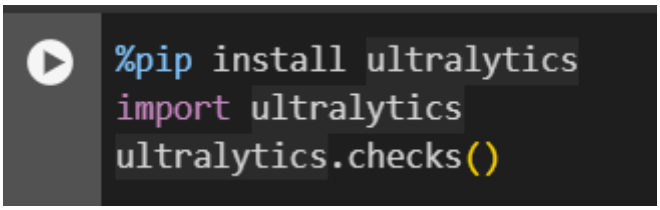


Рисунок 3.14 – Команда завантаження бібліотеки Ultralytics

Після завершення бібліотеки введемо команду тренування вже існуючої моделі YOLOv8 модифікації Medium за датасетом, що був використаний під час роботи у локальному середовищі (рис. 3.15).

Рисунок 3.15 – Команда тренування на базі існуючої моделі

Процес виконання команди такий самий, як і в випадку її виконання в локальному середовищі, розрізняються лише обчислювальні ресурси, задіяні в тренуванні, та кількість епох. Параметри тренування наведені в таблиці 3.3.

Таблиця 3.3 – Параметри тренування моделі

Параметр	Опис параметра
Кількість епох	150
Розмір батчу	16
Розмір зображення	640 · 640 пікселів
Швидкість навчання	0,01

Для навчання за таких параметрів на обладнанні, яке наведене в таблиці 3.4, знадобилось приблизно 20 хв або, згідно даних, виведених в інтерфейсі консолі після завершення, 0,248 год.

Таблиця 3.4 – Обладнання надане Google Colab

Обладнання	Опис
Центральний процесор	Intel(R) Xeon(R) CPU, 2,20 ГГц
Оперативна пам'ять	83 Гб
Жорсткий диск/SSD	Сервіс не надає інформації
Графічний процесор	NVIDIA A100-SXM4-40GB

Проведемо оцінку моделі завдяки команді val на оціночному датасеті та проаналізуємо графіки згенеровані у результаті оцінки (рис. 3.16-3.21). Параметри команди ідентичні з параметрами, використаними під час роботи у локальному середовищі.

```
!yolo task=detect mode=val model=/content/runs/detect/train/weights/best.pt data=https://universe.roboflow.com/ds/g9qP5ob2Fv?key=5AdOLMeIEf imgsz=640 batch=16
```

Рисунок 3.16 – Команда оцінки натренованої моделі

Згідно з графіком нормалізованої матриці помилок на рисунку 3.17, модель класифікує об’єкти правильно у 90 % випадків. Це є гарним результатом, що свідчить про низьку ймовірність помилкової ідентифікації об’єктів на фоні, які не є шуканим об’єктом, тобто яблуком.

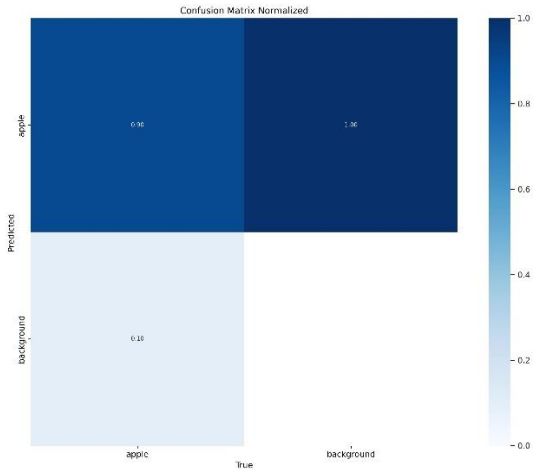


Рисунок 3.17 – Графік нормалізованої матриці помилок

Графік кривої F1-метрики по відношенню до впевненості на рисунку 3.18 показує, що максимальне значення F1 близько 0,86 досягається за порогу впевненості 0,338, що можна вважати гарним балансом між точністю та повнотою виявлення.

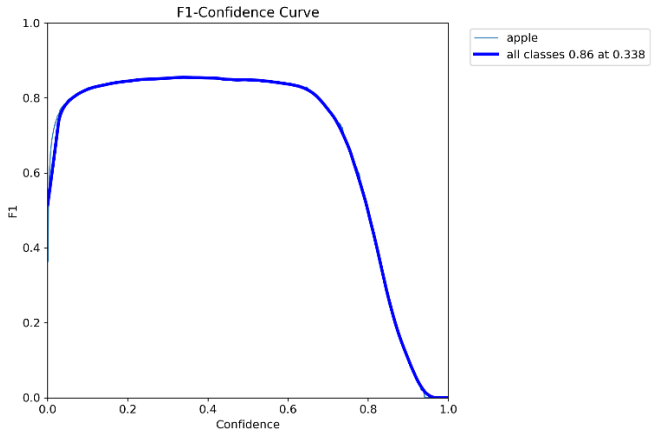


Рисунок 3.18 – Графік F1-метрики по відношенню до впевненості

Графік кривої співвідношення між точністю та впевненістю на рисунку 3.19 показує, що досягнення високої точності відбувається на високому рівні впевненості, близько 0,914, тоді як на низькому рівні впевненості спостерігається низька точність. Це свідчить про меншу ймовірність помилкової

ідентифікації об’єктів, у яких модель не впевнена, що є позитивним результатом під час практичного використання моделі.

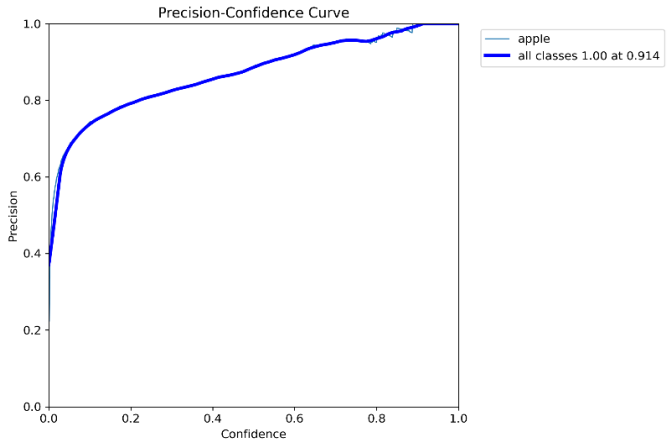


Рисунок 3.19 – Графік співвідношення між точністю та впевненістю

На графіку, зображеному на рисунку 3.20, крива відображає, що максимальна точність 0,913 досягається за повноти близько 0,5. Це означає, що для виявлення за такою точністю модель буде пропускати половину присутніх яблук на зображенні.

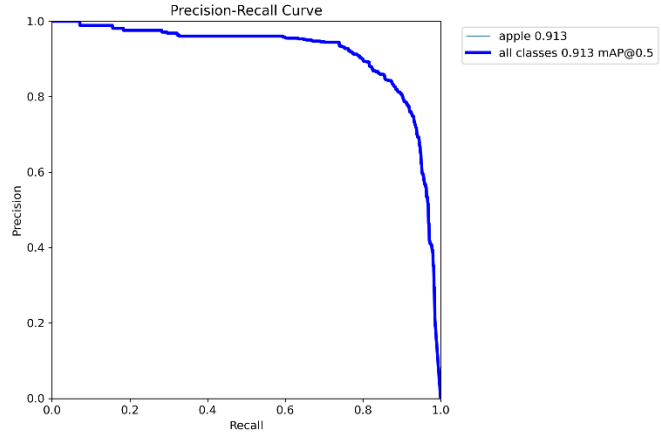


Рисунок 3.20 – Графік співвідношення між точністю та повнотою

Графік співвідношення між повнотою та впевненістю рисунку 3.21 плавно знижується, починаючи з високих значень повноти за низьких значень впевненості, що свідчить про наявність значної кількості помилкових виявлень,

і знижується до нуля за високих значень впевненості, що є ознакою того, що модель стає більш вибірковою, зменшуючи повноту, але потенційно збільшуючи точність.

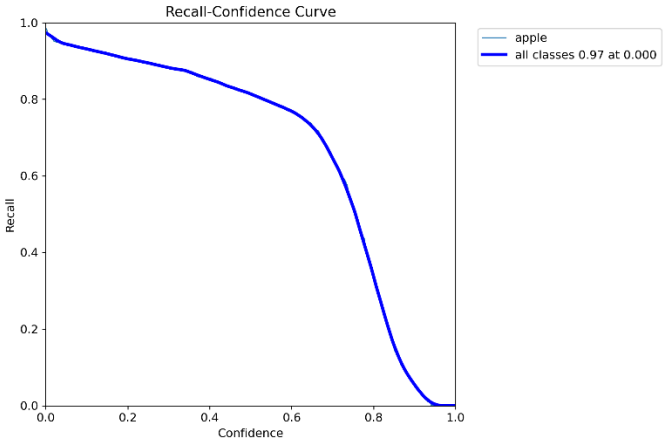


Рисунок 3.21 – Графік співвідношення між повнотою та впевненістю

Проаналізувавши графіки, можна зробити висновок, що результат тренування задовільний, але для підвищення ефективності роботи моделі треба знайти таке нижнє значення впевненості, за якого буде виявлено найбільша кількість об’єктів з найменшою кількістю помилок.

Застосуємо натреновану модель на декількох відео та проаналізуємо результати (рис. 3.22-3.23).

```
!yolo task=detect mode=predict model=/content/runs/detect/train/weights/best.pt source=/content/drive/MyDrive/Diploma/newdataset
```

Рисунок 3.22 – Команда застосування моделі на даних з папки newdataset

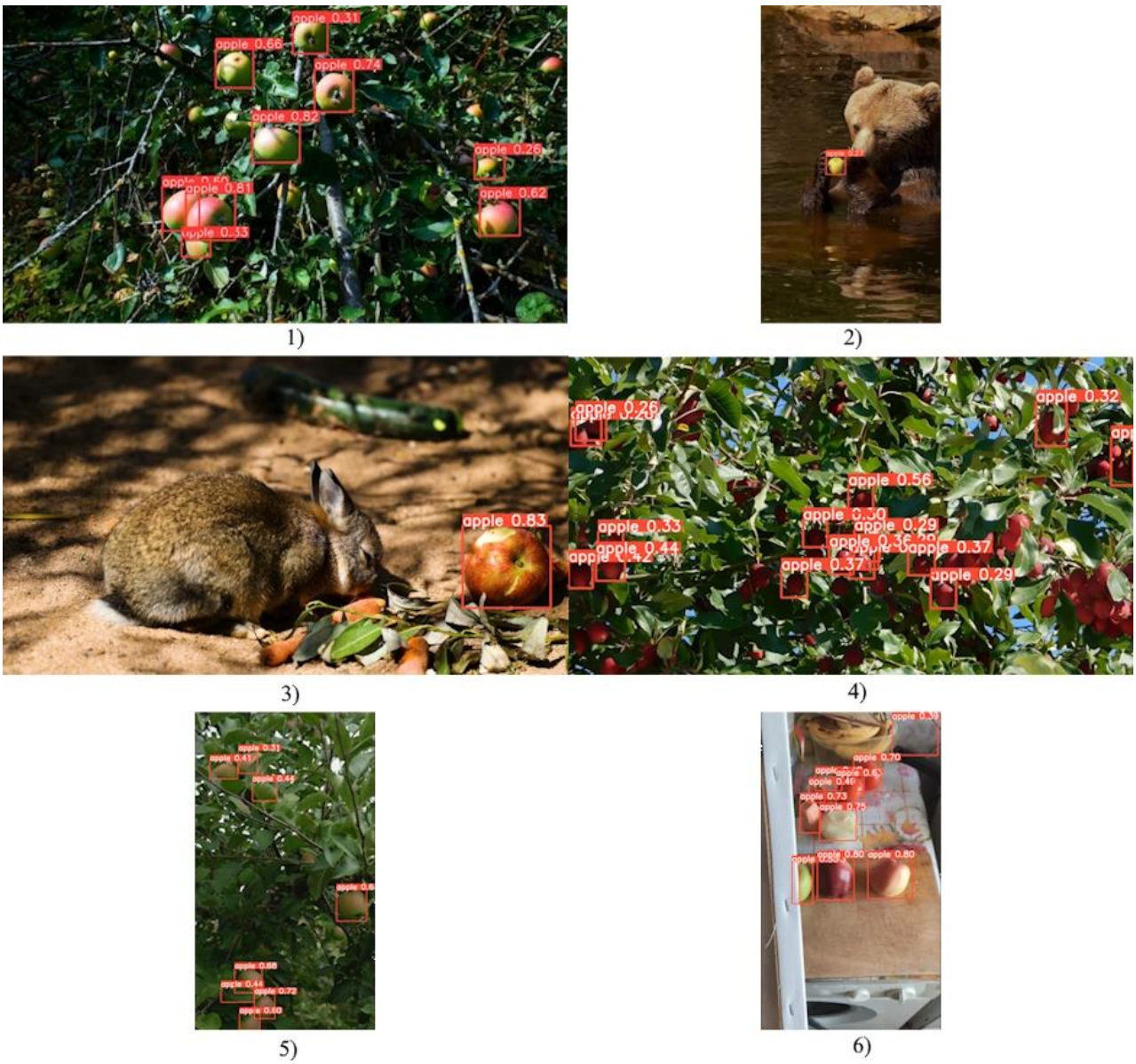


Рисунок 3.23 – Колаж результатів роботи моделі

На кадрі 1 модель виявила більше плодів та з більшою впевненістю, у порівнянні з результатом роботи в локальному середовищі. Зроблено 1 помилкову ідентифікацію.

На кадрі 2 система виявила яблуко, що тримає в лапах тварина, з низьким рівнем впевненості. Тварина хибно ідентифікована не була.

На кадрі 3 яблуко було ідентифіковано з високим рівнем впевненості. Помилкових ідентифікацій не було.

Кадр 4 демонструє ідентифікацію більшої кількості окремих яблук з невисокою впевненістю. Помилкових виявлень не було.

На кадрі 5 було виявлено всі яблука, не сховані за гілками або листям, з задовільним рівнем впевненості. Два листика помилково визначено як яблуко з низьким рівнем впевненості.

На кадрі 6 усі яблука були виявлені із виском рівнем впевненості. Об’єкти, які не є яблуками, але мають схожі риси, були помилково ідентифіковані з меншим рівнем впевненості.

Результат роботи моделі демонструє, що модель задовільно працює за невеликої кількості об’єктів, але позитивно ідентифікує елементи, які не є яблуками. Причинами таких результатів є модифікація передтренованої моделі, кількість класів, які ідентифікувала передтренована модель, та недостатня кількість епох.

Проаналізувавши результати роботи моделі, можна зазначити, що повнота та впевненість у вірних випадках виявлення об’єктів збільшились. Однак, з’явились нові помилкові ідентифікації, впевненість в яких незначна. Незважаючи на це, деякі проблеми, такі як помилкова ідентифікація та неповна ідентифікація об’єктів, все ще не вдалося повністю подолати. Проте, спостерігається помітне покращення у загальній продуктивності моделі.

Зазначимо, що під час практичного застосування встановлення відносно високого нижнього порогу впевненості дозволить уникнути більшості помилкових ідентифікацій. Це забезпечить більш надійне та точне виявлення об’єктів, мінімізуючи ризик хибних спрацьовувань.

3.3 Використання реалізованої нейронної мережі для ідентифікації об’єктів у реальному часі

Застосуємо реалізовану нейронну мережу для ідентифікації об’єктів у реальному часі. Забезпечуватиме зйомку зображень та їх передачу для подальшої обробки мікроконтролер ESP32-CAM.

ESP32-CAM – це мікроконтролер із зйомною камерою OV2640 2MP, що підтримує бездротову передачу даних через Wi-Fi, характеристики якого наведені в таблиці 3.5. Завдяки своїм можливостям, він є гарним вибором для реалізації системи ідентифікації об’єктів у реальному часі.

Таблиця 3.5 – Характеристики мікроконтролера ESP-32CAM

Характеристика	Значення
Напруга живлення	5 В
Бездротовий модуль	ESP32-S WiFi 802.11 b/g/n + Bluetooth 4.2 LE з друкованою антеною
Оперативна пам’ять	4 МБ PSRAM

Налаштуємо ESP32-CAM за допомогою середовища Arduino IDE. Arduino IDE є популярним середовищем розробки, яке підтримує широкий спектр мікроконтролерів, включаючи ESP32-CAM. Це середовище дозволяє розробляти, компілювати та завантажувати код на мікроконтролер, забезпечуючи простий і ефективний процес налаштування [19].

Спочатку необхідно завантажити та встановити Arduino IDE з офіційного сайту. Після цього слід додати підтримку плати ESP32, що виконується в менеджері плат Arduino IDE (рис. 3.24).

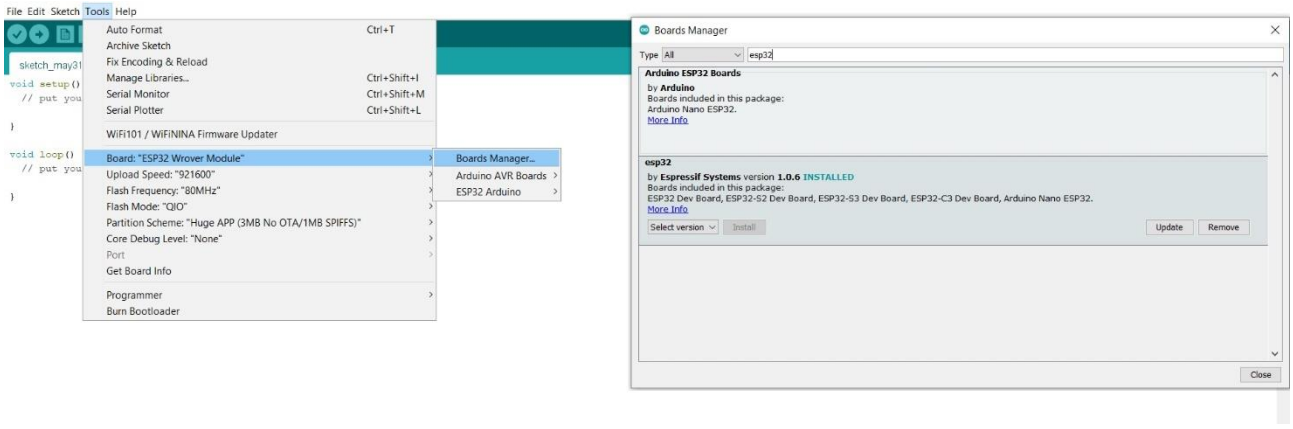


Рисунок 3.24 – Підготовка середовища Arduino

Після успішної підготовки середовища розробки виберемо готовий приклад скетчу під назвою CameraWebServer, введемо власні налаштування локальної мережі та встановимо налаштування плати (рис. 3.25). По завершенню встановлення перейдемо в монітор порта, в якому побачимо інформацію із посиланням на трансляцію камери (рис. 3.26).

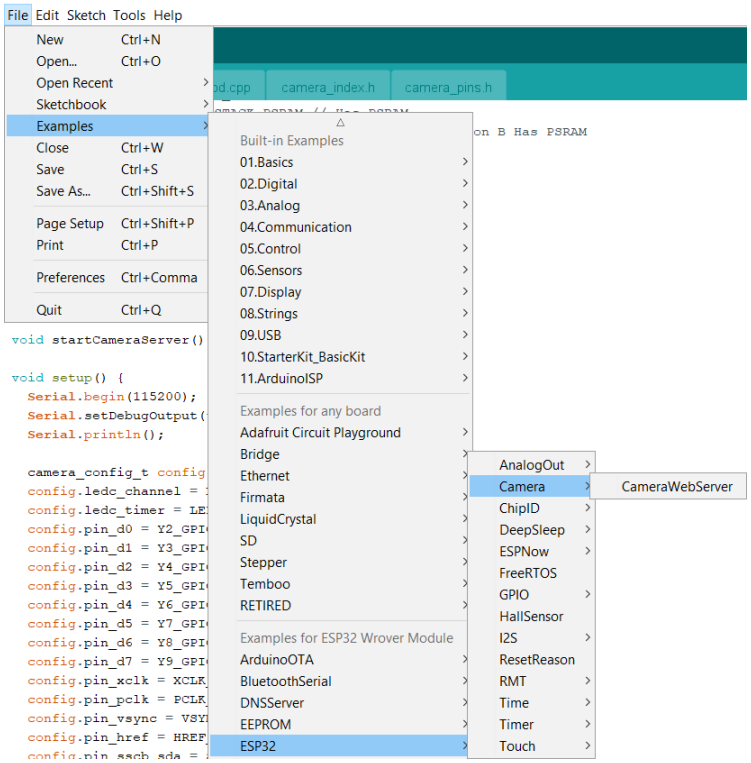


Рисунок 3.25 – Завантаження прикладу скетчу CameraWebServer

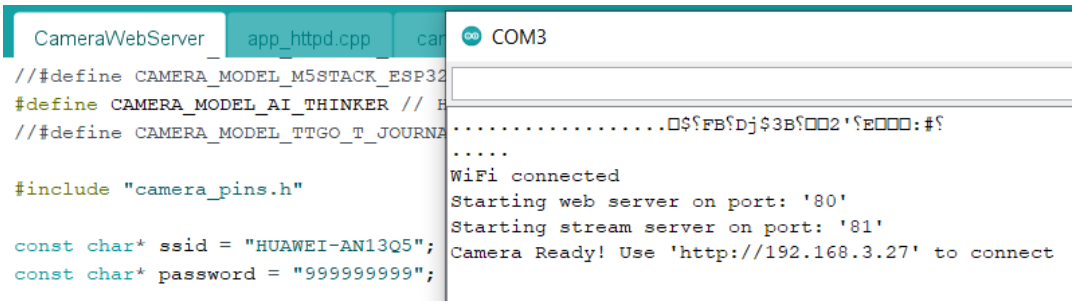


Рисунок 3.26 – Вікно монітору порта

На сторінці з трансляцією натиснемо кнопку «Start stream», натиснемо праву кнопку миші по зображенню, яке з’явиться, та скопіюємо посилання на зображення (рис. 3.27).

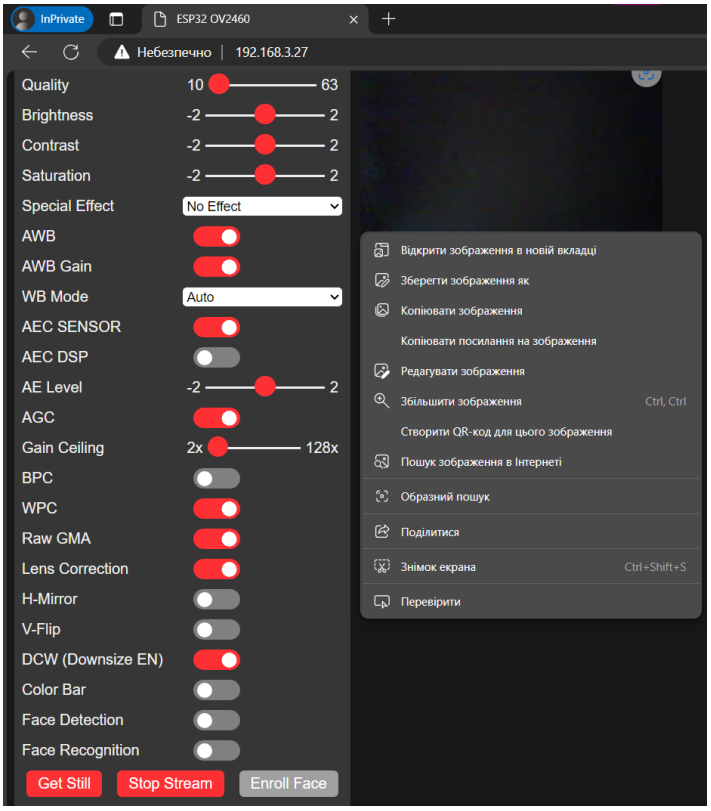


Рисунок 3.27 – Сторінка за посиланням отриманим з монітору порта

Після проведення цих дій матимемо посилання у форматі http://адреса:81/stream.

Напишемо сценарій на мові програмування Python, завдяки якому натренована модель буде виконувати ідентифікацію об’єктів у реальному часі із зображення, отриманого з ESP-32CAM по локальній мережі (рис. 3.28).

```
import keyboard
from ultralytics import YOLO

def main():
    model = YOLO('C:/ProgramData/anaconda3/envs/my_env/gcm/weights/best.pt')
    while True:
        results = model(source='http://192.168.3.27:81/stream', show=True, save=False)
        if keyboard.is_pressed('q'):
            print("Кнопка 'q' натиснута. Зупинка сценарія.")
            break

if __name__ == "__main__":
    main()
```

Рисунок 3.28 – Вигляд сценарію

Спочатку імпортуються бібліотеки keyboard для взаємодії з клавіатурою та YOLO з ultralytics для виявлення об’єктів на зображеннях або відео.

Далі визначається головна функція main(). Всередині цієї функції завантажується попередньо натренована модель YOLO з файлу 'best.pt', який знаходиться у вказаному шляху.

Після цього розпочинається нескінченний цикл while True:. У циклі модель YOLO використовується для виявлення об’єктів на відеопотоці, що надходить з заданого URL http://192.168.3.27:81/stream. Результати виявлення об’єктів виводяться на екран у реальному часі, але не зберігаються в файл.

Цикл переривається свою роботу, якщо користувач натискає клавішу 'q' на клавіатурі. У цьому випадку виводиться повідомлення "Кнопка 'q' натиснута. Зупинка сценарію.".

Останнім перевіряється, чи виконується код у головному потоці. Якщо так, то викликається функція main(). Це забезпечує, що код у функції main() виконається лише тоді, коли сценарій запускається безпосередньо, а не імпортується як модуль в іншій програмі.

Застосуємо сценарій на практиці. Щоб це зробити, відкриємо застосунок Anaconda Prompt, введемо команду для переходу в папку середовища «cd %CONDA_PREFIX%» та ініціюємо роботу сценарію «python ods.py», попередньо увімкнувши плату ESP-32CAM (рис. 3.29).

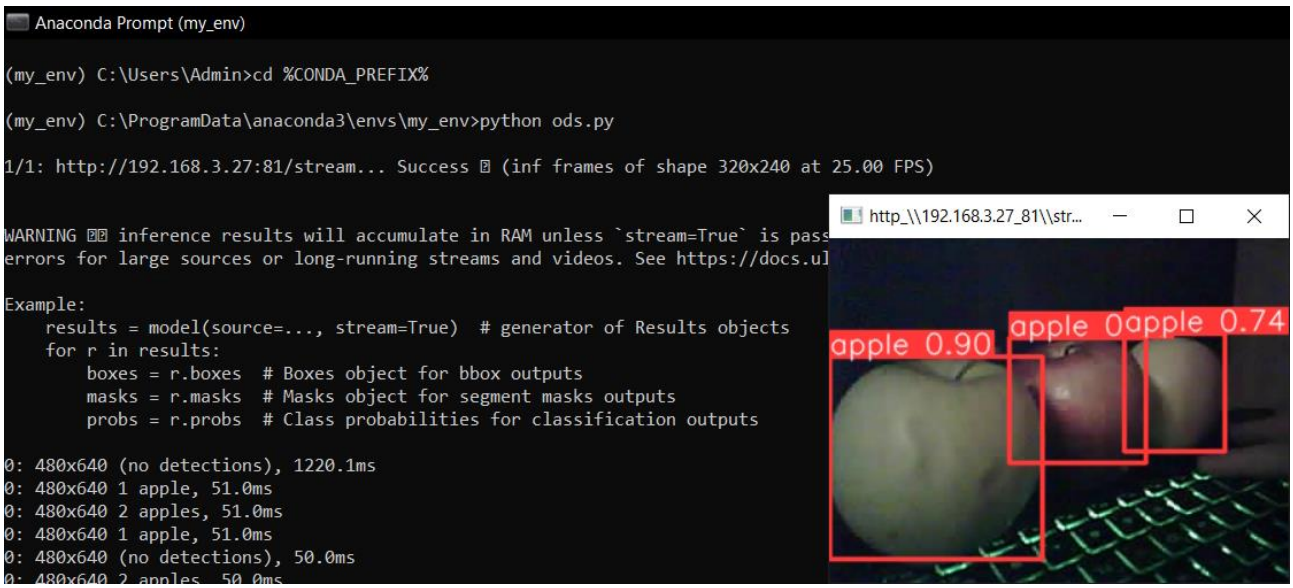


Рисунок 3.29 – Результат роботи сценарію

Після запуску сценарію та початку роботи ідентифікації було створено вікно розміром 320 пікселів × 240 пікселів, що відповідає розміру отриманого зображення. На зображенні об’єкти, визначені як яблуко, були окреслені червоними лініями та за умови переміщення камери або об’єктів процес виявлення продовжувався.

3.4 Аналіз стійкості системи передачі даних

Проведемо аналіз стійкості системи передачі даних з плати ESP-32CAM до ПК через локальну мережу.

Знайдемо передатні функції окремих компонентів системи: передатну функцію мережі, камери та передатну функції обробки даних на ПК.

Передатна функція локальної Wi-Fi мережі:

$$W_{wifi}(s) = \frac{e^{-Ts}}{T_{wifi}s+1}, \tag{3.1}$$

де τ – затримка в мережі;

T_{wifi} – постійна часу мережі.

Передатна функція ПК:

$$W_{\text{ПК}}(s) = \frac{1}{T_{\text{ПК}}s+1}, \tag{3.2}$$

де $T_{ПК}$ – постійна часу обробки даних ПК.

Передатна функція плати ESP-32CAM:

$$W_{ESP}(s) = \frac{1}{T_{ESP}s+1}, \tag{3.3}$$

де T_{ESP} – постійна часу захоплення й обробки відео платою ESP-32CAM.

Об’єднаємо ці окремі передатні функції в загальну передатну функцію системи:

$$W_{\text{C}}(s) = \frac{e^{-Ts}}{(T_{wifi}s+1)\cdot(T_{\text{ПК}}s+1)\cdot(T_{ESP}s+1)}. \tag{3.4}$$

Для оцінки стійкості системи використаємо алгебраїчний критерій Гурвіца. Критерій Гурвіца дозволяє визначити стійкість системи за допомогою аналізу коефіцієнтів характеристичного поліному [20]. Система є стійкою, якщо всі корені характеристичного поліному мають від’ємні дійсні частини.

Характеристичний поліном системи передачі даних $P(\lambda)$ визначається знаменником передатної функції системи. Для нашої системи:

$$P(\lambda) = \left(T_{wifi}\lambda + 1\right) \cdot (T_{\text{ПК}}\lambda + 1) \cdot (T_{ESP}\lambda + 1). \tag{3.5}$$

Розкриємо дужки:

$$P(\lambda) = T_{wifi}T_{\text{ПК}}T_{ESP} \cdot \lambda^3 + \left(T_{wifi}T_{\text{ПК}} + T_{wifi}T_{ESP} + T_{\text{ПК}}T_{ESP}\right) \cdot \lambda^2 + \\ + \left(T_{wifi}T_{\text{ПК}} + T_{\text{ПК}} + T_{ESP}\right)\lambda + 1. \quad (3.6)$$

Запишемо коефіцієнти полінома: $a_0 = 1;\ \ a_1 = T_{wifi}T_{\text{ПК}} + T_{\text{ПК}} + T_{ESP}$; $a_2 = T_{wifi}T_{\text{ПК}} + T_{wifi}T_{ESP} + T_{\text{ПК}}T_{ESP}$; $a_3 = T_{wifi}T_{\text{ПК}}T_{ESP}$.

Сформуємо матрицю Гурвіца:

$$\Delta_3 = \begin{vmatrix} a_1 & a_3 & 0 \\ a_0 & a_2 & 0 \\ 0 & a_1 & a_3 \end{vmatrix}. \quad (3.7)$$

Згідно з критерієм Гурвіца, система є стійкою, якщо всі головні мінори матриці Гурвіца є додатними. У нашому випадку необхідно, щоб виконувалися такі умови:

$$a_1 > 0, \Delta_2 = a_1a_2 - a_0a_3 > 0, a_3 > 0. \quad (3.8)$$

Умова стійкості $a_1 > 0$ виконується, якщо $T_{wifi}T_{\text{ПК}} + T_{\text{ПК}} + T_{ESP} > 0$. Значення $T_{wifi},\ T_{\text{ПК}},\ T_{ESP}$ завжди додатні, тому умова стійкості виконується.

Умова стійкості $a_3 > 0$ виконується, якщо $T_{wifi}T_{\text{ПК}}T_{ESP} > 0$. Значення $T_{wifi}, T_{\text{ПК}}, T_{ESP}$ завжди додатні, тому умова стійкості виконується.

Умова стійкості $a_1a_2 - a_0a_3 > 0$ виконується, якщо $(T_{wifi}T_{\text{ПК}} + T_{\text{ПК}} + \\ + T_{ESP})\left(T_{wifi}T_{\text{ПК}} + T_{wifi}T_{ESP} + T_{\text{ПК}}T_{ESP}\right) - T_{wifi}T_{\text{ПК}}T_{ESP} > 0$. Тобто потрібно щоб виконувалась умова $(T_{wifi}T_{\text{ПК}} + T_{\text{ПК}} + T_{ESP})\left(T_{wifi}T_{\text{ПК}} + T_{wifi}T_{ESP} + \\ + T_{\text{ПК}}T_{ESP}\right) > T_{wifi}T_{\text{ПК}}T_{ESP}$. Зазначимо, що постійні часу T не можуть бути меншими, ніж 1 мс або 0,001 с. Використаємо сценарій Scilab з додатку А для визначення, чи виконується ця умова стійкості для діапазону значень T від 0,001 до 5. Кінцеве значення 5 обумовлене тим, що за умови затримки, більшої за 5 с, система не зможе відповідати критерію швидкодії.

В результаті роботи сценарію (рис. 3.30) доведено, що умова стійкості $a_1 a_2 - a_0 a_3 > 0$ виконується за $T \in [0{,}001;\ 5]$, що є практично можливим діапазоном значень постійної часу *T*.

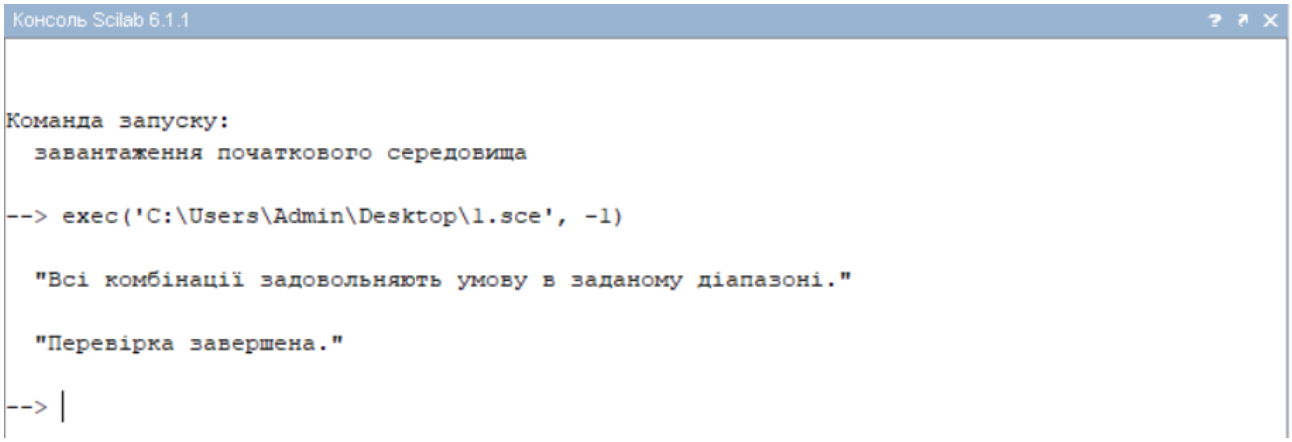


Рисунок 3.30 – Результат роботи сценарію Scilab

Оцінивши стійкість системи за алгебраїчним критерієм Гурвіца робимо висновок, що система стійка.

3.5 Заходи з охорони праці під час роботи з ПК

Під час роботи з ПК слід дотримуватись правил безпеки використання електричних пристроїв. Використання електронного обладнання може спричинити ураження електричним струмом.

Тривала робота за комп’ютером може призводити до перенапруження зору, болю у спині, синдрому карпального каналу. Можливість короткого замикання електронного обладнання може стати причиною пожежі. Використання хімічних речовин при виготовленні друкованих плат може призвести до отруєння чи алергічних реакцій [21].

При увімкненні або вимкненні комп’ютерів, іншої техніки та освітлення в електромережу слід братися тільки за ізольовані частини дротів.

Щоб уникнути розрядів статичної електрики, не допускається доторкання до екрана монітора.

При введенні даних, редагуванні програм, читанні інформації з екрана, безперервна тривалість роботи перед екраном не повинна перевищувати 1 год з наступними регламентованими перервами по 10 хв для відпочинку зору та виконуванню комплексу фізичних вправ, релаксаційної гімнастики й аутогенного тренування.

При увімкненому електроживленні комп’ютерів та оргтехніки:

– не розкривати захисні кожухи й кришки блоків ПК, не робити регулювання й чищення внутрішніх деталей, не змінювати запобіжники;

– не переключати сполучні дроти блоків;

– не змінювати встановлену конфігурацію робочого місця, не переставляти комп’ютери та інші електронні прилади;

– не робити вологе прибирання поверхонь комп’ютерів, моніторів та електронних приладів.

На робочому місці:

– не палити, не користуватися відкритим вогнем;

– не зберігати легкозаймисті, вибухонебезпечні і хімічно-активні, що руйнують ізоляцію, речовини;

– дотримуватись чистоти і порядку.

Забезпечення охорони праці під час розробки та експлуатації програмного забезпечення є критично важливим для запобігання виробничому травматизму, професійним захворюванням та зменшення негативного впливу на навколишнє середовище. Впровадження запропонованих заходів сприятиме підвищенню рівня безпеки та комфорту на робочих місцях[22].

ВИСНОВКИ

У ході виконання кваліфікаційної роботи проведено аналіз сучасних методів ідентифікації та проаналізовано архітектури нейронних мереж. Згідно з аналізом обрано відповідну вимогам архітектуру нейронної мережі. Обрано метод реалізації нейронної мережі.

Реалізовано нейронну мережу та проведено аналіз показників точності, швидкості та впевненості виявлення об’єктів реалізованою нейронною мережею. Впевненість ідентифікації об’єктів нейронною мережею варіювалась від 70 % до 90 %, що, разом з високою швидкістю ідентифікації, є задовільним результатом. Точність ідентифікації збільшувалась з підвищенням впевненості, але за низької впевненості відбувались хибні виявлення.

На основі аналізу цих показників було запропоновано способи покращення ефективності ідентифікації за допомогою нейронної мережі.

Заключним етапом було реалізовано використання розробленої нейронної мережі у реальному часі за зображенням, яке передається з зовнішнього пристрою локальною бездротовою мережею.

Після реалізації ідентифікації об’єктів у реальному часі проведено аналіз стійкості системи передачі даних з плати ESP-32CAM до ПК через локальну мережу за алгебраїчним критерієм Гурвіца. В ході аналізу було доведено, що система стійка.

Були наведені рекомендації з охорони праці під час роботи з комп’ютером та його периферійними пристроями

Розроблену нейронну мережу можна в подальшому покращити такими шляхами:

– провести додаткові тестування системи у різних умовах експлуатації, включаючи різне освітлення, погодні умови та варіації об’єктів, для забезпечення стабільної роботи у різних ситуаціях;

– створити зручний інтерфейс для користувачів, що дозволить легко налаштовувати систему та відслідковувати її роботу у реальному часі;

– розробити програмне забезпечення, оптимізоване для роботи на мобільних пристроях або вбудованих системах, що дозволить використовувати систему в умовах обмежених ресурсів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ДСТУ 3008-15. Документація. Звіти у сфері науки та техніки. Структура та правила оформлення. – Введ. 2015-06-22. – Держстандарт України, 2017 – 29 с.

2. Методичні вказівки з підготовки кваліфікаційної роботи для здобувачів першого (бакалаврського) рівня вищої освіти денної і заочної форми навчання спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» освітньої програми «Автоматизація та комп'ютерно-інтегровані технології» / Упоряд.: І. Ш. Невлюдов, О. І. Филипенко, О. В. Токарєва, С. П. Новоселов, О. В Сичова. – Харків: ХНУРЕ, 2023. – 64 с.

3. Glover Bill, RFID essentials [Електронний ресурс] / Glover Bill, Himanshu Bhatt. // O'Reilly Media, Inc., 2006. – 260 с. – Режим доступу: www/URL: https://books.google.com.ua/books?id=cKKZoH48D4cC. – Станом на 11.05.2024. – Назва з екрану.

4. What is a barcode and how does it work? [Електронний ресурс] / Irene Luong. – Режим доступу: www/URL: https://www.magestore.com/blog/what-is-a-barcode/. – Станом на 11.05.2024. – Назва з екрану.

5. QR code [Електронний ресурс]. – Режим доступу: www/URL: https://foxdesignsstudio.com/uploads/pdf/Three_QR_Code.pdf – Станом на 11.05.2024. – Назва з екрану.

6. Automated System Development for the Printed Circuit Boards Optical Inspection Using Machine Learning Methods / I. Nevliudov, I. Botsman, O. Chala, K. Khrustalev // INFORMATION SYSTEMS AND TECHNOLOGIES (IST'2021) : proceedings of the 10-th International Scientific and Technical Conference, September 13-19, Odessa, 2021. – С. 234-238.

7. Queries classification using machine learning for implementation in intelligent manufacturing / V. Bortnikova, V. Yevsieiev, I. Botsman, et al. // Methods

and tools in CAD – selected issues: monograph. Chapter 6. – Białystok : Publishing House of Bialystok University of Technology. – 2021. – c. 63-74.

8. Терейковський І. А. Штучні нейронні мережі: базові положення [Електронний ресурс]: навчальний посібник для здобувачів ступеня бакалавра за освітньою програмою «Системне програмування та спеціалізовані комп'ютерні системи» спеціальності 123 Комп'ютерна інженерія / І. А. Терейковський, Д. А. Бушуєв, Л. О. Терейковська; КПІ ім. Ігоря Сікорського. – Режим доступу: www/URL: https://ela.kpi.ua/handle/123456789/50135. – Станом на 11.05.2024. – Назва з екрану.

9. Аналіз генеративних моделей глибокого навчання та особливостей їх реалізації на прикладі WGAN [Електронний ресурс] / Ісаєнков Я. О., Мокін О. Б. – Режим доступу: www/URL: https://visnyk.vntu.edu.ua/index.php/visnyk/article/view/2737. – Станом на 11.05.2024. – Назва з екрану.

10. Conv2d: Finally Understand What Happens in the Forward Pass [Електронний ресурс] / Axel Thevenot. – Режим доступу: www/ URL: https://towardsdatascience.com/conv2d-to-finally-understand-what-happens-in-the-forward-pass-1bbaafb0b148. – Станом на 11.05.2024. – Назва з екрану.

11. Використовуємо CNN для обробки зображень. Частина перша. [Електронний ресурс] / Малишев О. – Режим доступу: www/URL: https://dou.ua/forums/topic/48368/. – Станом на 11.05.2024. – Назва з екрану.

12. Overview of two-stage object detection algorithms [Електронний ресурс] / Du Lixuan, Zhang Rongyu, Wang Xiaotian // Journal of Physics: Conference Series. – 2020. – Вип. 1544, стаття 012033. – Режим доступу: www/URL: https://www.researchgate.net/publication/341871095_Overview_of_two-stage_object_detection_algorithms – Станом на 11.05.2024. – Назва з екрану.

13. Yang, L., Zhao, D., Yan, X. Analysis of microbial diversity in soil samples from the Yellow River Delta [Електронний ресурс] / L. Yang, D. Zhao, X. Yan //

Scientific Reports. – 2020. – Вип. 10, стаття 79243. – Режим доступу: https://doi.org/10.1038/s41598-020-79243-9. – Станом на 11.05.2024. – Назва з екрану.

14. CUDA [Електронний ресурс]. – Режим доступу: www/URL: https://uk.wikipedia.org/wiki/CUDA. – Станом на 01.06.2024. – Назва з екрану.

15. Ласкаво просимо до Colab! [Електронний ресурс]. – Режим доступу: www/URL: https://colab.research.google.com. – Станом на 01.06.2024. – Назва з екрану.

16. We’re not just a company; we’re a movement [Електронний ресурс]. – Режим доступу: www/URL: https://www.anaconda.com/about-us. – Станом на 01.06.2024. – Назва з екрану.

17. Why TensorFlow [Електронний ресурс]. – Режим доступу: www/URL: https://www.tensorflow.org/about. – Станом на 01.06.2024. – Назва з екрану.

18. PyTorch [Електронний ресурс]. – Режим доступу: www/URL: https://www.nvidia.com/en-us/glossary/pytorch/. – Станом на 01.06.2024. – Назва з екрану.

19. About Arduino [Електронний ресурс]. – Режим доступу: www/URL: https://www.arduino.cc/en/about. – Станом на 01.06.2024. – Назва з екрану.

20. Теорія автоматичного управління (збірник задач) [Текст]: навч. посіб. для студентів спеціальності 151 Автоматизація та комп’ютерно-інтегровані технології / І. Ш. Невлюдов, О. В. Токарєва; Харків. нац. ун-т радіоелектроніки. – Харків: Панов А.М., 2020. – 240 с.

21. Які хвороби розвиваються через роботу за комп’ютером [Електронний ресурс]. – Режим доступу: www/URL: https://ukr.media/medicine/376451/#google_vignette – Станом на 01.06.2024. – Назва з екрану.

22. Методичні вказівки до виконання розділу «Охорона праці» у випускних роботах ОКР "бакалавр" усіх форм навчання / упоряд. В. А. Айвазов,

Т. Є. Стиценко, Н. Л. Березуцька; М-во освіти і науки України, ХНУРЕ. – Харків: ХНУРЕ, 2018. – 28 с. – 1,81.