Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютеризованих технологій
(повна назва)

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

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

другий (магістерський)
(рівень вищої освіти)

Розроблення програмного нейромережевого модуля для виявлення дронів на основі YoloV5
(тема)

Виконав:
здобувач 2 року навчання,
групи КТРСм-23-2

Сагула Олександр Олександрович

Спеціальності 174 Автоматизація, комп'ютерно-інтегровані технології та робототехніка

Тип програми Освітньо-професійна

Освітня програма Комп'ютеризовані та робототехнічні системи

Керівник доц. Демська Н. П.

Допускається до захисту
Зав. кафедри КІТАР _____ Невлюдов І. Ш.
(підпис) (прізвище, ініціали)

2025р.

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Факультет АКТ
Кафедра КІТАР
Рівень вищої освіти другий (магістерський)
Спеціальність 174 Автоматизація, комп'ютерно-інтегровані технології та робототехніка
Тип програми Освітньо-професійна
Освітня програма Комп'ютеризовані та робототехнічні системи
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри КІТАР _____
(підпис)
« ___» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Сагулі Олександру Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розроблення програмного нейромережевого модуля для виявлення дронів на основі YoloV5

Затверджена наказом по університету від 25.11.2024 р. № 1239 Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 20.01.2025 р.

3. Вихідні дані до роботи
3.1 Для навчання моделі використовуємо датасет, який складається з 1012 зображень для тренувальної вибірки та 347 фотографій
3.2 Використане програмне забезпечення: мова програмування Python; бібліотека OpenCV; бібліотеки TensorFlow, PyTorch; бібліотека NumPy;
3.3 Для розробки та тестування моделей буде використовуватися VSCode як основне середовище для програмування;
3.4 Для обчислювальної потужності та проведення експериментів використано сервіс Google Colab

4. Перелік питань, що потрібно опрацювати в роботі
4.1 Вступ; 4.2 Аналіз технічного завдання та постановка задач дослідження; 4.3 Підготовка навчального набору даних; 4.4 Налаштування середовища Google Colab; 4.5 Навчання моделі; 4.6 Аналіз результатів навчання моделі; 4.7 Валідація отриманої моделі та аналіз даних; 4.8 Реалізація нейромережевого модуля; 4.9 Інтеграція навченої моделі у застосунок; 4.10 Тестування нейромережевого модуля

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій

Графічний матеріал у вигляді презентації – 15 арк. ф. А 4

6. Консультанти розділів роботи

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

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Приміт-ка
1	Аналіз існуючих технологій для виявлення дронів та вибір оптимальної технології	01.09.24-25.09.24	виконано
2	Підготовка навчального набору даних та налаштування середовища для навчання моделі	26.09.24-25.10.24	виконано
3	Навчання моделі YOLOv5 та валідація результатів	26.10.24-25.11.24	виконано
4	Інтеграція навченої моделі у застосунок для виявлення дронів у реальному часі	26.11.24-25.12.24	виконано
5	Тестування системи, оптимізація та підготовка висновків	26.12.24-16.01.25	виконано
6	Подання роботи на перевірку Інтернет-сервісом StrikePlagiarism	17.01.25	виконано
7	Оформлення пояснювальної записки	18.01.25	виконано
8	Подання роботи на рецензію	19.01.25	виконано
9	Подання роботи на підпис зав. кафедри	20.01.25	виконано
11	Подання кваліфікаційної роботи в ЕК	21.01.25	виконано

Дата видачі завдання 01.09.2024 р.

Здобувач _____Сагула Олександр Олександрович
(підпис)

Керівник роботи _____ доц. Демська Н. П.
(підпис) (посада, прізвище, ініціали)

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

18 січня 2025 р.  Сагула О. О.

РЕФЕРАТ

Пояснювальна записка: 67 с., 31 рис., 3 дод., 22 джерела.

КОМП'ЮТЕРНИЙ ЗІР, НЕЙРОМЕРЕЖЕВІ МОДУЛІ, РАДІОЛОКАЦІЙНІ СИСТЕМИ, ВІДЕОСПОСТЕРЕЖЕННЯ, АКУСТИЧНІ СИСТЕМИ, ГЛИБОКЕ НАВЧАННЯ, ОБРОБКА ЗОБРАЖЕНЬ.

Об’єкт дослідження – процес автоматичного виявлення дронів у відеопотоці, який включає методи комп'ютерного зору та глибокого навчання для ідентифікації об’єктів на основі нейронних мереж. Виявлення дронів є важливим етапом у забезпеченні безпеки стратегічних об’єктів, а також для моніторингу повітряного простору.

Предмет дослідження – програмний модуль, що реалізує архітектуру нейронної мережі для виявлення дронів у відеопотоці в реальному часі. Зокрема, досліджуються методи налаштування нейромережевої архітектури для точного виявлення малорозмірних та швидкорухомих об’єктів (дронів), способи оптимізації моделі для підвищення продуктивності, зменшення кількості хибнопозитивних спрацьовувань, а також забезпечення високої швидкості обробки даних.

Метою роботи є створення нейромережевого модуля на основі архітектури YOLOv5 для моніторингу повітряного простору та забезпечення безпеки об’єктів, який зможе оперативно реагувати на появу дронів, зокрема у випадках несанкціонованого проникнення.

В першому розділі проведено детальний аналіз існуючих методів виявлення дронів. Розглянуто пульсову допплерівську радіолокацію, безперервні хвильові радари, відеоспостереження з комп'ютерним зором та акустичні системи. Було встановлено, що комп'ютерний зір є найбільш

ефективним для задачі виявлення дронів завдяки високій точності та адаптивності.

В другому розділі було виконано аналіз технічного завдання та сформульовано задачі дослідження. Було визначено ключові вимоги до системи виявлення дронів, що зумовило вибір підходу на основі комп'ютерного зору.

В третьому розділі описано процес навчання моделі YOLOv5 для виявлення дронів. Проведено підготовку навчального набору даних, налаштовано середовище Google Colab, виконано навчання та валідацію моделі. Отримані результати продемонстрували високу точність моделі при виявленні дронів.

У результаті розроблена система виявлення дронів на основі нейронної мережі, інтегрована в програмний додаток. Після цього система була успішно протестована.

Також, отримані результати роботи можна віднести до цілей сталого розвитку 9 «Промисловість, інновації та інфраструктура», а саме до пункту 9.4 «Сприяти прискореному розвитку високо- та середньо- високотехнологічних секторів переробної промисловості, які формуються на основі використання ланцюгів «освіта – наука – виробництво» та кластерного підходу за напрямами: розвиток інформаційно-телекомунікаційних технологій (ІКТ); застосування ІКТ в АПК, енергетиці, транспорті та промисловості; високотехнологічне машинобудування».

ABSTRACT

Explanatory note: 67 p., 31 figures, 3 appendices, 22 sources.

COMPUTER VISION, NEURAL NETWORK MODULES, RADAR SYSTEMS, VIDEO SURVEILLANCE, ACOUSTIC SYSTEMS, DEEP LEARNING, IMAGE PROCESSING.

The object of study is the process of automatic drone detection in a video stream, which includes computer vision and deep learning methods for object identification based on neural networks. Drone detection is an important step in ensuring the security of strategic facilities, as well as for airspace monitoring.

The subject of the research is a software module that implements the architecture of a neural network for detecting drones in a real-time video stream. In particular, the methods of tuning the neural network architecture for accurate detection of small and fast-moving objects (drones), ways to optimize the model to improve performance, reduce the number of false positives, and ensure high data processing speed are investigated.

The aim of the work is to develop a neural network module based on the YOLOv5 architecture capable of detecting drones in a real-time video stream. This will create a tool for airspace monitoring and facility security that will be able to respond quickly to the appearance of drones, in particular in cases of unauthorized intrusion.

The first section provides a detailed analysis of existing drone detection methods. Pulse Doppler radar, continuous wave radar, video surveillance with computer vision, and acoustic systems are considered. It was found that computer vision is the most effective for drone detection due to its high accuracy and adaptability.

The second section analyzes the terms of reference and formulates the

research objectives. The key requirements for a drone detection system were identified, which led to the choice of a computer vision-based approach.

Section 3 describes the process of training the YOLOv5 model for drone detection. The training dataset was prepared, the Google Colab environment was set up, and the model was trained and validated. The results demonstrated the high accuracy of the model in detecting drones.

As a result, a neural network-based drone detection system was developed and integrated into a software application. The system was then successfully tested.

Also, the results of the work can be attributed to the Sustainable Development Goals 9 "Industry, Innovation and Infrastructure", namely to item 9.4 "Promote the accelerated development of high- and medium-high-tech sectors of the processing industry, which are formed on the basis of the use of the "education – science – production" chains and a cluster approach in the following areas: development of information and telecommunications technologies (ICT); application of ICT in agro-industrial complex, energy, transport and industry; high-tech mechanical engineering."

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

БПЛА – безпілотний літальний апарат;

ПДР – пульсова допплерівська радіолокація;

CUDA – Compute Unified Device Architecture;

CW – Continuous Wave;

FPS – кадри в секунду;

GFLOPS – Giga Floating Point Operations Per Second;

GHz – гігагерц;

GPU – графічний процесор;

IoU – перетин на одиницю;

mAP – середня точність;

MP – змішана точність;

NMS – невід'ємне подавлення;

P – точність;

PD – Pulsed Doppler;

R – відгук, повнота;

SP – одинична точність;

TFLOPS – Tera Floating Point Operations Per Second;

TPU – тензорний процесор;

YOLO – You Only Look Once.

ВСТУП

Сучасний світ постійно стикається з новими викликами у сфері безпеки. Однією з головних загроз стали безпілотні летальні апарати (дрони), які широко використовуються для різноманітних цілей, що вимагає постійного удосконалення технік виявлення та знешкодження небезпечних об'єктів. Особливо актуальним це стає у військовій та цивільній сферах, де використання дронів створює ризики для громадської безпеки [1].

У зв'язку з цим виникає потреба у розробленні ефективних інструментів для виявлення та спостереження за дронами. Особливою увагою користуються технології нейромереж, які зарекомендували себе як ефективні та достовірні засоби автоматизованого виявлення та класифікації об'єктів.

Метою даної кваліфікаційної роботи є розроблення програмного нейромережевого модуля для виявлення дронів на основі технології YOLOv5. Даний модуль повинен забезпечити швидке та точне виявлення дронів з метою їх нейтралізації та забезпечення безпеки у різних сферах діяльності.

У роботі буде здійснено аналіз існуючих технік виявлення дронів, проведено обгрунтування вибору нейромережі YOLOv5 та розроблено прототип системи, що демонструє її ефективність в різних умовах.

Для розроблення прототипу системи необхідно навчити модель на основі наборів даних (датасету), та оцінити отриману навчену модель на тестовому наборі даних використовуючи такі метрики, як середня точність (mAP), точність (Precision або P), пригадування (Recall або R), та оцінку F1.

Мета нейромережевого модуля для виявлення дронів включає такі ключові аспекти:

– швидке та точне виявлення дронів;

– автоматизація процесу виявлення;

– адаптивність до різних умов;

– інтеграція з іншими системами безпеки;

– масштабованість та продуктивність.

Таким чином, метою роботи є створення нейромережевого модуля на основі архітектури YOLOv5 для моніторингу повітряного простору та забезпечення безпеки об’єктів, який зможе оперативно реагувати на появу дронів, зокрема у випадках несанкціонованого проникнення

Об’єктом дослідження є процеси автоматичного виявлення дронів за допомогою нейромережевих моделей глибокого навчання.

Предметом дослідження є програмний нейромережевий модуль для виявлення дронів на основі YOLOv5, його архітектура, алгоритми обробки зображень та методи навчання нейронної мережі.

Для досягнення поставленої мети необхідно виконати наступні завдання:

– провести аналіз сучасних методів та технологій для виявлення дронів за допомогою нейромереж;

– дослідити архітектуру YOLOv5 та визначити її переваги для задачі виявлення дронів;

– зібрати та підготувати навчальний датасет із зображеннями дронів у різних умовах;

– налаштувати та навчити модель YOLOv5 на підготовленому датасеті;

– провести тестування та оцінку ефективності розробленого модуля, включаючи аналіз точності, швидкості виявлення та рівня помилкових спрацьовувань;

– інтегрувати модель у програмне забезпечення, використовуючи мову python;

– протестувати отриманий застосунок та розробити рекомендації щодо подальшого вдосконалення та адаптації під різні умови експлуатації.

Робота виконана згідно [1-2]. Результати роботи опубліковані в [3].

1 АНАЛІЗ ІСНУЮЧИХ СПОСОБІВ ВИЯВЛЕННЯ ДРОНІВ

1.1 Використання радіолокаційних систем

Радіолокаційні системи широко використовуються для виявлення та ідентифікації дронів у сучасних оборонних системах та цивільному авіаційному контролю. Радіолокаційні системи базуються на принципі емісії і прийому радіохвиль для визначення місцезнаходження та характеристики цілей.

Для виявлення дронів використовуються різні типи радарів, зокрема:

– Pulsed Doppler (PD) радари;

– Continuous Wave (CW) радари.

1.1.1 Пульсова допплерівська (PD) радіолокація

Пульсова доплерівська радіолокація (ПДР) є одним із ефективних методів виявлення безпілотних літальних апаратів (БПЛА). Цей метод використовує принципи пульсової радіолокації в поєднанні з доплерівським ефектом для визначення не лише наявності, а й швидкості руху об'єктів [4-5]. Блок діаграма PD радару наведена на рис. 1.1.

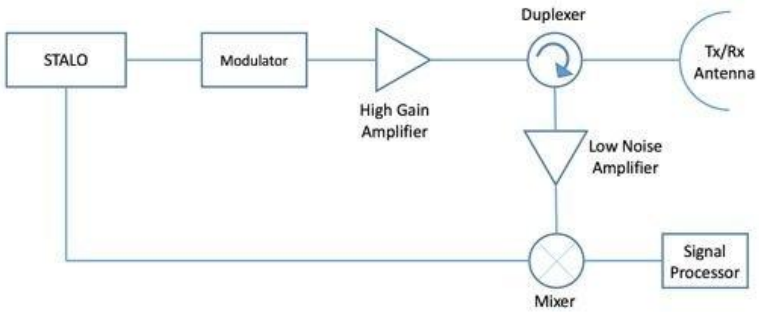

Рисунок 1.1 – Блок діаграма PD радару

Принцип роботи ПДР:

– радар посилає короткі імпульси радіохвиль у напрямку до цілі (БПЛА),

коли імпульс відбивається від об'єкта, він повертається назад до радару;

– якщо БПЛА рухається, частота повернутого сигналу змінюється в залежності від швидкості та напрямку руху дрону. Це зміщення частоти (доплерівський зсув) дозволяє радару визначити швидкість дрону;

– сигнали, що повернулися, обробляються за допомогою спеціальних алгоритмів, які аналізують характеристики відбитих хвиль. Це дозволяє не лише виявити об'єкт, але й оцінити його траєкторію та швидкість [6].

Пульсова доплерівська радіолокація має кілька ключових переваг, які роблять її ефективним інструментом для виявлення БПЛА. По-перше, вона здатна виявляти дрони на великих відстанях, що забезпечує своєчасне попередження про можливі загрози. По-друге, метод є стійким до зовнішніх завад, таких як погані погодні умови або інші радіосигнали, оскільки здатний фільтрувати шум і акцентувати увагу на реальних доплерівських змінах. Крім того, здатність визначати швидкість і напрямок руху БПЛА дозволяє проводити детальний моніторинг його траєкторії.

Попри численні переваги, пульсова доплерівська радіолокація має й свої недоліки. Одним із основних є складність у розробці та впровадженні таких систем, що може вимагати значних фінансових та ресурсних витрат. Крім того, точність виявлення може знижуватися, якщо БПЛА рухається дуже повільно або зависає, оскільки в таких випадках доплерівський ефект стає менш помітним. Це обмежує можливості системи в ситуаціях, коли дрон не має помітної швидкості, що може стати проблемою для безперервного моніторингу повітряного простору.

1.1.2 Безперервні хвильові (CW) радари

Безперервні хвильові радари (Continuous Wave, CW) – це тип радіолокаційних систем, які використовують постійно генеровані радіохвилі для виявлення і відстеження об'єктів. На відміну від пульсових радарів, які посилають імпульси радіохвиль, CW радари випромінюють сигнал постійно, що дозволяє їм отримувати інформацію про об'єкти в режимі реального часу.

CW радари працюють на основі генерації безперервного радіосигналу, який відбивається від об'єктів (наприклад, літаків або дронів) і повертається назад до приймальної антени. Зміна частоти повернутого сигналу, що виникає внаслідок доплерівського ефекту (якщо об'єкт рухається), дозволяє визначити швидкість об'єкта. Для вимірювання відстані до об'єкта часто використовуються технології фазової різниці або імпульсного модулювання. Блок діаграму CW радару наведено на рис. 1.2.

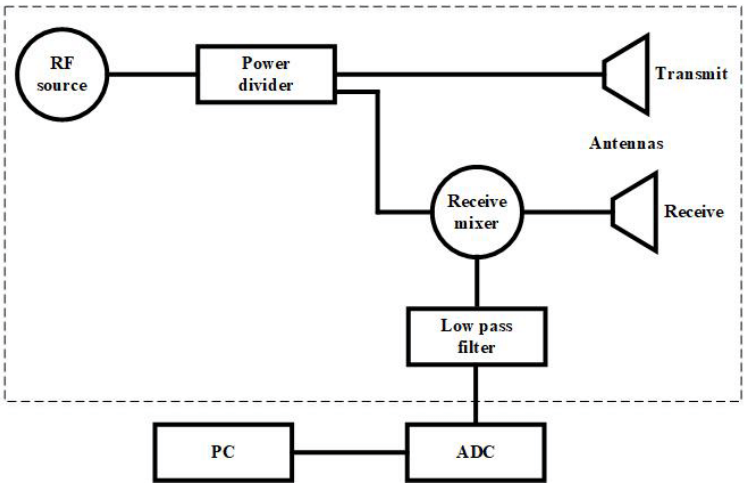

Рисунок 1.2 – Блок діаграма CW радару [7]

Безперервні хвильові радари (CW) забезпечують високу чутливість до малих швидкостей, що дозволяє їм виявляти навіть повільно рухомі об'єкти, такі як безпілотники. Вони також здійснюють моніторинг у реальному часі, постійно передаючи сигнал і оперативно реагуючи на зміни в навколишньому середовищі.

Однак CW радари мають і свої недоліки. По-перше, вони стикаються з обмеженнями у визначенні відстані до об'єкта, оскільки не використовують імпульсну техніку, що може вимагати додаткових методів для точного вимірювання. По-друге, їхня постійна генерація сигналів робить їх вразливими до радіозавад і перешкод. Також, для точного оцінювання відстані CW радари часто потребують інтеграції з іншими системами, що ускладнює їх використання.

1.2 Відеоспостереження та комп'ютерний зір

Відеоспостереження та комп'ютерний зір є важливими технологіями для виявлення дронів у сучасних системах безпеки.

Технології відеоспостереження та комп'ютерного зору для виявлення дронів використовують наступні методи:

– комбінація статичних ширококутних камер та камер з меншим кутом огляду на поворотній башті;

– аналіз зображень та відеопотоків у реальному часі для виявлення аномалій та ідентифікації об'єктів;

– алгоритми глибокого навчання для класифікації зображень, виявлення об'єктів та розуміння сцени;

– сегментація та відстеження руху об'єктів для точного визначення місцезнаходження дронів.

Системи відеоспостереження, оснащені камерами високої роздільної здатності, захоплюють зображення навколишнього середовища. Комп'ютерний зір, за допомогою алгоритмів машинного навчання, аналізує ці зображення для виявлення характерних ознак дронів, таких як форма, розмір та рух. Алгоритми, такі як YOLO, забезпечують швидку та точну детекцію об'єктів, дозволяючи ідентифікувати дрони серед інших об'єктів. Приклад роботи YOLO для визначення дронів наведено на рис. 1.3.

Рисунок 1.3 – Приклад визначення дрону за допомогою YOLO

Пов’язуючи дані від нейронної мережі, яка відіграє роль класифікатора, простих алгоритмів виявлення та відслідковування на виході ми отримаємо вектор що містить наступні параметри: швидкість, поточні координати в кадрі, клас цілі, імовірність приналежності до класу [8].

Використання відеоспостереження з комп'ютерним зором має ряд переваг. По-перше, ці системи забезпечують високу точність виявлення, завдяки можливості навчання на великій кількості даних. По-друге, вони можуть працювати в режимі реального часу, що дозволяє швидко реагувати на появу дронів у повітряному просторі. Також системи комп'ютерного зору можуть адаптуватися до різних умов освітлення та фону, що підвищує їхню ефективність у різних ситуаціях.

Проте, цей підхід також має свої недоліки. По-перше, точність виявлення може знижуватися в умовах поганого освітлення або при наявності великої кількості перешкод на фоні. По-друге, системи комп'ютерного зору потребують значних обчислювальних ресурсів [8] для обробки відеопотоків, що може бути обмеженням для деяких застосувань, але одним з варіантів вирішення цієї проблеми є використання хмарних сервісів, що дають змогу використовувати обчислювальну потужності для навчання нейронної мережі. Крім того, навчання моделей на великих наборах даних може вимагати часу та зусиль, що може ускладнити швидке впровадження системи в експлуатацію.

1.3 Акустичні системи виявлення

Акустичні системи виявлення використовують звукові сигнали для виявлення безпілотних літальних апаратів (БПЛА). Цей підхід базується на здатності дронів створювати характерні звукові хвилі під час польоту, які можуть бути захоплені спеціалізованими мікрофонами та оброблені для визначення наявності та місцезнаходження дронів [9].

По-перше, система складається з масиву мікрофонів, які розміщуються в різних точках для забезпечення максимального охоплення зони

спостереження. Мікрофони можуть бути як стаціонарними, так і мобільними, в залежності від цілей виявлення. Коли дрон пролітає поблизу, він створює звукові хвилі, які виникають внаслідок роботи двигунів, обертання пропелерів та інших механізмів.

Наступний етап полягає в аналізі звукових сигналів. Зібрані звукові сигнали передаються до центрального процесора або комп'ютера, де проходять обробку. Система використовує алгоритми обробки сигналів, які можуть включати фільтрацію шуму, виділення характерних частот, амплітуду та інші параметри звуку. Використовуючи методи спектрального аналізу, система може виявляти специфічні звукові шаблони, які відрізняються від фонових шумів (наприклад, звуки автомобілів, птахів, вітру тощо).

На основі аналізу звукових сигналів система може визначити різні характеристики дронів. Це включає тип дрону, оскільки різні моделі можуть мати характерні звуки. Також система може оцінювати швидкість руху дрону, використовуючи зміни частоти звукових сигналів (доплерівський ефект) [10]. Хоча акустичні системи мають обмежену дальність виявлення, за допомогою аналізу часу прибуття звуку до різних мікрофонів можна оцінити місцезнаходження дрону.

Останній етап полягає в інтерпретації даних та сповіщенні. Після обробки даних система може генерувати оповіщення або звукові сигнали для операторів, що вказують на наявність дрону [11]. Це може бути важливим для систем безпеки, моніторингу та контролю повітряного простору. В деяких випадках акустичні системи можуть бути інтегровані з відеоспостереженням або радіолокацією для створення багаторівневої системи виявлення, що підвищує загальну ефективність. Процес виявлення дронів за допомогою акустичної системи наведено на рис. 1.4.

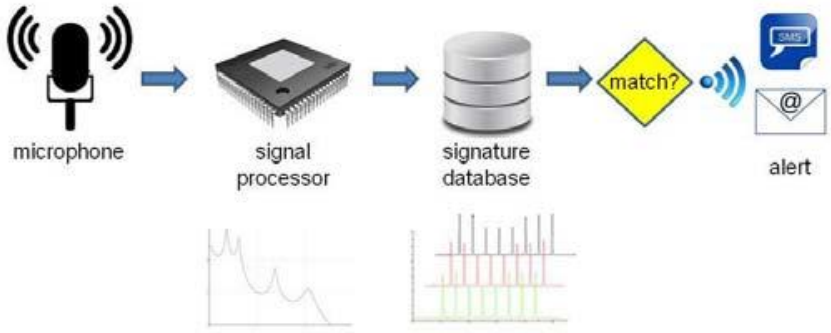

Рисунок 1.4 – Процес виявлення дронів за допомогою акустичної системи

1.4 Висновки до розділу

У розділі про способи виявлення дронів розглянуті різні технології, кожна з яких має свої переваги і недоліки.

Пульсова доплерівська радіолокація забезпечує дальнє виявлення, але може мати проблеми з низькошвидкісними об'єктами.

Безперервні хвильові радари демонструють високу чутливість, але обмежені в точності відстані.

Відеоспостереження з комп'ютерним зором забезпечують високу точність і адаптивність, але потребують значних обчислювальних ресурсів і можуть бути менш ефективними при поганому освітленні.

Акустичні системи виявлення здатні ідентифікувати дрони без візуальної прив'язки, але їхня чутливість знижується в умовах сильного шуму.

З метою досягнення оптимальних результатів прийнято рішення на користь комп'ютерного зору, тому що ця технологія забезпечує високу точність виявлення дронів завдяки здатності аналізувати зображення в режимі реального часу, надаючи умовному «оператору» візуальні вихідні дані.

2 АНАЛІЗ ТЕХНІЧНОГО ЗАВДАННЯ ТА ПОСТАНОВКА ЗАДАЧ ДОСЛІДЖЕННЯ

Розробка ефективної системи виявлення дронів вимагає використання сучасних технологій, включаючи комп'ютерний зір, радіолокаційні системи, акустичні сенсори та інші методи збору даних [12]. Розглянемо основні засоби розробки та технології, які будуть використані для реалізації системи, а також чітко сформулюємо постановку задачі.

Для розробки алгоритмів виявлення та обробки даних буде використовуватись наступне програмне забезпечення:

– мова програмування Python, котра забезпечує високий рівень гнучкості та підтримку багатьох бібліотек для роботи з машинним навчанням, комп'ютерним зором тощо;

– бібліотека OpenCV, що дозволить оброблювати зображення або відеопоток, застосовувати різні методи виявлення об'єктів [13]. На рис. 2.1 наведено приклад роботи з бібліотекою OpenCV для контролю трафіку;

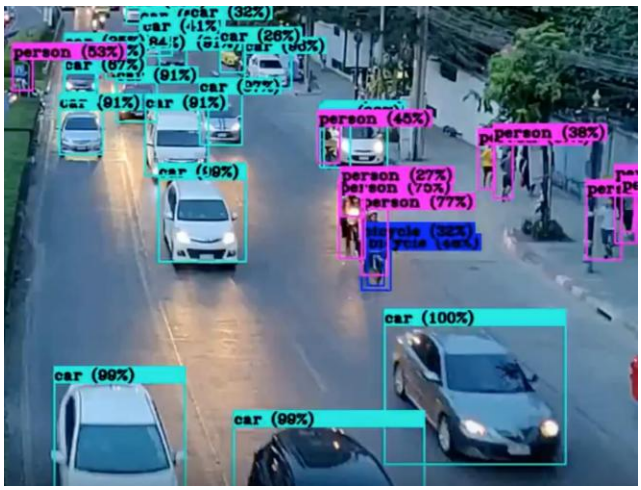

Рисунок 2.1 – Приклад роботи OpenCV бібліотеки

– бібліотеки TensorFlow, PyTorch для глибокого навчання, які здатні ефективно виявляти та класифікувати об'єкти на основі заданих даних [14]. На рис. 2.2 наведено приклад побудованої за допомогою PyTorch нейронної

мережі для класифікації цифрових зображень;

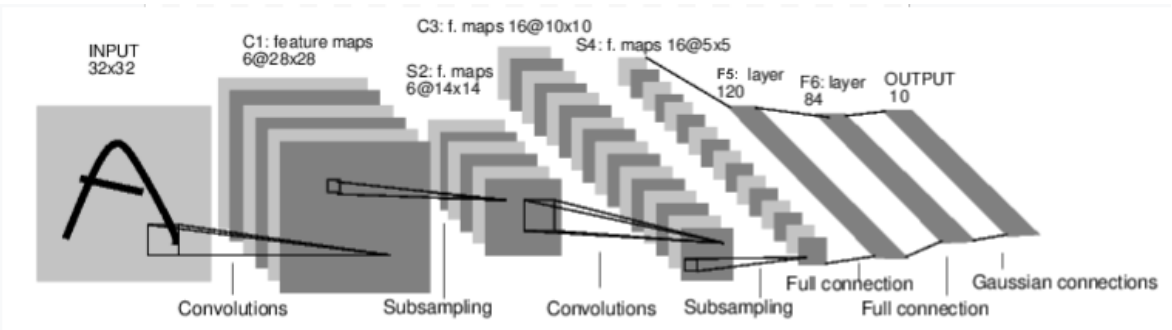

Рисунок 2.2 – Побудована за допомогою PyTorch нейронна мережа

– бібліотека NumPy, що дасть змогу робити математичні та статистичні обчисленя, обробляти великий потік даниї та проводити аналіз [15].

Основною задачею дослідження є розробка інтегрованої системи виявлення дронів, яка в собі високу точность та надійность виявлення. Для цього необхідно вирішити кілька важливих підзадач:

– підготовка набору даних для навчання моделі;

– навчання та тестування моделі;

– оцінка ефективності системи;

– оптимізація системи.

Для зручності розробки та тестування моделей буде використовуватися VSCode як основне середовище для програмування, що дозволяє ефективно працювати з кодом, інтегрувати бібліотеки та підтримує інструменти для налагодження [16]. Це середовище підтримує роботу з Python і різними бібліотеками, що значно прискорює процес розробки.

Для обчислювальної потужності та проведення експериментів будемо використовувати сервіс Google Colab, який надає доступ до потужних графічних процесорів (GPU) та тензорних процесорів (TPU), що дозволяє виконувати обчислення з високою продуктивністю без необхідності в дорогому апаратному забезпеченні [17]. Google Colab дозволяє ефективно працювати з великими наборами даних, тренувати складні моделі та значно знижує час на виконання обчислень.

Colab надає доступ до різних типів GPU, таких як NVIDIA Tesla K80, T4, P4 та P100. Тип доступного GPU може змінюватися в залежності від навантаження на сервери та доступності ресурсів. У безкоштовній версії доступ до GPU обмежений, а в платних підписках доступ до більш потужних GPU розширюється [18]. В нашому випадку будемо використовувати безкоштовку підписку, що надає необмежений доступ до GPU Nvidia Tesla T4. Характеристики Tesla T4 наведено нижче:

– 320 Turing тензерних ядер;

– 2560 Nvidia CUDA ядер;

– 8,1 TFLOPS одиничної точності (SP);

– 65 TFLOPS змішаної точності (MP) FP16/FP32;

– 130 TOPS INT8;

– 260 TOPS INT4;

– 16 гігабайт відеопам’яті типу GDDR6 з пропусною здатність 300 ГБ/с;

– пам'ять коду з корекцією помилок [19].

«PyTorch» дає можливість при роботі з YoloV5 використовувати як відеокарту (GPU) та її CUDA ядра, якщо відеокарта від виробника NVIDIA та має CUDA ядра, бо NVIDIA є засновником цієї технології, так і процесор.

Для порівняння мій персональний комп’ютер має процесор Intel Core i5-14600kf, та відеокарту NVIDIA RTX 4070 Super. Процесор має 14 ядер і 20 потоків з тактовою частотою до 5,3 гігагерц, а відеокарта побудована на архітектурі Ada Lovelace та має 12 ГБ GDDR6X пам’яті і 7168 CUDA ядер. Хоча вона і має вищу продуктивність у задачах, що не обмежуються використанням тільки FP16/FP32 обчислень (як у випадку з Tesla T4), Colab надає зручний доступ до сервісів, зокрема при потребі у TPU для глибокого навчання, де 4070 Super може поступатись за обчислювальною потужністю.

У другому розділі було розглянуто основні технології та програмні засоби, що використовуються для створення нейромережевого модуля для виявлення дронів. Проаналізовано переваги та недоліки різних методів.

Для розробки алгоритмів виявлення дронів були визначені наступні інструменти:

– мова програмування Python для гнучкої розробки та інтеграції з різними бібліотеками;

– бібліотека OpenCV для обробки зображень і відеопотоків;

– бібліотеки TensorFlow та PyTorch для глибокого навчання і класифікації об’єктів;

– бібліотека NumPy для обробки великих обсягів даних та виконання математичних обчислень.

Основні задачі дослідження включають підготовку набору даних, навчання та тестування моделі, оцінку її ефективності, а також оптимізацію системи. Використання VSCode забезпечить зручне середовище для розробки, а Google Colab з доступом GPU NVIDIA Tesla T4 дозволить швидко і ефективно виконувати обчислення.

У другому розділі було розглянуто технічне завдання та сформульовано основні задачі для розробки системи виявлення дронів. Було обрано сучасні технології та програмні засоби, зокрема мову Python, бібліотеки OpenCV, TensorFlow, PyTorch і NumPy, які забезпечують ефективну обробку даних та машинне навчання.

Для розробки використовується середовище VSCode, а для обчислень Google Colab з доступом до GPU, що дозволяє швидко виконувати необхідні обчислення. Основні задачі включають підготовку даних, навчання, тестування та оптимізацію моделі, що забезпечить високу точність і надійність системи.

3 НАВЧАННЯ МОДЕЛІ З ВИКОРИСТАННЯМ НАБОРУ ДАНИХ

3.1 Підготовка навчального набору даних

Для навчання моделі виявлення дронів було зібрано датасет, який складається з 1012 зображень для тренувальної вибірки та 347 фотографій для валідаційної вибірки. Датасет також містить відповідні анотації для кожного зображення, що дозволяє моделі ефективно навчатися та оцінювати свою продуктивність. На рис. 3.1 наведено приклад зображення що буде використовувати для навчання моделі.

Рисунок 3.1 – Приклад зображення що буде використовуватись для навчання моделі

Перед початком навчання дані повинні бути попередньо оброблені. Це включає кілька основних етапів, а саме:

– розділення даних на тренувальну та валідаційну вибірки;

– обробка анотацій.

Зібрані зображення вже розподілені на дві категорії: для навчання (train) та для перевірки якості моделі (val).

Анотації для кожного зображення будуть зчитуватися та перетворюватися у формат, придатний для подачі в модель, з урахуванням необхідних нормалізованих координат.

Структура набору даних наведена на рис. 3.2.

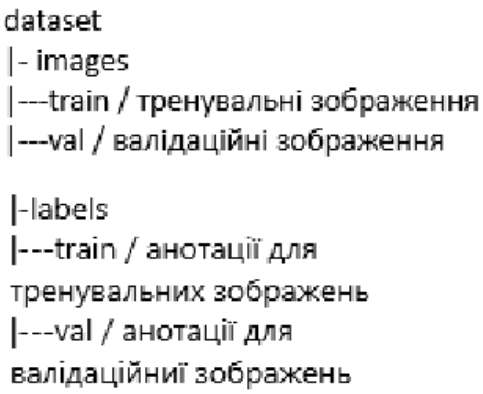

Рисунок 3.2 – Структура набору даних

Анотації зберігаються в окремих текстових файлах для кожного зображення. Кожен файл містить координати обмежувального прямокутника, що окреслює об'єкт на зображенні, а також ідентифікатор класу об'єкта (у даному випадку, дрон). Формат анотацій для одного зображення має наступний вигляд: «0 0.532 0.501 0.936 0.631».

Перше значення вказує на клас об’єкта. В нашому випадку – 0, тому що клас лише один, а саме дрон. В майбутньому модель можна буде допрацювати не тільки на виявлення дронів, а й на їх класифікацію, або класифікацію разних літальних об’єктів.

Наступні чотири числа є координатами обмежувального прямокутника в нормалізованих координатах:

– «0.532» координата X центру прямокутника;

– «0.501» координата Y центру прямокутника;

– «0.936» ширина прямокутника (відносна до ширини зображення);

– «0.631» висота прямокутника (відносна до висоти зображення).

Ці анотації необхідні для правильної підготовки даних для моделі, оскільки вони дозволяють моделі навчатися виявляти об'єкти на зображеннях шляхом регресії до координат обмежувального прямокутника [20].

3.2 Налаштування середовища Google Colab

Для ефективного навчання моделі виявлення дронів, використовується обчислювальне середовище Google Colab. Це дозволяє швидко розгорнути необхідну інфраструктуру та використовувати потужні апаратні ресурси, зокрема графічні процесори (GPU) та тензорні процесори (TPU).

Першим етапом необідно підключитись до Colab та створити новий блокнот (рис. 3.3). У полі Hardware Accelerator необхідно обрати Tesla T4. Після цього ми можемо побачити ресурси, які нам доступні, а саме 12,7 гігабайт оперативної пам’яті, 15 гігабайт відеопам’яті та 112,6 гігібайт дискового простору.

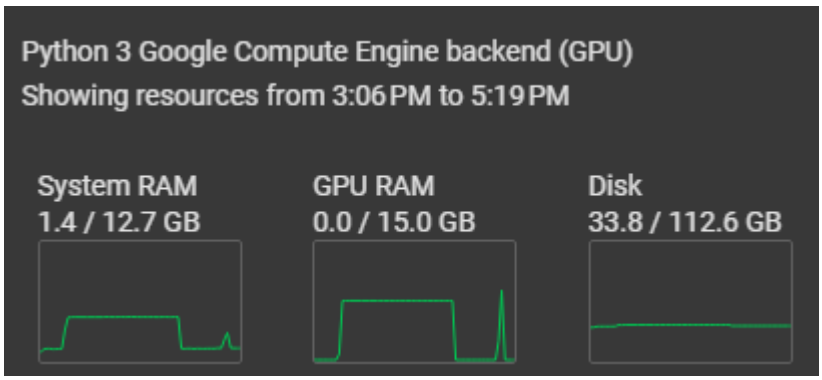

Рисунок 3.3 – Доступні для роботи ресурси Google Colab

Після цього необхідно підключити Google диск до Google Colab, що робиться для того щоб використовувати хмарне сховище для всього процесу навчання моделі. Тобто модель YoloV5, датасет, і весь код для навчання моделі зберігається в хмарному сховищу. Для підключення Google диску використовуємо команду, наведену на рисунку 3.4.

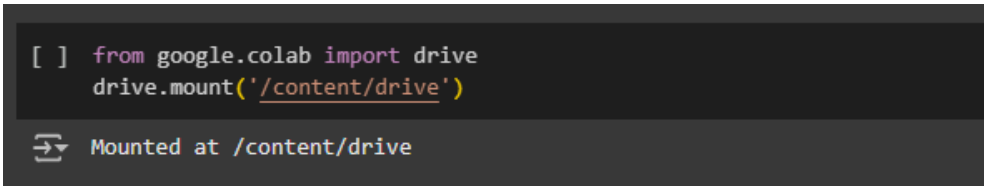

Рисунок 3.4 – Підключення Google диску до Colab

Наступним кроком є завантаження та розпакування датасету. Для цього використовуємо команду наведену на рисунку 3.5.

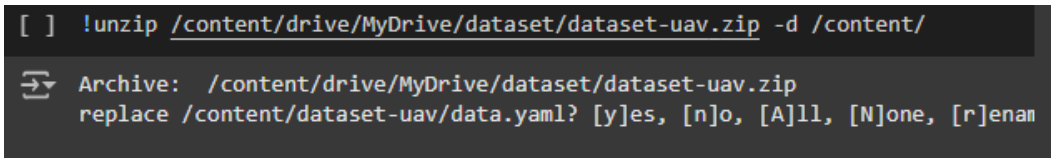

Рисунок 3.5 – Розпакування датасету в середовище Google Colab

Після цього відбувається перевірка поточного робочого каталогу, що виконується за допомогою команди наведеної на рисунку 3.6.

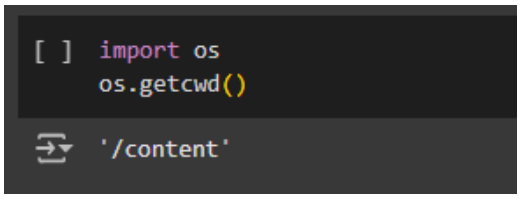

Рисунок 3.6 – Перевірка поточного робочого каталогу

Наступним кроком необхідно завантажити останню версію YoloV5. Для цього клонуємо за допомогою Git офіційний репозиторій. Для цього використовуємо команду наведену на рисунку 3.7.

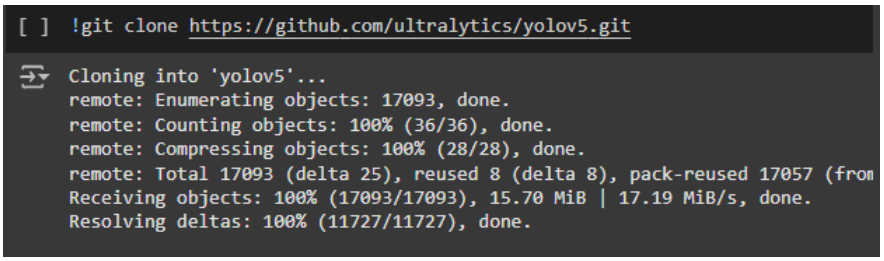

Рисунок 3.7 – Клонування репозиторію YoloV5 за допомогою Git

Після клонування переходимо до каталогу з YoloV5, використовуючи команду «%cd yolov5». Також для коректної роботи з Yolo, необхідно встановити усі необхідні залежності, використовуючи pip (менеджер пакетів Python), використовуючи команду наведену на рисунку 3.8.

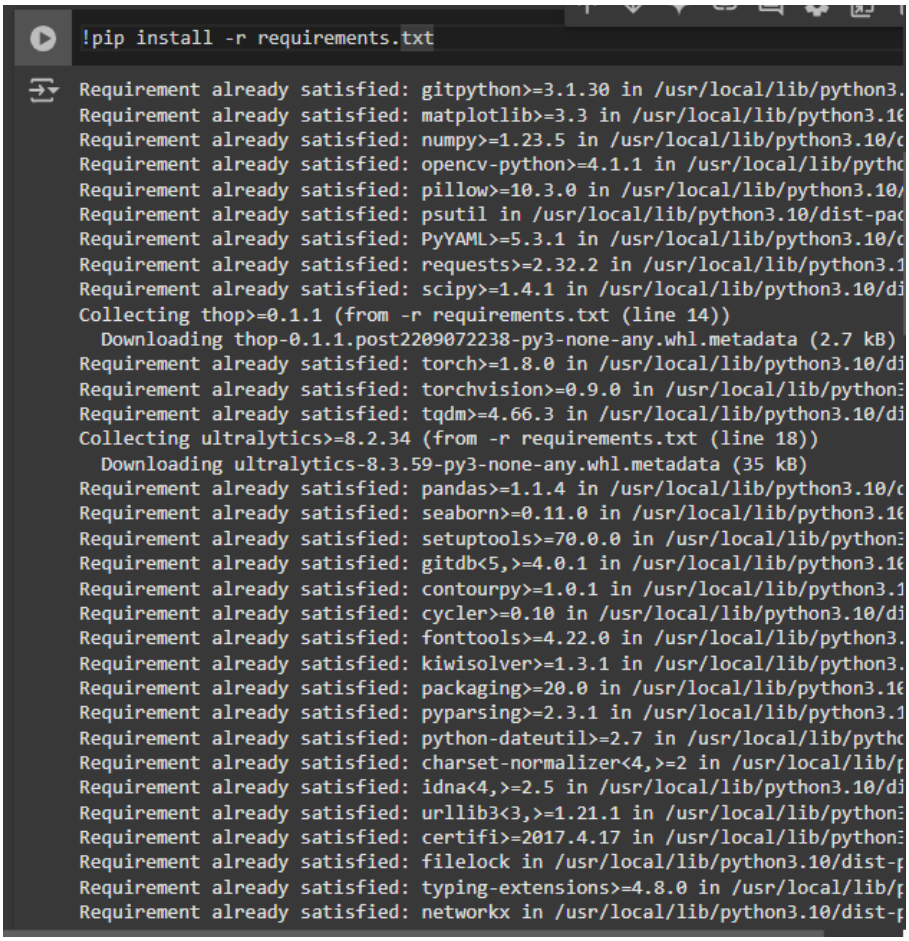

Рисунок 3.8 – Встановлення залежностей Yolo

3.3 Навчання моделі

Після налаштування середовища та підготовки датасету, наступним етапом є навчання моделі YOLOv5. Для цього використовується команда, яка запускає процес тренування з певними параметрами – «!python train.py --img 416 --batch 16 --epochs 100 --data data.yaml --weights yolov5x.pt --cache --name uav_1st».

Параметри навчання моделі:

– «--img 416» вказує розмір вхідного зображення, тобто 416 на 416 пікселів. Цей параметр задає розмір зображення, на якому буде проводитись тренування. В Yolo 416 вжається стандартним, і саме цей розмір оптимізано під навчання Yolo;

– «--batch 16» вказує на розмір пакету (batch size), тобто кількість зображень що буде оброблятись під час кожного кроку навчання;

– «--epochs 100» вказує на кількість епох, тобто моделі необхідно буде тренуватись на всіх зображеннях 100 разів для досягнення кращої точності. Це значення підбирається під кожне навчання окремо, тому що модель або або недовчити, або перевчити. Після декількох спроб, кожна з яких займає приблизно 2 години, було прийнято рішення що 100 епох дають найкращі результати;

– «--weights yolov5x.pt» вказує на ваги моделі, які будуть завантажуватись з вказаного файлу. YoloV5x це найбільша попередньо навчена модель з Yolo, яка буде використовуватись для подальшого навчання;

– «--cache» вказує на те, що датасет буде завантажено в пам’ять для прискорення процесу навчання;

– «--name uav_1st» вказує на ім’я сессії тренування моделі. Тобто після закінчення тренування моделі результати будуть збережені з відповідною назвою.

На рис. 3.9 наведено приклад вихідних даних під час навчання моделі.

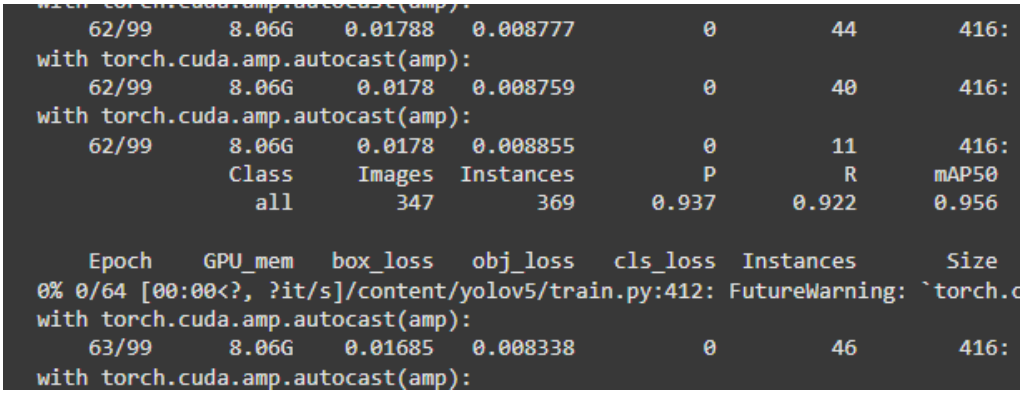

Рисунок 3.9 – Вихідні дані під час навчання моделі

Після успішного закінчення навчання моделі отримуємо наступні результати (рис. 3.10).

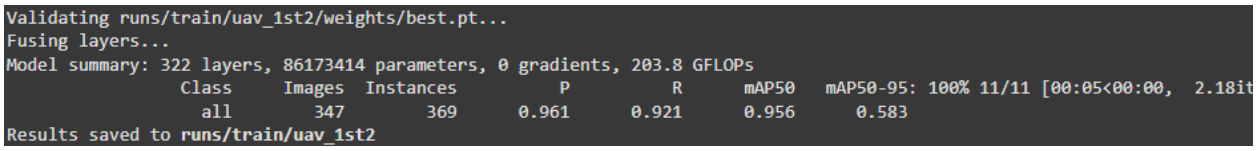

Рисунок 3.10 – Результати навчання моделі

3.4 Аналіз результатів навчання моделі

Зведення моделі:

– кількість шарів складає 322;

– кількість параметрів складає 86173414;

– градієнти 0, що свідчить про успішну оптимізацію моделі;

– обчислювальна складність 203,8 GFLOPs.

Метрики продуктивності:

– «images» вказує на те, що 347 зображень було використано для валідації;

– «instances» вказує на те, що 369 об’єктів було виявлено.

Основні метрики:

– P (Precision) вказує на точність, тобто відсоток правильно ідентифікованих дронів серед усіх передбачених позитивних прикладів [21] і складає 0,961;

– R (Recall) вказує на відтворюваність, та відображає відсоток правильно ідентифікованих дронів серед усіх реальних позитивних прикладів [21] і складає 0,921;

– mAP50 вказує на середню точність на основі обмежуючих рамок при порозі IoU = 0,5 [21] і складає 0,956;

– mAP50-95 вказує на середню точність при різних порогах IoU від 0,5 до 0,95 з кроком 0,05, що надає більш повну оцінку продуктивності моделі [21]

і становить 0,583.

Для більш детального аналізу продуктивності моделі, нижче наведено:

– матрицю плутанини (рис. 3.11), що показує кількість правильних та неправильних класифікацій для кожного класу;

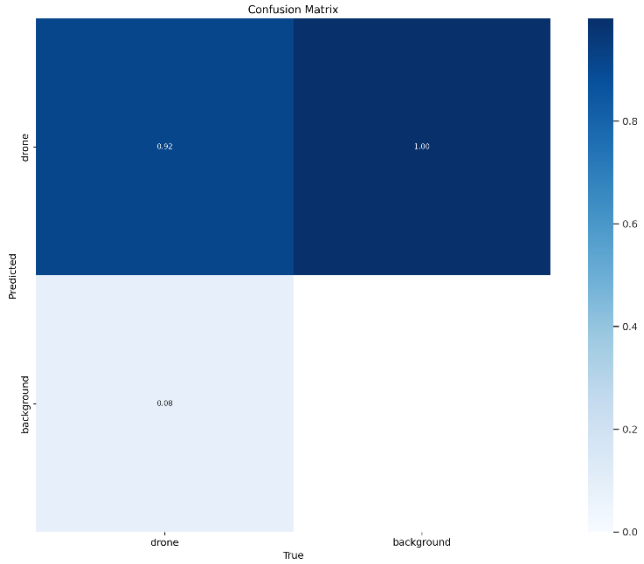

Рисунок 3.11 – Матриця плутанини

– графік зміни F1-міри (гармонійного середнього між точністю і повнотою) (рис. 3.12) в залежності від порогу виявлення;

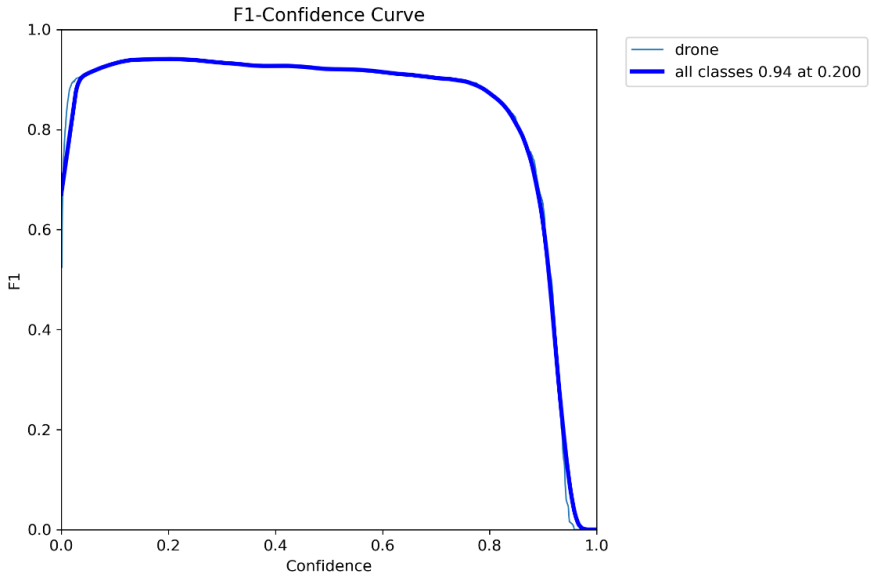

Рисунок 3.12 – Графік зміни F1-міри

– графік, що відображає розподіл міток (кількість прикладів кожного класу в наборі даних) (рис. 3.13);

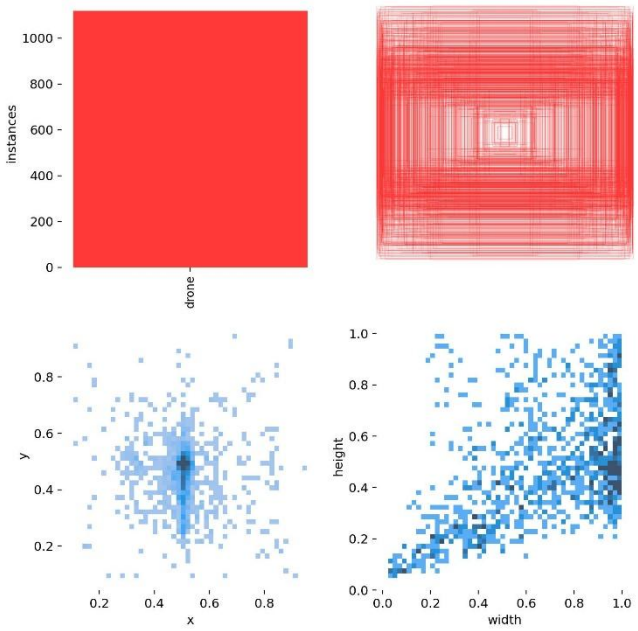

Рисунок 3.13 – Графік відображення розподілу міток

– корелограма міток (рис. 3.14), що показує взаємозв'язки між різними класами у раборі даних;

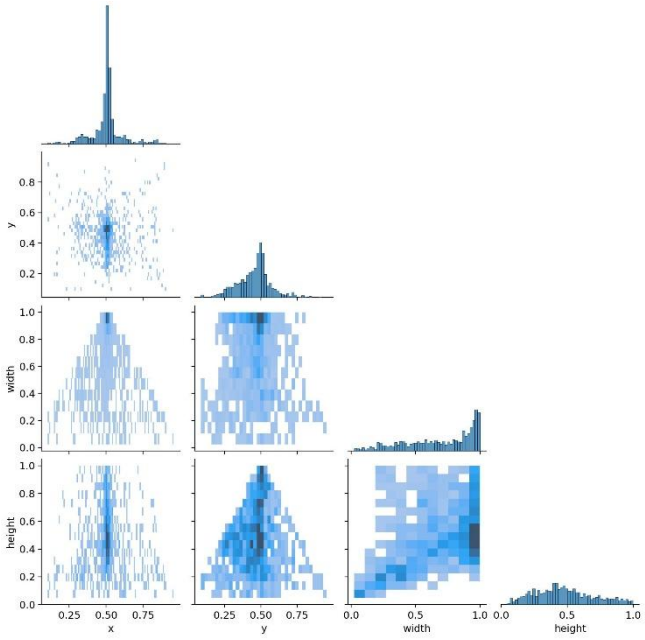

Рисунок 3.14 – Корелограма міток

– крива точності (Precision) (рис. 3.15) в залежності від порогу виявлення;

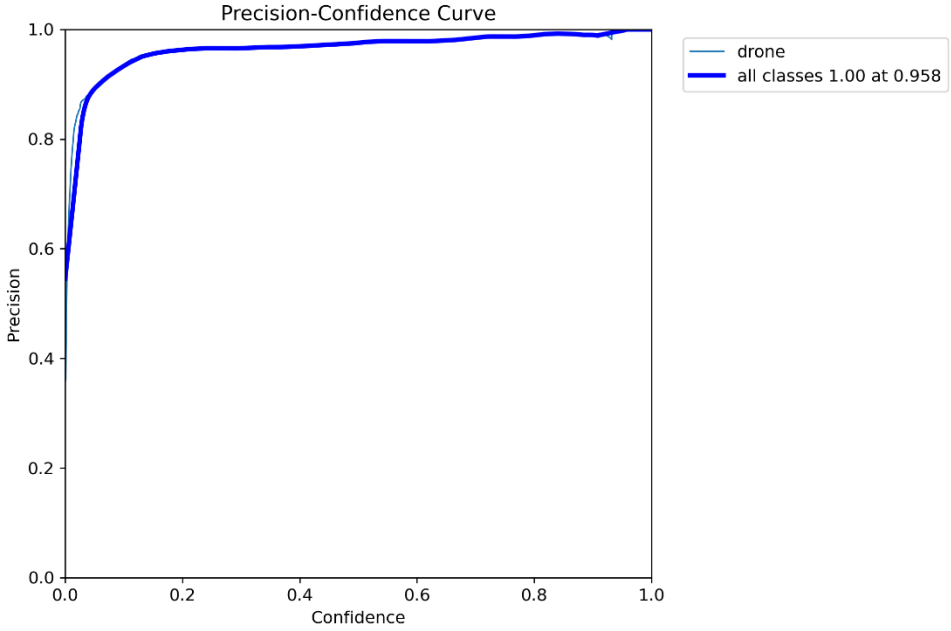

Рисунок 3.15 – Крива точності

– крива залежності між точністю (Precision) і повнотою (Recall) (рис. 3.16), що дозволяє оцінити баланс між цими метриками;

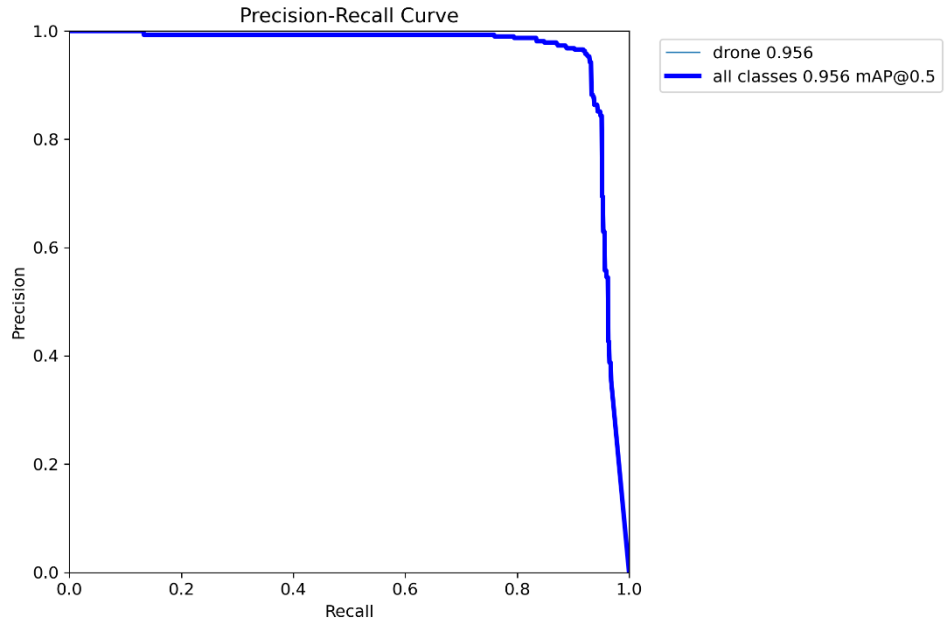

Рисунок 3.16 – Крива залежності між точністю (Precision) і повнотою (Recall)

– крива повноти (Recall) (рис. 3.17) в залежності від порогу виявлення;

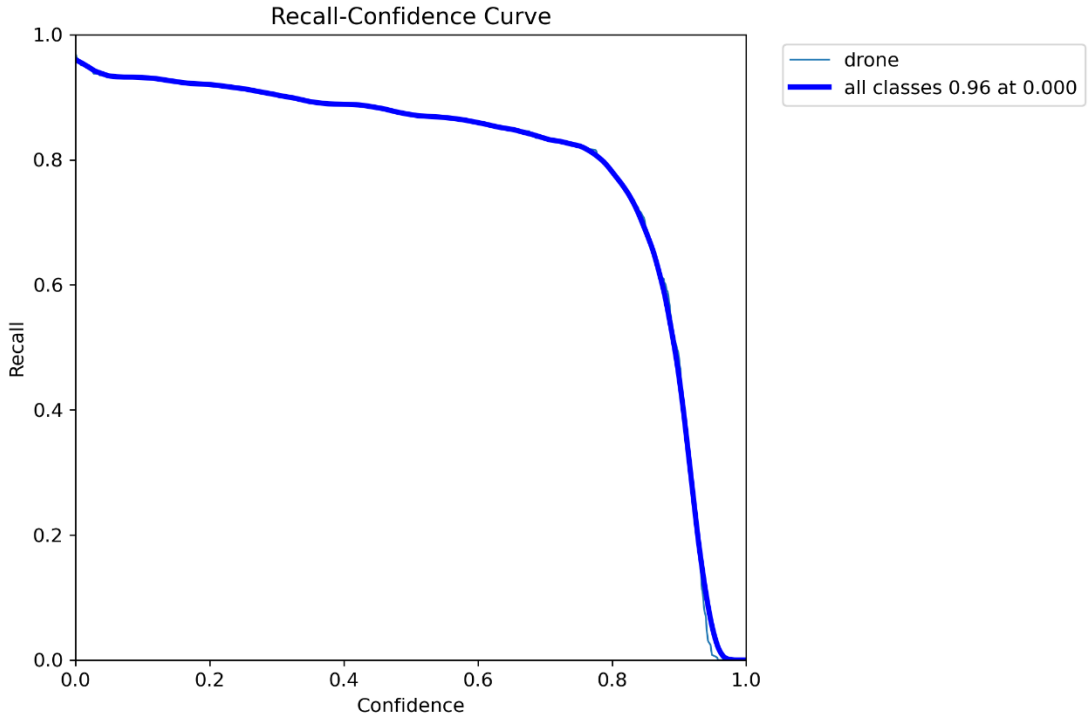

Рисунок 3.17 – Крива повноти

– загальний огляд результатів навчання (рис. 3.18), що включає графіки втрат, точності та повноти за епохами;

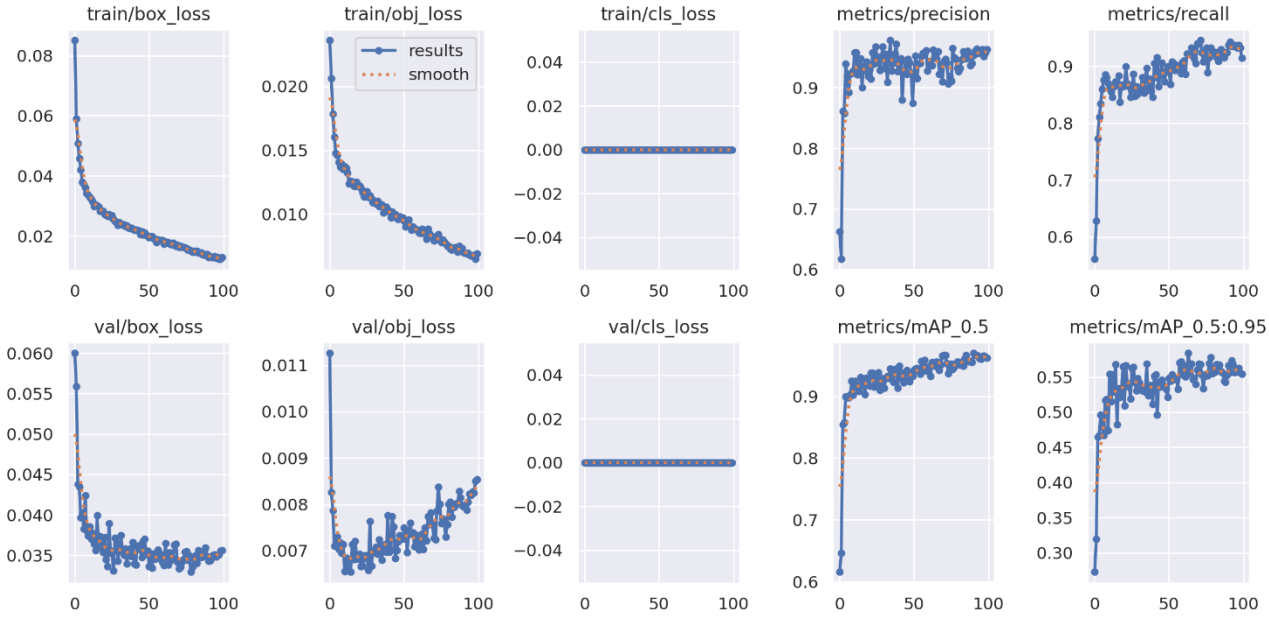

Рисунок 3.18 – Загальний огляд результатів навчання

– приклади зображень із навчальної вибірки з накладеними мітками та передбаченнями моделі на різних стадіях навчання (рис. 3.19);

Рисунок 3.19 – Приклади зображень із навчальної вибірки з накладеними мітками

– приклади зображень із валідаційної вибірки з накладеними мітками істинних класів (рис. 3.20);

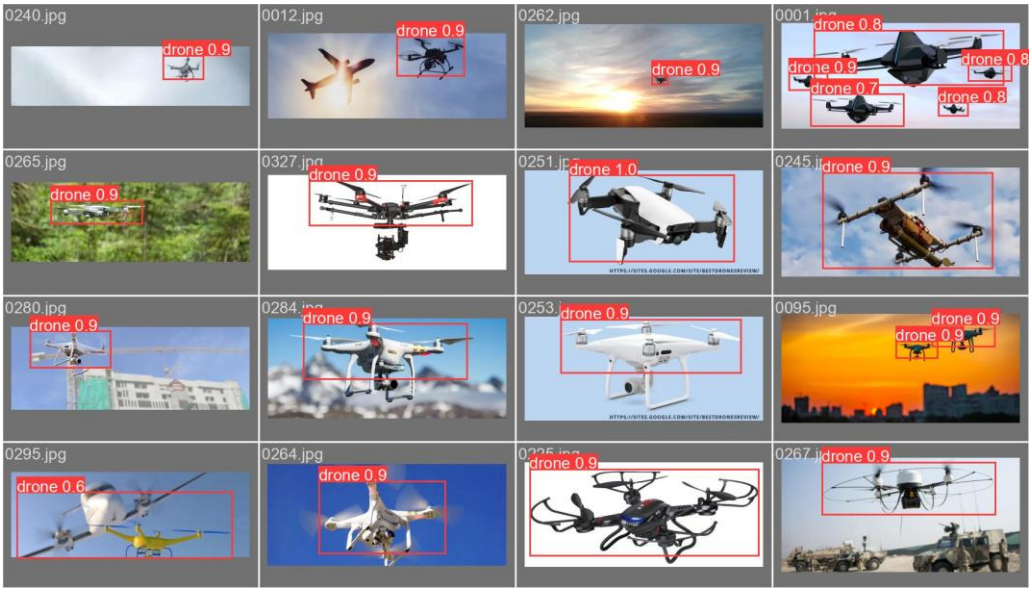

Рисунок 3.20 – Приклади зображень із валідаційної вибірки з накладеними мітками істинних класів

– приклади зображень із валідаційної вибірки з накладеними передбаченнями моделі (рис. 3.21).

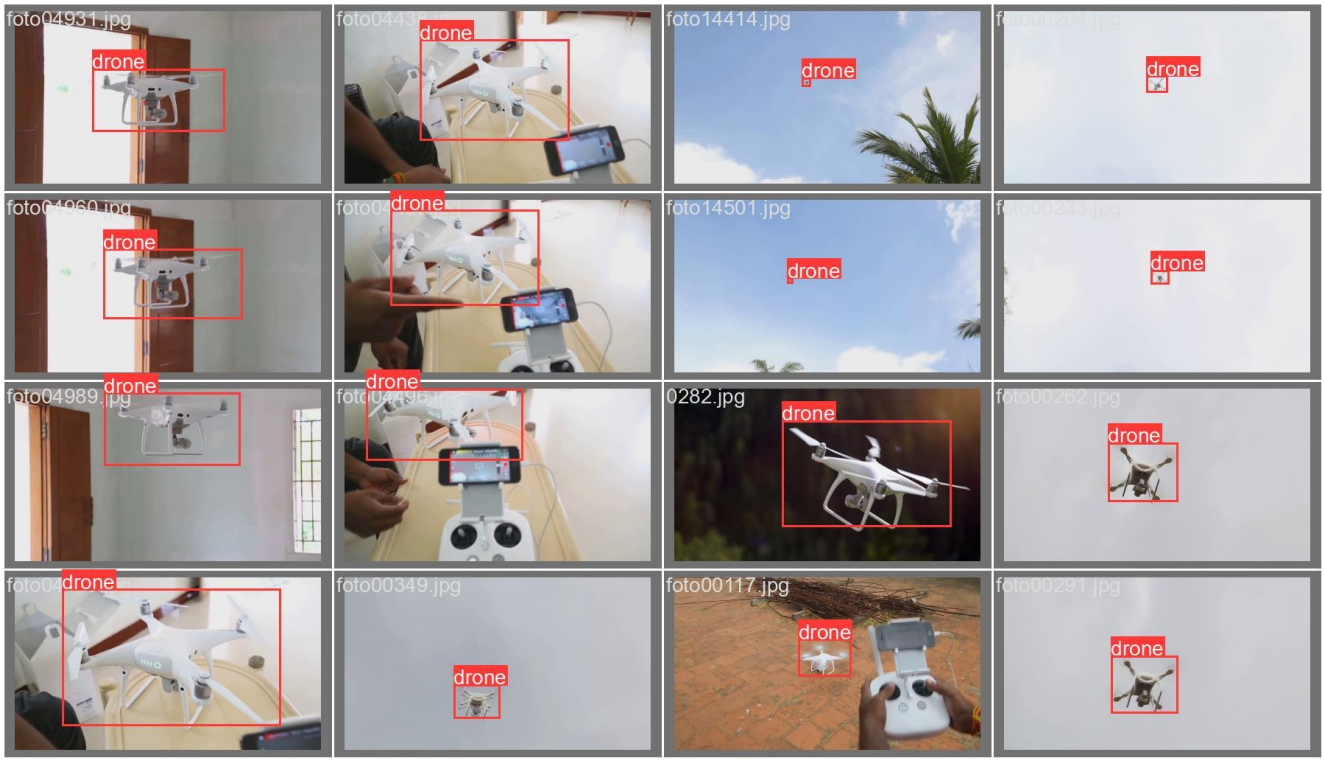

Рисунок 3.21 – Приклади зображень із валідаційної вибірки з накладеними передбаченнями моделі

Ці результати демонструють високу ефективність моделі, особливо за такими метриками, як точність (P) та середня точність (mAP50). Це свідчить про те, що модель здатна надійно виявляти дрони з високою точністю та відтворюваністю.

3.5 Валідація отриманої моделі та аналіз даних

Після завершення навчання моделі, для оцінки її продуктивності на валідаційному наборі даних використовується наступна команда: «!python val.py --weights runs/train/uav_1st2/weights/best.pt --data data.yaml».

Параметри валідації:

– дані, а саме файл конфігурації data.yaml, який містить інформацію про валідаційний набір даних;

– ваги, а саме файл best.pt, сформований після навчання моделі і являє собою найкращі ваги моделі;

– пакетна обробка складає 32 зображення на одну ітерацію;

– розмір зображення 640 на 640 пікселів;

– поріг впевненності 0,001;

– поріг IoU 0,6;

– максимальна кількість об’єктів на зображення 300;

– 8 робочих процесів;

– пристрій CUDA (Tesla T4).

Результати валідації:

– P 0,846;

– R 0,803;

– mAP 0,901;

– mAP50-95 0,442;

– швидкість обробки складає 0,7 мс на зображення;

– інференс 36,9 на зображення;

– NMS 4,9 мс на зображення.

3.6 Висновки до розділу

У третьому розділі було проведено навчання моделі YOLOv5 для виявлення дронів, включаючи підготовку навчального набору даних, налаштування середовища Google Colab, виконання навчання та валідації моделі.

Було використано структурований набір даних, який включає 1012 зображень для навчання і 347 зображень для валідації. Всі зображення мають відповідні анотації, що дозволяє моделі ефективно навчатися.

Google Colab було налаштовано для використання обчислювальних потужностей Tesla T4, що забезпечує швидке і ефективне навчання моделі. Інтеграція з Google Диском дозволила зберігати всі необхідні файли та результати навчання в хмарному середовищі.

Навчання було виконано з використанням базових ваг YOLOv5x і параметрів, оптимальних для завдання виявлення дронів. Параметри навчання були налаштовані на 100 епох з розміром зображень 416 пікселів і з загальною кількостю тренувальних об’єктів 16.

Модель показала високий рівень точності map50 при виявленні дронів. Однак, mAP50-95 вказує на можливість покращення продуктивності моделі при вищих порогах IoU.

Використання Google Colab з GPU Tesla T4 значно прискорило процес навчання та валідації моделі.

4 РЕАЛІЗАЦІЯ НЕЙРОМЕРЕЖЕВОГО МОДУЛЯ

4.1 Інтеграція навченої моделі у застосунок

Для інтеграції навченої моделі YoloV5 у застосунок необхідно налаштувати робоче середовище, встановити необхідні для роботи модуля бібліотеки та реалувати код для завантаження і використання моделі.

Першим кроком є встановлення Python версії 3.9. На даний час існує більш актуальна версія Python, а саме 3.13, але для використання бібліотеки pytorch найстабільнішою все ще залишається версія 3.9. На рис. 4.1 наведено процес встановлення Python 3.9.

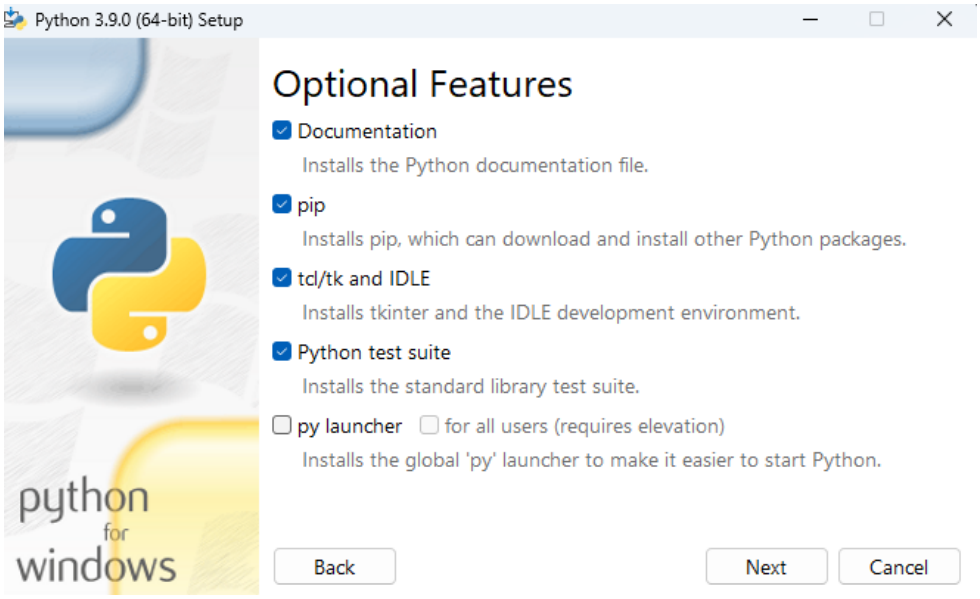

Рисунок 4.1 – Процес встановлення Python 3.9

Також для коректної роботи необхідно встановити Git версії 2.47.1 для клонування необхідних для роботи модуля репозиторіїв. На рис. 4.2 наведено процес встановлення Git.

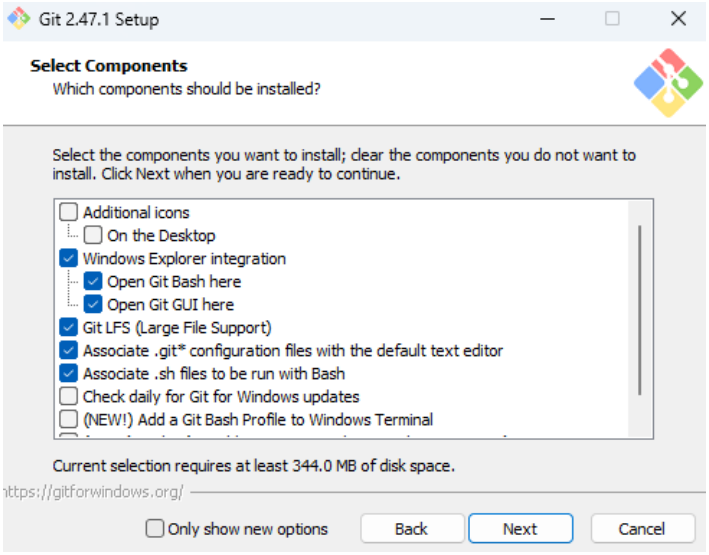

Рисунок 4.2 – Процес встановлення Git 2.47.1

Наступним кроком є встановлення середовища розробки Visual Studio Code. Середовище розробки VSCode дозволяє ефективно працювати з кодом і інтегрувати додаткові інструменти для налагодження та тестування. Також після встановлення VSCode для роботи з бібліотекою PyTorch і використання CUDA ядер, необхідно встановити Toolkit для роботи з CUDA від NVIDIA. Наразі одною з актуальних версій є CUDA 11.8.0 522.06 випуску 2022 року. Процес встановлення Toolkit'у для роботи з CUDA наведено на рис. 4.3.

Рисунок 4.3 – Процес встановлення CUDA Toolkit

Після успішного встановлення CUDA Toolkit переходимо до середовища розробки, створюємо каталог для застосунку та віртуальне середовище. Для створення віртуального середовища використовуємо наступну команду:

```
Python -m venv .venv
.venv\Scripts\activate
```

Після цього встановлюємо необхідні бібліотеки для роботи

```
pip3 install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu118
```

Необхідно використовувати саме версію CUDA 11.8 для роботи с CUDA Toolkit 11.8.

Також необхідно встановити бібліотеки OpenCV для роботи з камерою і зображеннями в цілому, pathlib для роботи з шляхамих, та Tkinter для створення графічного інтерфейсу користувача. Для цього використовуємо наступні команди:

```
pip install opencv-python
pip install pathlib
```

Після цього можна переходити до реалізації коду нейромережевого модуля. Починаємо з імпорту бібліотек:

```
import cv2
import torch
import tkinter as tk
from tkinter import messagebox
from threading import Thread
from PIL import Image, ImageTk
import time
```

Після цього необхідно змінити поведінку модуля pathlib, тимчасово підмінивши PosixPath на WindowsPath для коректної інтеграції моделі:

```
import pathlib
temp = pathlib.PosixPath
```

```
pathlib.PosixPath = pathlib.WindowsPath
```

Наступним кроком реалізуємо завантаження моделі Yolov5:

```
# завантаження yolov5
model = torch.hub.load('ultralytics/yolov5', 'custom', path='best.pt', force_reload=True)
```

force_reload=True змушує перевантажити модель навіть якщо вона вже завантажена, 'custom' вказує на те що ми використовуємо не стандартну модель yolov5, а файл 'best.pt' і є нашою моделю для виявлення дронів.

Після цього створюємо глобальні зміні:

```
cap = None
frame = None
```

Змінна «cap» це змінна для зберігання об’єкта захоплення відео, а «frame» це змінна для збереження поточного кадру відео.

Після цього створюємо функцію обробки відео та виявлення дронів:

```
# функція обробки відео та знаходження дронів
def detect_drone():
    global cap, frame
    cap = cv2.VideoCapture(0)  # відкриття камери
    if not cap.isOpened():
        messagebox.showerror("Ошибка", "Не удается открыть камеру.")
        return
    # оновлення фреймів в Tkinter
    update_frame()
```

Функція відкриває камеру, а якщо камеру відкрити не вдається, відображається повідомлення про помилку. Після успішного виконання викликається функція «update_frame» для обробки кадрів.

Наступним кроком є написання функції для оновлення зображення:

```
# функція оновлення зображення
def update_frame():
    global frame
```

```
        ret, new_frame = cap.read()
        if ret:
            # перетворюємо зображення в формат для моделі
            results = model(new_frame)
            # виводимо результати на зображення
            new_frame = results.render()[0]
             # перетворюємо зображення OpenCV в формат підходящий для
Tkinter
            new_frame_rgb = cv2.cvtColor(new_frame, cv2.COLOR_BGR2RGB)
            img_pil = Image.fromarray(new_frame_rgb)
            img_tk = ImageTk.PhotoImage(image=img_pil)
            # оновлення зображення якщо воно змінилось
            if frame != img_tk:  # перевірка зміни кадру
                label.imgtk = img_tk
                label.configure(image=img_tk)
                frame = img_tk
        # виклик кадру кожні 30 мілісекунд (30-35fps)
        window.after(33, update_frame)
```

Логіка функція полягає в захоплені нового кадру з камери. Після функція проганяє кадр через модель для виявлення дронів, виконує рендер результатів візуалізації, конвертує кадр з OpenCV формату до формату, сумісного з Tkinter і оновлює зображення в інтерфейсі кожні 33 мілісекунди (це приблизно 30 FPS).

Після чого реалізуємо функцію відкриття камери:

```
# функція відкриття камери
def open_camera():
    # функція знаходження дронів в окремому потоці
    detection_thread = Thread(target=detect_drone)
    detection_thread.start()
```

Функція «open_came» запускає функцію «detect_drone» в окремому

потоці для уникнення блокування головного потоку інтерфейсу.

Також необхідно реалузівати функцію закриття програми:

```
# функція закриття программи
def close_program():
    global cap
    if cap is not None:
        cap.release()  # чистимо камеру перед закриттям програми
    window.quit()
```

Логіка цієї функція полягає в звільненні камери та закритті програми.

Останнім кроком переходимо до створення графічного інтерфейсу:

```
# створення окна програми
window = tk.Tk()
window.title("Drone Detection")
# мітка для відображення зображення
label = tk.Label(window)
label.pack()
# кнопка відкриття камери
open_button = tk.Button(window, text="Открыть камеру", command=open_camera)
open_button.pack(pady=10)
# кнопка закриття програми
close_button = tk.Button(window, text="Закрыть программу", command=close_program)
close_button.pack(pady=10)
# запуск інтерфейса
window.mainloop()
```

Інтерфейс програми простий і складається з:

– мітки для відображення відео «label»;

– кнопки для відкриття камери «open_button»;

– кнопки для закриття програми «close_button»;

– циклу обробки подій інтерфейс «window.mainloop()».

4.2 Тестування нейромережевого модуля

Після інтеграції навченої моделі у застосунок необхідно виконати її тестування, щоб переконатися в правильності роботи та відповідності очікуванням. Тестування нейромережевого модуля включає наступні кроки:

– перевірка коректності завантаження моделі;

– перевірка роботи камери;

– тестування функція виявлення.

Після успішного тестування коректності завантаження моделі та роботи камери, нейромережевий модуль було протестовано на зображеннях з датасету та відеозаписів дронів з вільного доступу. На рис. 4.4 та рис. 4.5 наведено результати тестування нейромережевого модуля.

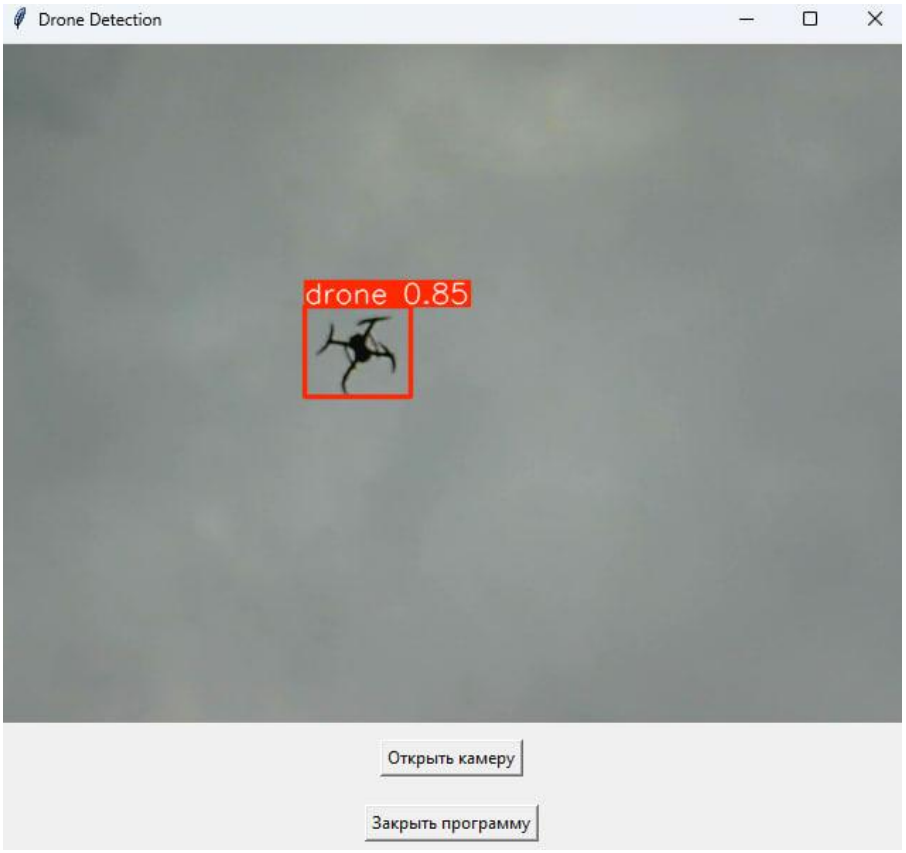

Рисунок 4.4 – Результат тестування нейромережевого модуля на фотографії з валідаційного набору даних

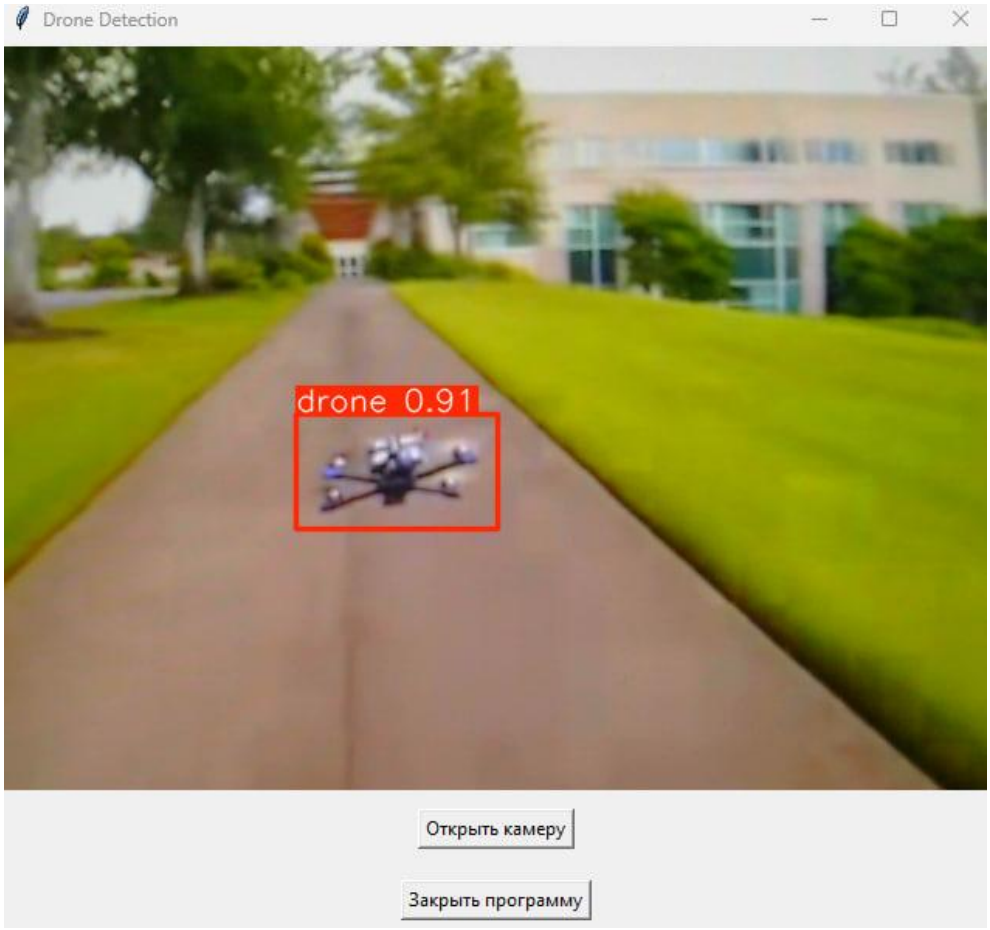

Рисунок 4.5 – Результат тестування нейромережевого модуля на відеозаписі з вільного доступу

4.3 Забезпечення безпечних умов праці при розробці системи

При розробці та впровадженні системи виявлення дронів необхідно забезпечити дотримання всіх вимог охорони праці та техніки безпеки. Це включає організацію безпечного робочого середовища, захист від електромагнітного випромінювання, пожежну безпеку, безпеку роботи з електричними приладами [22].

Для організації безпечного робочого середовища необхідно забезпечити наступне:

– висота стільця повинна відповідати висоті робочого столу, щоб забезпечити правильне положення рук і спини. Висота сидіння повинна бути 40-50 см;

– рівень освітлення на робочому місці повинен бути в межах 300-500 люкс. Наприклад, при площі приміщення 20 м², потрібна освітленість 6000-10000 люмен, що буде забезпечено кількома світильниками загальною потужністю близько 100 Вт (LED лампи).

Для підтримки комфортного мікроклімату слід враховувати, що об'єм повітря становить 54 м³, якщо приміщення має площу м² і висоту стелі 2,7 м. За нормами повітрообмін повинен становити 30 м³/год на людину. Для двох людей потрібно забезпечити обмін 60 м³/год [22].

Допустимий рівень електромагнітного випромінювання для персоналу:

– робоча зона 5-25 В/м для частот 30 кГц - 300 МГц;

– у місцях з великою кількістю обчислювальної техніки слід забезпечити екранування обладнання або збільшити відстань між робочими місцями та технікою.

Необхідні засоби пожежогасіння для приміщення площею 20 м² з електричним обладнанням:

– один порошковий вогнегасник типу ОП-5 (площа захисту до 20 м²);

– встановлення автоматичної пожежної сигналізації, що реагує на дим або температуру.

Наступним кроком є розрахунок навантаження на електричну мережу. Якщо в приміщенні встановлено 5 комп'ютерів з середнім споживанням 200 Вт кожен і додаткове обладнання (монітори, принтери) загальною потужністю 500 Вт, загальне споживання становить 1500 Вт. Враховуючи стандартну напругу 220 В, загальний струм складе 6,8 А. Необхідно забезпечити, щоб розетки і проводка витримували цей струм [22].

Забезпечення охорони праці потребує врахування всіх наведених аспектів для створення безпечного та комфортного робочого середовища. Розрахунки підтверджують, що приміщення відповідає нормам з точки зору площі, освітлення, електробезпеки та пожежної безпеки.

4.4 Розрахунок стійкості системи

Стійкість нейромережевого моудля для виявлення дронів визначається здатністю системи зберігати свою функціональність при різних зовнішних впливах, збої в обладанні та високих навантаженнях.

Основні аспекти стійкості системи включають:

– надійність компонентів системи;

– безперервність роботи при збої;

– енергоефективність та резервне живлення.

Надійність системи залежить від тривалості безвідмовної роботи кожного компонента. Припустимо, що середній час безвідмовної роботи (MTBF) для кожного з основних компонентів становить:

– 50000 годин процесора (CPU);

– 40000 годин відеоадаптера (GPU);

– 30000 годин оперативної пам’яті (RAM);

– 20000 годин жорсткого/твердотільного накопичувача (HDD/SDD).

Загальна надійність системи «R» при паралельному підключенні компонентів розраховується як:

$$R = 1 - \left(1 - R_{cpu}\right) \cdot \left(1 - R_{gpu}\right) \cdot (1 - R_{ram}) \cdot (1 - R_{hdd}).$$

Припускаючи, що надійність кожного компонента за час «t» обчислюється як:

$$R_i = e^{-\lambda it},$$

де $\lambda_i = \frac{1}{MTBF}$, розрахуємо надійність системи для періоду 1 року (8760 годин).

Формула надійності для кожного компонента:

$$R_i(t) = e^{-\lambda it},$$

де $\lambda_i = \frac{1}{MTBF_i}$ – інтенсивність відмов компонента;

t – час у годинах (8760 годин).

Розрахунок надійності для CPU:

$$\lambda_{cpu} = \frac{1}{50000},$$

$$R_{cpu}(8760) = e^{-\frac{8760}{50000}} = e^{-0,1752} = 0,839.$$

Розрахунок надійності для GPU:

$$\lambda_{gpu} = \frac{1}{40000},$$

$$R_{gpu}(8760) = e^{-\frac{8760}{40000}} = e^{-0,219} = 0,803.$$

Розрахунок надійності для RAM:

$$\lambda_{ram} = \frac{1}{30000},$$

$$R_{ram}(8760) = e^{-\frac{8760}{30000}} = e^{-0,292} = 0,746.$$

Розрахунок надійності для HDD:

$$\lambda_{hdd} = \frac{1}{20000},$$

$$R_{hdd}(8760) = e^{-\frac{8760}{20000}} = e^{-0,438} = 0,645.$$

Таким чином загальна надійність система розраховується наступним чином:

$$R = 1 - (1 - 0,839) \cdot (1 - 0,803) \cdot (1 - 0,746) \cdot (1 - 0,645) = 1 - (0,161 \cdot 0,197 \cdot 0,254 \cdot 0,355) = 1 - 0,0028 = 0,9972.$$

Надійність системи за 1 рік роботи становить приблизно 0,9972 або 99,72%. Це означає, що система має високий рівень стійкості та ймовірність безвідмовної роботи протягом року дуже висока.

Для забезпечення безперервності роботи при збої використовуються резервні системи та механізми відновлення. Дані моделі та конфігурації зберігаються в хмарному сховищі, що забезпечує миттєве відновлення у випадку локального збою.

Для оцінки енергоефективності необхідно розрахувати сумарне енергоспоживання системи на основі технічних характеристик кожного компонента. Маємо наступні вихідні дані:

– споживання процесору 95 Вт;

– споживання графічного процесору 120 Вт;

– споживання оперативної пам’яті 10 Вт;

– споживання жорсткого або твердотільного накопичувача 6 Вт;

– споживання інших компонентів (мережева плата, вентилятори) 20 Вт.

Сумарне споживання становить 251 Вт.

Для розрахунку енергоспоживання за рік (8760 годин), використовуємо формулу:

$$E_{total} = P_{total} \cdot t,$$

де t – час у годинах (8760 годин).

$$E_{total} = 251 \cdot 8760 = 2197260 \frac{\text{Вт}}{\text{год}} = 2197{,}26 \frac{\text{кВт}}{\text{год}}.$$

Для забезпечення безперебійної роботи системи в разі відключення електроенергії необхідно обрати джерело безперебійного живлення (ДБЖ), здатне підтримувати роботу системи на необхідний час.

Вихідні дані для вибору ДБЖ:

– час автономної роботи 30 хвилин;

– потужність системи 251 Вт.

Розрахунок ємності батареї ДБЖ використовуємо формулу:

$$C_{battery} = \frac{P_{total} \cdot t_a}{n \cdot V},$$

де P_{total} – потужність системи;

t_a – час автономної роботи;

n – коєфіцієнт ефективності ДБЖ (в середньому 0,9);

V – напруга батареї (12 В).

Отже, розрахунок ємності батареї ДБЖ:

$$C_{battery} = \frac{251 \cdot 0{,}5}{0{,}9 \cdot 12} = \frac{125{,}5}{10{,}8} = 11{,}62 \text{ А/год.}$$

Отже, для забезпечення 30 хвилин автономної роботи системи необхідна батарея ємністю приблизно 11,62 А/год.

Виходячи з цього, робимо висновок що для забезпечення безперебійної роботи системи в разі відключення електроенергії необхідне джерело безперебійного живлення з батареєю ємністю приблизно 11,62 А на годину для підтримання автономної роботи протягом 30 хвилин.

4.5 Висновки до розділу

У цьому розділі успішно інтегровано навчений модуль YOLOv5 у застосунок для виявлення дронів.

Також реалізовано простий інтерфейс користувача, що дозволяє обробляти відеопотік у реальному часі.

Тестування підтвердило стабільну роботу системи та ефективність виявлення дронів у різних умовах.

Виявлено можливості для подальшої оптимізації продуктивності та точності. Загалом, нейромережевий модуль готовий до використання з

потенціалом для покращень.

Також було враховано усі моменти охорони праці, включаючи організацію безпечного робочого середовища, захист від електромагнітного випромінювання, пожежну безпеку та безпеку роботи з електричними приладами.

ВИСНОВКИ

При виконані кваліфікаційної роботи було проведено комплексний аналіз технологій для виявлення дронів, серед яких обрано комп'ютерний зір завдяки його високій точності та адаптивності.

Також було виконано навчання моделі YOLOv5 з використанням структурованого набору даних (1012 зображень для тренування і 347 для валідації) з використанням обчислювальних ресурсів Google Colab з GPU Tesla T4, що забезпечило ефективне та швидке навчання.

Модель показала високі результати за метриками точності, зокрема mAP50, але залишаються можливості для покращення при вищих порогах IoU.

Було описано умови забезпечення безпечних умов праці при розробці системи, також було розрахована стійкість системи.

Успішно інтегровано навчений модуль у застосунок з реальним часом обробки відеопотоку, підтверджено стабільну роботу системи. Виявлено можливості для подальшої оптимізації продуктивності та точності моделі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Методичні вказівки з підготовки та захисту кваліфікаційної роботи здобувачами другого (магістерського) рівня вищої освіти спеціальності 174 Автоматизація, комп'ютерно-інтегровані технології та робототехніка, освітньо-професійних програм: «Комп'ютерно-інтегровані технологічні процеси і виробництва», «Комп'ютеризовані та робототехнічні системи» / Упоряд. І. Ш. Невлюдов, Р. В. Артюх, В. В. Безкоровайний, Н. П. Демська, В. В. Євсєєв, О. І. Филипенко, О. М. Цимбал. Харків: ХНУРЕ, 2024. 57 с.

2. ДСТУ 3008: 2015. Інформація та документація. Звіти у сфері науки і техніки. Структура і правила оформлення. К.: ДП «УкрНДНЦ». 2016. 30 с.

3. Сагула О. О. Аналіз програмного нейромережевого модуля для виявлення дронів на основі YOLOV5 // Автоматизація та Приладобудування («Automation and Development of Electronic Devices» ADED-2024) [Електронний ресурс] : збірник студентських наукових статей. Харків : ХНУРЕ, 2024. Вип. 2. С. 130-137.

4. Singha, S.; Aydin, B. Automated Drone Detection Using YOLOv4. Drones 2021, 5, 95.

5. Wang, Chenxing, et al. «Deep learning-based UAV detection in pulse-Doppler radar.» IEEE Transactions on Geoscience and Remote Sensing 60 (2021): 1-12.

6. Tian, Jiangmin, et al. «Fully Convolutional Network-Based Fast UAV Detection in Pulse Doppler Radar.» IEEE Transactions on Geoscience and Remote Sensing 62 (2024): 1-12.

7. Liang, Cang, et al. «UAV detection using continuous wave radar.» 2018 IEEE International Conference on Information Communication and Signal Processing (ICICSP). IEEE, 2018.

8. Соколов, К. А. «Візуальне виявлення та відстеження малогабаритних рухомих об'єктів на основі функціональних особливостей зорового апарату та

особливостях сприйняття людини.» Вчені записки (2023): 5202375.

9. Sedunov, Alexander, et al. «UAV passive acoustic detection.» 2018 IEEE International Symposium on Technologies for Homeland Security (HST). IEEE, 2018.

10. Borysov, O., Yu Artabaiev, and A. Surma. «Кібербезпека безпілотних військових апаратів: методи захисту від перехоплення та дистанційного управління.» Computer-integrated technologies: education, science, production 56 (2024): 117-125.

11. Sathyamoorthy, Dinesh. (2015). A Review of Security Threats of Unmanned Aerial Vehicles and Mitigation Steps. The Journal of Defence and Security. 6. In press.

12. Laktionov, Oleksandr & Boryak, Bohdyan & Pedchenko, Nazar & Mykhailichenko, Oleksiy. (2023). Огляд алгоритмів комп'ютерного зору для виявлення небезпечних об'єктів дронами. Системи управління, навігації та зв’язку. Збірник наукових праць. 3. 120-122. 10.26906/SUNZ.2023.3.120.

13. Lee, Dongkyu, Woong Gyu La, and Hwangnam Kim. «Drone detection and identification system using artificial intelligence.» 2018 International Conference on Information and Communication Technology Convergence (ICTC). IEEE, 2018.

14. Hashem, Ahmed, and Thomas Schlechter. «Drone Detection Using Deep Learning: A Benchmark Study.» International Conference on Computer Aided Systems Theory. Cham: Springer Nature Switzerland, 2022.

15. Srigrarom, Sutthiphong, and Photchara Ratsamee. "Development of UAV for fire detection and for object detection and tracking of flying object." AIAA Aviation 2019 Forum. 2019.

16. Sawadogo, Abdoul Moumouni, et al. «Dynamic Fire Detection and Alerting Using Image Processing for Drone Applications.» 2024 International Conference on Electrical Electronics and Computing Technologies (ICEECT). Vol. 1. IEEE, 2024.

17. Menon, Hema P., et al. «A Study on YOLOv5 for Drone Detection with

Google Colab Training.» 2023 2nd International Conference on Automation, Computing and Renewable Systems (ICACRS). IEEE, 2023.

18. Colaboratory. Frequently Asked Questions URL: https://research.google.com/colaboratory/faq.html?hl=uk&utm

19. NVIDIA T4 Tensor Core Datasheet. URL: https://www.nvidia.com/en-us/data-center/tesla-t4/

20. Diwan, Tausif, G. Anirudh, and Jitendra V. Tembhurne. «Object detection using YOLO: Challenges, architectural successors, datasets and applications.» multimedia Tools and Applications 82.6 (2023): 9243-9275.

21. Ali, Usman, et al. «Performance Evaluation of YOLO Models in Plant Disease Detection.» Journal of Informatics and Web Engineering 3.2 (2024): 199-211.

22. Методичні вказівки до виконання розділу "Охорона праці" у випускних роботах ОКР "бакалавр" усіх форм навчання / упоряд.: В. А. Айвазов. Т. Є. Стиценко., Н. Л. Березуцька ; М-во освіти і науки України, ХНУРЕ. – Харків : ХНУРЕ, 2018. – 28 с. – 1,81.