МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили

Факультет комп'ютерних наук

Кафедра інтелектуальних інформаційних систем

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри інтелектуальних інформаційних систем

_____Юрій КОНДРАТЕНКО

«____»_____ 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА

НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

СИСТЕМА ІДЕНТИФІКАЦІЇ ТА РОЗПІЗНАВАННЯ ДРОНІВ МЕТОДАМИ ШТУЧНОГО ІНТЕЛЕКТУ

Спеціальність 122 Комп'ютерні науки

Освітня програма «Інтелектуальні інформаційні системи»

Здобувач _____Сергій ДАНКОВИЧ

«___»_____ 2024 р.

Керівник канд. техн. наук, доцент _____Євген СІДЕНКО

«___»_____ 2024 р.

Миколаїв – 2024

Чорноморський національний університет імені Петра Могили
(повне найменування закладу вищої освіти)

Факультет	Комп’ютерних наук
Кафедра	Інтелектуальних інформаційних систем
Рівень вищої освіти	Другий (магістерський)
Освітній ступень	Магістр
Спеціальність	122 Комп’ютерні науки
Освітня програма	Інтелектуальні інформаційні системи

ЗАТВЕРДЖУЮ

Завідувач кафедри інтелектуальних інформаційних систем
_____Юрій КОНДРАТЕНКО
«____» _____ 2024 р.

ЗАВДАННЯ
на кваліфікаційну роботу здобувача

Данковича Сергія Юрійовича
((прізвище, ім’я, по батькові здобувача)

1. Тема кваліфікаційної роботи: «Система ідентифікації та розпізнавання дронів методами штучного інтелекту».

Керівник роботи: Сіденко Євген Вікторович, доцент кафедри інтелектуальних інформаційних систем, канд. техн. наук, доцент.

Затверджена наказом ЧНУ ім. Петра Могили від «03» червня 2024 р. № 140/1.

2. Строк представлення кваліфікаційної роботи «16» грудня 2024 р.

3. Очікуваний результат роботи та початкові дані, якщо такі потрібні: система ідентифікації та розпізнавання дронів з використанням методів штучного інтелекту та набір даних із розміченими обмежувальними рамками об’єктів та їх класами.

4. Перелік питань, що підлягають розробці: аналіз предметної області та аналогічних систем; визначення ключових вимог до задачі; дослідження методів штучного інтелекту; створення набору даних із використанням синтетичних даних; навчання моделей, здатних розпізнавати дрони з високою точністю за різних умов; тестування моделей та аналіз результатів; розробка системи, що використовує створені моделі ідентифікації та розпізнавання дронів з урахуванням вимог до швидкості, точності та адаптивності.

5. Перелік графічних матеріалів: презентація.

Керівник роботи _____ Євген СІДЕНКО
(Особистий підпис) *(Власне ім'я ПРІЗВИЩЕ)*

Здобувач _____ Сергій ДАНКОВИЧ
(Особистий підпис) *(Власне ім'я ПРІЗВИЩЕ)*

Дата видачі завдання «07» червня 2024 р.

КАЛЕНДАРНИЙ ПЛАН
кваліфікаційної роботи

Тема: Система ідентифікації та розпізнавання дронів методами штучного інтелекту

№	Найменування роботи	Початок	Закінчення	Примітки
1	Отримання завдання на виконання КР	03.06.2024	07.06.2024	виконано
2	Аналіз предметної області та постановка задачі	10.06.2024	20.06.2024	виконано
3	Огляд літературних джерел за темою кваліфікаційної роботи, зокрема аналіз публікацій та аналогічних систем	21.06.2024	01.07.2024	виконано
4	Дослідження сучасних архітектури нейронних мереж, технологій формування наборів даних та інструментів для створення системи	01.09.2024	25.10.2024	виконано
5	Розробка, тестування та оптимізація запропонованої системи, проведення тестування та аналіз отриманих результатів	26.10.2024	21.11.2024	виконано
6	Перший попередній захист КР на засіданні комісії кафедри	22.11.2024	22.11.2024	виконано
7	Корегування роботи за результатами попереднього захисту	23.11.2024	05.12.2024	виконано
8	Другий попередній захист КР на засіданні комісії кафедри	06.12.2024	06.12.2024	виконано
9	Доробка та остаточне оформлення КР	07.12.2024	10.12.2024	виконано
10	Подання КР, її електронної копії та інших документів (відгуку, рецензії) до захисту	16.12.2024	17.12.2024	виконано

Керівник роботи _____ Євген СІДЕНКО
(Особистий підпис) *(Власне ім'я ПРІЗВИЩЕ)*

Здобувач _____ Сергій ДАНКОВИЧ
(Особистий підпис) *(Власне ім'я ПРІЗВИЩЕ)*

Дата складання календарного плану
«19» червня 2024 р.

АНОТАЦІЯ

до кваліфікаційної роботи
здобувача групи 601м ЧНУ ім. Петра Могили

Данковича Сергія Юрійовича

на тему: **“СИСТЕМА ІДЕНТИФІКАЦІЇ ТА РОЗПІЗНАВАННЯ ДРОНІВ МЕТОДАМИ ШТУЧНОГО ІНТЕЛЕКТУ”**

Актуальність зумовлена зростаючим попитом на системи для ідентифікації та розпізнавання дронів у зв’язку з їх широким використанням у різних галузях.

Об’єкт дослідження – процес ідентифікації та розпізнавання дронів методами штучного інтелекту.

Предметом дослідження є методи штучного інтелекту для створення системи ідентифікації та розпізнавання дронів.

Мета роботи є підвищення точності ідентифікації та розпізнавання та ідентифікації дронів шляхом використання сучасних методів штучного інтелекту.

В результаті виконання роботи було досліджено моделі для виявлення об’єктів, навчено на створеному наборі синтетичних даних, перевірено та проаналізовано результати на реальних зображеннях, а також розроблено систему, в якій інтегровано навчені моделі.

Робота складається з чотирьох розділів. У першому розділі розглянуто сучасний стан задачі ідентифікації та розпізнавання дронів методами штучного інтелекту. Другий розділ присвячено дослідженню методам та технологій, на яких базуватиметься система ідентифікації та розпізнавання дронів. У третьому розділі наведено проєктування цієї системи. Четвертий розділ містить результати розробленої системи та її аналіз. Загальний обсяг роботи – 85 сторінок. Кваліфікаційна робота містить 1 додаток, 65 рисунків, 7 таблиць і 45 джерел посилання.

Ключові слова: система, дрони, ідентифікація, розпізнавання, набір даних, штучний інтелект.

ABSTRACT

to the qualification work by the student of the group 601m of Petro Mohyla Black Sea National University

Serhii Dankovych

“SYSTEM OF IDENTIFICATION AND RECOGNITION OF DRONES USING ARTIFICIAL INTELLIGENCE METHODS”

The relevance is due to the growing demand for systems for identifying and recognizing drones due to their widespread use in various industries.

The object of the study is the process of identifying and recognizing drones using artificial intelligence methods.

The subject of the study is artificial intelligence methods for creating a drone identification and recognition system.

The purpose of the work is to increase the accuracy of identification and recognition and identification of drones using modern artificial intelligence methods.

As a result of the work, models for detecting objects were studied, trained on the created set of synthetic data, the results were tested and analyzed on real images, and a system was developed in which the trained models were integrated.

The work consists of four sections. The first section considers the current state of the problem of identifying and recognizing drones using artificial intelligence methods. The second section is devoted to the study of methods and technologies on which the drone identification and recognition system will be based. The third section presents the design of this system. The fourth section contains the results of the developed system and their analysis.

The overall scope of the work is 85 pages. Thesis contains 1 application, 65 figures, 7 tables and 45 references in it.

Key words: system, drones, identification, recognition, dataset, artificial intelligence.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ШІ – Штучний інтелект

CNN – Convolutional Neural Network

YOLO – You Only Look Once

CV – Computer vision

ORT – ONNX Runtime

WASM – WebAssembly

PWA – Progressive Web App

ВСТУП

За останні роки дрони набули широкого використання в різних галузях завдяки своїм багатофункціональним можливостям. Зростаюче використання дронів піднімає питання безпеки та вимагає розробки ефективних методів розпізнавання та ідентифікації для визначення потенційних загроз.

Одним з методів штучного інтелекту, що дозволяє отримувати інформацію з візуальних даних є комп'ютерний зір (CV). Комп'ютерний зір застосовується для виявлення об'єктів за допомогою попередньо навчених моделей на даних з розміченими зображеннями або використовуючи глибокі нейроні мережі, що дозволяють виявити специфічні особливостей з таких зображень [1].

Дрони можуть суттєво відрізнятися за своїми функціями та зовнішнім виглядом. Вони можуть бути представлені в різних категоріях, таких як багатороторні безпілотні літальні апарати або дрони з нерухомим крилом.

Складність задачі полягає в тому, що дрони – це рухомі об'єкти в повітряному просторі, які можуть з'являтися в різноманітних умовах, відсутність якісних наборів даних у вільному доступі та обмеженість можливостей створення власних ускладнює розробку надійних систем ідентифікації та розпізнавання дронів.

Система повинна розпізнавати та ідентифікувати дрони у режимі реального часу використовуючи методи штучного інтелекту (ШІ). Для розробки такої системи важливо зосередитися на варіативності вхідних даних, які будуть використовуватися для навчання, тестування та оцінки моделей, а також на виборі методів штучного інтелекту для підвищення точності для задачі ідентифікації та розпізнавання дронів.

Завдяки розробленої системи можна підвищити рівень безпеки у цивільних та у військових цілях, що робить систему актуальною в сучасному світі.

1 АНАЛІЗ СУЧАСНОГО СТАНУ ЗАДАЧІ СТВОРЕННЯ СИСТЕМИ ІДЕНТИФІКАЦІЇ ТА РОЗПІЗНАВАННЯ ДРОНІВ

1.1 Опис предметної області

Розвиток дронів зумовив необхідність створення надійних та ефективних систем для ідентифікації та розпізнавання. Дрони – це рухомий апарат, який працює без наявності пілоту на борту [2]. Дані засоби можуть мати дистанційне керування або бути повністю автономними апаратами. Вони здатні аналізувати зовнішнє оточення за допомогою сенсорів та комп’ютерного зору, здійснювати навігацію та приймати рішення самостійно.

Методи боротьби з дронами включають спектр рішень, що дозволяє розпізнати та нейтралізувати безпілотні літальні апарати.

Обладнання моніторингу за дронами може бути пасивним або активним та може виконувати функції:

- виявлення;
- класифікації або ідентифікації;
- розташування та відстеження;
- оповіщення.

Системи виявлення дронів можуть використовувати радар, радіочастотні датчики, акустичні датчики та оптичні системи для виявлення дронів у повітряному просторі [3].

Оптичні системи можуть включати камери та тепловізори, що забезпечують візуальне підтвердження та використовуються для виявлення, переслідування та ідентифікації дронів на основі їх видових характеристик. Даний вид системи можна побудувати використовуючи дані з камер за допомогою методів штучного інтелекту.

Дрони розрізняються за типом конструкції. Дрони з нерухомим крилом нагадують традиційні літальні апарати, що здатні здійснювати тривалий політ та мають високу швидкість. Дрони з ротаційним крилом мають можливість

вертикального зльоту та посадки. Також існують гібридні варіанти, що поєднують у собі характеристики різних типів дронів.

Дрони використовуються у різних сферах. Цивільні дрони можуть бути використані для професійної фотографії, картографування, моніторингу сільськогосподарських угідь та доставлення товарів. Промислові дрони застосовуються в будівництві, енергетиці, гірничодобувній промисловості та для інспекцій інфраструктури. Військові дрони використовуються для розвідки, спостереження та ураження цілі вбудованою бойовою частиною.

Система для ідентифікації та розпізнавання дронів може зменшити ризик, який становлять дрони, захищаючи критичну інфраструктуру, громадські заходи та чутливі зони від потенційних загроз, пов’язаних з нанесення пошкоджень дронами.

1.2 Аналіз наявних досліджень та публікацій

У публікації [4] автори пропонують систему комп’ютерного зору в режимі реального часу, здатну виявляти дрони, ідентифікувати їх відносне розташування та класифікувати їх за певними категоріями. Система використовує архітектуру згорткової нейронної мережі типу YOLO, щоб класифікувати дрони на три категорії (військові, спостереження та дрони доставлення).

Пропонується у публікації [5] спільна мережа класифікації на основі використання радарів та камер. Радарна мережа використовує часові особливості із радіолокаційної доріжки, а мережа камер виділяє характеристики із зображення. Радарна класифікація використовує комбінацію фільтрів IMM та RNN. Для розпізнавання використовується архітектура YOLO. Комбінована класифікаційна мережа оцінюється на польовому наборі даних.

Автори публікації [6] пропонують метод виявлення та класифікації дронів з використанням синтезу датчиків. Набори даних для виявлення та класифікації дронів збиралися шляхом вимірювань фактичних дронів за допомогою оптичної камери, радара та мікрофона. Класифікація дронів здійснюється за допомогою моделей CNN з окремо навчених оптичними зображеннями, радіолокаційними

картами та аудіоспектрограмами. Щоб підвищити точність спостереження дронів вихідні значення об'єднуються за допомогою логістичної регресії.

У публікації [7] автори запропонували багатокадрову модель ідентифікації. Даних підхід складає кадри поверх інших кадрів з ширококутної фотокамери. Метод проводить початкову ідентифікацію на площині основного зображення та ідентифікацію на площині розширеного зображення одночасно, що зменшує обчислювальне навантаження. Система використовує методи глибокого навчання для точної класифікації та відстеження дронів у режимі реального часу.

Підхід для створення спеціально направленого набору даних було представлено у публікації [8], що включає набори як для навчання та тестування. Набір даних складається із зображень дронів та птахів у різних середовищах. Набір даних було створено за допомогою вебзастосунку Roboflow, використовуючи сегментацію для надання моделі YOLO для більш точної інформації для навчання.

У публікації [9] авторами розглядалося дослідження, пов'язана з виявленням та класифікацією дронів на основі глибоких нейронних мереж. Дослідження полягало в аналізі порівняння впливу використання спектрів величини та фази як вхідних даних для класифікатора на основі аудіо. Результати публікації вказують на кращі передбачення при використанні спектра величини.

У публікації [10] автори розглядають методи класифікації. Було використано метод згорткової нейронної мережі, метод опорних векторів (SVM) та метод найближчого сусіда для задачі розпізнавання дронів з використанням камери з об'єктивом типу риб'яче око. Результати дослідження показують, що CNN має найвищу загальну точність.

Запропоновано модель виявлення об'єктів на основі архітектури YOLO для виявлення та класифікації дронів у публікації [11]. Було використано модель YOLOV5 та PANet (Path Aggregation Network) з метою покращення виділення ознак та підвищення точності виявлення об'єктів у різних масштабах.

Авторами у публікації [12] пропонується метод на основі глибокого навчання для ефективного виявлення та розпізнавання двох типів дронів та птахів.

Запропонований метод здатний розпізнавати та розрізняти два види дронів, відрізняти їх від птахів. Набір даних складається з 10 000 зображень.

Публікація [13] представляє огляд на задачу виявлення та відстеження дронів. Проаналізовано та порівняно відстеження дронів на основі лінійних фільтрів Калмана (LKF) у порівнянні з відстеженням за допомогою методів нелінійної поліноміальної регресії (NPR).

Запропоновано використовувати двоетапний підхід на основі сегментації, використовуючи просторово-часові сигнали у публікації [14]. На першому етапі, фіксується детальна контекстна інформація на картах згорткових функцій за допомогою пірамідного об’єднання. На другому етапі досліджуються нові ймовірні місця розташування дрона. Для виявлення нових місць розташування дронів використовуються межі руху.

Представлено авторами у публікації [15] розпізнавання декілька типів дронів (багатороторних, з нерухомим крилом, гелікоптерів, вертикальних літальних апаратів) використовуючи модель на основі YOLO та навчання за допомогою набору даних з 26 000 зображень. В публікації досліджували зміну кількості згорткових шарів для більш точної та детальної семантичної характеристики.

Автори у публікації [16] представили CNN, використовуючи дерево рішень та структуру ансамблю для розпізнавання повної характеристики дронів у польоті. Система визначає тип дрона, його орієнтацію, виконує сегментацію для класифікації різних частин дрона (двигунів, корпусу та камери).

Публікація [17] досліджує виявлення дронів. Авторами представлено структуру аналітики даних у реальному часі з використанням глибокого навчання. Система складається з рекурентних нейронних мереж (RNN). Дослідження включає збір інформації з мережі та її розбиття за допомогою перевірки інформації для виявлення невідповідностей.

У публікації [18] пропонується модель виявлення дронів з використанням низької якості знімків. Використано Bi-PAN-FPN, що поєднує в собі елементи

двонаправленої пірамідної мережі (Bi-FPN) та мережі агрегування шляхів (PAN) для досягнення більш точної моделі на основі архітектури YOLO.

Сучасні наукові дослідження у сфері виявлення та класифікації дронів активно використовують підходи глибокого навчання, зокрема згорткові нейронні мережі (CNN) та моделі YOLO. Методи забезпечують високу точність та продуктивність в режимі реального часу.

На основі аналізу публікацій основні тенденції розвитку технологій і методів ідентифікації та розпізнавання дронів включають:

- інтеграцію сенсорних даних для підвищення точності;
- удосконалення моделей YOLO та CNN з урахуванням специфіки задачі;
- створення спеціалізованих, варіативних наборів даних;
- ефективну обробку даних низької якості;
- подолання проблем, пов’язаних із недостатньою варіативністю даних;
- відстеження дронів у режимі реального часу.

Деякі з цих напрямів є особливо перспективними. Планується розглянути та впровадити використання моделей YOLO та CNN, створення варіативних наборів даних та відстеження дронів у режимі реального часу.

1.3 Огляд наявних технологій для розв’язання поставленої задачі

Комп’ютерний зір став ключовим інструментом для ідентифікації та розпізнавання дронів. Комп’ютерний зір передбачає вилучення, аналіз та розуміння інформації із зображень та відео, що дозволяє інтерпретувати та приймати рішення на основі візуальних даних [19]. Системи комп’ютерного зору можуть вчитися на великих наборах даних, ідентифікувати закономірності та з часом підвищувати свою точність.

Розглянемо різні типи моделей для виявлення об’єктів та наявні архітектури для задачі ідентифікації та розпізнавання дронів.

Одноетапне виявлення об’єктів використовує один прохід вхідного зображення для прогнозування наявності та розташування об’єктів на зображенні.

Воно обробляє все зображення за один прохід, що робить їх обчислювально ефективними. Одноразове виявлення об’єктів зазвичай менш точне, ніж інші методи та менш ефективне для виявлення невеликих об’єктів. Такі алгоритми можна використовувати для виявлення об’єктів у реальному часі в середовищах з обмеженими ресурсами [20].

Двоетапне виявлення об’єктів використовує два проходи вхідного зображення для прогнозування наявності та розташування об’єктів. Перший прохід використовується для створення набору пропозицій або потенційних місць розташування об’єктів, а другий прохід використовується для уточнення цих пропозицій та створення остаточних прогнозів. Даний підхід більш точний, ніж одноразове виявлення об’єкта, але потребує складніших обчислень.

YOLO – це модель виявлення об’єктів у режимі реального часу (рис. 1.1), яка розглядає виявлення об’єктів як єдину задачу регресії, починаючи від пікселів зображення до координат обмежувальної рамки та ймовірностей класу [21]. Модель YOLO ділить вхідне зображення на сітку та передбачає обмежувальні рамки та ймовірності для кожної комірки сітки. Модель відома своєю швидкістю та ефективністю, що робить її придатним для застосунків, які потребують виявлення в реальному часі. Ключовою перевагою YOLO є здатність швидко обробляти зображення без значної втрати для точності. Модель може мати проблеми з виявленням невеликих об’єктів або об’єктів, що перекриваються, порівняно з іншими моделями.

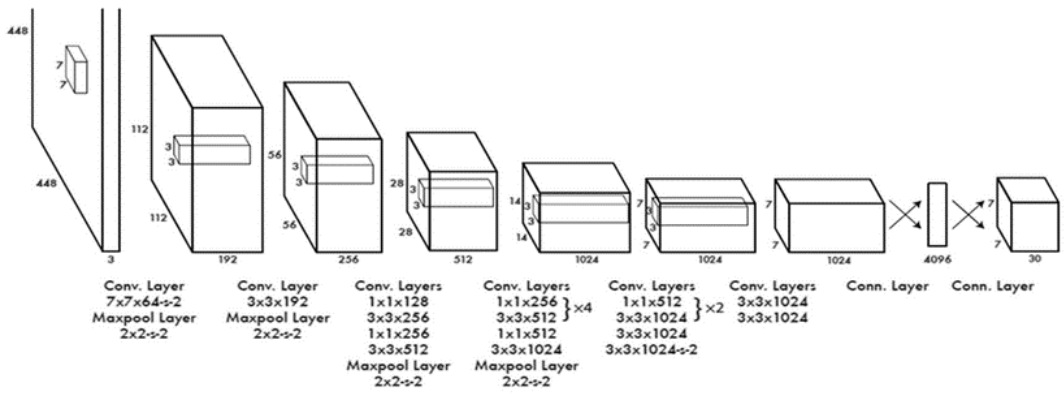

Рисунок 1.1 – Архітектура YOLO

Faster R-CNN – це двоетапна модель виявлення об’єктів, яка спочатку генерує пропозиції регіону (RP), а потім класифікує ці пропозиції [22]. Перший етап, відомий як мережа регіональних пропозицій (RPN), що пропонує регіони, які можуть містити об’єкти. Потім дані пропозиції передаються на другий етап, де згорточна нейронна мережа класифікує пропозиції та уточнює їх обмежувальні рамки (рис. 1.2). Faster R-CNN є високоточним та ефективним для виявлення об’єктів у складних сценах. Модель є повільнішою ніж одноетапні детектори через її двоетапний процес. Широко використовується в застосунках, де точність має вирішальне значення.

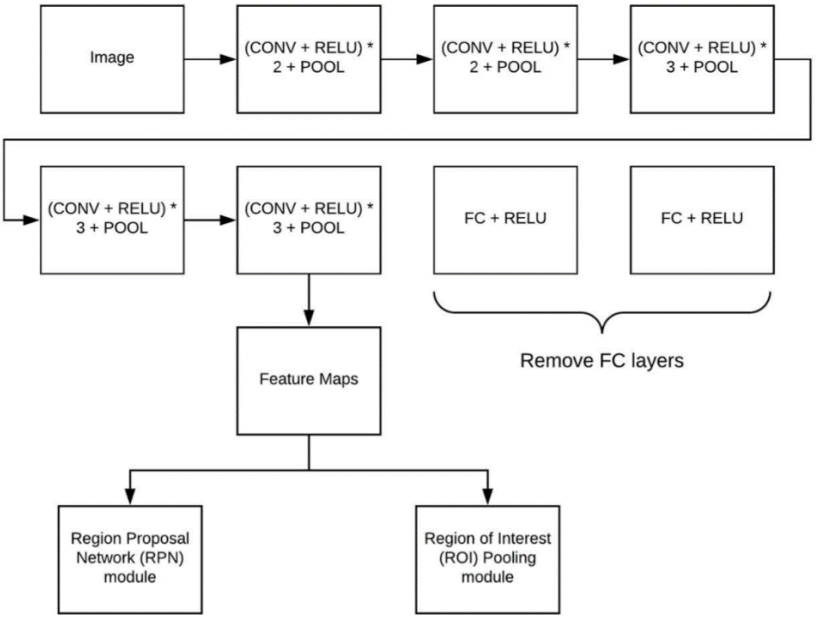

Рисунок 1.2 – Архітектура Faster R-CNN

RetinaNet – це одноетапний детектор об’єктів, який поєднує швидкість одноетапних моделей та точність двоетапних детекторів (рис. 1.3). Її ключовою інновацією є функція Focal Loss, яка розв’язувати проблему дисбалансу класів шляхом зменшення ваг втрат [23]. RetinaNet забезпечує баланс між швидкістю та точністю, що дає змогу моделі бути придатною для застосунків у режимі реального часу. Модель є ефективною у задачах, пов’язаних із щільними та малими об’єктами.

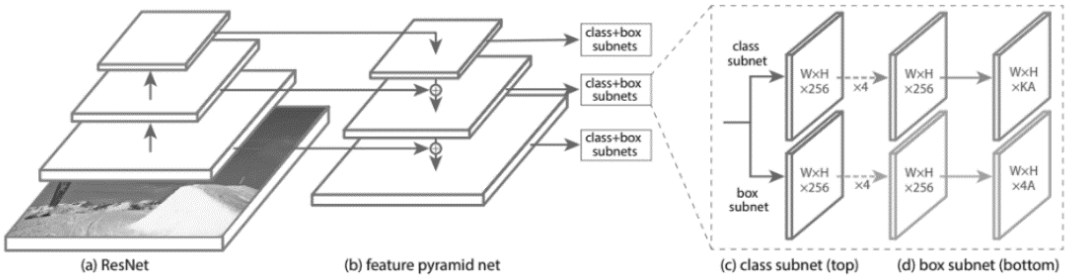

Рисунок 1.3 – Архітектура RetinaNet

SSD – це одноетапний детектор об’єктів, який виконує локалізацію та класифікацію об’єктів за один прямий прохід мережі (рис. 1.4). Для виявлення об’єктів різного розміру він використовує низку полів із різними пропорціями та масштабами в кожному місці карти об’єктів [24]. SSD об’єднує прогнози з кількох карт функцій різної роздільної здатності для ефективної обробки об’єктів різного розміру. Модель відома своєю швидкістю та точністю, що робить її придатною для використання в реальному часі. SSD може не мати високої точності для невеликих об’єктів.

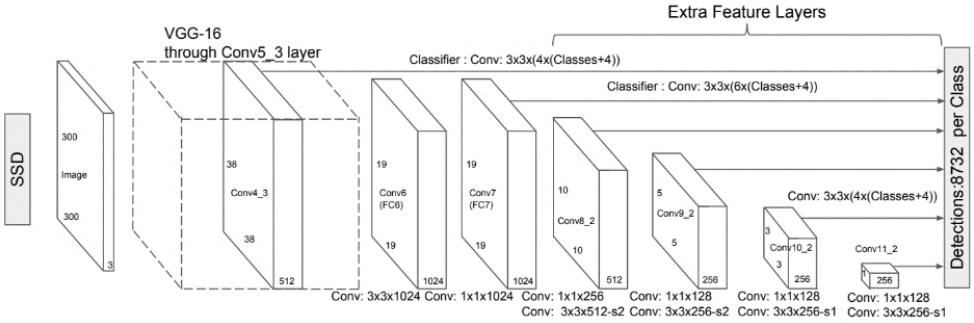

Рисунок 1.4 – Архітектура SSD

1.4 Аналіз наявних аналогів

Dedrone є компанією у сфері виявлення дронів та використання протидронових технологій. Dedrone використовує передові алгоритми комп’ютерного зору у своєму програмному забезпеченні DedroneTracker.AI. Система забезпечує виявлення, ідентифікацію та відстеження дронів у режимі

реального часу (рис. 1.5). Однією з сильних сторін підходу Dedrone є можливість інтеграції PTZ (панорамно-нахильних) камер, що підвищує ефективність у складних умовах безпеки. Серед недоліків DedroneTracker.AI можна виділити його спрямованість на певні типи дронів, наприклад, FPV (з видом від першої особи). Дрони типу FPV становлять значну загрозу, але ефективність може зменшуватися проти інших класів дронів, що мають інший тип конструкції. Дане обмеження може вплинути на універсальність системи, без можливості використання в різноманітних сценаріях.

Рисунок 1.5 – Система DedroneTracker.AI

OSL (Optical Systems Lab) компанія для виявлення та класифікації дронів, що має інноваційні рішення INSIGHT та SKYSIGHT (рис. 1.6). Система INSIGHT являти собою модуль класифікації та аналізу для виявлення об’єктів шляхом отримання даних з наявних оптичних сенсорів. Система SKYSIGHT, система з двома камерами, поєднує ширококутну камеру для моніторингу неба з інтелектуальною PTZ камерою для детального відстеження та класифікації, що інтегрується для наявних екосистем безпеки. Системи орієнтовані на використання штучного інтелекту та оптичних сенсорів для розв’язання проблем безпеки, пов’язаних із дронами. Системи мають суттєві обмеження. INSIGHT залежить від

наявності високоякісних або спеціалізованих оптичних сенсорів, що може обмежувати його ефективність. SKYSIGHT потребує значних затрат в обладнання, зокрема кількох камер та потужних обчислювальних можливостей, що може зробити його доступним лише для великих організацій. Недоліки підкреслюють потребу в більшій адаптивності та зменшенні залежності від обладнання.

Рисунок 1.6 – Системи INSIGHT та SKYSIGHT

ELYSION – це система, розроблена для управління та інтеграції різних оборонних та спостережних пристроїв, яка використовує передові алгоритми для обробки даних у реальному часі та аналізу загроз (рис. 1.7). Система дозволяє безперешкодно інтегрувати пристрої, ефективно поєднуючи дані з різних сенсорів, таких як радари, камери, акустичні сенсори та дрони. Система надає візуалізацію операцій у реальному часі. Система має інтуїтивно зрозумілий інтерфейс користувача, надаючи карту для відображення оточення та управлінням. Якщо розглядати ELYSION для цивільного застосування для ідентифікації та розпізнавання дронів, то є суттєві недоліки. Залежність від спеціалізованого обладнання та інтеграція з пристроями військового класу можуть призвести до високих витрат та обмеженої доступності.

Рисунок 1.7 – Система ELYSION

Проведено порівняння розглянутих альтернатив (табл. 1.1). Система DedroneTracker.AI демонструє високу ефективність у задачах реального часу, хоча її можливості щодо виявлення різних типів дронів залишаються обмеженими. Системи INSIGHT та SKYSIGHT (OSL) краще підходять для організацій, які мають доступ до високопродуктивного обладнання для моніторингу. Їх основними обмеженнями є здатність розпізнавати лише багатороторні дрони, певні труднощі з ідентифікацією інших типів. Система ELYSION вирізняється широким функціоналом, особливо в оборонних цілях та в завданнях відстеження, але її висока вартість та залежність від спеціалізованих пристроїв значно звужують можливості застосування.

Ідеальна альтернатива повинна передбачати використання широкого спектра дронів для ідентифікації та розпізнавання об’єктів за допомогою технологій комп’ютерного зору. Важливо, щоб система була легко інтегрованою та не залежала виключно від оптичного обладнання. Ключовою перевагою має стати здатність працювати в реальному часі, забезпечуючи оперативність та ефективність.

Таблиця 1.1 – Порівняння розглянутих альтернатив

Критерії	**DedroneTracker.AI**	**OSL рішення**	**ELYSION**
Призначення	Виявлення, ідентифікація та відстеження дронів у реальному часі	Виявлення, моніторинг дронів за допомогою оптичних сенсорів та камер	Інтеграція оборонних пристроїв для обробки та аналізу загроз у реальному часі
Технології	Комп'ютерний зір, PTZ-камери	Оптичні сенсори, PTZ-камери	Радари, камери, акустичні сенсори, інтелектуальні алгоритми
Основні переваги	Ефективність у реальному часі, інтеграція з PTZ-камерами	Комбінування ширококутних, інтеграція з PTZ-камерами або іншими камерами	Інтеграція різних сенсорів, карта оточення для візуалізації
Основні недоліки	Виявлення комерційних типів дронів	Залежність від спеціалізованих сенсорів, висока вартість обладнання	Висока вартість обладнання, обмежена доступність для цивільного застосування
Галузь	Безпека у реальному часі	Інтелектуальні системи безпеки для великих організацій	Військові та оборонні операції, розширене спостереження

1.5 Постановка задачі

Актуальність зумовлена зростаючим попитом на системи для ідентифікації та розпізнавання дронів у зв'язку з їх широким використанням у різних галузях.

Об'єкт дослідження – процес ідентифікації та розпізнавання дронів методами штучного інтелекту.

Предметом дослідження є методи штучного інтелекту для створення системи ідентифікації та розпізнавання дронів.

Мета роботи є підвищення точності ідентифікації та розпізнавання та ідентифікації дронів шляхом використання сучасних методів штучного інтелекту.

Система для ідентифікації та розпізнавання дронів має відповідати кільком важливим вимогам, щоб забезпечити її ефективність та універсальність. По-перше,

вона повинна демонструвати високу точність навіть у складних умовах, таких як низька видимість або зміни освітлення. По-друге, система має працювати в реальному часі, що дозволить швидко й точно виявляти дрони, забезпечуючи оперативність.

Для досягнення цих цілей необхідно використовувати різноманітні сценарії у створенні наборів даних, використовуючи сучасні методи штучного інтелекту. Також важливо забезпечити легкість інтеграції різних моделей та незалежність системи від оптичного обладнання.

Завдання, які необхідно виконати для досягнення поставленої мети:

– проаналізувати предметну область та визначити ключові вимоги до системи ідентифікації та розпізнавання дронів;

– оцінити та вибрати відповідні методи штучного інтелекту;

– створити набір даних, використовуючи автоматизовані підходи до створення синтетичних даних для тренування та оцінки моделей;

– навчити моделі, які здатні розпізнавати дрони з високою точністю в різних умовах та середовищах;

– провести тестування та оцінку моделей, проаналізувати результати, визначивши ключові метрики;

– розробити систему, яка використовує розроблені моделі ідентифікації та розпізнавання дронів в режимі реального часу.

Висновки до розділу 1

Ідентифікація та розпізнавання дронів є ключовими елементами для забезпечення безпеки та контролю повітряного простору, оскільки зростання кількості дронів підвищує ризик потенційних загроз. Сучасні системи ідентифікації та розпізнавання дронів використовують різноманітні технології, включаючи радіолокацію, акустичні датчики, камери та тепловізори. Однак комп’ютерний зір, виявляється одним із найбільш ефективних підходів для виявлення об’єктів.

Аналіз сучасних наукових досліджень показує, що більшість систем покладаються на архітектури, як згорткові нейронні мережі (CNN) та моделі виявлення об’єктів YOLO. Розглянуті моделі демонструють високу ефективність та здатні виконувати виявлення дронів у режимі реального часу.

Аналіз наявних аналогів виявив низку недоліків, які планується врахувати. Основні проблеми стосуються обмежень у виявленні об’єктів різного типу та необхідності використання спеціалізованого обладнання, що ускладнює інтеграцію таких систем, збільшує витрати.

Виходячи з цього, перспективним напрямком є навчання моделей YOLO та Faster R-CNN на варіативному наборі даних для розширення можливостей ідентифікації та розпізнавання більшої кількості класів, що дозволить покращити точність та надійність систем у різноманітних сценаріях їх застосування.

2 МЕТОДИ ТА ТЕХНОЛОГІЇ ДЛЯ РОЗВ'ЯЗАННЯ ЗАДАЧІ ІДЕНТИФІКАЦІЇ ТА РОЗПІЗНАВАННЯ ДРОНІВ

2.1 Використання синтетичних даних у комп'ютерному зорі

Моделі комп'ютерного зору створені для виконання завдань розпізнавання чи класифікації об'єктів. Для того, щоб моделі могли ефективно виконувати завдання, вони спочатку повинні бути навчені. Однією з основних проблем у цьому процесі є дефіцит якісних навчальних даних. Для досягнення високої точності моделі, необхідно, щоб дані включали різноманітні зразки об'єктів, відображаючи їхню варіативність [24].

У зв'язку з цим було прийнято рішення про створення синтетичних даних для моделей задачею яких є ідентифікація та розпізнавання дронів. Створення синтетичних зображень для навчання моделі зменшує потребу в масштабному зборі реальних даних та спрощують процес розмітки зображень. Реальні зображення часто мають текстові водяні знаки та логотипи, що може заважати навчанню моделі, особливо в задачах виявлення об'єктів. Якщо логотип нагадує об'єкт, наприклад дрон, модель може помилково визначити його як справжній об'єкт, що призведе до помилкових спрацьовувань. Під час анотації фокус зазвичай зосереджується на позначенні реальних об'єктів, але наявність логотипів або водяних знаків може ускладнити розрізнення моделі між об'єктом та даними артефактами. В такій ситуації треба редагувати вхідні зображення, що збільшить складність та час обробки зображень.

Одним із підходів до генерації синтетичних даних є метод 3D-візуалізації, що використовує 3D-моделі об'єктів для створення реалістичних зображень або відео. Для генерації синтетичних даних необхідно мати колекцію 3D-моделей та сценаріїв [25]. Продемонстровано 3D-моделі об'єктів (рис. 2.1), наведено приклади сценаріїв, що можуть використовуватися для створення синтетичного набору даних (рис. 2.2).

Рисунок 2.1 – 3D-моделі об’єктів

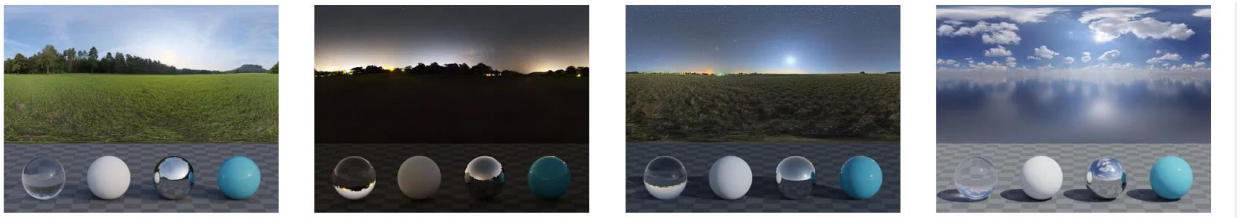

Рисунок 2.2 – Сценарії

Зображено процес генерації синтетичних даних методом 3D-візуалізації. 3D-моделі являються об’єктами для подальшого розпізнавання, що формують основу для створення різноманітних наборів даних (рис. 2.3). Умовами є фактори навколишнього середовища або чинники, які можуть впливати на відтворення 3D-моделей. Це такі аспекти, як освітлення, погода, розташування камери, розмиття під час руху, текстури та інші параметри, які змінюють зовнішній вигляд моделей.

Програмне забезпечення 3D-візуалізації діє як центральний компонент у цьому процесі. Воно використовує 3D-моделі створення реалістичних зображень. Кінцевий результат складається зі згенерованих синтетичних зображень або відео, які можна використовувати для навчання моделей для задачі виявлення об’єктів.

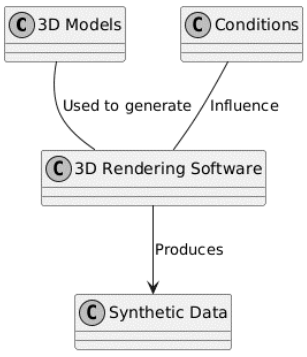

Рисунок 2.3 – Створення синтетичних наборів даних

У якості програмного забезпечення для 3D-візуалізації було використано Blender, що являє собою потужний пакет для створення 3D-графіки з відкритим вихідним кодом. Blender дозволяє моделювати, візуалізувати 3D-об’єкти, що робить його придатним для створення синтаксичних даних.

Blender має два вбудовані рушії для візуалізацій. Розширені можливості освітлення, тіней та текстур підвищують якість візуалізації об’єктів. Використання Python скриптів у Blender надає можливість в автоматизації створення синтаксичного набору даних. За допомогою скрипту можна створювати зображення декілька видів моделей дронів у різних умовах, з різних кутів та при різному освітленні.

Синтетичні дані дозволяють створювати контрольовані набори даних, що сприяє більш збалансованим наборам даних для навчання моделей. Синтетичні набори даних є критично важливими для навчання моделей комп’ютерного зору, оскільки вони надають різноманітні, розмічені дані, які можуть покращити здатність моделі розпізнавати дрони в реальних умовах.

2.2 Архітектура YOLO

Було використано архітектуру YOLO (You Only Look Once), що є однією з найпопулярніших моделей для виявлення об’єктів на зображеннях та відео, оскільки дозволяє виконувати цю задачу в реальному часі [27]. Її головна особливість полягає в тому, що процес виявлення об’єктів розглядається як задача регресії, модель одночасно передбачає класи об’єктів та координати їхніх обмежувальних рамок для всього зображення, що значно підвищує швидкість роботи порівняно з іншими методами.

Архітектура YOLO побудована з трьох ключових компонентів (рис. 2.4). Основа (backbone) відповідає за витягування основних ознак з вхідного зображення за допомогою згорткових шарів, що дозволяє моделі будувати уявлення про різні аспекти зображення. Шия (neck) поєднує функції, отримані з різних рівнів основи, допомагаючи моделі обробляти інформацію з різними масштабами та контекстами.

Голова (head) відповідає за остаточне передбачення класів об’єктів та їхніх координат у вигляді обмежувальних рамок.

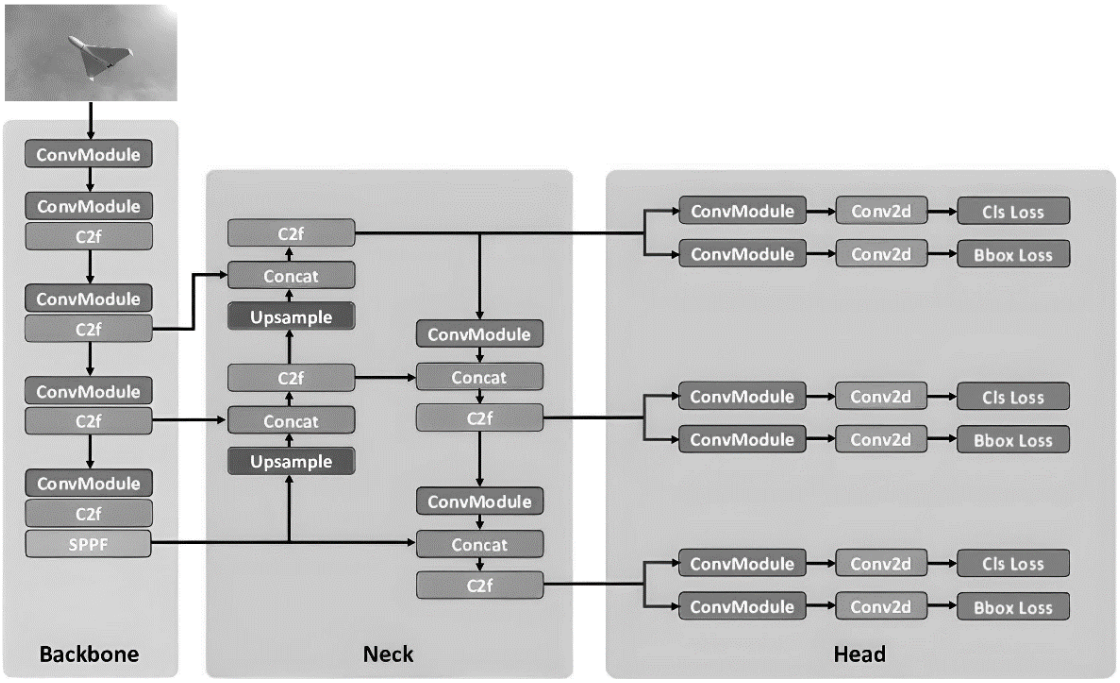

Рисунок 2.4 – Основні блоки архітектури YOLO

Однією з важливих частин архітектури є шар Conv2D, що виконує згорткову операцію для витягування ознак. У цьому шарі використовуються фільтри, що ковзають по зображенню та витягують важливу локальну інформацію. Крок фільтра визначає швидкість його переміщення по зображенню, а додавання нульових пікселів по краях зображення дозволяє зберегти його просторові розміри. Дані потім проходять через рівень BatchNorm2D, який нормалізує активації, покращуючи стабільність і швидкість навчання.

В архітектурі YOLO використовується функція активації SiLU (Swish), що сприяє плавному передаванню градієнтів через шари та допомагає уникати проблем, пов’язаних зі зникненням градієнтів [28]. Блок Bottleneck з’єднує шари за допомогою механізму швидкого підключення, що дозволяє покращити передавання градієнтів через мережу та сприяє ефективнішому навчанню моделі.

У моделі також реалізовано блок C2f, який поділяє карту функцій на дві частини. Одну спрямовує до блоку Bottleneck, а іншу до блоку Concat, що дозволяє моделі ефективно поєднувати різні ознаки та покращувати якість виявлення об’єктів.

Блок SPPF (Spatial Pyramid Pooling Fast) використовується для обробки зображень різних розмірів та захоплення масштабної інформації, що покращує здатність моделі працювати із зображеннями різної роздільної здатності.

Для зменшення розміру карти ознак застосовується шар MaxPool2D, який виконує вибір максимального значення з певної області зображення, зберігаючи важливі елементи. Даний підхід зменшує кількість обчислень, необхідних для подальшої обробки, допомагає виділити домінуючі ознаки на різних рівнях.

У YOLO блок виявлення об’єктів побудовано за принципом без прив’язок (anchor-free), модель передбачає центр об’єкта без використання попередньо заданих прив’язок [29]. В такий спосіб зменшується кількість обчислень та спрощує обробку результатів, роблячи модель швидшою та більш ефективною.

Загалом, архітектура YOLO є надзвичайно гнучкою й ефективною, що робить її ідеальним вибором для застосувань, де потрібне виявлення об’єктів у реальному часі, особливо в умовах обмежених обчислювальних ресурсів, таких як мобільні пристрої або вбудовані системи.

2.3 Архітектура Faster R-CNN

Для порівняння результатів було обрано двоетапну модель для виявлення об’єктів. Faster R-CNN (Region-based Convolutional Neural Network) – це сучасна архітектура для виявлення об’єктів, яка суттєво покращує попередні моделі, інтегруючи мережу пропозицій регіонів (RPN) для генерації кандидатних обмежувальних рамок об’єктів. Архітектура складається з двох основних компонентів (рис. 2.5). RPN – генерує пропозиції регіонів, що містять потенційні об’єкти, та мережі Faster R-CNN, яка виконує виявлення об’єктів на цих запропонованих регіонах [30].

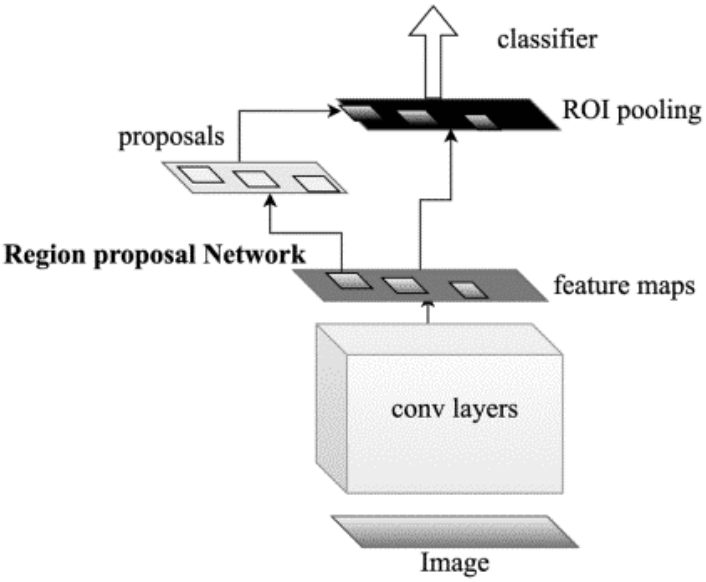

Рисунок 2.5 – Архітектура Faster R-CNN

В основі Faster R-CNN лежить основа, якою може бути попередньо навчена згорткова нейронна мережа (CNN), така як VGG16, ResNet або MobileNet. Ця мережа витягує високорозмірні карти ознак з вхідних зображень, підкреслюючи важливі шаблони, корисні для виявлення об’єктів. RPN є меншою мережею, яка ковзає по картам ознак, використовуючи підхід ковзного вікна для прогнозування пропозицій регіонів. Для кожного вікна RPN генерує кілька якорів, попередньо визначених обмежувальних рамок різних масштабів та співвідношень сторін, щоб покрити різноманітні розміри та форми об’єктів. Кожному якорю присвоюється оцінка, що вказує на ймовірність того, що він містить об’єкт, а координати якоря уточнюються через регресію обмежувальної рамки.

Після генерації пропозицій регіонів вони проходять через шар ROI (Region of Interest) (рис. 2.6). Даний шар перетворює змінні розміри запропонованих регіонів у фіксований розмір, що дозволяє наступним шарам обробляти їх однорідно. Потім відібрані ознаки передаються до мережі Faster R-CNN, де вони підлягають подальшій обробці через зв’язкові шари [31]. Компонент класифікує об’єкти в запропонованих регіонах та прогнозує координати обмежувальних рамок для

кожної пропозиції, використовуючи softmax для розподілу ймовірностей по класах об’єктів та регресійний шар для остаточних коригувань обмежувальних рамок.

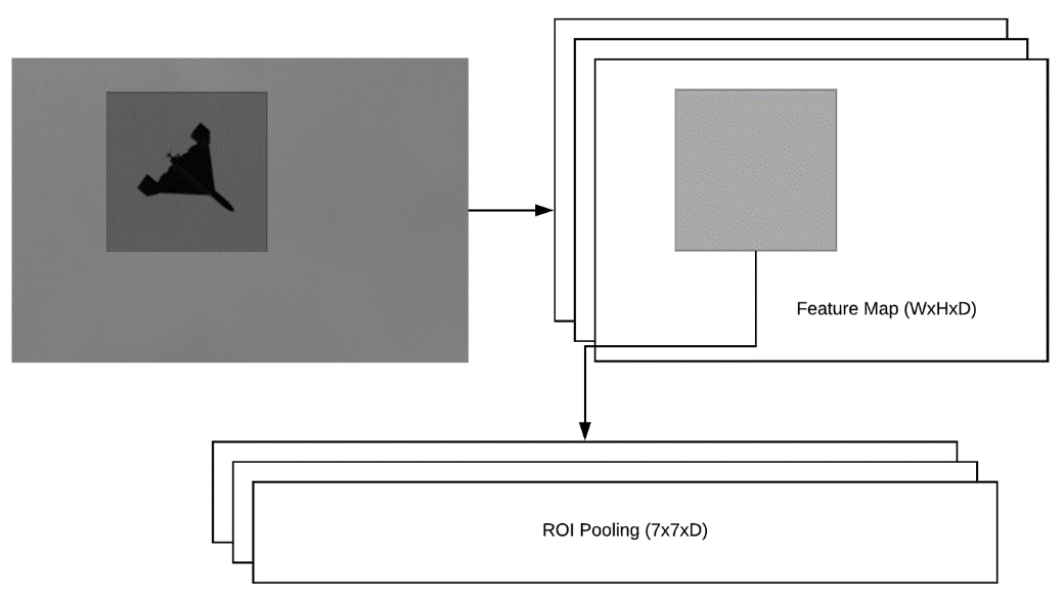

Рисунок 2.6 – ROI (Region of Interest)

Faster R-CNN використовує двоступеневий процес навчання. Спочатку RPN навчена за допомогою обмежувальних рамок істини, з функцією втрат, яка поєднує класифікацію (відмінність між об’єктом та не об’єктом) і втрати регресії обмежувальної рамки. Після навчання RPN компонент Fast R-CNN навчається за допомогою пропозицій, згенерованих RPN, знову використовуючи класифікацію та втрати регресії обмежувальної рамки [32]. Вхідне зображення обробляється через мережу для отримання карт ознак. Потім RPN генерує пропозиції регіонів з цих карт ознак, які потім обробляються в однакові розміри. Відібрані ознаки класифікують та уточнюють обмежувальні рамки, що призводить до точного виявлення об’єктів.

Faster R-CNN має переваги у швидкості та точності завдяки інтеграції RPN, що дозволяє ефективно генерувати пропозиції регіонів та навчати мережу у кінцевому результаті. Дана архітектура знайшла застосування в різних сферах, включаючи автономні транспортні засоби, системи спостереження, медичну

візуалізацію, робототехніку та аналіз відео, що свідчить про її універсальність і ефективність у завданнях виявлення об’єктів у реальному часі.

2.4 Transfer learning

Обрані моделі потребують використання трансферного навчання для задачі виявлення об’єктів. Трансферне навчання являє собою техніку в машинному навчанні, яка використовує попередньо навчені моделі для покращення результатів на новому, часто пов’язаному завданні. Даний підхід є особливо корисним у ситуаціях, коли кількість доступних даних для навчання є обмеженою. Замість того, щоб навчати модель з нуля, трансферне навчання дозволяє використовувати модель, яка вже була навчена на великому наборі даних. Попередньо навчена модель захоплює знання, включаючи низько рівневі ознаки та високо рівневі ознаки, які можуть бути перенесені для виконання нового завдання.

Представлено процес трансферного навчання для обраних моделей (рис. 2.7).

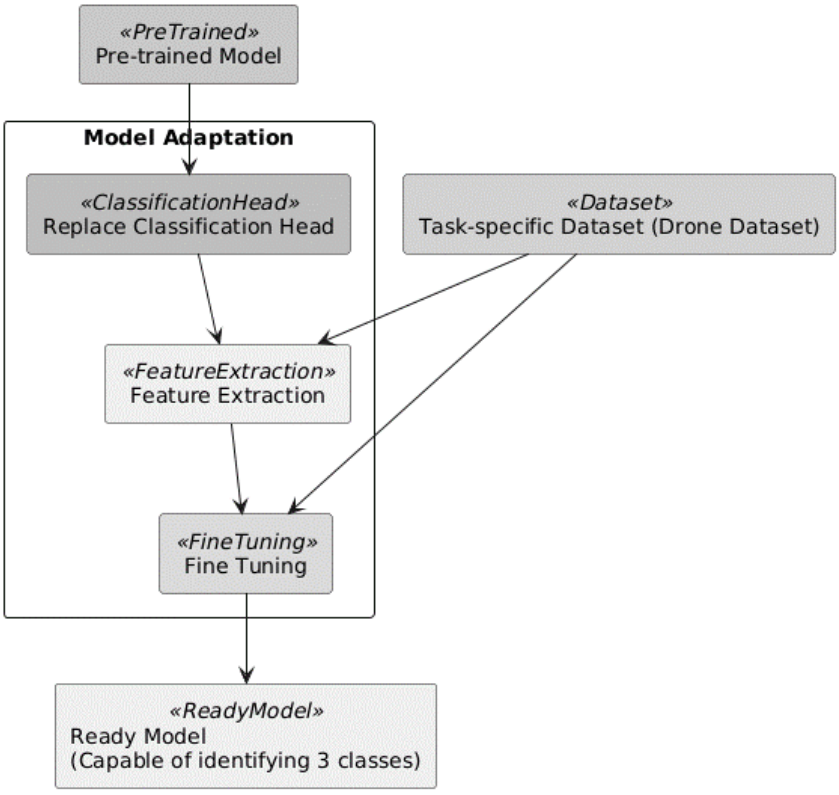

Рисунок 2.7 – Transfer learning

У трансферному навчанні зазвичай існують дві стратегії, а саме вилучення ознак або тонке налаштування. Вилучення ознак передбачає використання попередньо навченої моделі як фіксованого екстрактора ознак, де навчені моделі ознаки подаються на новий класифікатор, пристосований до конкретного завдання, що вимагає менше обчислювальних ресурсів, оскільки тренується лише останній класифікаційний шар. Тонке налаштування передбачає розморожування деяких шарів у попередньо навченої моделі та спільне їх навчання з новим завданням, що може призвести до кращих результатів, оскільки дозволяє моделі адаптувати свої навчені ознаки більш тісно до специфіки нових даних.

Реалізацію трансферного навчання можна виконати за допомогою фреймворку PyTorch, що забезпечує надійну підтримку для завантаження попередньо навчених моделей з бібліотеки torchvision, яка включає популярні архітектури, такі як ResNet, VGG та MobileNet. Модульний дизайн PyTorch полегшує модифікацію попередньо навчених моделей для конкретних завдань шляхом налаштування вихідного шару або додавання нових шарів. PyTorch дозволяє модифікувати моделі, надаючи прості методи для заморожування та розморожування шарів під час навчання.

Фреймворк дозволяє ефективно керувати швидкостями навчання для різних шарів, що є важливим для підтримання продуктивності попередньо навченої моделі під час адаптації до нового завдання. Завдяки наведеним перевагам PyTorch став вибором для трансферного навчання в задачі ідентифікації та розпізнавання дронів.

2.5 Виконання моделей штучного інтелекту в браузері

Щоб реалізувати систему ідентифікації та розпізнавання дронів, необхідно знайти ефективний спосіб виконання розроблених моделей для виявлення об’єктів. Одним із найбільш зручних та сучасних підходів є виконання моделей безпосередньо в браузері.

Next.js фреймворк розроблений для створення сучасних вебзастосунків. Він підтримує серверну візуалізацію та статичну генерацію вебзастосунків. Дані варіанти візуалізації дозволяють створювати високопродуктивні вебзастосунки, які швидко завантажуються та добре підходять для пошукової оптимізації.

Однією з визначних функцій фреймворку є система маршрутизації на основі файлів. Сторінки організовують у спеціальній директорії, що автоматично генерує необхідні маршрути. Такий підхід спрощує управління маршрутизацією та робить навігацію між сторінками безперешкодною.

Next.js також підтримує серверне та клієнтське отримання даних. Наявний функціонал для створення маршрутів API у вебзастосунку, що дозволяє повноцінно працювати з серверною та клієнтською частиною в одному проєкті.

Крім своїх основних функцій, фреймворк включає вбудовану підтримку CSS та Sass, що дозволяє стилізувати свої застосунки без необхідності в складній конфігурації. Фреймворк також автоматично оптимізує зображення, що допомагає підвищити продуктивність, особливо на повільних мережах або мобільних пристроях.

ONNX Runtime (ORT) – це мультиплатформовий інструмент для виконання моделей машинного навчання, що підтримує різні популярні фреймворки, такі як TensorFlow, PyTorch та SciKit-Learn [33]. Він забезпечує можливість виконувати моделі на різних платформах та пристроях, полегшуючи роботу з різноманітними середовищами розробки. ONNX Runtime розширив свої можливості на мобільні пристрої та веббраузер через технологію ORT Web (рис. 2.8).

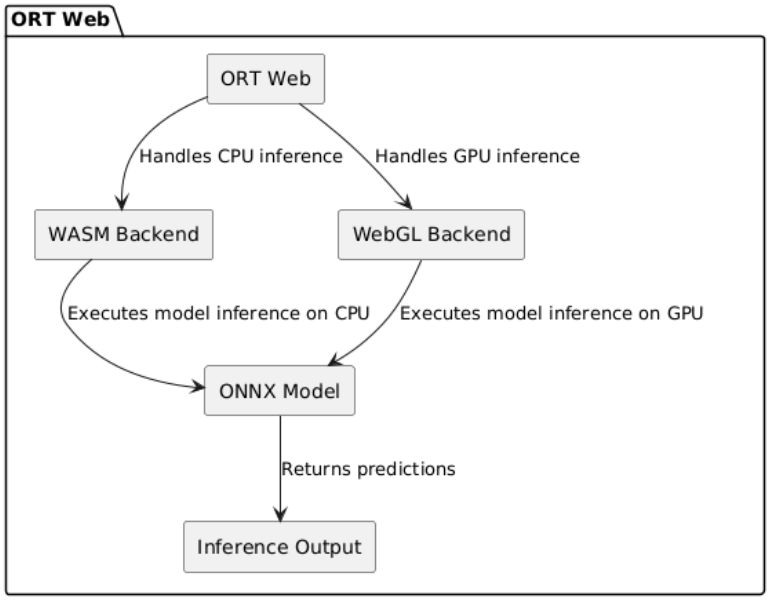

Рисунок 2.8 – ONNX Runtime в браузері

ORT Web дозволяє виконувати моделі у браузері на центральних процесорах (CPU) та на графічних процесорах (GPU), використовуючи два різних підходи. WebAssembly (WASM) для CPU та WebGL для GPU. WebAssembly дозволяє виконувати серверний код на стороні клієнта в браузері з більшою ефективністю та підтримує багато поточність, що значно підвищує швидкість обробки великих обсягів даних [34].

У ORT Web, основний двигун ONNX Runtime компілюється у WebAssembly за допомогою Emscripten, що дозволяє використовувати всі основні компоненти ONNX Runtime.

WebGL дозволяє використовувати GPU для обробки моделей у браузері та забезпечує прискорення на основних платформах. Він також включає оптимізації продуктивності, такі як режим пакування даних, кешування та оптимізація коду.

Завдяки такій комбінації технологій Next.js та ONNX Runtime, система для виявлення об’єктів може ефективно працювати у браузері, забезпечуючи швидке і плавне виконання моделей штучного інтелекту без значних витрат на ресурсах.

2.6 Метрики для оцінки ефективності моделей

Моделі YOLO та Faster R-CNN широко використовуються завдяки їхній продуктивності у виявленні об’єктів на зображеннях. Дані моделі відрізняються за архітектурою, але мають спільну мету яка полягає в точному виявлення об’єктів, мінімізуючи обчислювальні витрати.

В задачах виявлення об’єктів оцінка продуктивності моделі передбачає кількісне порівняння її прогнозів із фактичними (ground-truth) мітками. Для цього визначають такі терміни:

– істинно позитивний результат (True Positive): істинно позитивний результат виникає, коли модель правильно ідентифікує та локалізує об’єкт. Якщо фактична мітка вказує на наявність об’єкта в конкретному місці та модель передбачає той самий результат з достатньою точністю;

– істинно негативний результат (True Negative): істинно негативний результат представляє ситуацію, коли модель правильно прогнозує відсутність об’єкта. Це стосується областей зображення або всього зображення, де об’єкти відсутні та модель утримується від помилкових прогнозів;

– хибно позитивний результат (False Positive): хибно позитивний результат виникає, коли модель прогнозує наявність об’єкта там, де його немає. Модель помилково інтерпретує особливості фону як об’єкт або надмірно впевнена у неоднозначних областях;

– хибно негативний результат (False Negative): хибно негативний результат трапляється, коли модель не виявляє об’єкт взагалі. Може статися через низьку впевненість у прогнозах або труднощі з ідентифікацією менших або менш виразних об’єктів.

Для оцінки продуктивності моделей виявлення об’єктів, таких як YOLO та Faster R-CNN використовуються кілька стандартних метрик. Метрики кількісно оцінюють точність, якість та надійність прогнозів моделей. Найбільш поширені метрики:

– Precision (точність);
– Recall (повнота);
– Intersection over Union (IoU);
– Average Precision (AP);
– Mean Average Precision (mAP).

IoU є мірою перекриття між передбаченою обмежувальною рамкою та обмежувальною рамкою з розмітки (рис. 2.9). Міра обчислюється за формулою:

$$IoU = \frac{Area\ of\ Intersection}{Area\ of\ Union} \quad (2.1)$$

Значення IoU варіюється від 0 до 1. IoU рівний 1 вказує на ідеальне збіг між передбаченою і фактичною обмежувальною рамкою. IoU рівний 0 означає відсутність перекриття. Поріг зазвичай встановлюється IoU ≥ 0,5 для визначення та вважається виявлення коректним [35].

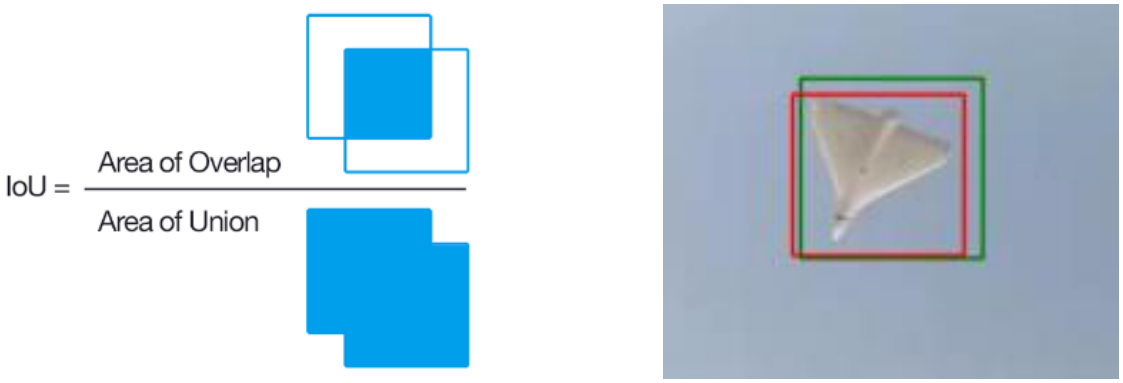

Рисунок 2.9 – Міра перекриття

Показник точності вимірює скільки з передбачених обмежувальних рамок правильно класифіковані, фокусуючись на якості виявлень. Він обчислюється як:

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \quad (2.2)$$

Високий показник точності означає, що більшість передбачень моделі правильні з меншим числом хибно позитивних виявлень.

Показник повноти вимірює здатність моделі виявити всі релевантні об’єкти, фокусуючись на кількості правильних виявлень. Він обчислюється за формулою:

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \quad (2.3)$$

Високий показник повноти свідчить про те, що модель виявляє більшість фактичних об’єктів, але може включати більше хибнопозитивних результатів.

Показник AP підсумовує криву Precision-Recall, яка відображає точність і повноту при різних порогах впевненості. Вона обчислюється як площа під кривою Precision-Recall.

AP для конкретного класу визначається за формулою:

$$AP = \int_0^1 P\,recision(Recall)\ dRecall \quad (2.4)$$

AP обчислюється для кожного класу, вищий показник AP означає, що модель краще виявляє об’єкти цього класу на різних рівнях впевненості.

mAP є найпоширенішою метрикою для оцінки моделей виявлення об’єктів. Це середнє значення AP для кожного класу в наборі даних. mAP обчислюється за формулою:

$$mAP = \frac{1}{N}\sum_{i=1}^{N} A\,P_i\ , \quad (2.5)$$

де N – кількість класів, AP_i – середня точність для класу.i

mAP надає єдине значення, яке відображає загальну продуктивність моделі для всіх класів об’єктів. Вищий показник mAP свідчить про кращу загальну продуктивність виявлення.

Для оцінки моделей, таких як YOLO або Faster R-CNN, набір даних зазвичай поділяється на тренувальні, валідаційні та тестові набори. Модель навчається на тренувальних даних, а її продуктивність оцінюється на валідаційних та тестових даних за допомогою вищезгаданих метрик.

Оцінка моделей виявлення об’єктів, таких як YOLO та Faster R-CNN, сильно залежить від співвідношення точності та повноти, а також розрахунку IoU, mAP. Дані метрики надають стандартизований спосіб оцінки продуктивності моделей, що дозволяє порівнювати різні моделі та вибирати яка найкраще відповідає їхнім цілям, це чи виявлення в реальному часі за допомогою YOLO чи висока точність виявлення з Faster R-CNN.

Висновки до розділу 2

Підсумовуючи, було обрано створення синтетичних даних за допомогою 3D-візуалізацій з використанням обраних 3D-моделей у різних середовищах для автоматизації процесу створення набору даних без необхідності в анотації зображень. Згенеровані зображення зберігаються з відповідними обмежувальними рамками та мітками класу, що спрощує подальшу обробку даних.

Детально було досліджено архітектури двох моделей YOLO та Faster R-CNN. Їх обрали для навчання та оцінки та надалі для інтеграції для розробленої системи. Досліджено методику навчання цих моделей із застосуванням трансферного навчання.

Система повинна бути реалізована у вигляді вебзастосунка, що використовуватиме технології для виконання моделей виявлення об’єктів безпосередньо в браузері.

.

3 ПРОЄКТУВАННЯ СИСТЕМИ ДЛЯ ІДЕНТИФІКАЦІЇ ТА РОЗПІЗНАВАННЯ ДРОНІВ

3.1 Проєктування архітектури системи

Для задачі ідентифікації та розпізнавання дронів було використано діаграму архітектури системи для відображення структури проєкту (рис. 3.1). Діаграма починається з інтерфейсу де компоненти представляють інтерфейс користувача. Дані компоненти дозволяють користувачам завантажувати зображення або відео та отримувати доступ до відеокамери. Інший компонент відповідає за візуалізацію обмежувальних рамок навколо виявлених об’єктів. Виконання моделей було реалізовано в браузері за допомогою WebAssembly (Wasm), що дозволяє запускати низькорівневий код у браузері.

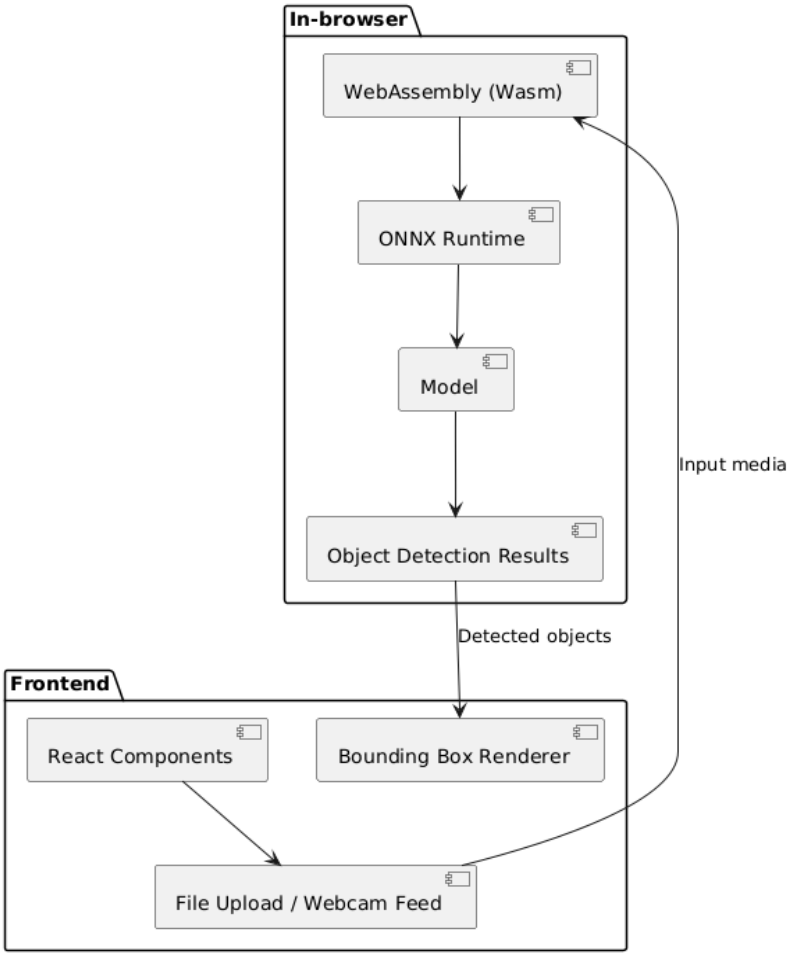

Рисунок 3.1 – Діаграма архітектури системи для ідентифікації та дронів

WebAssembly використовується для виконання ONNX Runtime, фреймворку, відповідального за запуск моделей машинного навчання, таких як моделі YOLO та Faster R-CNN у браузері [35]. Моделі приймають медіадані, надані користувачем (відео або зображення) та обробляє їх для виявлення об’єктів. Вихідні дані складаються з обмежувальних рамок навколо виявлених об’єктів разом із їхніми мітками, які потім відображаються у браузері через компоненти інтерфейсу.

Архітектура системи розроблена для обробки на стороні клієнта для виявлення об’єктів безпосередньо в браузері користувача, що усуває потребу у внутрішніх серверах.

На діаграмі прецедентів відображається взаємодія з системою (рис. 3.2). Користувач може завантажити відео, зображення або почати трансляцію використовуючи вебкамеру. Інтерфейс вебзастосунка взаємодіє з вхідними даними, що надсилаються до середовища виконання ONNX. Коли вхідні дані отримано, вони обробляються моделлю для виявлення об’єктів. Даний процес охоплює розбиття вхідних даних на кадри та застосування алгоритму для ідентифікації об’єктів та їх положення у формі обмежувальних рамок. Результати процесу виявлення об’єкта надсилаються назад до вебзастосунка. Обмежувальні рамки разом із відповідними мітками відображаються безпосередньо на відео чи зображенні в браузері, дозволяючи користувачеві візуалізувати виявлені об’єкти.

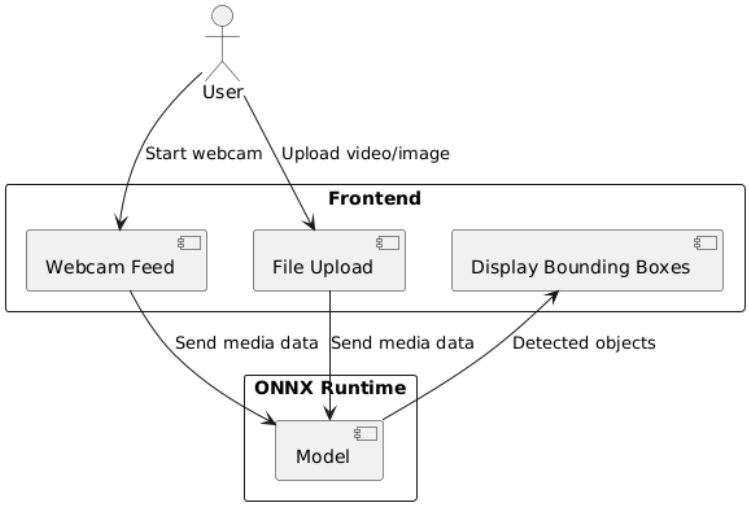

Рисунок 3.2 – Діаграма прецедентів для системи

3.2 Автоматизація створення синтетичних даних

Описано процес створення набору даних у Blender, починаючи з вибору моделі дрона (рис. 3.3) до готового набору даних. Можна використовувати кілька моделей дронів, кожна модель поєднується з іншим середовищем, яке визначає умови освітлення. Після вибору моделі дрона та середовища моделі призначається певний шлях анімації. Шлях визначає, як дрон буде рухатися в межах сцени. Після встановлення шляху відбуваються два процеси. Перший процес полягає в тому, що модель рухається заданим шляхом та зображення відтворюються кадр за кадром, фіксуючи модель у русі з ефектами розмиття, створюється файл з обмежувальними рамками та міткою для кожного кадру, позначаючи місце розташування дрона на зображенні та його клас. Відтворені зображення та мітки обмежувальної рамки зберігаються в вихідній папці, організованій середовищем. Даний процес повторюється для кожної комбінації моделей дронів та середовищ. В результаті отримується повний набір даних із з попередньо встановленими умовами освітлення та варіативним положенням дронів.

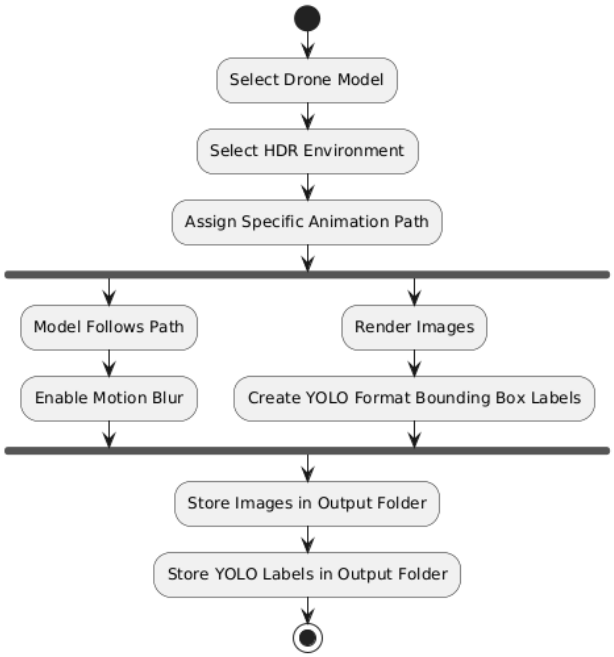

Рисунок 3.3 – Створення синтетичних даних на основі методу 3D-візуалізації

Для отримання обмежувальної рамки треба перетворити точки у 3D світовому просторі у 2D простір камери (рис. 3.4) за допомогою перетворень, що враховують положення та орієнтацію камери та проєкцію 3D простору на площину [36].

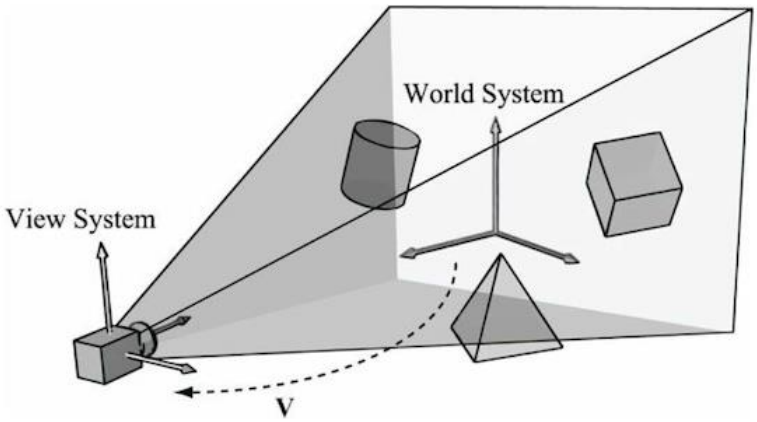

Рисунок 3.4 – Точки у 3D світовому просторі у 2D простір камери

Для перетворення світових координат точки в локальну систему координат камери здійснюється матричним перетворенням, яке комбінує трансляцію та обертання на основі положення й орієнтації камери у сцені [37]. Після цього кроку 3D точка знаходиться у локальному просторі камери, тобто координати точки відносні до камери.

$$P_{camera} = T_{camera} \times P_{world}, \tag{3.1}$$

де P_{world} – це точка у світових координатах;

T_{camera} – це матриця перетворення, яка комбінує трансляцію (положення) та обертання камери (орієнтацію);

P_{camera} – це перетворена точка у локальній системі координат камери.

Для проєкції 3D точки у просторі камери на 2D площину зображення використовується проєкційною матрицею. Проєкційна матриця застосовує поділ перспективи:

$$x' = \frac{x_{camera}}{z_{camera}} \quad \text{та} \quad y' = \frac{y_{camera}}{z_{camera}}, \tag{3.2}$$

де x_{camera}, y_{camera}, z_{camera} – це координати точки у просторі камери;

x' і y' – це спроєктовані 2D координати у просторі проєкції (нормалізовані координати).

Після проєкції координати точки ще знаходяться у просторі проєкції, але їх потрібно нормалізувати, щоб вони відповідали вікну камери. На цьому етапі область, яка визначає, що бачить камера масштабується до одиничного квадрата.

Функція приймає вектор (x, y, z). Параметри x і y – це нормалізовані 2D координати точки у просторі камери (від 0 до 1). Параметр z – це глибина точки відносно камери (позитивні значення означають, що точка знаходиться перед камерою, а негативні – що вона знаходиться позаду камери).

Для перетворення координат об’єктів у вихідних формат треба виконати ряд обчислень. Формат зберігає інформацію про об’єкти у зображенні за допомогою їхніх координат у нормалізованій формі та записуються у файл у форматі <class_index> <x_norm> <y_norm> <width_norm> <height_norm>.

Для обчислення координат центру об’єкта використовується формула:

$$x_{center} = \frac{x_{min}+x_{max}}{2} \text{ та } y_{center} = \frac{y_{min}+y_{max}}{2} \tag{3.3}$$

Для обчислення ширини та висоти об’єкта використовується формула:

$$width = x_{max} - x_{min} \text{ та } height = y_{max} - y_{min} \tag{3.4}$$

Для нормалізації координат <x_norm> <y_norm> використовуються формули:

$$x_{norm} = \frac{x_{center}}{image\_width} \text{ та } y_{norm} = \frac{y_{center}}{image\_height} \quad (3.5)$$

Для нормалізації координат <width_norm> <height_norm> використовуються формули:

$$width_{norm} = \frac{width}{image\_width} \text{ та } height_{norm} = \frac{height}{image\_height} \quad (3.6)$$

Для кожної вершини обчислено двовимірні координати в просторі за допомогою функції яка проєктує положення тривимірної вершини на перспективу камери. Виконується перевірка видимості, що вершина знаходиться в межах видимої області огляду камери. Координати x і y мають бути в діапазоні [0, 1].

Висновки до розділу 3

Розглянуто архітектуру та використання системи. Розроблені компоненти дозволяють користувачам завантажувати зображення або відео та отримувати доступ до відеокамери та візуалізувати результати у вигляді обмежувальних рамок та міток класів дронів навколо об’єкту. Архітектура системи розроблена для обробки на стороні клієнта для виявлення об’єктів безпосередньо в браузері користувача. Взаємодію з системою описано за допомогою діаграми прецедентів.

Автоматизація створення синтетичних даних у Blender дозволяє отримати набір даних для навчання моделей, враховуючи різноманітні умови освітлення та траєкторії руху дронів. Досліджено процес обчислення обмежувальної рамки за допомогою перетворень точок у 3D світовому просторі в 2D простір камери.

4. РЕАЛІЗАЦІЯ СИСТЕМИ ДЛЯ ІДЕНТИФІКАЦІЇ ТА РОЗПІЗНАВАННЯ ДРОНІВ

4.1 Створення синтетичного набору даних за допомогою Blender

Для створення набору даних для задачі ідентифікації та розпізнавання дронів було використано синтаксичні дані створені використовуючи Blender. Готовий набір даних повинен складатися з зображень та обмежувальних рамок.

Процес створення набору даних починається з завантаження зображень середовища та текстур. Кожне середовище використовується як фонове зображення. Щоб підвищити різноманітність набору даних, застосовано різну текстуру до дрона для кожного середовища (рис. 4.2). Текстури можуть включати базовий колір, нормаль, шорсткість, металік й карти зміщення, що призначаються для дрона, створюючи різноманітність у візуальному вигляді об’єкта (рис. 4.3). Для кожного середовища створюється окрема папка для зберігання відповідних зображень та обмежувальних рамок з класом.

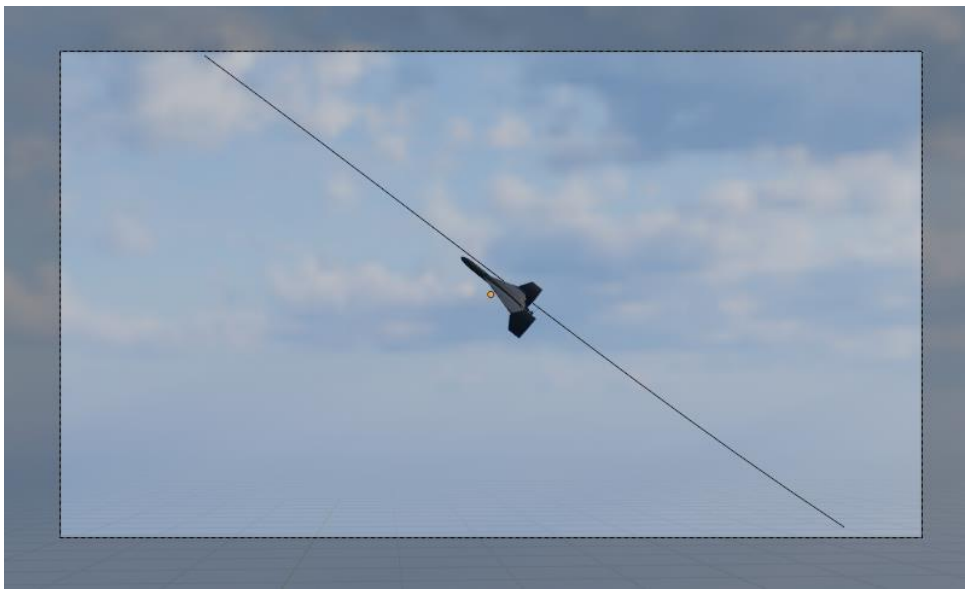

Рисунок 4.2 – Модель дрону у відповідному середовищі

Рисунок 4.3 – Модель дрону з використанням текстур

Для автоматизації руху дрона в 3D-просторі були створені криві траєкторії, за якими слідує модель дрона під час анімації (рис. 4.4). Доступні різні криві, для кожного середовища вибирається окрема траєкторія. Криві анімуються протягом заданої кількості кадрів, можуть мати різну форму та довжину та розміщення відносно камери.

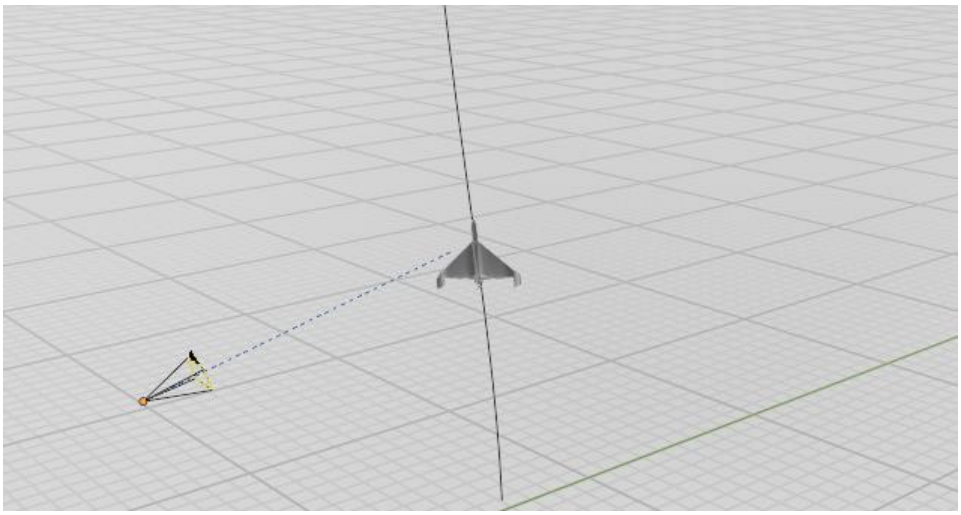

Рисунок 4.4 – Модель дрону в 3D-просторі Blender

Задачею було створення зображень схожих на зображення при їх зібранні в реальних умовах. Тому до руху дрона було застосовано ефект розмиття, що додає реалістичності (рис. 4.5). Відтворені зображення зберігаються у форматі PNG у відповідній папці, забезпечуючи високу якість запису дрона в різних умовах освітлення та середовища.

Рисунок 4.5 – Імітація реальних зображень

Для спрощення процесу анотування набору даних було використано автоматичне створення сумісних із YOLO міток обмежувальних рамок. Обмежувальна рамка обчислюється та перетворюється у формат YOLO. Для кожного відтвореного зображення створюється відповідний файл міток у форматі YOLO, який містить ідентифікатор класу та координати обмежувальної рамки [38].

Для кожного класу наведено приклад створених зображень в різних середовищах (рис. 4.6).

Рисунок 4.6 – Створені зображення за класом

Створений набір даних складається з 2331 екземпляру розміром 1920x1080 (рис. 4.7), які належать до трьох різних класів та охоплюють 50 унікальних середовищ із різним освітленням, а також 25 траєкторій для польоту моделей дронів та 5 різних текстур для них.

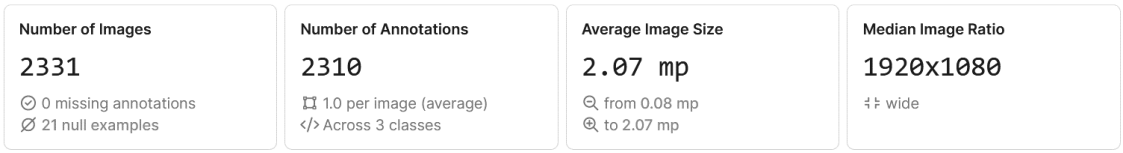

Рисунок 4.7 – Характеристики набору даних

Розподіл класів становить приблизно 35% для класу Shahed 136, 33% для класу FPV та 32,00% для класу ZALA 421-16e. Набір даних є збалансованим, з мінімальними варіаціями в кількості екземплярів у класах (рис. 4.8). Такий баланс забезпечує меншу ймовірність того, що процес навчання моделі буде зміщений щодо будь-якого конкретного класу, що призводить до більш надійних результатів [39].

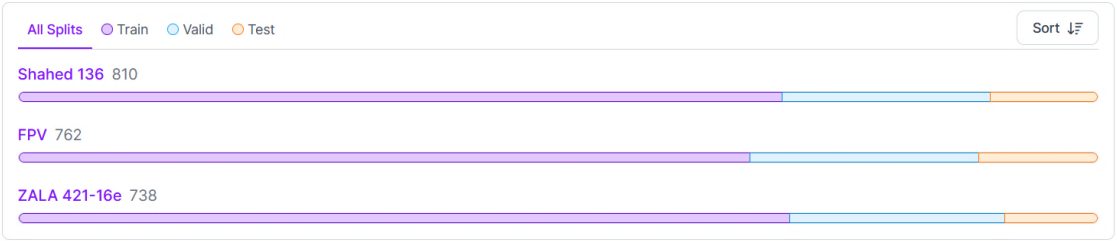

Рисунок 4.8 – Розподіл класів в наборі даних

Досліджено теплову карту анотацій (рис. 4.9), яка дає змогу візуалізувати розподіл анотацій у зображеннях набору даних. Аналіз дозволяє виявити просторові зміщення, що дозволило оцінити область покриття анотацій [40]. У наборі даних не виявлено прогалин, рідкісних чи нерівномірних зон покриття. Також не зафіксовано аномалій у вигляді анотацій, зосереджених лише на краях зображень або розташованих у незвичних візерунках.

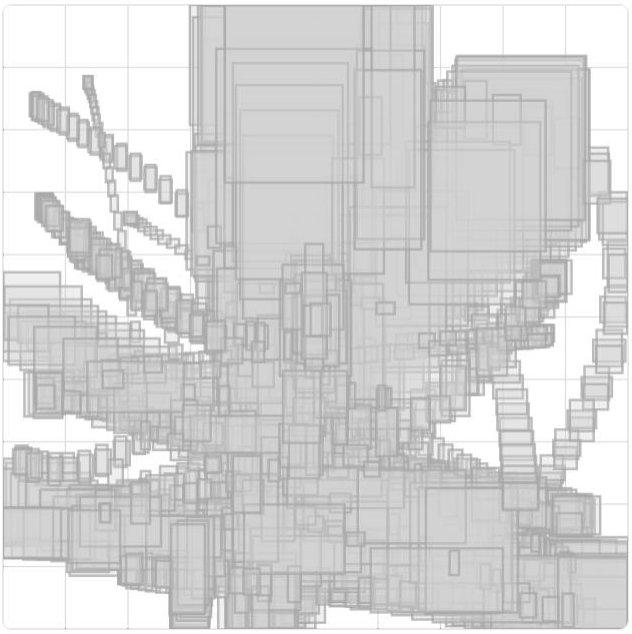

Рисунок 4.9 – Теплова карта анотацій

Після попередньої обробки та аугментації набір даних розширено до 3968 зображень для подальшого навчання моделі (рис. 4.10).

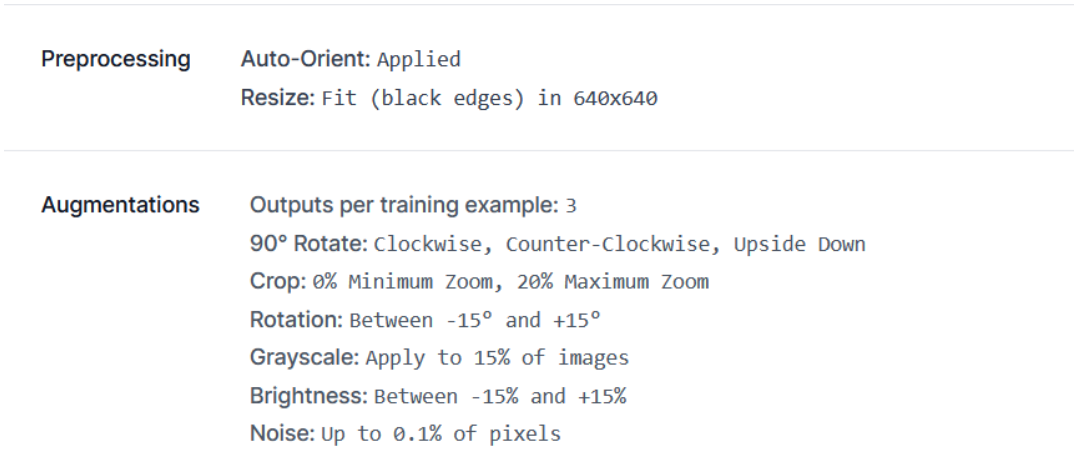

Рисунок 4.10 – Попередня обробка зображень та аугментація

Для набору даних було додано зображення без анотацій для зменшення кількості хибнопозитивних спрацьовувань (FP). Фонові зображення, які не містять об’єктів, використовуються для покращення точності розпізнання, оскільки вони допомагають моделі правильно розрізняти зображення, де відсутні об’єкти.

Рекомендується додавати 0-10% фонових зображень до набору даних, оскільки це дозволяє знизити кількість хибнопозитивних виявлень [41], тому було додано 1% зображення без анотацій (рис. 4.11). Для фонових зображень не вимагаються анотації, оскільки вони не містять об’єктів для позначення.

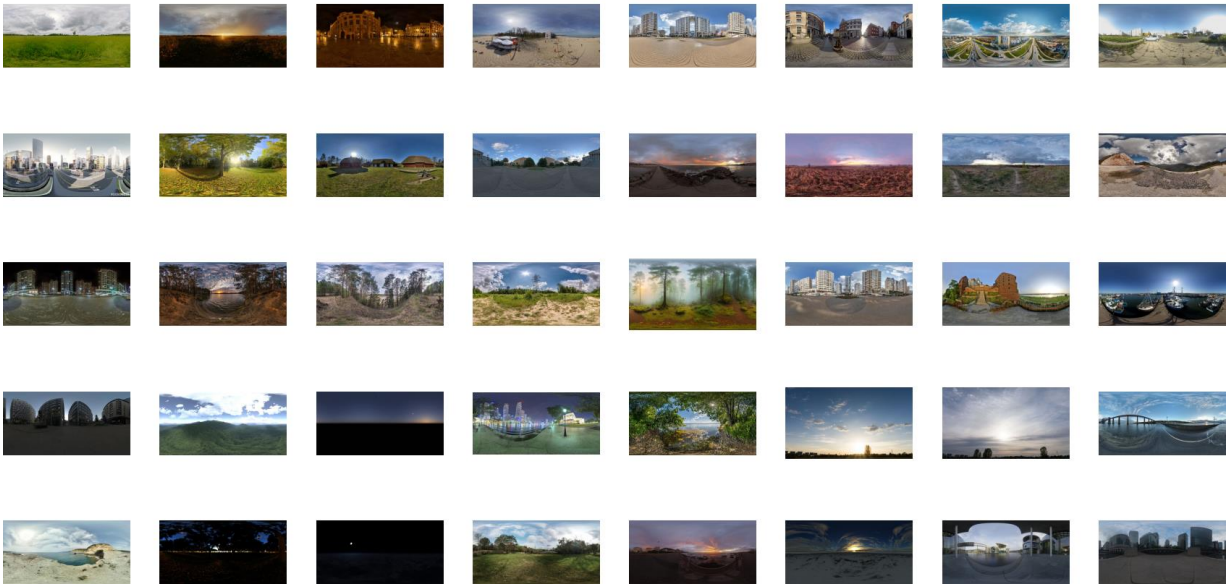

Рисунок 4.11 – Зображення без анотацій

Розподіл набору даних обрано 70/20/10 (рис. 4.12). У такому порядку 70% даних використовується для навчання моделі, що дозволяє їй вивчати шаблони та зв’язки в наборі даних. Зарезервовано 20% для валідації, що допомагає оцінити продуктивність моделі під час навчання. Решта 10% відкладено як тестовий набір, який використовується для оцінки продуктивності моделі на невидимих даних.

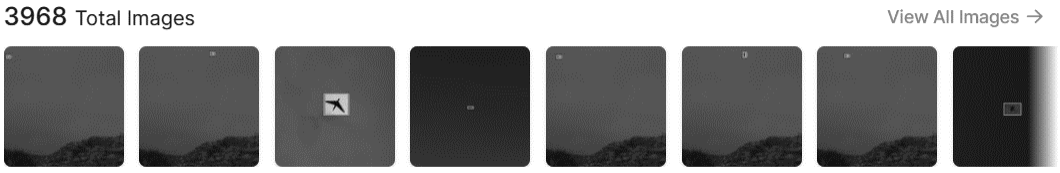

Рисунок 4.12 – Розподіл набору даних 70/20/10

Представлено готовий набір даних (рис. 4.13) для задачі ідентифікації та розпізнання дронів використовуючи обрані моделі виявлення об’єктів.

Рисунок 4.13 – Готовий набір даних

Було також створено набір даних, використовуючи реальні зображення з відкритих джерел, які були вручну анотовані (рис. 4.14). Даний набір даних потрібен для подальших досліджень, аналізу результатів та оцінки моделі, навченої на синтетичних даних для перевірки її ефективності на реальних даних.

Рисунок 4.14 – Набір даних з реальними зображеннями

4.2 Навчання моделі YOLO

Для розв'язання поставленої задачі було використано попередньо натреновану модель YOLOv10m на основі набору даних COCO (рис. 4.15). Використання YOLOv10m дозволило скористатися перевагами трансферного навчання, що забезпечило високу точність та суттєво скоротило час тренування моделі [42].

Model	Input Size	AP^{val}	FLOPs (G)	Latency (ms)
YOLOv10-N	640	38.5	**6.7**	**1.84**
YOLOv10-S	640	46.3	21.6	2.49
YOLOv10-M	640	51.1	59.1	4.74
YOLOv10-B	640	52.5	92.0	5.74
YOLOv10-L	640	53.2	120.3	7.28
YOLOv10-X	640	**54.4**	160.4	10.70

Рисунок 4.15 – Метрики попередньо навчених моделей YOLOv10

Для моделі було використано задану конфігурацію для навчання (рис. 4.16). Параметр task=detect, що визначає задачу виявлення об'єктів, mode=train, який вмикає режим навчання [43]. Модель навчали протягом 200 епох із розміром партії (batch size) 16. Попередньо навчені ваги завантажувалися з файлу yolov10m.pt, що слугувало початковою точкою. Зображення масштабували до 640x640 пікселів, а для ефективної обробки даних використовувалися два робітники завантажувача.

```
!yolo task=detect mode=train \
    epochs=200 \
    batch=16 \
    plots=True \
    model={HOME}/weights/yolov10m.pt \
    data={dataset.location}/data.yaml \
    augment=True \
    hsv_h=0.015 \
    hsv_s=0.7 \
    hsv_v=0.4 \
    degrees=10 \
    translate=0.1 \
    scale=0.5 \
    fliplr=0.5 \
    mosaic=1.0 \
    mixup=0.1 \
    flipud=0.1
```

Рисунок 4.16 – Конфігурація моделі YOLO

Застосовувалася аугментація даних, яка включала: регулювання кольору у просторі HSV, випадкові обертання до 10°, зсуви на 10%, масштабування до 50%, горизонтальні відображення (ймовірність 50%), вертикальні відображення (ймовірність 10%) та мозаїчну аугментацію зі змішуванням (mixup) на рівні 0.1

Продемонстровано архітектуру попередньо натреновану модель YOLOv10m (рис. 4.17).

```
Overriding model.yaml nc=80 with nc=3

                  from  n    params  module                                       arguments
  0                 -1  1      1392  ultralytics.nn.modules.conv.Conv             [3, 48, 3, 2]
  1                 -1  1     41664  ultralytics.nn.modules.conv.Conv             [48, 96, 3, 2]
  2                 -1  2    111360  ultralytics.nn.modules.block.C2f             [96, 96, 2, True]
  3                 -1  1    166272  ultralytics.nn.modules.conv.Conv             [96, 192, 3, 2]
  4                 -1  4    813312  ultralytics.nn.modules.block.C2f             [192, 192, 4, True]
  5                 -1  1     78720  ultralytics.nn.modules.block.SCDown          [192, 384, 3, 2]
  6                 -1  4   3248640  ultralytics.nn.modules.block.C2f             [384, 384, 4, True]
  7                 -1  1    228672  ultralytics.nn.modules.block.SCDown          [384, 576, 3, 2]
  8                 -1  2   1689984  ultralytics.nn.modules.block.C2fCIB          [576, 576, 2, True]
  9                 -1  1    831168  ultralytics.nn.modules.block.SPPF            [576, 576, 5]
 10                 -1  1   1253088  ultralytics.nn.modules.block.PSA             [576, 576]
 11                 -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 12            [-1, 6]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 13                 -1  2   1993728  ultralytics.nn.modules.block.C2f             [960, 384, 2]
 14                 -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 15            [-1, 4]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 16                 -1  2    517632  ultralytics.nn.modules.block.C2f             [576, 192, 2]
 17                 -1  1    332160  ultralytics.nn.modules.conv.Conv             [192, 192, 3, 2]
 18           [-1, 13]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 19                 -1  2    831744  ultralytics.nn.modules.block.C2fCIB          [576, 384, 2, True]
 20                 -1  1    152448  ultralytics.nn.modules.block.SCDown          [384, 384, 3, 2]
 21           [-1, 10]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 22                 -1  2   1911168  ultralytics.nn.modules.block.C2fCIB          [960, 576, 2, True]
 23       [16, 19, 22]  1   2284450  ultralytics.nn.modules.head.v10Detect        [3, [192, 384, 576]]
YOLOv10m summary: 498 layers, 16487602 parameters, 16487586 gradients, 64.0 GFLOPs
```

Рисунок 4.17 – Звіт моделі YOLO

Оптимізатором обрано Adam зі швидкістю навчання 0,001429 та моментом імпульсу 0,9. Кількість класів моделі скорегована до трьох відповідно до специфіки набору даних.

Конфігураційний файл data.yaml містив інформацію про класи та шляхи до файлів (рис. 4.18).

data.yaml

```
names:
- FPV
- Shahed 136
- ZALA 421-16e
nc: 3

test: ../test/images
train: ../train/images
val: ../valid/images

```

Рисунок 4.18 – Конфігураційний файл даних для моделі YOLO

Продемонстровано значення втрат під час навчання моделі (рис. 4.19). Значення втрат, таких як train/box_loss та train/box_loss, демонструють, як модель поступово покращує прогнозування координат рамок об’єктів. Зменшення значень втрат train/cls_loss та train/cls_loss свідчить про покращення точності класифікації об’єктів. Втрати train/dfl_loss та train/dfl_loss, які відповідають за локалізацію об’єктів, також зменшуються, що свідчить про покращення точності визначення розташування об’єктів.

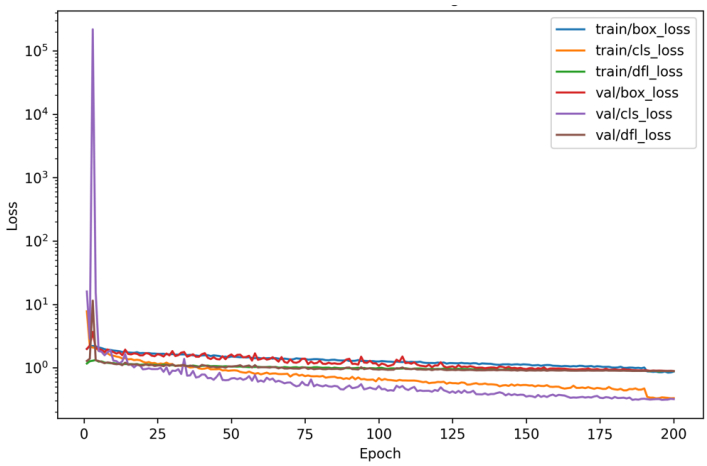

Рисунок 4.19 – Значення втрат моделі YOLO

Продемонстровано значення метрик оцінки під час навчання моделі (рис. 4.20). Графік точності (precision) демонструє, що кількість хибнопозитивних спрацьовувань зменшується. Графік повноти (recall) демонструє, що модель все краще виявляє всі об’єкти на зображеннях, оскільки значення повноти постійно

зростає [44]. Графіки mAP50 та mAP50-95 показують середню точність при різних порогах IoU, їхній стабільний ріст вказує на покращення загальної точності моделі у виявленні об’єктів.

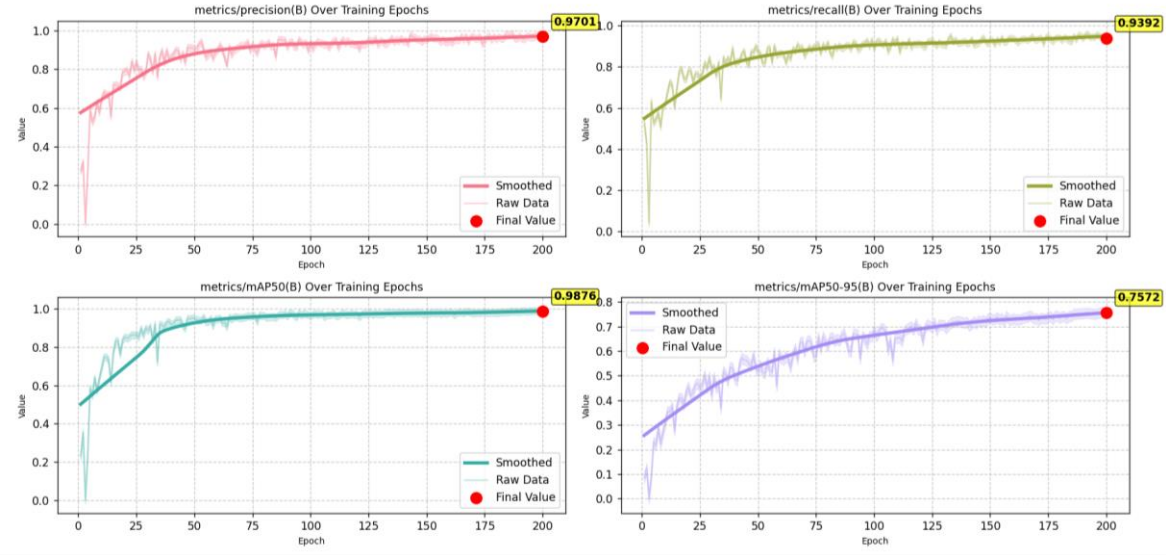

Рисунок 4.20 – Метрики оцінки точності моделі YOLO

Під час навчання моделі точність (precision) зростає до 0,97015. Повнота (recall) збільшується до 0,93919, що відображає помітне покращення здатності моделі ідентифікувати відповідні об’єкти. Показник mAP50 зростає до 0,98763, а mAP50-95 до 0,75716, демонструючи суттєве покращення здатності моделі виявляти об’єкти за різних порогів IoU.

Продемонстровано значення метрик оцінки точності під час тестування моделі (рис. 4.21). Оцінено модель за допомогою тестових зображень з набору даних та отримано точність (precision) 0,9295, також повноту (recall) 0,9782. F1-міра становить 0.9532, що вказує на збалансовану продуктивність між точністю та повнотою. Модель досягає mAP50 зі значенням 0,9825 та mAP50-95 на рівні 0,6912.

Аналізуючи продуктивність для кожного класу, модель демонструє кращі результати для класу FPV. Наступним за показниками йде клас Shahed 136 та клас ZALA 421-e. Збалансована продуктивність для різних класів вказує на відсутність значного перекосу.

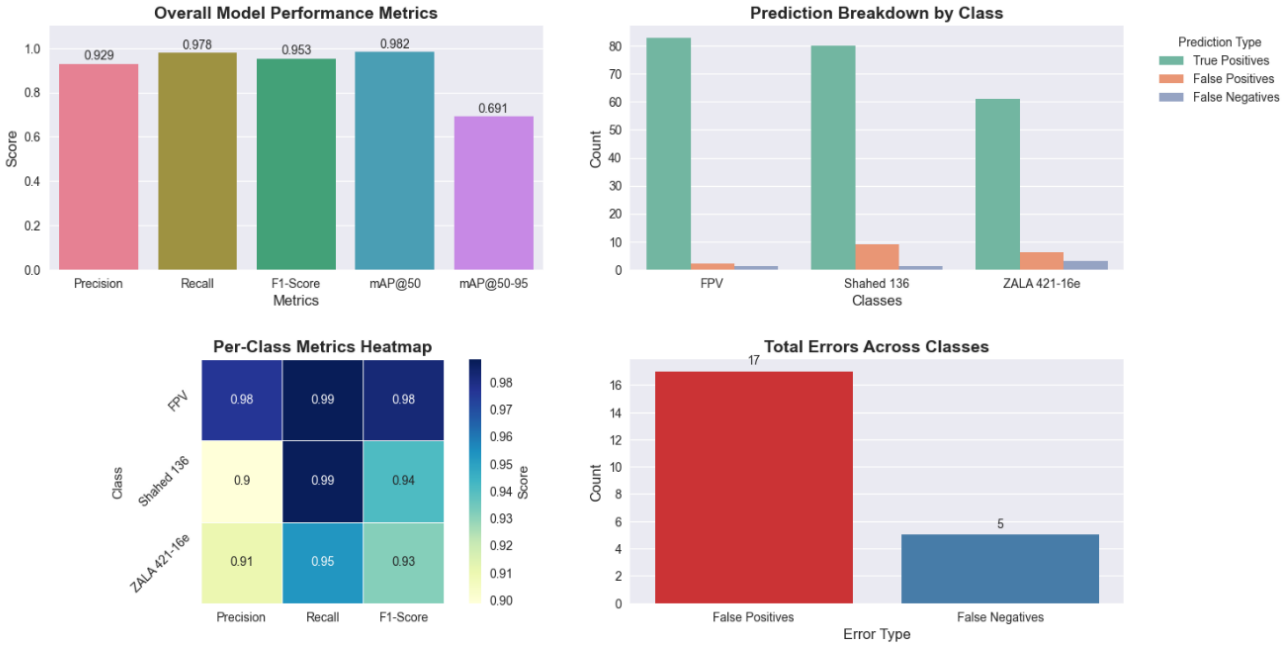

Рисунок 4.21 – Оцінка продуктивність моделі YOLO

Попри високу ефективність, було отримано 17 хибнопозитивних та 5 хибнонегативних результатів, що свідчить про потенціал для покращення, особливо у зменшенні кількості хибнонегативних прогнозів.

4.3 Навчання моделі Faster R-CNN

Для розв'язання поставленої задачі також використано попередньо навчену модель Faster R-CNN [45] з основою MobileNetV3-Large та мережею піраміди ознак (FPN) на основі набору даних COCO (рис. 4.22).

box_map (on COCO-val2017)	32.8
categories	__background__, person, bicycle, ... (88 omitted)
min_size	height=1, width=1
num_params	19386354
recipe	link
GFLOPS	4.49

Рисунок 4.22 – Метрики попередньо навченої моделі Faster R-CNN

Навчання проводилося протягом 100 епох із розміром партії (batch size) 32 (рис. 4.23). Швидкість навчання становила 0,001, а для оптимізації застосовувався метод SGD. Зображення змінювалися до розміру 640x640 пікселів. Застосовувалося

раннє зупинення. Навчання завершилося на 70-й епосі через відсутність покращення показника mAP на валідаційних даних протягом 10 епох поспіль.

```
!python train.py
--data data_configs/custom_data.yaml
--epochs 100
--model fasterrcnn_mobilenetv3_large_fpn
--name custom_training --batch 32
```

Рисунок 4.23 – Конфігурація моделі Faster R-CNN

Продемонстровано архітектуру попередньо натреновану модель YOLOv10m (рис. 4.24).

FasterRCNN (FasterRCNN)	[32, 3, 640, 640]	[0, 4]	--
├─GeneralizedRCNNTransform (transform)	[32, 3, 640, 640]	[32, 3, 640, 640]	--
├─BackboneWithFPN (backbone)	[32, 3, 640, 640]	[32, 256, 10, 10]	--
│ └─IntermediateLayerGetter (body)	[32, 3, 640, 640]	[32, 960, 20, 20]	--
│ │ └─Conv2dNormActivation (0)	[32, 3, 640, 640]	[32, 16, 320, 320]	(432)
│ │ └─InvertedResidual (1)	[32, 16, 320, 320]	[32, 16, 320, 320]	(400)
│ │ └─InvertedResidual (2)	[32, 16, 320, 320]	[32, 24, 160, 160]	(3,136)
│ │ └─InvertedResidual (3)	[32, 24, 160, 160]	[32, 24, 160, 160]	(4,104)
│ │ └─InvertedResidual (4)	[32, 24, 160, 160]	[32, 40, 80, 80]	(9,960)
│ │ └─InvertedResidual (5)	[32, 40, 80, 80]	[32, 40, 80, 80]	(20,432)
│ │ └─InvertedResidual (6)	[32, 40, 80, 80]	[32, 40, 80, 80]	(20,432)
│ │ └─InvertedResidual (7)	[32, 40, 80, 80]	[32, 80, 40, 40]	30,960
│ │ └─InvertedResidual (8)	[32, 80, 40, 40]	[32, 80, 40, 40]	33,800
│ │ └─InvertedResidual (9)	[32, 80, 40, 40]	[32, 80, 40, 40]	31,096
│ │ └─InvertedResidual (10)	[32, 80, 40, 40]	[32, 80, 40, 40]	31,096
│ │ └─InvertedResidual (11)	[32, 80, 40, 40]	[32, 112, 40, 40]	212,280
│ │ └─InvertedResidual (12)	[32, 112, 40, 40]	[32, 112, 40, 40]	383,208
│ │ └─InvertedResidual (13)	[32, 112, 40, 40]	[32, 160, 20, 20]	426,216
│ │ └─InvertedResidual (14)	[32, 160, 20, 20]	[32, 160, 20, 20]	793,200
│ │ └─InvertedResidual (15)	[32, 160, 20, 20]	[32, 160, 20, 20]	793,200
│ │ └─Conv2dNormActivation (16)	[32, 160, 20, 20]	[32, 960, 20, 20]	153,600
│ └─FeaturePyramidNetwork (fpn)	[32, 160, 20, 20]	[32, 256, 10, 10]	--
│ │ └─ModuleList (inner_blocks)	--	--	(recursive)
│ │ └─ModuleList (layer_blocks)	--	--	(recursive)
│ │ └─ModuleList (inner_blocks)	--	--	(recursive)
│ │ └─ModuleList (layer_blocks)	--	--	(recursive)
│ │ └─LastLevelMaxPool (extra_blocks)	[32, 256, 20, 20]	[32, 256, 20, 20]	--
├─RegionProposalNetwork (rpn)	[32, 3, 640, 640]	[0, 4]	--
│ └─RPNHead (head)	[32, 256, 20, 20]	[32, 15, 20, 20]	--
│ │ └─Sequential (conv)	[32, 256, 20, 20]	[32, 256, 20, 20]	590,080
│ │ └─Conv2d (cls_logits)	[32, 256, 20, 20]	[32, 15, 20, 20]	3,855
│ │ └─Conv2d (bbox_pred)	[32, 256, 20, 20]	[32, 60, 20, 20]	15,420
│ │ └─Sequential (conv)	[32, 256, 20, 20]	[32, 256, 20, 20]	(recursive)
│ │ └─Conv2d (cls_logits)	[32, 256, 20, 20]	[32, 15, 20, 20]	(recursive)
│ │ └─Conv2d (bbox_pred)	[32, 256, 20, 20]	[32, 60, 20, 20]	(recursive)
│ │ └─Sequential (conv)	[32, 256, 10, 10]	[32, 256, 10, 10]	(recursive)
│ │ └─Conv2d (cls_logits)	[32, 256, 10, 10]	[32, 15, 10, 10]	(recursive)
│ │ └─Conv2d (bbox_pred)	[32, 256, 10, 10]	[32, 60, 10, 10]	(recursive)
│ └─AnchorGenerator (anchor_generator)	[32, 3, 640, 640]	[13500, 4]	--
├─RoIHeads (roi_heads)	[32, 256, 20, 20]	[0, 4]	--
│ └─MultiScaleRoIAlign (box_roi_pool)	[32, 256, 20, 20]	[0, 256, 7, 7]	--
│ └─TwoMLPHead (box_head)	[0, 256, 7, 7]	[0, 1024]	--
│ │ └─Linear (fc6)	[0, 12544]	[0, 1024]	12,846,080
│ │ └─Linear (fc7)	[0, 1024]	[0, 1024]	1,049,600
│ └─FastRCNNPredictor (box_predictor)	[0, 1024]	[0, 4]	--
│ │ └─Linear (cls_score)	[0, 1024]	[0, 4]	4,100
│ │ └─Linear (bbox_pred)	[0, 1024]	[0, 16]	16,400

Рисунок 4.24 – Звіт моделі Faster R-CNN

У конфігурації моделі використовувався файл data.yaml, який містив інформацію про класи й шляхи до файлів (рис. 4.25).

```
TRAIN_DIR_IMAGES: 'custom_data/train'
TRAIN_DIR_LABELS: 'custom_data/train'
VALID_DIR_IMAGES: 'custom_data/valid'
VALID_DIR_LABELS: 'custom_data/valid'


CLASSES: [
    '__background__',
    'FPV', 'Shahed 136', 'ZALA 421-16e'
]


NC: 4

SAVE_VALID_PREDICTION_IMAGES: True
```

Рисунок 4.25 – Конфігураційний файл даних для моделі Faster R-CNN

Під час навчання загальна втрата (train_loss) поступово зменшувалася (рис. 4.26). Класифікаційні втрати (train_cls_loss) демонстрували стабільне покращення, що свідчить про вдосконалення здатності моделі розпізнавати об’єкти в межах рамок. Втрати регресії обмежувальних рамок (train_box_reg_loss) також зменшувалися, що свідчить про покращення точності у визначенні розташування об’єктів. Втрата предметності (train_obj_loss) швидко стабілізувалася і залишалася низькою, вказуючи на здатність моделі відрізняти об’єкти від фону. Втрати регіональної мережі пропозицій (train_rpn_loss) залишалися стабільними й низькими, демонструючи ефективність у генерації пропозицій вже на ранніх етапах навчання. Існують сплески навколо 35 епохи, що є результатом до навчання.

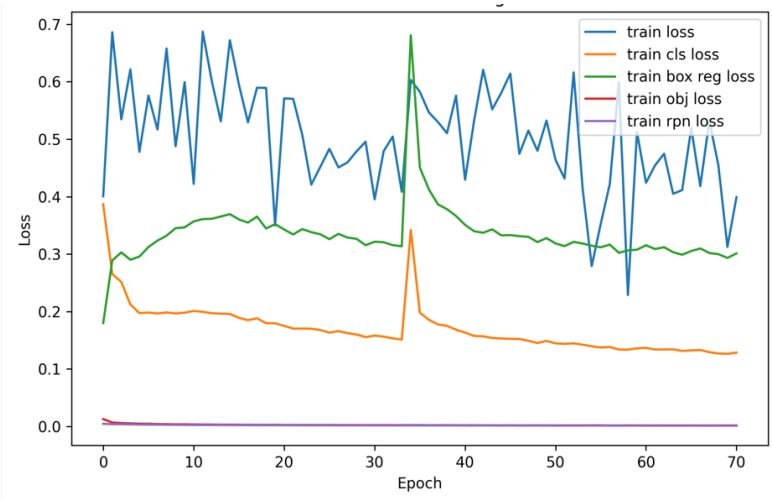

Рисунок 4.26 – Значення втрат моделі Faster R-CNN

Продемонстровано значення метрик оцінки під час навчання моделі (рис. 4.27). Навчальні показники mAP50 та mAP50-95, демонструють послідовне покращення протягом 70 епох. Модель досягла mAP50-95 на рівні 0,5548, а mAP50 піднявся до 0,9419.

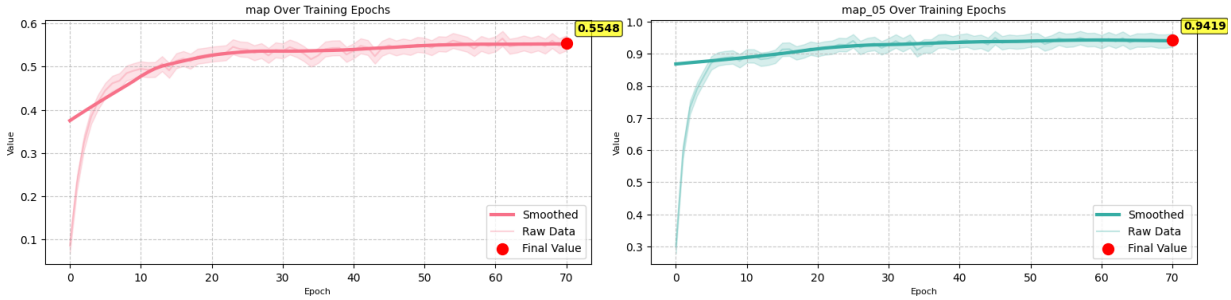

Рисунок 4.27 – Метрики оцінки точності моделі Faster R-CNN

Додатково продемонстровано значення метрик оцінки під час навчання моделі для визначення ефективності моделі для виявлення об’єктів різного розміру (рис. 4.28).

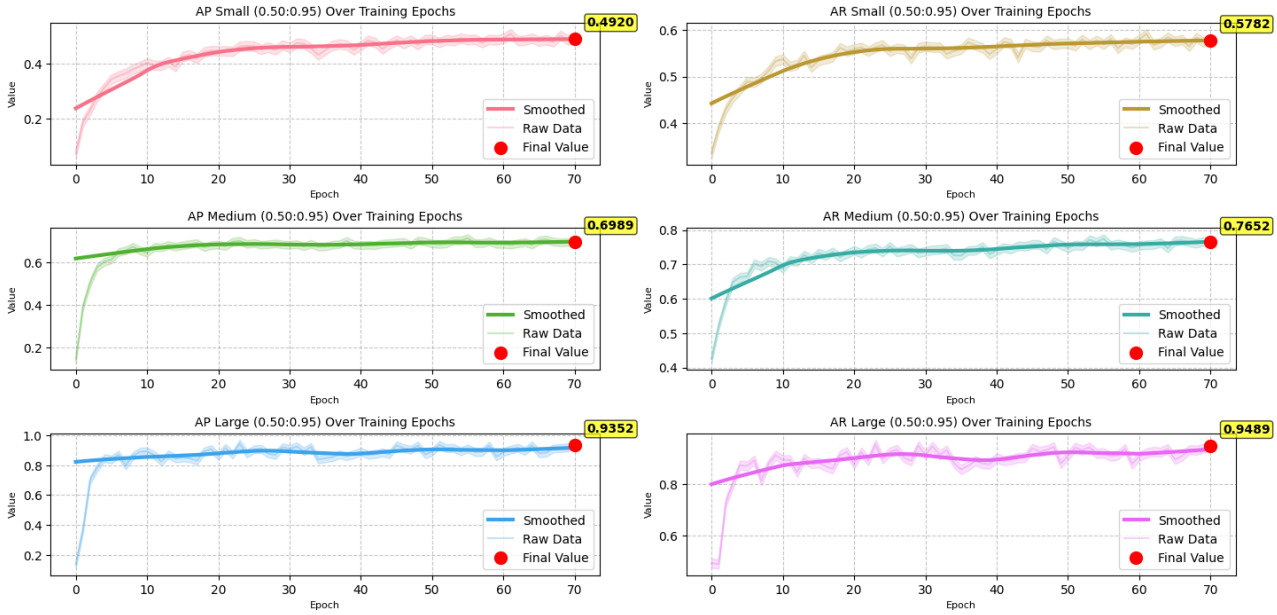

Рисунок 4.28 – Оцінка виявлення об’єктів різного розміру моделі Faster R-CNN

Аналіз продуктивності показав різні результати залежно від розмірів об’єктів. Для малих об’єктів mAP50-95 становив 0,4920, а mAR50-95 – 0,5782, що вказує на складність ідентифікації дрібних об’єктів. Для середніх об’єктів mAP50-95 піднявся до 0,6989, а mAR50-95 – до 0,7652. Великі об’єкти були виявлені з найбільшою точністю, показник mAP50-95 дорівнював 0,9352, а mAR50-95 – 0,9489, що свідчить про впевнену здатність моделі працювати з великими об’єктами.

На тестових даних загальна середня точність mAP50-95 досягла 0,5211, а mAP50 – 0,9458 (рис. 4.29).

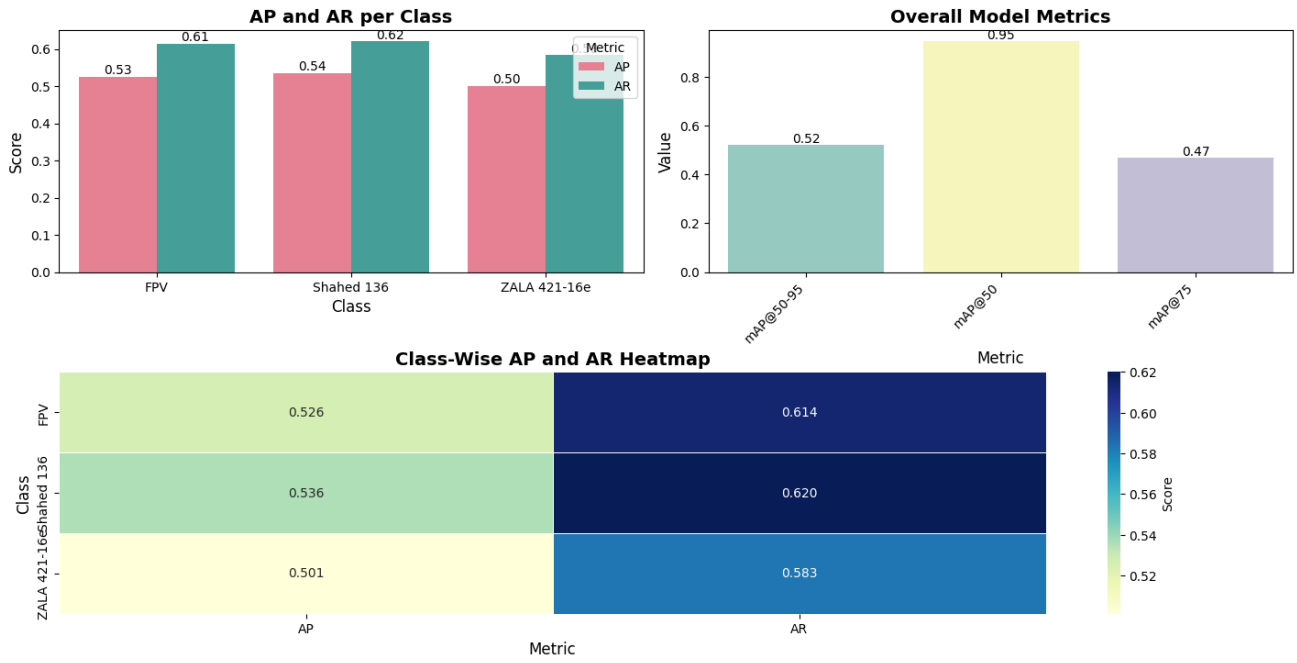

Рисунок 4.29 – Оцінка продуктивність моделі Faster R-CNN

Результати оцінки по класах були збалансовані, показник mAP50-95 становив 0,5261, 0,5358 та 0,5013 для класів. За класами значення показника mAR50-95 були відповідно 0,6142, 0,6199 та 0,5830, що підкреслює послідовну продуктивність моделі.

4.4 Аналіз результатів моделей

Сформовано таблицю, яка відображає результати продуктивності моделі YOLO на валідаційних даних під час навчання (табл. 4.1). У ній наведено кінцеві значення метрик, а також найкращі значення, досягнуті на певній епосі.

Модель YOLO показує високу продуктивність як на тестових, так і на валідаційних даних, з невеликими відмінностями у метриках (табл. 4.2).

Таблиця 4.1 – Результати оцінювання метрик на етапі навчання моделі YOLO

Метрика	**Кінцеве значення**	**Найкраще значення**	**Найкраща епоха**
Precision	0,97015	0,98203	186
Recall	0,93919	0,96071	191
mAP50	0,98763	0,98966	195
mAP50-95	0,75716	0,75729	197

Таблиця 4.2 – Порівняння метрик тестування та навчання моделі YOLO

Метрика	**Продуктивність на тестових даних**	**Продуктивність на валідаційних даних**	**Різниця у продуктивності**
Precision	0,9295	0,98203	-0,05253
Recall	0,9782	0,96071	+0,01749
mAP50	0,9825	0,98966	-0,00716
mAP50-95	0,6912	0,75729	-0,06609

Завантажено базові мітки істинності та зроблено прогнози для тестових зображень з метою візуального порівняння результатів моделі YOLO (рис. 4.30). Зелені обмежувальні рамки вказують на фактичне розташування об’єкту, а червоні є результатом виявлення об’єкту моделлю. В легенді зазначено фактична мітка класу та прогнозована моделлю з показником впевненості.

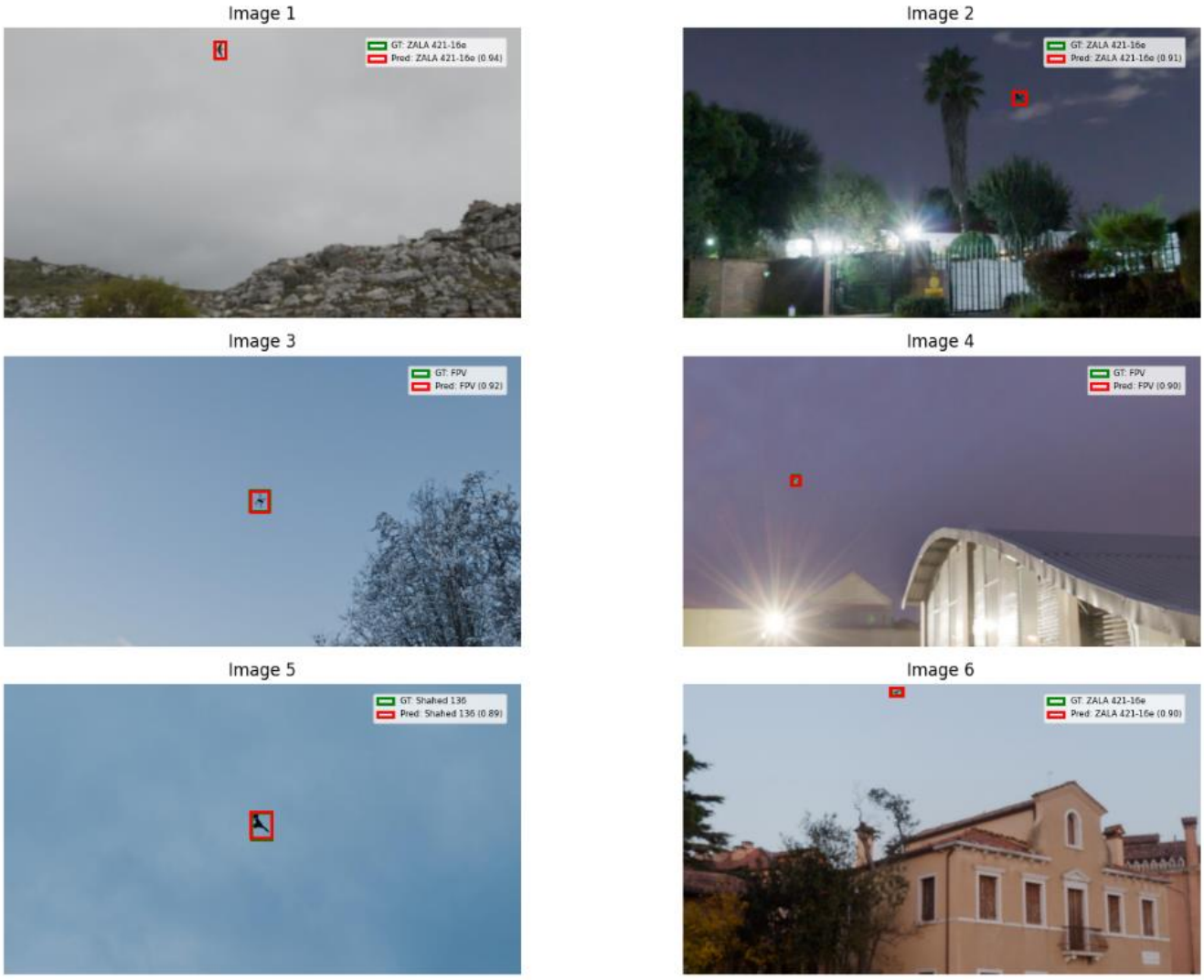

Рисунок 4.30 – Прогнози для тестових зображень моделі YOLO

Модель демонструє високу точність на тестових зображеннях, ефективно справляючись із виявленням об’єктів у різних середовищах та з різними розмірами об’єктів. Завдяки цьому модель здатна адаптуватися до змінних умов та точно ідентифікувати цілі навіть за складних варіантів зображень.

Завантажено базові мітки істинності та зроблено прогнози для реальних зображень з метою візуального порівняння результатів моделі YOLO (рис. 4.31).

Рисунок 4.31 – Прогнози для реальних зображень моделі YOLO

На реальних зображеннях спостерігається зниження впевненості моделі щодо класу об’єкта порівняно з тестовими зображеннями. Однак, модель точно виявляє об’єкти, що свідчить про її ефективність навіть в умовах реальних сценаріїв.

Сформовано таблицю, яка відображає результати продуктивності моделі Faster R-CNN на валідаційних даних під час навчання (табл. 4.3). У ній наведено кінцеві значення метрик, а також найкращі значення, досягнуті на певній епосі.

Модель Faster R-CNN показує високу продуктивність як на тестових, так і на валідаційних даних, з невеликими відмінностями у метриках (табл. 4.4).

Таблиця 4.3 – Результати метрик на етапі навчання моделі Faster R-CNN

Метрика	**Кінцеве значення**	**Найкраще значення**	**Найкраща епоха**
mAP50-95	0,5548	0,5652	61
mAP50	0,9419	0,9500	57
Small Objects mAP50-95	0,4920	0,5093	61
Medium Objects mAP50-95	0,6989	0,7150	55
Large Objects mAP50-95	0,9352	0,9434	23
mAR50-95	0,6321	0,6396	61
Small Objects mAR50-95	0,5782	0,5918	61
Medium Objects mAR50-95	0,7652	0,7746	55
Large Objects mAR50-95	0,9489	0,9533	23

Таблиця 4.4 – Порівняння метрик тестування і навчання моделі Faster R-CNN

Метрика	**Продуктивність на тестових даних**	**Продуктивність на валідаційних даних**	**Різниця у продуктивності**
mAP50-95	0,5211	0,5652	-0,0441
mAP50	0,9458	0,9500	-0,0042
Small Objects mAP50-95	0,4699	0,5093	-0,0394
Medium Objects mAP50-95	0,6516	0,7150	-0,0634
Large Objects mAP50-95	0,8934	0,9434	-0,05
Small Objects mAR50-95	0,5535	0,5918	-0,0383
Medium Objects mAR50-95	0,7309	0,7746	-0,0437
Large Objects mAR50-95	0,9122	0,9533	-0,0411

Завантажено базові мітки істинності та зроблено прогнози для тестових зображень з метою візуального порівняння результатів моделі Faster R-CNN (рис. 4.32). Зелені обмежувальні рамки вказують на фактичне розташування об’єкту, а червоні є результатом виявлення об’єкту моделлю. В легенді зазначено фактичну мітку класу та прогнозована моделлю з показником впевненості.

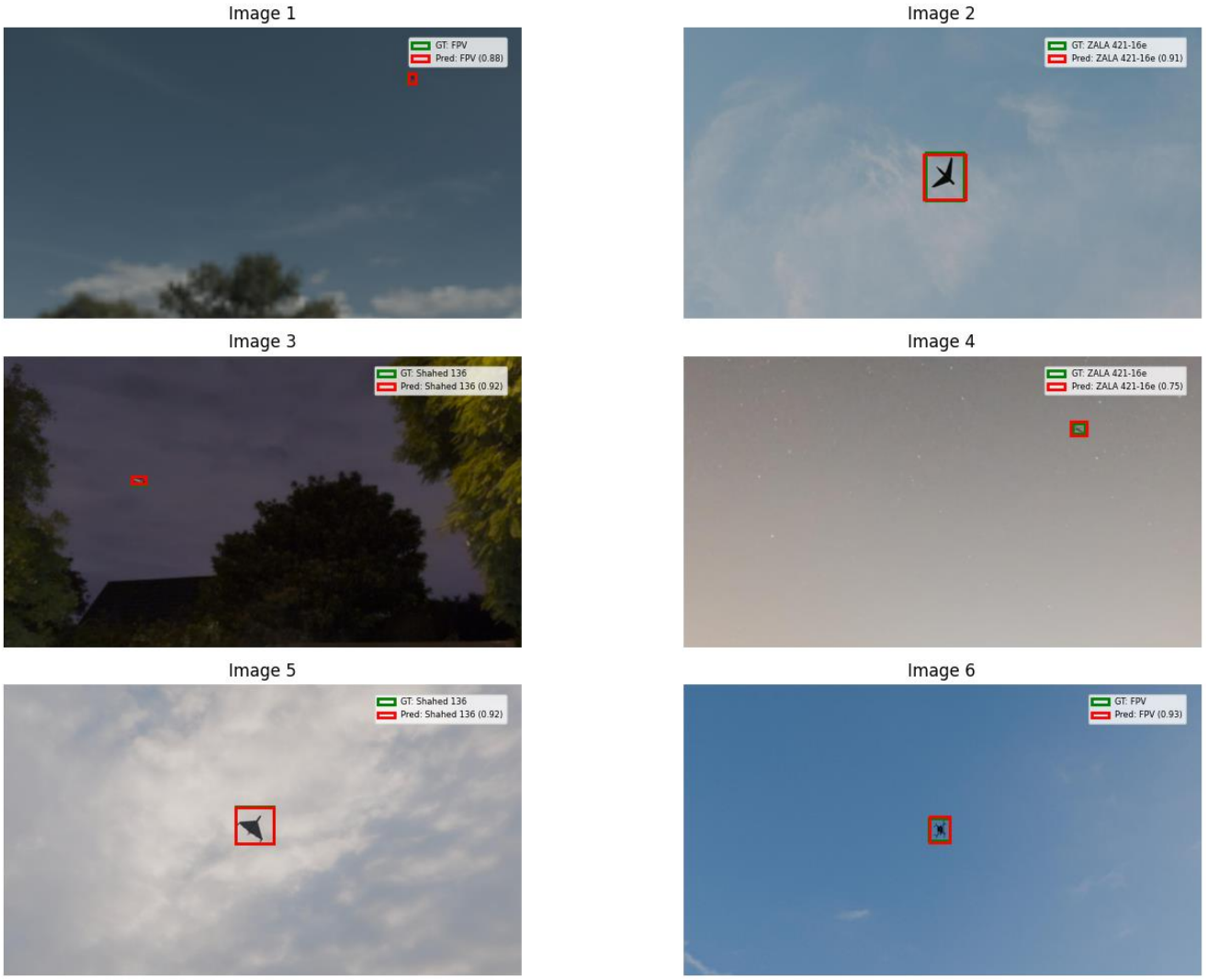

Рисунок 4.32 – Прогнози для тестових зображень моделі Faster R-CNN

Модель демонструє високу точність на тестових зображеннях, ефективно справляючись із виявленням об’єктів у різних середовищах та з різними розмірами об’єктів. Завдяки цьому модель здатна адаптуватися до змінних умов і точно ідентифікувати цілі навіть за складних варіантів зображень.

Завантажено базові мітки істинності та зроблено прогнози для реальних зображень (рис. 4.33) з метою візуального порівняння результатів моделі Faster R-CNN.

Рисунок 4.33 – Прогнози для реальних зображень моделі Faster R-CNN

Аналізуючи прогнози та результати оцінки (табл. 4.5) можна зробити висновок, що обидві моделі, YOLO та Faster R-CNN, мають схожі риси та поведінку на тестових та реальних зображеннях.

На тестових зображеннях модель YOLO для показника mAP50 досягає значення 0,9825, що вище за результат Faster R-CNN в умовах виявлення об’єктів з меншою мінімальною площею (IoU > 50%). Що стосується метрики mAP50-95, то YOLO має результат 0.6912, тоді як Faster R-CNN 0.5211. YOLO є більш ефективним у виявленні об’єктів з низьким IoU, що є важливою характеристикою в реальних сценаріях, де об’єкти можуть бути складнішими для виявлення.

Порівняння продуктивності моделей YOLO та Faster R-CNN на реальних даних, зібраних із приблизно 100 зображень продемонстрували високий рівень точності для класів FPV та Shahed 136 (табл. 4.6). Для класу ZALA 421-16e було зафіксовано суттєве зниження точності. Показник AP50 виявився приблизно у 2

рази нижчим порівняно з іншими класами для обох моделей. Така різниця може бути пов’язана з особливостями набору даних для тестування на реальних даних. Faster R-CNN продемонструвала вищу впевненість в ідентифікації та розпізнаванні дронів ніж модель YOLO.

Отже, моделі справляються з обробкою реальних зображень, однак результати для класу ZALA 421-16e вказують на необхідність детальнішого дослідження причин низької продуктивності, що може включати аналіз якості даних та можливі коригування в процесі навчання моделей.

Таблиця 4.5 – Продуктивність моделей YOLO та Faster R-CNN на тестових даних

Метрика	**YOLO**	**Faster R-CNN**	**Різниця у продуктивності**
mAP50	0,9825	0,9458	-0,0367
mAP50-95	0,6912	0,5211	-0,1701

Таблиця 4.6 – Продуктивність моделей YOLO та Faster R-CNN на реальних даних

Клас	**AP50 (YOLO)**	**AP50 (Faster R-CNN)**	**Різниця у продуктивності**
FPV	0,8366	0,9221	+0,0855
Shahed 136	0,7759	0,8846	+0,1087
ZALA 421-16e	0.3268	0.4342	+0,1074

Одним з основних недоліків моделей (рис. 4.34) YOLO та Faster R-CNN, є їхня схильність до хибнопозитивних результатів у ситуаціях, коли потрібно розпізнати об’єкти, схожі за виглядом на дрони такі як логотипи або текст. Ще одним недоліком є те, що набір даних, на яких тренуються моделі, часто охоплюють лише певний шар варіативності дронів. Об’єкти, що знаходяться близько до камери, можуть не розпізнаватися належним чином або були розпізнані з помилками через відсутність достатньої кількості даних.

Рисунок 4.34 – Спільні недоліки моделей YOLO та Faster R-CNN

Перевагами моделей YOLO та Faster R-CNN (рис. 4.35) є висока точність на реальних даних при навчанні на моделей лише на синтетичних даних. Моделі можуть розпізнавати навіть малі об’єкти, що є складною задачею для багатьох задач. Також, моделі здатні виявляти об’єкти, зображення яких можуть відрізнятися від тих, що були в навчальному наборі даних, що свідчить про їх здатність до адаптації до нових умов, що дозволяє моделі бути більш універсальною при виявленні різноманітних об’єктів у реальних сценаріях.

Рисунок 4.35 – Спільні переваги моделей YOLO та Faster R-CNN

4.4 Реалізація системи ідентифікації та розпізнавання дронів

Система ідентифікації та розпізнавання об’єктів представлена у вигляді вебзастосунка. Система працює повністю на стороні клієнта, використовуючи поєднання технологій виконання моделей безпосередньо у браузері. Система дозволяє користувачам завантажувати зображення або відео, а також використовувати камеру для виявлення об’єктів у реальному часі.

На головній сторінці відображаються варіанти для завантаження зображення або відеофайлу, а також можливість використовувати камеру для роботи в реальному часі.

Розглянемо завантаження зображення до системи (рис. 4.36). Для цього треба вибрати зображення зі свого пристрою (рис. 4.37).

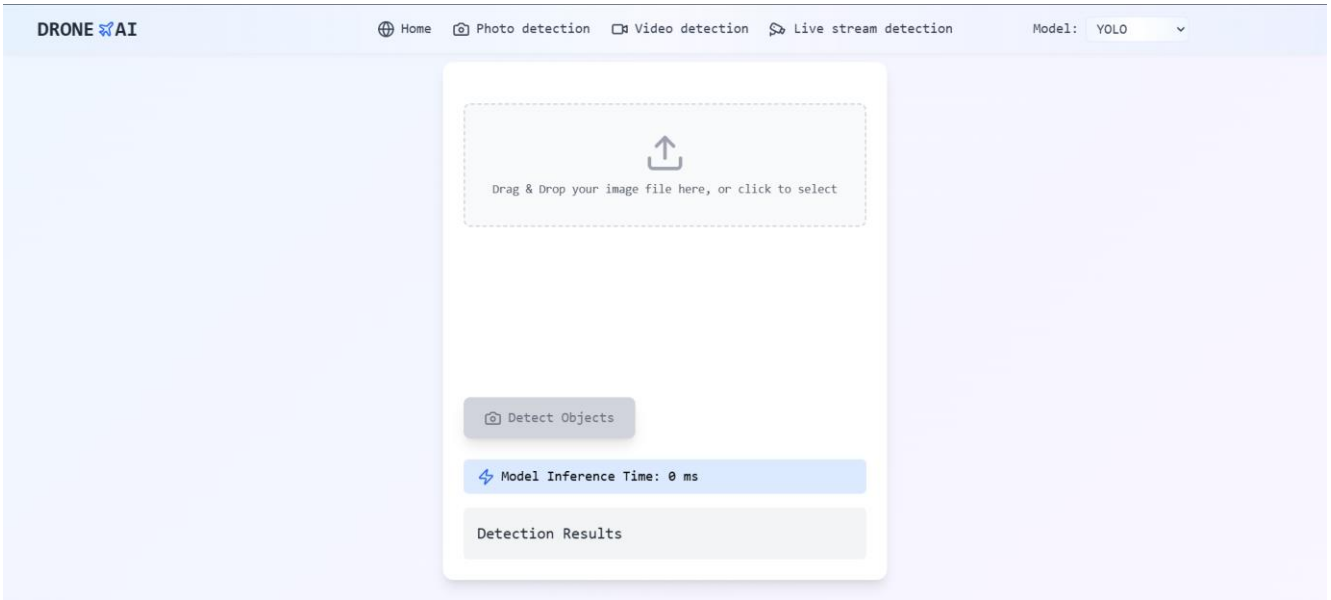

Рисунок 4.36 – Завантаження зображення до системи

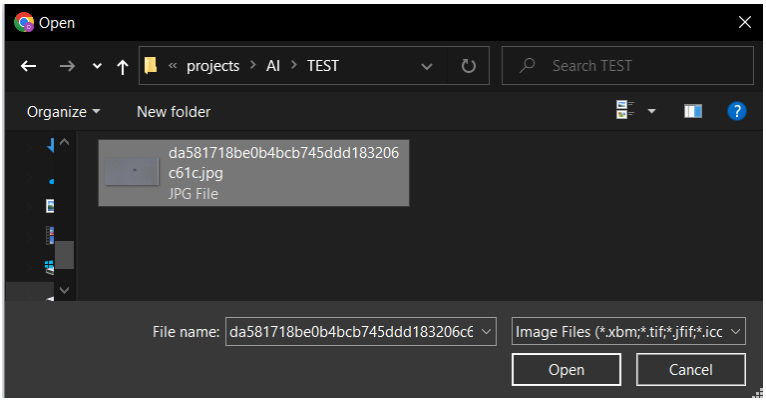

Рисунок 4.37 – Вибір зображення з пристрою

Після вибору зображення з пристрою, зображення відобразиться та система буде готовою для ідентифікації та розпізнавання дронів (рис. 4.38). Система використовує модель ONNX, яка виконується безпосередньо через WebAssembly (Wasm), забезпечуючи локальне виконання всіх обчислень. ONNX Runtime у браузері обробляє кадри відео або зображення, визначаючи об’єкти. Коли модель розпізнає об’єкти, вона прогнозує координати обмежувальних рамок, що оточують

виявлені дрони, разом із відповідними мітками (рис. 4.39). Вихідні дані з моделі дублюються знизу для кращого сприйняття ідентифікованого об’єкта.

Рисунок 4.38 – Відображення зображення до системи

Рисунок 4.39 – Результат роботи системи з зображенням

Також можна завантажити відео, обмежувальні рамки та мітки класу накладаються на кожен кадр під час відтворення відео (рис. 4.40). Для цього треба

вибрати відео зі свого пристрою (рис. 4.41). Для підвищення швидкості роботи системи використовується зображення 320x320 пікселів на вхід до моделі для обробки відео файлу.

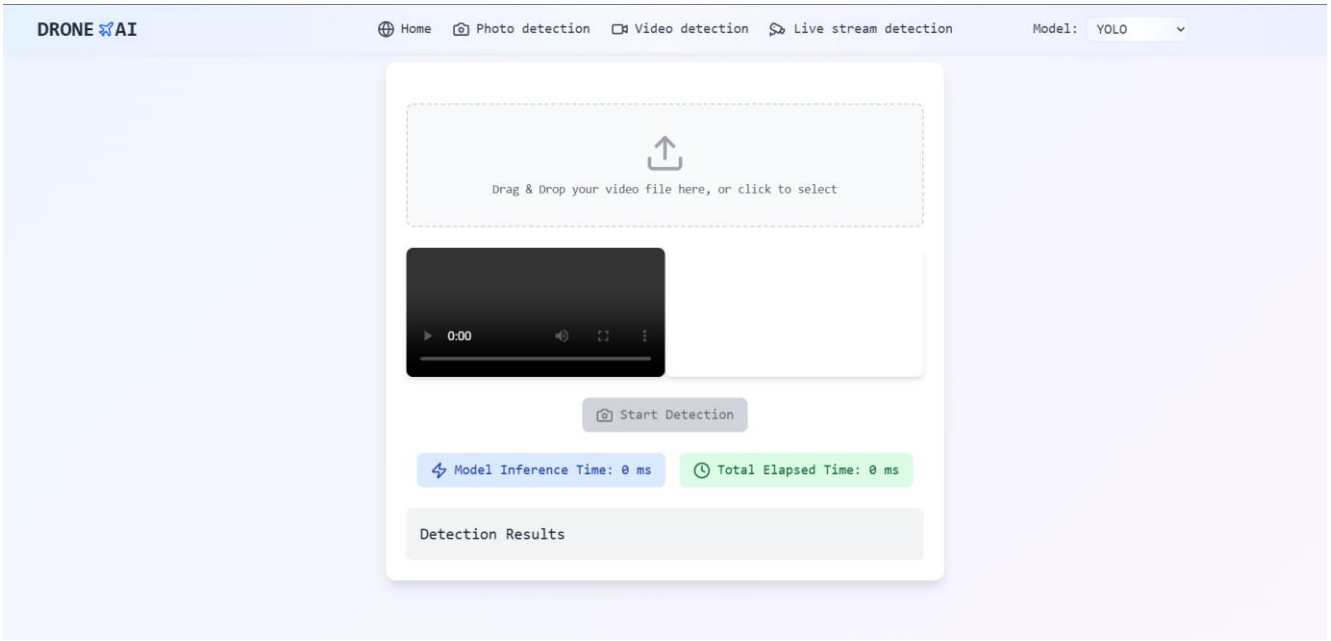

Рисунок 4.40 – Завантаження відео файлу до системи

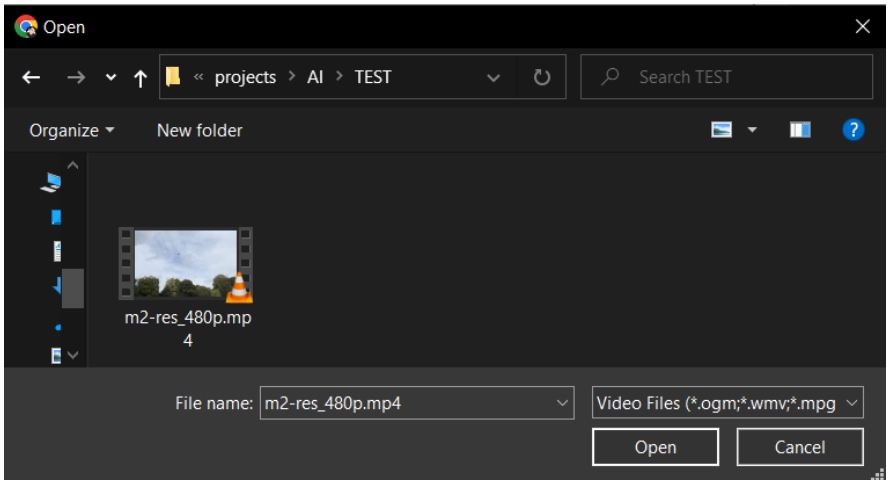

Рисунок 4.41 – Вибір відео файлу з пристрою

Після вибору відео файлу з пристрою, відео відобразиться та система буде готовою для ідентифікації та розпізнавання дронів (рис. 4.42).

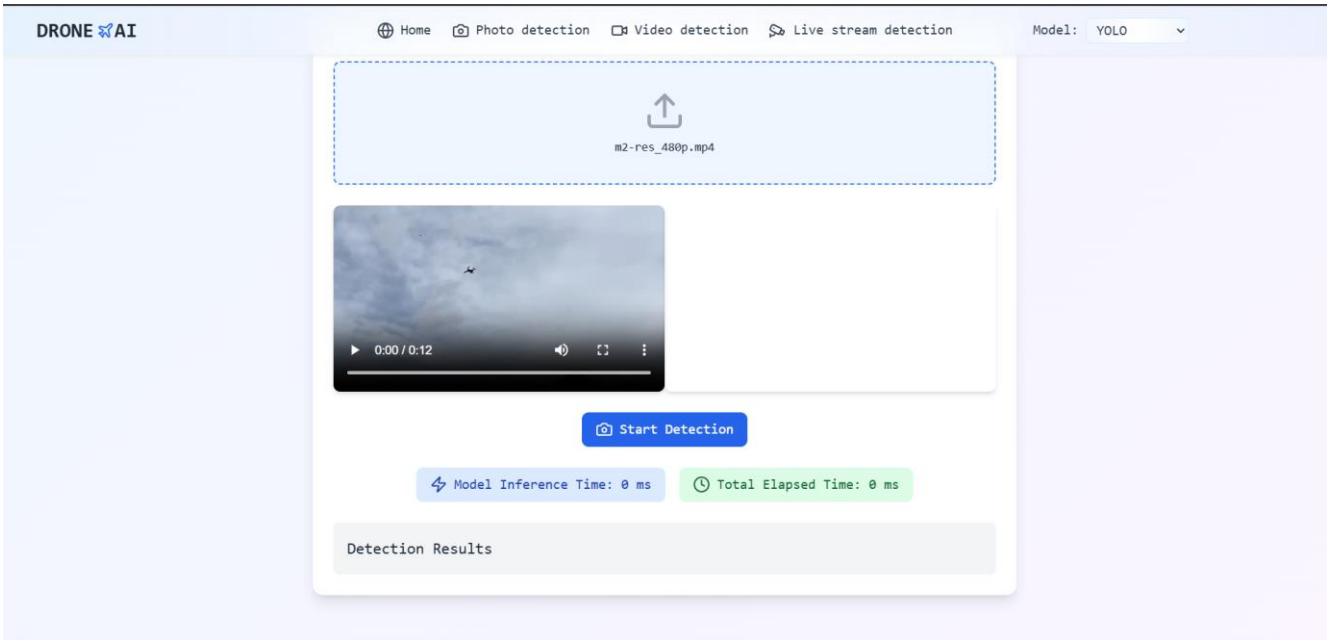

Рисунок 4.42 – Вибір відео файлу з пристрою

Система відображає кадри вхідного відео та кадри з обмежувальними рамками, на яких позначено класи об’єктів. Також реалізовані функції роботи з відео для призупинення, поновлення та повної зупинки процесу розпізнавання дронів (рис. 4.43).

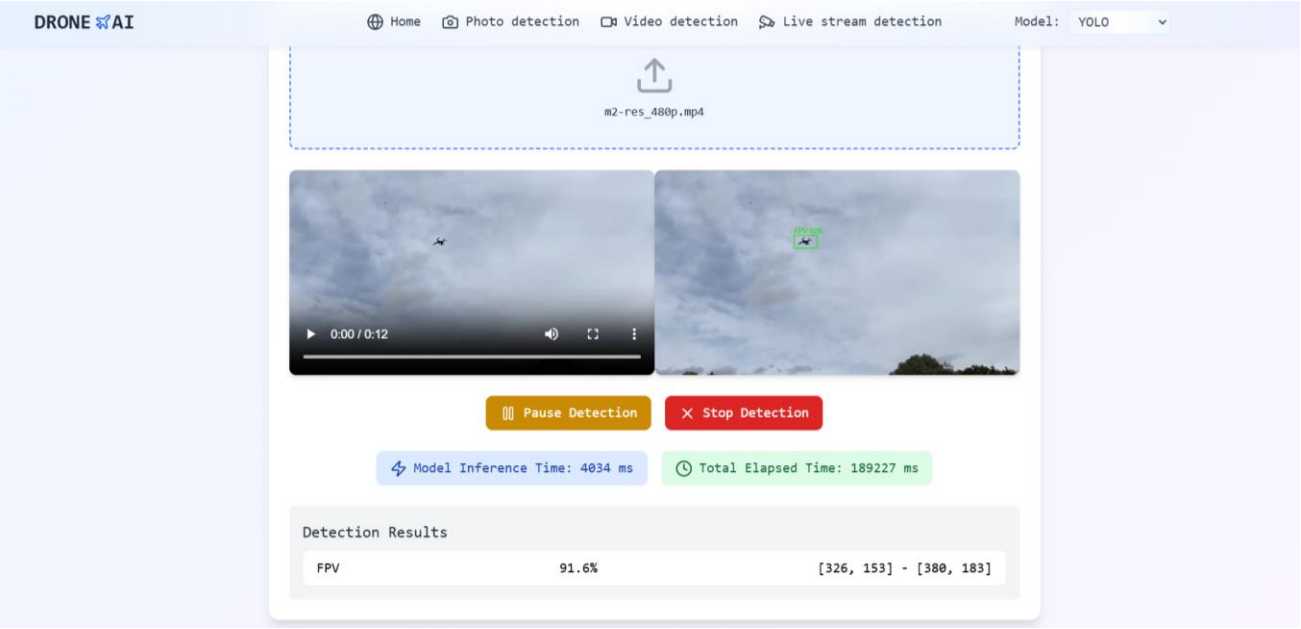

Рисунок 4.43 – Результат роботи системи з відео

Оскільки ONNX Runtime з WebAssembly використовує процесор (CPU), важливо відстежувати його завантаження, щоб зрозуміти, яку обчислювальну потужність модель споживає. При запуску ONNX-моделі для виявлення об’єктів у браузері з використанням ONNX Runtime можна отримати кілька ключових показників продуктивності (рис. 4.44). Час виконання означає – це час необхідний для обробки вхідних даних, таких як зображення або кадри з відео та для отримання вихідних прогнозів, які включають рамки обмежень, класи об’єктів та оцінки впевненості.

Рисунок 4.44 – Результат роботи системи з відео

Наведено результати виконання моделі в браузері для зображень різних класів (рис. 4.45).

Рисунок 4.45 – Приклади роботи системи з реальними зображеннями

Систему також можна встановити на пристрій (комп’ютер чи мобільний) як прогресивну вебпрограму (PWA) та запускати з головного екрана (рис. 4.46).

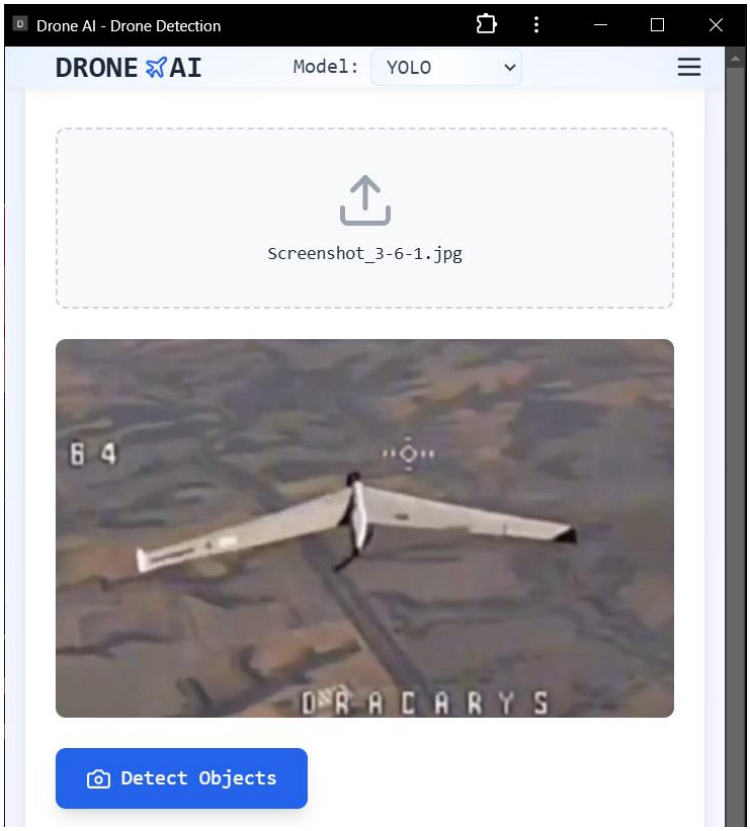

Рисунок 4.46 – PWA застосунок

Висновки до розділу 4

У результаті проведеної роботи був створений набір даних, охоплюючи різноманітні умови освітлення, траєкторії польоту дронів та текстур. Навчені моделі YOLO та Faster R-CNN показали високу точність на тестових зображеннях, ефективно розпізнаючи об’єкти в різних умовах. У реальних сценаріях моделі мають зниження впевненості моделі щодо класу об’єкта порівняно з тестовими зображеннями. Однак, моделі зберігають можливість розпізнавати та ідентифікувати об’єкти, що свідчить про її ефективність навіть в умовах реальних сценаріїв.

Моделі показали схожу поведінку на тестових зображеннях. YOLO перевершує Faster R-CNN за метриками точності, зокрема виявленням об’єктів із низьким IoU на тестових даних.

На реальних зображеннях модель Faster R-CNN демонструє вищу впевненість для ідентифікації дронів, особливо коли об’єкт є малим або перекритим.

Обидві моделі мають обмеження у вигляді хибнопозитивних результатів. Проте їхні переваги, зокрема висока точність та здатність адаптуватися до нових умов, роблять дані моделі ефективними для задач розпізнавання об’єктів у реальних сценаріях.

Створена система ідентифікації та розпізнавання об’єктів надає можливість завантажувати зображення та відео або здійснювати виявлення об’єктів у реальному часі за допомогою камери. Система дозволяє ідентифікувати та розпізнавати дрони у реальному часі без необхідності в серверних обчисленнях.

ВИСНОВКИ

У результаті було досягнуто поставленої мети – підвищення точності ідентифікації та розпізнавання дронів завдяки використанню сучасних методів штучного інтелекту за умов обмеженого доступу до якісних даних для навчання. Цього досягнуто завдяки створенню унікального набору даних із синтетичними зображеннями, який охоплює різноманітні умови освітлення, траєкторії польоту дронів та текстури. Такий підхід дозволив моделям ефективно працювати у різних сценаріях.

Було навчено моделі YOLO та Faster R-CNN, які продемонстрували високу точність на тестових та реальних зображеннях. Модель YOLO виявилася більш ефективною в тестових умовах, перевершивши Faster R-CNN за метриками точності. Водночас Faster R-CNN показала кращі результати на реальних зображеннях.

Проведено аналіз помилкових виявлень та оцінку переваг моделей. До основних переваг належать висока точність, здатність ефективно ідентифікувати та розпізнавати малі об’єкти, а також обробляти зображення, які суттєво відрізняються від представлених у навчальній вибірці.

Розроблена система працює на стороні клієнта, яка забезпечує ідентифікацію та розпізнавання дронів у реальному часі за допомогою відеофайлів або камер, без необхідності у використанні серверних обчислень. Створені інструменти для завантаження медіафайлів та візуалізації результатів роботи моделей.

Таким чином, усі поставлені завдання виконані в повному обсязі. Створена система забезпечує точну ідентифікацію та розпізнавання дронів, адаптована до роботи на різних пристроях, функціонує в реальному часі. Використання сучасних методів штучного інтелекту та розробка синтетичного набору даних дозволили створити ефективний інструмент для забезпечення безпеки та контролю повітряного простору.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. D. Rajawat, B. P. Lohani, A. Rana, A. Srivastava, P. Yadav and S. Gupta, "Object Detection in Images and Videos Using OpenCV: A Comparative Study of Deep Learning and Traditional Computer Vision Techniques," 2023 10th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON), Gautam Buddha Nagar, India, 2023, pp. 141-146, doi: 10.1109/UPCON59197.2023.10434536.

A. Rouhi et al., "Long-Range Drone Detection Dataset," 2024 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 2024, pp. 1-6, doi: 10.1109/ICCE59016.2024.10444135.

2. Z. Tan and M. Karaköse, "Survey and Comparative Study for Drone Detection Using Deep Learning," 2022 International Conference on Data Analytics for Business and Industry (ICDABI), Sakhir, Bahrain, 2022, pp. 234-238, doi: 10.1109/ICDABI56818.2022.10041658.

3. R. Valaboju, Vaishnavi, C. Harshitha, A. R. Kallam and B. S. Babu, "Drone Detection and Classification using Computer Vision," 2023 7th International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, 2023, pp. 1320-1328, doi: 10.1109/ICOEI56765.2023.10125737.

4. V. Mehta, F. Dadboud, M. Bolic and I. Mantegh, "A Deep Learning Approach for Drone Detection and Classification Using Radar and Camera Sensor Fusion," 2023 IEEE Sensors Applications Symposium (SAS), Ottawa, ON, Canada, 2023, pp. 01-06, doi: 10.1109/SAS58821.2023.10254123.

5. H. Lee et al., "CNN-Based UAV Detection and Classification Using Sensor Fusion," in IEEE Access, vol. 11, pp. 68791-68808, 2023, doi: 10.1109/ACCESS.2023.3293124.

6. H. Ahmad, M. Farhan, and U. Farooq, “Computer Vision Techniques for Military Surveillance Drones”, WJCMS, vol. 2, no. 2, pp. 53–59, Jun. 2023, doi: 10.31185/wjcms.148.

7. Shishir Kumar Shandilya, Aditya Srivastav, Kyrylo Yemets, Agni Datta, Atulya K. Nagar, YOLO-based segmented dataset for drone vs. bird detection for deep and machine learning algorithms, Data in Brief, Volume 50, 2023, 109355, ISSN 2352-3409, https://doi.org/10.1016/j.dib.2023.109355.

8. Almubairik, N.A., El-Alfy, ES.M. (2024). RF-Based Drone Detection with Deep Neural Network: Review and Case Study. In: Luo, B., Cheng, L., Wu, ZG., Li, H., Li, C. (eds) Neural Information Processing. ICONIP 2023. Communications in Computer and Information Science, vol 1969. Springer, Singapore. https://doi.org/10.1007/978-981-99-8184-7_2

9. F. Mahdavi and R. Rajabi, "Drone Detection Using Convolutional Neural Networks," 2020 6th Iranian Conference on Signal Processing and Intelligent Systems (ICSPIS), Mashhad, Iran, 2020, pp. 1-5, doi: 10.1109/ICSPIS51611.2020.9349620.

10. F. Dadboud, V. Patel, V. Mehta, M. Bolic and I. Mantegh, "Single-Stage UAV Detection and Classification with YOLOV5: Mosaic Data Augmentation and PANet," 2021 17th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Washington, DC, USA, 2021, pp. 1-8, doi: 10.1109/AVSS52988.2021.9663841.

11. Samadzadegan F, Dadrass Javan F, Ashtari Mahini F, Gholamshahi M. Detection and Recognition of Drones Based on a Deep Convolutional Neural Network Using Visible Imagery. Aerospace. 2022; 9(1):31. https://doi.org/10.3390/aerospace9010031.

12. Zitar, R.A., Mohsen, A., Seghrouchni, A.E. et al. Intensive Review of Drones Detection and Tracking: Linear Kalman Filter Versus Nonlinear Regression, an Analysis Case. Arch Computat Methods Eng 30, 2811–2830 (2023). https://doi.org/10.1007/s11831-023-09894-0

13. Ashraf, M. W., Sultani, W., & Shah, M. (2021). Dogfight: Detecting drones from drones videos. arXiv. https://arxiv.org/abs/2103.17242

14. Dadrass Javan F, Samadzadegan F, Gholamshahi M, Ashatari Mahini F. A Modified YOLOv4 Deep Learning Network for Vision-Based UAV Recognition. Drones. 2022; 6(7):160. https://doi.org/10.3390/drones6070160

15. S. Scholes, A. Ruget, G. Mora-Martín, F. Zhu, I. Gyongy and J. Leach, "DroneSense: The Identification, Segmentation, and Orientation Detection of Drones via Neural Networks," in IEEE Access, vol. 10, pp. 38154-38164, 2022, doi: 10.1109/ACCESS.2022.3162866.

16. Sucharitha, Y., Reddy, P. C. S., & Suryanarayana, G. (2023). Network intrusion detection of drones using recurrent neural networks. In Drone Technology (Chap. 15, pp. 375–392). John Wiley & Sons. https://doi.org/10.1002/9781394168002.ch15

17. Li Y, Fan Q, Huang H, Han Z, Gu Q. A Modified YOLOv8 Detection Network for UAV Aerial Image Recognition. Drones. 2023; 7(5):304. https://doi.org/10.3390/drones7050304

18. D. T. Wei Xun, Y. L. Lim and S. Srigrarom, "Drone detection using YOLOv3 with transfer learning on NVIDIA Jetson TX2," 2021 Second International Symposium on Instrumentation, Control, Artificial Intelligence, and Robotics (ICA-SYMP), Bangkok, Thailand, 2021, pp. 1-6, doi: 10.1109/ICA-SYMP50206.2021.9358449.

19. L. Pei, G. Cheng, X. Sun, Q. Li, M. Zhang and S. Miao, "Multi-Scale Bidirectional Feature Fusion for One-Stage Oriented Object Detection in Aerial Images," 2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS, Brussels, Belgium, 2021, pp. 2592-2595, doi: 10.1109/IGARSS47720.2021.9555142.

20. O. Sahin and S. Ozer, "YOLODrone: Improved YOLO Architecture for Object Detection in Drone Images," 2021 44th International Conference on Telecommunications and Signal Processing (TSP), Brno, Czech Republic, 2021, pp. 361-365, doi: 10.1109/TSP52935.2021.9522653.

21. Z. Wang, Y. Cao and J. Li, "A Detection Algorithm Based on Improved Faster R-CNN for Spacecraft Components," 2023 IEEE International Conference on Image

Processing and Computer Applications (ICIPCA), Changchun, China, 2023, pp. 1-5, doi: 10.1109/ICIPCA59209.2023.10257992.

22. M. S. Jagadeesh, A. S. Veliyathuparamban, N. K. V. Manasa, A. S. Menon, B. N. Jagadesh and D. R. Kumar Raja, "A Unified Approach for Weed Detection in Arable Acreage Using RetinaNet Architecture," 2024 2nd International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT), Bengaluru, India, 2024, pp. 641-647, doi: 10.1109/IDCIoT59759.2024.10467578.

23. D. A. Navastara, N. Musyafira, C. Fatichah and S. Maharani, "Video-Based License Plate Recognition Using Single Shot Detector and Recurrent Neural Network," 2021 13th International Conference on Information & Communication Technology and System (ICTS), Surabaya, Indonesia, 2021, pp. 151-154, doi: 10.1109/ICTS52701.2021.9608790.

24. L. C. Adi and T. M. Cheng, "Analysis of Impact of Synthetic Image Data with Multiple Randomization Strategies on Object Detection Performance," 2022 IEEE 4th Eurasia Conference on IOT, Communication and Engineering (ECICE), Yunlin, Taiwan, 2022, pp. 206-210, doi: 10.1109/ECICE55674.2022.10042887.

25. N. Reddy Nandyala and R. Kumar Sanodiya, "Underwater Object Detection Using Synthetic Data," 2023 11th International Symposium on Electronic Systems Devices and Computing (ESDC), Sri City, India, 2023, pp. 1-6, doi: 10.1109/ESDC56251.2023.10149870.

26. S. D. N, R. M. Pai, S. N. Bhat and M. P. M. M, "Synthetic Vertebral Column Fracture Image Generation by Deep Convolution Generative Adversarial Networks," 2021 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT), Bangalore, India, 2021, pp. 1-4, doi: 10.1109/CONECCT52877.2021.9622527.

27. B. Balakrishnan, R. Chelliah, M. Venkatesan and C. Sah, "Comparative Study On Various Architectures Of Yolo Models Used In Object Recognition," 2022 International Conference on Computing, Communication, and Intelligent Systems

(ICCCIS), Greater Noida, India, 2022, pp. 685-690, doi: 10.1109/ICCCIS56430.2022.10037635.

28. Y. Zhao, "Design and Implementation of Pedestrian Target Tracking System Based on YOLO Architecture," 2023 International Conference on Artificial Intelligence and Automation Control (AIAC), Xiamen, China, 2023, pp. 368-373, doi: 10.1109/AIAC61660.2023.00024.

29. Z. Sun, "Research on the Application of Convolutional Neural Network Based on the YOLO Algorithm in Airport Intelligent Monitoring," 2023 3rd Asia-Pacific Conference on Communications Technology and Computer Science (ACCTCS), Shenyang, China, 2023, pp. 103-108, doi: 10.1109/ACCTCS58815.2023.00071.

30. T. Sriranjani, K. P. Senthil Kumar and G. Thiruksha, "Varicose Vein Detection and Real-Time Integration using Faster R-CNN Algorithm," 2024 Third International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS), Krishnankoil, Virudhunagar district, Tamil Nadu, India, 2024, pp. 01-06, doi: 10.1109/INCOS59338.2024.10527750.

31. N. Rafi, Z. Zainuddin and I. Nurtanio, "Betta Fish Classification Using Faster R-CNN Approach with Multi-Augmentation," 2024 International Seminar on Intelligent Technology and Its Applications (ISITIA), Mataram, Indonesia, 2024, pp. 500-505, doi: 10.1109/ISITIA63062.2024.10668167.

A. Darnilasari, Indrabayu and I. S. Areni, "Implementation of Faster R-CNN with Colour and Blur Augmentation For Differentiate Cloves From Debris," 2023 2nd International Conference on Computer System, Information Technology, and Electrical Engineering (COSITE), Banda Aceh, Indonesia, 2023, pp. 72-77, doi: 10.1109/COSITE60233.2023.10249515.

32. S. Y. Kim, J. Lee, C. H. Kim, W. J. Lee and S. W. Kim, "Extending the ONNX Runtime Framework for the Processing-in-Memory Execution," 2022 International Conference on Electronics, Information, and Communication (ICEIC), Jeju, Korea, Republic of, 2022, pp. 1-4, doi: 10.1109/ICEIC54506.2022.9748444.

33. ONNX Runtime Web. URL: https://onnxruntime.ai/docs/build/web.html (дата звернення: 21.10.2024).

34. M. Garofalo, M. Colosi, A. Catalfamo and M. Villari, "Web-Centric Federated Learning over the Cloud-Edge Continuum Leveraging ONNX and WASM," 2024 IEEE Symposium on Computers and Communications (ISCC), Paris, France, 2024, pp. 1-7, doi: 10.1109/ISCC61673.2024.10733614.

35. C. Chen et al., "A Unified Interactive Model Evaluation for Classification, Object Detection, and Instance Segmentation in Computer Vision," in IEEE Transactions on Visualization and Computer Graphics, vol. 30, no. 1, pp. 76-86, Jan. 2024, doi: 10.1109/TVCG.2023.3326588.

36. Kho, Dickson & Sazali, Norazlianie & Kettner, Maurice & Friedrich, Christian & Schempp, Constantin & Salim, Naqib & Ishak, Ismayuzri & Ghani, Saiful. (2024). Synthetic Image Data Generation via Rendering Techniques for Training AI-Based Instance Segmentation. Journal of Advanced Research in Applied Sciences and Engineering Technology. 158-169. 10.37934/araset.62.1.158169.

37. Projecting 3D Points into a 2D Screen. URL: https://skannai.medium.com/projecting-3d-points-into-a-2d-screen-58db65609f24 (date of access: 21.10.2024).

38. Man K, Chahl J. A Review of Synthetic Image Data and Its Use in Computer Vision. Journal of Imaging. 2022; 8(11):310. https://doi.org/10.3390/jimaging8110310.

39. H. Azad, V. Mehta, F. Dadboud, M. Bolic and I. Mantegh, "Air-to-Air Simulated Drone Dataset for AI-powered problems," 2023 IEEE/AIAA 42nd Digital Avionics Systems Conference (DASC), Barcelona, Spain, 2023, pp. 1-7, doi: 10.1109/DASC58513.2023.10311339.

40. A. Rouhi et al., "Long-Range Drone Detection Dataset," 2024 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 2024, pp. 1-6, doi: 10.1109/ICCE59016.2024.10444135.

41. K. V. S. Reddy, V. R. Molabanti, S. T. P. Dumpala and U. R. Nelakuditi, "YOLOV3 based Real Time Drone Detection for Counter Drone System," 2023 IEEE

3rd International Conference on Technology, Engineering, Management for Societal impact using Marketing, Entrepreneurship and Talent (TEMSMET), Mysuru, India, 2023, pp. 1-5, doi: 10.1109/TEMSMET56707.2023.10149935.

42. YOLOv10: Real-Time End-to-End Object Detection. URL: https://docs.ultralytics.com/models/yolov10/ (дата звернення: 20.11.2024).

43. Model Training with Ultralytics YOLO. URL: https://docs.ultralytics.com/modes/train/ (дата звернення: 21.11.2024).

44. YOLO Performance Metrics. URL: https://docs.ultralytics.com/guides/yolo-performance-metrics/ (дата звернення: 22.11.2024).

45. Faster R-CNN models. URL: https://pytorch.org/vision/main/models/faster_rcnn.html (дата звернення: 25.11.2024).

ДОДАТОК А

Апробація роботи

Робота пройшла апробацію під час Всеукраїнської науково-практичної конференції молодих вчених, аспірантів і студентів (рис. А.1), 2-4 грудня 2024 р., м. Миколаїв.

Міністерство освіти і науки України
Чорноморський національний університет ім. Петра Могили
Факультет комп'ютерних наук
Кафедра інтелектуальних інформаційних систем

Інформаційний лист

Всеукраїнська науково-практична конференція молодих вчених, аспірантів і студентів

Інтелектуальні інформаційні системи

2 – 4 грудня 2024 року

Миколаїв

УДК 004.93

Данкович С. Ю.
магістрант,
Сіденко Є. В.
канд. техн. наук., доцент,
ЧНУ імені Петра Могили, Миколаїв, Україна

СИСТЕМА ІДЕНТИФІКАЦІЇ ТА РОЗПІЗНАВАННЯ ДРОНІВ МЕТОДАМИ ШТУЧНОГО ІНТЕЛЕКТУ

В доповіді розглядається розробка системи для ідентифікації та розпізнавання дронів з використанням методів штучного інтелекту. Система реалізована у вигляді вебзастосунку, що включає модель, яка навчена на спеціально створеному наборі даних. Основною метою системи є забезпечення швидкого та точного виявлення дронів у різних середовищах.

Одним із основних методів штучного інтелекту, застосовуваних для отримання інформації з візуальних даних є комп'ютерний зір. Він використовується для виявлення дронів на основі моделей, навчених на розмічених зображеннях, або з використанням глибоких нейронних мереж, які дозволяють виділяти специфічні ознаки дронів [1]. Дрони мають різноманітні форми та функціональні особливості, включаючи багатороторні безпілотні літальні апарати та дрони з нерухомими крилами. Їхня різноманітність ускладнює процес розпізнавання [2].

Однією з основних проблем, яку вирішує система, полягає в тому, що дрони є рухомими об'єктами, які можуть з'являтися в різних умовах, таких як погодні умови, різний час доби та складні середовища, зокрема міські райони або ліси. Відсутність якісних наборів даних у відкритому доступі та обмеженість ресурсів для створення власних наборів також значно ускладнює розробку надійних систем для розпізнавання дронів [3].

Розроблена система представлена у вигляді вебзастосунку (рис. 1).

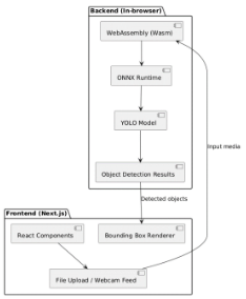

Рисунок 1 – Система ідентифікації та розпізнання дронів

Рисунок А.1 – Апробація роботи

ДОДАТОК Б

Лістинг програмного коду

```
import { useState, useEffect } from "react";
import ndarray from "ndarray";
import ops from "ndarray-ops";
import { Tensor } from "onnxruntime-web";
import PhotoUploadDetection from "./PhotoUploadDetection";
import VideoUploadDetection from "./VideoUploadDetection";
import CameraDetection from "./CameraDetection";
import { droneClasses } from "../data/drone_classes";
import { runModelUtils } from "../utils";

const RES_TO_MODEL: [number[], string][] = [
  [[320, 320], "yolov10m200e.onnx"],
  [[640, 640], "yolov10m200e.onnx"],
  [[640, 640], "fasterrcnn70e.onnx"],
];

const COMPONENTS = {
  photo: PhotoUploadDetection,
  video: VideoUploadDetection,
  camera: CameraDetection,
};

const Models = (props: any) => {
   const [modelResolution, setModelResolution] = useState<number[]>(
    RES_TO_MODEL[props.modelIndex][0]
  );

  const [modelName, setModelName] = useState<string>(
    RES_TO_MODEL[props.modelIndex][1]
  );

  const [session, setSession] = useState<any>(null);
  const [detectionResults, setDetectionResults] = useState<
    Array<{
      className: string;
      score: number;
      boundingBox: [number, number, number, number];
    }>
  >([]);

  useEffect(() => {
    const getSession = async () => {
      const session = await runModelUtils.createModelCpu(
        `./_next/static/chunks/pages/${modelName}`
```

```
    );
    setSession(session);
  };
  getSession();
}, [modelName, modelResolution]);

const resizeCanvasCtx = (
  ctx: CanvasRenderingContext2D,
  targetWidth: number,
  targetHeight: number,
  inPlace = false
) => {
  let canvas: HTMLCanvasElement;

  if (inPlace) {
    canvas = ctx.canvas;
    canvas.width = targetWidth;
    canvas.height = targetHeight;
    ctx.scale(
      targetWidth / canvas.clientWidth,
      targetHeight / canvas.clientHeight
    );
  } else {
    canvas = document.createElement("canvas");
    canvas.width = targetWidth;
    canvas.height = targetHeight;
    canvas
      .getContext("2d")!
      .drawImage(ctx.canvas, 0, 0, targetWidth, targetHeight);
    ctx = canvas.getContext("2d")!;
  }

  return ctx;
};

const preprocess = (ctx: CanvasRenderingContext2D) => {
  const resizedCtx = resizeCanvasCtx(
    ctx,
    modelResolution[0],
    modelResolution[1]
  );

  const imageData = resizedCtx.getImageData(
    0,
    0,
    modelResolution[0],
    modelResolution[1]
  );
  const { data, width, height } = imageData;
```

```
  const dataTensor = ndarray(new Float32Array(data), [width, height, 4]);
  const dataProcessedTensor = ndarray(new Float32Array(width * height * 3), [
    1,
    3,
    width,
    height,
  ]);

  ops.assign(
    dataProcessedTensor.pick(0, 0, null, null),
    dataTensor.pick(null, null, 0)
  );
  ops.assign(
    dataProcessedTensor.pick(0, 1, null, null),
    dataTensor.pick(null, null, 1)
  );
  ops.assign(
    dataProcessedTensor.pick(0, 2, null, null),
    dataTensor.pick(null, null, 2)
  );

  ops.divseq(dataProcessedTensor, 255);

  const tensor = new Tensor("float32", new Float32Array(width * height * 3), [
    1,
    3,
    width,
    height,
  ]);

  (tensor.data as Float32Array).set(dataProcessedTensor.data);
  return tensor;
};

const conf2color = (conf: number) => {
  const r = Math.round(255 * (1 - conf));
  const g = Math.round(255 * conf);
  return `rgb(${r},${g},0)`;
};

const postprocess = async (
  tensor: Tensor,
  inferenceTime: number,
  ctx: CanvasRenderingContext2D,
  modelName: string
) => {
  const results = [];
  const dx = ctx.canvas.width / modelResolution[0];
```

```
  const dy = ctx.canvas.height / modelResolution[1];

  for (let i = 0; i < tensor.dims[1]; i += 6) {
    const [x0, y0, x1, y1, score, cls_id] = tensor.data.slice(i, i + 6);

    if (score < 0.5) continue;

    const scaledX0 = x0 * dx;
    const scaledY0 = y0 * dy;
    const scaledX1 = x1 * dx;
    const scaledY1 = y1 * dy;

    results.push({
      className: droneClasses[cls_id],
      score: Math.round(score * 100 * 10) / 10,
      boundingBox: [
        Math.round(scaledX0),
        Math.round(scaledY0),
        Math.round(scaledX1),
        Math.round(scaledY1),
      ],
    });

    const color = conf2color(score);
    ctx.strokeStyle = color;
    ctx.lineWidth = 2;
    ctx.strokeRect(
      scaledX0,
      scaledY0,
      scaledX1 - scaledX0,
      scaledY1 - scaledY0
    );

    ctx.font = "16px Arial";
    ctx.fillStyle = color;
    ctx.fillText(
      `${droneClasses[cls_id]} ${Math.round(score * 100)}%`,
      scaledX0,
      scaledY0 - 5
    );
  }

  setDetectionResults(results);
};

const Component = COMPONENTS[props.mode];

return Component ? (
  <Component
```

```
      width={props.width}
      height={props.height}
      preprocess={preprocess}
      postprocess={postprocess}
      session={session}
      detectionResults={detectionResults}
      currentModelResolution={modelResolution}
      modelName={modelName}
    />
  ) : (
    <div>Invalid mode</div>
  );
};

export default Models;
```