МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
КРИВОРІЗЬКИЙ ФАХОВИЙ КОЛЕДЖ
ДЕРЖАВНОГО НЕКОМЕРЦІЙНОГО ПІДПРИЄМСТВА
«ДЕРЖАВНИЙ УНІВЕРСИТЕТ «КИЇВСЬКИЙ АВІАЦІЙНИЙ ІНСТИТУТ»
Циклова комісія *комп'ютерних систем та мереж*
(повна назва циклової комісії)

Допустити до захисту
Голова випускової циклової комісії
комп'ютерних систем та мереж
(повна назва циклової комісії)
(підпис) *Ірина КРАВЧУК* (ім'я, ПРІЗВИЩЕ)
« 10 » 06 2025 р.

КВАЛІФІКАЦІЙНА РОБОТА
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬО-ПРОФЕСІЙНОГО СТУПЕНЯ ФАХОВИЙ МОЛОДШИЙ БАКАЛАВР

Тема: *Інтеграція комп'ютерного зору та штучного інтелекту для автоматичного розпізнавання об'єктів дронами в задачах моніторингу*

Група: *3-012* Спеціальність: *123 «Комп'ютерна інженерія»*

Здобувач освіти (підпис) *Єгор ЛІЗКО* (ім'я, ПРІЗВИЩЕ)

Керівник роботи (підпис) *Артем КУТІН* (ім'я, ПРІЗВИЩЕ)

Консультант з оформлення пояснювальної записки (підпис) *Оксана ОСАДЧА* (ім'я, ПРІЗВИЩЕ)

Кривий Ріг 2025 р.

КРИВОРІЗЬКИЙ ФАХОВИЙ КОЛЕДЖ
ДЕРЖАВНОГО НЕКОМЕРЦІЙНОГО ПІДПРИЄМСТВА
«ДЕРЖАВНИЙ УНІВЕРСИТЕТ «КИЇВСЬКИЙ АВІАЦІЙНИЙ ІНСТИТУТ»

Відділення *комп'ютерної та програмної інженерії*
Циклова комісія *комп'ютерних систем та мереж*
Освітньо-професійний ступінь *фаховий молодший бакалавр*
Спеціальність *123 «Комп'ютерна інженерія»*

ЗАТВЕРДЖУЮ
Голова випускової циклової комісії
комп'ютерних систем та мереж
(повна назва циклової комісії)
(підпис) *Ірина КРАВЧУК*
(ім'я, ПРІЗВИЩЕ)
« *01* « *03* 2025 р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ОСВІТИ

ЛІЗКО Єгора Тимофійовича
(прізвище, ім'я, по батькові)

1.Тема роботи *Інтеграція комп'ютерного зору та штучного інтелекту для автоматичного розпізнавання об'єктів дронами в задачах моніторингу*

Керівник роботи *Кутін Артем Ілліч, викладач вищої категорії*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по коледжу від « *04* « *04* 2025 року № *50-ст*

2. Строк подання здобувачем освіти роботи з *01.03.2025* по *15.06.2025*

3. Вихідні дані до роботи *системи розпізнавання об'єктів, мова програмування Python, бібліотека OpenCV*

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

Теоретичні основи інтеграції комп'ютерного зору та штучного інтелекту в дронах. Аналіз і вибір технологій. Розробка системи автоматичного в розпізнавання об'єкті.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Презентація Microsoft PowerPoint

6. Консультанти розділів роботи (проекту)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	*Узгодження технічного завдання з керівником дипломної роботи*	*01.03.2025*	*виконано*
2	*Підбір та вивчення науково-технічної літератури за темою дипломної роботи*	*15.03.2025*	*виконано*
3	*Розділ 1. Теоретичні основи інтеграції комп'ютерного зору та штучного інтелекту в дронах*	*28.04.2025*	*виконано*
4	*Розділ 2. Аналіз і вибір технологій*	*14.05.2025*	*виконано*
5	*Розділ 3. Розробка системи автоматичного розпізнавання об'єктів*	*26.05.2025*	*виконано*
6	*Підготовка матеріалів до презентації*	*30.05.2025*	*виконано*
7	*Написання та оформлення пояснювальної записки*	*06.06.2025*	*виконано*
8	*Захист дипломної роботи*		

Здобувач освіти _____ *Єгор ЛІЗКО*
(підпис) (ім'я, ПРІЗВИЩЕ)

Керівник роботи _____ *Артем КУТІН*
(підпис) (ім'я, ПРІЗВИЩЕ)

StrikePlagiarism.com

Дата звіту 5/27/2025
Дата редагування ---

Звіт не був оцінений

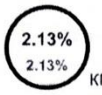
Звіт подібності

метадані

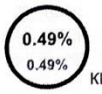
Назва організації
Ukrainian national aviation university
Заголовок
Лізко 3-012 Кваліфікаційна робота
Автор Науковий керівник / Експерт
ЛізкоКлименко С
підрозділ
Криворізький Фаховий коледж

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.



КП 1



КЦ

25	11382	86969
Довжина фрази для коефіцієнта подібності 2	Кількість слів	Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв		0
Інтервали		0
Мікропробіли		2
Білі знаки		0
Парафрази (SmartMarks)		10

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз — Колір тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	2022_M_KITAM_KITПВм_21_1_Брюховецкий_О_А [redacted] 5/30/2024 Kharkiv National University of Radio Electronics (Kharkiv National University of Radio Electronics)	79 0.69 %
2	2022_M_KITAM_KITПВм_21_1_Брюховецкий_О_А [redacted] 5/30/2024 Kharkiv National University of Radio Electronics (Kharkiv National University of Radio Electronics)	19 0.17 %

РЕФЕРАТ

Дипломна робота «Інтеграція комп'ютерного зору та штучного інтелекту для автоматичного розпізнавання об'єктів дронами в задачах моніторингу» містить 83 сторінок, 28 рисунків, 8 таблиці, 30 використаних джерел.

СИСТЕМА КОМП'ТЕРНОГО ЗОРУ, ШТУЧНИЙ ІНТЕЛЕКТ, НЕЙРОННА МЕРЕЖА, НАВЧАННЯ З ВЧИТЕЛЕМ, БЕЗПІЛОТНИЙ ЛІТАЛЬНИЙ АПАРАТ, DATASET IMAGENET, БІБЛІОТЕКА TENSORFLOW, МОДЕЛЬ ГЛИБОКОГО НАВЧАННЯ YOLOV8

Дипломна робота «Інтеграція комп'ютерного зору та штучного інтелекту для автоматичного розпізнавання об'єктів дронами в задачах моніторингу» присвячена розробці та впровадженню інтелектуальної системи, що забезпечує обробку візуальної інформації в реальному часі. В роботі проведено аналіз методів комп'ютерного зору (OpenCV) та моделей глибокого навчання (YOLOv8), досліджено апаратні й програмні компоненти інтегрованої системи, описано процеси збору та анотації даних, а також тренування ШІ-моделі. Результатом роботи стала працездатна архітектура, що дозволяє дрону ефективно виявляти об'єкти під час польоту.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БПЛА – безпілотний літальний апарат;

ДДЗ – датчик динамічного зору;

ЛА – літальний апарат;

ПЗ – програмне забезпечення;

ПК – персональний комп’ютер;

ШІ – штучний інтелект;

API – Application Programming Interface;

CAA – Civil Aviation Authority;

CCD – charge-coupled device;

CNN – Convolutional neural network;

CV – computer vision;

DoD – Department of Defense;

GNSS – Global Navigation Satellite System;

ILSVRC – ImageNet Large Scale Visual Recognition Challenge;

IMU – inertial measurement unit.

IoU – Intersection Over Union;

LiDAR – Light Detection and Ranging;

SLAM – simultaneous localization and mapping;
UAV – unmanned aerial vehicle.

ВСТУП

В епоху стрімкого технологічного розвитку безпілотні літальні апарати (БпЛА), широко відомі як дрони, трансформуються з нішевих пристроїв у багатофункціональні інструменти з широким спектром застосувань. Їхня здатність оперативно збирати дані з висоти відкриває нові горизонти для моніторингу різноманітних об'єктів та територій. Одночасно, прогрес у галузях комп'ютерного зору та штучного інтелекту (ШІ) надає потужні засоби для автоматизованої обробки та аналізу візуальної інформації, що надходить з камер дронів. Інтеграція цих технологій дозволяє не просто фіксувати зображення, а й автоматично розпізнавати на них конкретні об'єкти, що має критичне значення для ефективного вирішення завдань моніторингу в численних сферах, починаючи від сільського господарства та екологічного контролю, закінчуючи забезпеченням безпеки та оборонними потребами. Особливої актуальності ця проблематика набуває в контексті необхідності швидкого та точного реагування на динамічні зміни обстановки, де людський ресурс є обмеженим або його застосування пов'язане з високими ризиками.

Незважаючи на значні досягнення, сфера автоматичного розпізнавання об'єктів дронами стикається з низкою невирішених проблем. До них належать складність ідентифікації об'єктів в умовах обмеженої видимості, за наявності перешкод, при зміні ракурсів та масштабів зйомки. Також актуальними залишаються питання оптимізації алгоритмів для роботи в режимі реального часу на борту дрона з обмеженими обчислювальними ресурсами, адаптації моделей до різноманітних типів об'єктів та мінливих умов навколишнього середовища.

Метою дипломної роботи є підвищення ефективності моніторингових операцій шляхом розробки та дослідження інтегрованої системи комп'ютерного зору та штучного інтелекту для автоматичного розпізнавання об'єктів дронами. Для досягнення поставленої мети необхідно вирішити наступні завдання:

- Проаналізувати сучасний стан досліджень та існуючі рішення в галузі інтеграції комп'ютерного зору та штучного інтелекту для розпізнавання об'єктів за допомогою БпЛА.

-Визначити ключові виклики та обмеження, що виникають при автоматичному розпізнаванні об'єктів дронами в задачах моніторингу.

- Розробити архітектуру інтегрованої системи, що поєднує можливості комп'ютерного зору та алгоритмів штучного інтелекту для детекції та класифікації об'єктів.

- Підібрати або розробити ефективні алгоритми обробки зображень та машинного навчання для розпізнавання цільових об'єктів.

- Провести експериментальні дослідження ефективності запропонованої системи на прикладі конкретних завдань моніторингу та оцінити точність і швидкість розпізнавання об'єктів.

-Сформулювати рекомендації щодо практичного застосування розробленої системи.

Об'єктом дослідження є процес автоматичного розпізнавання об'єктів на зображеннях, отриманих з безпілотних літальних апаратів, в задачах моніторингу. Предметом дослідження є методи та алгоритми комп'ютерного зору та штучного інтелекту, а також їх інтеграція для підвищення точності, швидкості та надійності розпізнавання об'єктів дронами.

Наукова новизна отриманих результатів полягає у:

- Удосконаленні підходів до інтеграції алгоритмів комп'ютерного зору та моделей штучного інтелекту, адаптованих для специфічних умов роботи дронів (зміна висоти, ракурсу, освітлення).

- Розробці модифікованих або комбінованих методів розпізнавання, що дозволяють підвищити ефективність детекції та класифікації об'єктів в умовах обмежених обчислювальних ресурсів БпЛА та зашумлених даних.

- Обґрунтуванні вибору оптимальних параметрів та архітектур нейронних мереж для конкретних завдань моніторингу з використанням дронів.

РОЗДІЛ 1 ТЕОРЕТИЧНІ ОСНОВИ ІНТЕГРАЦІЇ КОМП’ЮТЕРНОГО ЗОРУ ТА ШІ В ДРОНАХ

1.1 Основи комп’ютерного зору: принципи роботи, обробка зображень, методи розпізнавання об’єктів.

Моніторинг різноманітних об’єктів та територій є однією з ключових задач у багатьох сферах людської діяльності, включаючи безпеку, сільське господарство, екологію, інспекцію інфраструктури та пошуково-рятувальні операції. Традиційні методи моніторингу часто є затратними за часом, ресурсами та можуть бути небезпечними для виконавців. З розвитком технологій безпілотні літальні апарати (БПЛА), або дрони, стали потужним інструментом для вирішення цих завдань, надаючи можливість оперативного та гнучкого збору візуальної інформації з повітря. Однак, для перетворення зібраних даних на корисну інформацію та забезпечення автономності моніторингових місій, необхідна інтеграція передових технологій обробки зображень та аналізу даних. Саме тут на передову виходять комп’ютерний зір та штучний інтелект.

Комп’ютерний зір (КЗ) надає дронам здатність «бачити» та інтерпретувати візуальну інформацію з навколишнього середовища. Він дозволяє обробляти зображення та відео, виділяти з них значущі ознаки та структуру. Штучний інтелект (ШІ), зокрема методи машинного та глибокого навчання, забезпечує аналіз цих ознак для автоматичного розпізнавання об’єктів.

Процес комп’ютерного зору зазвичай починається із захоплення зображення або послідовності зображень (відео) за допомогою сенсора (камери). Цифрове зображення є дискретним представленням візуальної інформації, що складається з пікселів (елементів зображення). Кожен піксель має своє місцезнаходження (координати) та значення, яке може представляти інтенсивність світла (для чорно білих зображень) або комбінацію значень для різних колірних каналів (наприклад, RGB – червоний, зелений, синій).

$$I(x, y) = value \quad (1.1)$$

де I – функція зображення, (x,y) – координати пікселя, value – значення пікселя (наприклад, інтенсивність або колірний вектор).

Після захоплення зображення проходить низку етапів обробки з метою виділення корисної інформації та зменшення впливу небажаних факторів (шум, зміни освітлення тощо). Загалом, процес комп'ютерного зору можна представити як послідовність операцій, що трансформують сирі дані від сенсора у більш абстрактні

та осмислені представлення.

Процес комп'ютерного зору можна представити наступним чином (рис. 1.1):

Рис. 1.1 – Схема процесу обробки візуальної інформації в системах комп'ютерного зору

На відміну від людського зору, який є надзвичайно адаптивним і ефективним, комп'ютерним системам бракує вродженого розуміння контексту та тривимірної структури світу лише з плоского зображення.

Обробка зображень – це сукупність методів та алгоритмів, спрямованих на покращення якості зображення, трансформацію його вигляду або підготовку до подальшого аналізу. Цей етап є необхідним попереднім кроком перед спробою витягти з зображення семантичний зміст. Ключові задачі обробки зображень включають:

попереднє оброблення (Pre-processing) – це покращити якість зображення та усунути спотворення, спричинені процесом захоплення або умовами середовища.

шумозаглушення (Noise Reduction) зображення – це випадкові флуктуації яскравості або кольору (шум), які можуть ускладнити подальший аналіз. Методи шумозаглушення включають лінійні фільтри (гауссівський фільтр) та нелінійні фільтри (медіанний фільтр).

Приклад дії гауссівського або медіанного фільтра представленні на (рис. 1.2).

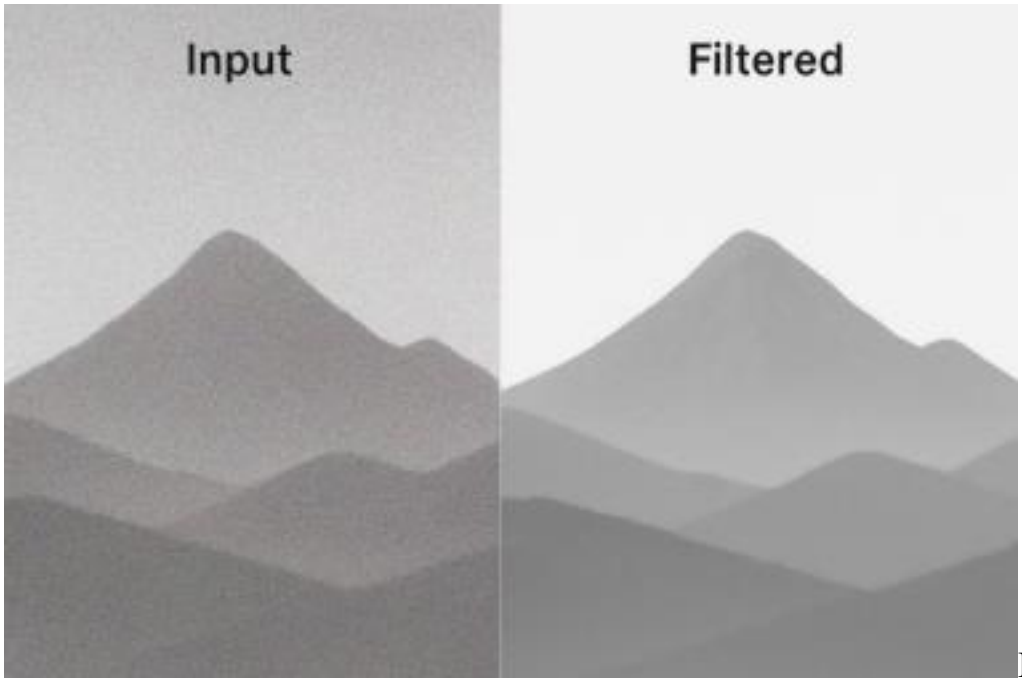


Рис. 1.2 – Приклад дії гауссівського або медіанного фільтра на зображенні із шумом

корекція освітлення та контрасту – це нерівномірне освітлення або низький контраст можуть ускладнити виділення об'єктів. Методи, такі як гістограмна еквалізація, адаптивна гістограмна еквалізація (CLAHE) або корекція гамми, можуть покращити розподіл яскравостей.

геометричні перетворення – це виправлення спотворень, спричинених оптикою камери або рухом (наприклад, дисторсія об'єктива, афінні перетворення для вирівнювання).

фільтрація (Filtering) – це застосування масок або ядер (kernel) до зображення для виконання певних операцій над кожним пікселем на основі його околу. Фільтрація є основою для багатьох операцій, включаючи шумозаглушення, виділення границь, підвищення різкості тощо. Ось кілька прикладів ядер фільтрів для типових задач:

Таблиця 1.1 – Приклади ядер фільтрів для різних задач.

Задача обробки зображення	Ядро фільтра (Матриця 3x3)	Призначення
Згладжування (Усереднення)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	Зменшує шум та деталі шляхом усереднення значень сусідніх пікселів.

Виділення горизонтальних границь (Sobel X)	-1 0 1 [ ] -2 0 2 -1 0 1	Виділяє ділянки зображення зі швидкою зміною інтенсивності по горизонталі
Виділення вертикальних границь (Sobel Y)	-1 -2 -1 [ ] 0 0 0 1 2 1	Виділяє ділянки зображення зі швидкою зміною інтенсивності по вертикалі.
Підвищення різкості	0 -1 -0 [ ] -1 5 -1 0 -1 0	Підкреслює деталі та границі, роблячи зображення більш чітким.

Ключова операція – згортка (convolution), у вигляді рівння:

$$(I * K)(x,y) = \sum_i \sum_j I(x-i, y-j) K(i,j) \quad (1.2)$$

де I – вхідне зображення,

K – ядро фільтра (маска),

(x,y) – координати пікселя, (i,j) – координати в ядрі.

виділення ознак (Feature Extraction) – це перетворення сирих піксельних даних у більш абстрактні та інформативні представлення, які є стійкими до незначних змін (наприклад, невеликих поворотів, змін масштабу, освітлення). Ознаки можуть бути низькорівневими (графічні примітиви) або високорівневими (більш комплексні дескриптори).

виділення границь (Edge Detection) – є важливими ознаками, оскільки часто відповідають контурам об'єктів. Вони несуть багато інформації про форму та структуру об'єктів на зображенні. Виділення границь є фундаментальним кроком у багатьох задачах комп'ютерного зору, таких як сегментація зображень, розпізнавання об'єктів та вимірювання.

Для виділення границь використовуються різні оператори, серед яких популярні: Sobel, Prewitt, Roberts, Canny. Оператор Canny є одним з найефективніших та найпоширеніших методів виділення границь. Він включає кілька послідовних етапів обробки:

- Шумозаглушення.

- Обчислення градієнтів.

- Придушення немаксимумів (Non-maximum Suppression).

- Порогова фільтрація з гістерезисом (Hysteresis Thresholding). На Рис. 1.3 показано приклад застосування оператора Canny для виділення границь на зображенні.

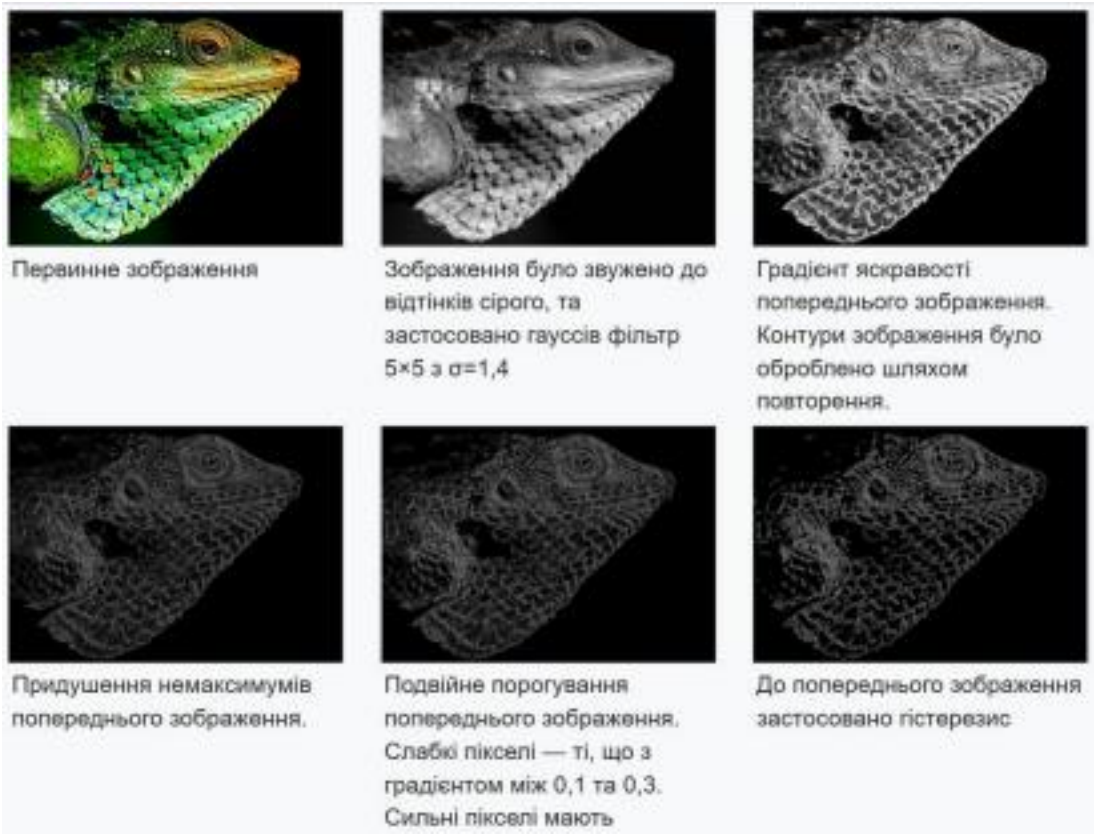


Рис. 1.3 – Приклад виділення границь оператором Canny на зображенні

- виділення кутів (Corner Detection) – є точками з високою локальною зміною інтенсивності у кількох напрямках і є стійкими до повороту.

- виділення текстур (Texture Analysis) – це аналіз просторового розподілу інтенсивності або кольору для опису поверхні об'єкта.

- високорівневі дескриптори – це більш складні методи, що описують локальні ділянки зображення таким чином, щоб вони були інваріантними або стійкими до

різних перетворень.

- сегментація (Segmentation) – це поділ зображення на області або об'єкти.

Розпізнавання об’єктів (Object Recognition) – це задача визначення, чи присутній певний об’єкт або об’єкти на зображенні, та, у разі присутності, визначення його/їх класу. Більш складна задача – детектування об’єктів (Object Detection) – включає не тільки визначення класу, але й локалізацію об’єкта на зображенні за допомогою обмежувального прямокутника (bounding box). Традиційні методи розпізнавання об'єктів, що передували епосі глибокого навчання, часто покладалися на вручну розроблені ознаки та класичні алгоритми машинного навчання:

1. На основі шаблонів (Template Matching): Пошук на зображенні ділянок, які максимально схожі на заздалегідь визначений шаблон об'єкта. Метод простий, але дуже чутливий до змін масштабу, повороту, освітлення та деформацій об'єкта.

2. На основі ознак (Feature-based Matching): Цей підхід передбачає виділення на зображенні та на еталонному зображенні об'єкта стійких локальних ознак (SIFT - Scale-Invariant Feature Transform).

На Рис. 1.4 показано приклад процесу зіставлення ознак SIFT для знаходження та розпізнавання об'єкта на іншому зображенні.

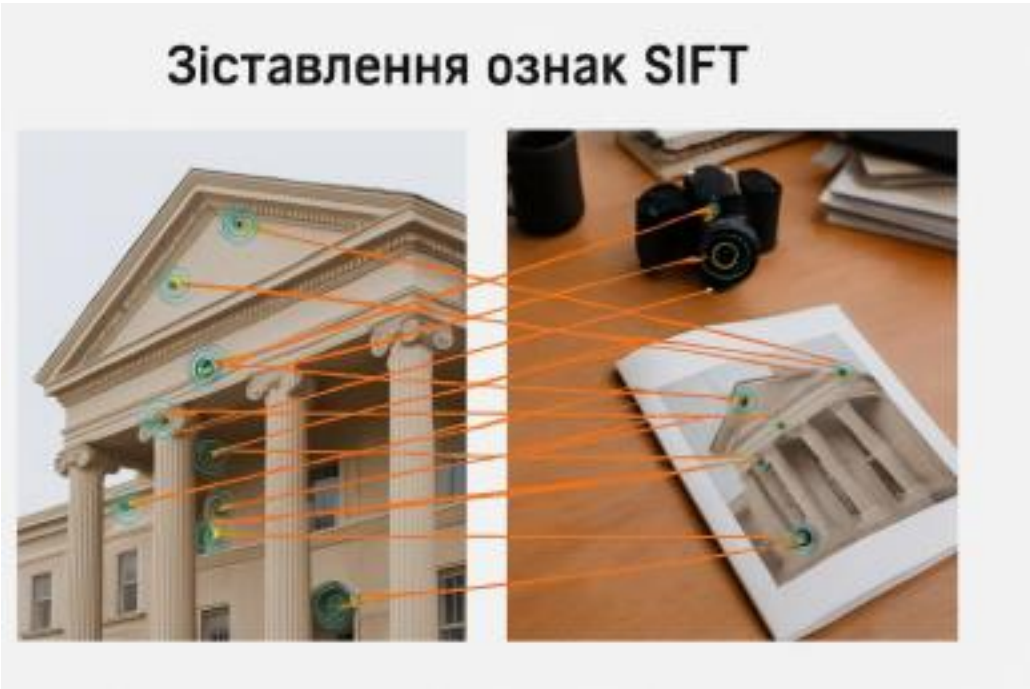


Рис. 1.4 – Приклад зіставлення ознак SIFT для розпізнавання об'єкта на іншому зображенні

3. Статистичні методи та класифікатори: після виділення ознак з зображення, їх

можна використовувати як вхід для статистичних класифікаторів, навчених розрізняти різні класи об'єктів.

Використання комп'ютерного зору на дронах ставить низку специфічних викликів, пов'язаних з умовами зйомки та обмеженнями платформи: 1. Зміни ракурсу та масштабу: Висота польоту та траєкторія дрона постійно змінюються, що призводить до значних варіацій у масштабі та ракурсі об'єктів на зображеннях.

2. Динамічні зміни освітлення: Умови освітлення можуть швидко змінюватися залежно від часу доби, погодних умов та руху дрона (тіні від хмар, власна тінь). 3. Рух та вібрації: Платформа дрона не є повністю стабільною, що спричиняє розмиття зображень (motion blur) та вібрації.

4. Обмежена роздільна здатність та якість зображення: Залежно від висоти та типу камери, об'єкти можуть бути зображені з низькою роздільною здатністю. 5. Оклюзії: Об'єкти можуть бути частково або повністю перекриті іншими об'єктами, рослинністю, будівлями тощо.

6. Різноманітність об'єктів: Об'єкти одного класу можуть мати значні відмінності у зовнішньому вигляді (наприклад, різні моделі автомобілів, різні види тварин).

7. Необхідність роботи в реальному часі: Для деяких задач моніторингу (напр., слідування за об'єктом, уникнення перешкод) обробка повинна відбуватися з мінімальною затримкою.

Ці виклики вимагають використання надійних та ефективних алгоритмів комп'ютерного зору, які можуть впоратися з такими варіаціями. Як буде показано у наступному розділі, методи, засновані на глибокому навчанні, виявилися особливо ефективними для подолання багатьох з цих проблем завдяки їх здатності автоматично навчатися складних та стійких ознак.

1.2 Штучний інтелект у задачах розпізнавання: машинне навчання, нейронні мережі (CNN, YOLO, SSD)

Штучний інтелект (ШІ) – це широка галузь комп'ютерних наук, що займається створенням систем, здатних виконувати завдання, які зазвичай вимагають людського інтелекту, такі як сприйняття, прийняття рішень та розпізнавання образів.

У контексті розпізнавання об’єктів на зображеннях, найбільш значущим підрозділом ШІ є машинне навчання.

Машинне навчання (Machine Learning, ML) – це підхід до створення ШІ, при якому системи вчаться на даних, а не програмуються на виконання конкретних завдань. Існують різні парадигми машинного навчання:

- навчання з учителем (Supervised Learning). Алгоритм навчається на наборі даних, що містить як вхідні дані, так і відповідні їм правильні «мітки» (наприклад, зображення та клас об'єкта на ньому).

- навчання без учителя (Unsupervised Learning). Алгоритм навчається на наборі даних без міток.

- навчання з підкріпленням (Reinforcement Learning). Агент навчається діяти в певному середовищі, отримуючи «винагороду» або «покарання» за свої дії, з метою максимізації сукупної винагороди.

У задачах розпізнавання об'єктів домінує навчання з учителем, де система навчається на великих наборах мічених зображень (ImageNet, COCO), що містять тисячі або мільйони зображень з обмежувальними прямокутниками та класами об'єктів.

Глибоке навчання (Deep Learning, DL) є підгалуззю машинного навчання, що ґрунтується на використанні штучних нейронних мереж з багатьма (глибокими) шарами. Ключова відмінність DL від традиційного ML полягає в тому, що глибокі нейронні мережі здатні автоматично навчатися ієрархічних представлень (ознак) даних без необхідності їх ручної інженерії. Це робить глибоке навчання надзвичайно ефективним для роботи зі складними, багатовимірними даними, такими як зображення, аудіо та текст.

Штучна нейронна мережа (ШНМ) – це обчислювальна модель, натхненна структурою та функціонуванням біологічних нейронних мереж. Вона складається з великої кількості взаємопов'язаних штучних нейронів (або перцептронів), організованих у шари.

Нейрон – це базовий обчислювальний елемент. Він приймає набір вхідних значень, множить їх на відповідні ваги, додає зміщення (bias), сумує ці значення і пропускає результат через функцію активації.

$$z = \sum_{i=1}^{n} x_i w_i + b \quad (1.3)$$

$$a = f(z) \quad (1.4)$$

де xi – вхідні значення, wi – ваги, b – зміщення, z – зважена сума, f – функція активації, a – вихід нейрона.

Навчання (Тренування): Процес налаштування ваг та зміщень мережі шляхом подачі великої кількості мічених даних. Мережа робить прогноз, результат порівнюється з правильним значенням за допомогою функції втрат (Loss Function), яка вимірює помилку. Градієнт функції втрат обчислюється за допомогою алгоритму зворотного поширення помилки (Backpropagation), і ваги коригуються у напрямку зменшення помилки за допомогою оптимізаційних алгоритмів (напр., градієнтний спуск або його варіанти: Adam, SGD з імпульсом).

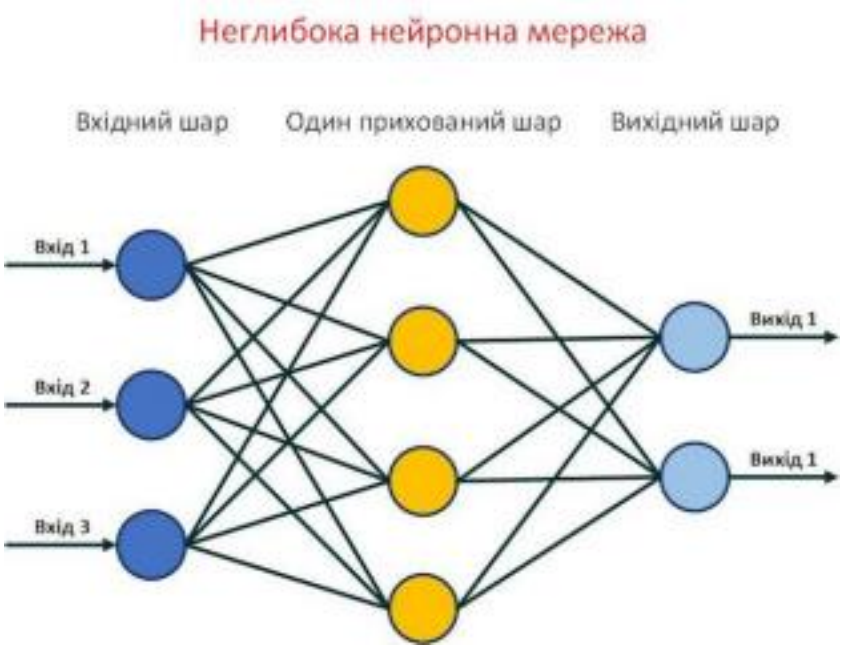


Рис. 1.5 – Спрощена схема багатошарової нейронної мережі

Згорткові нейронні мережі (CNN або ConvNet) – це спеціалізований тип нейронних мереж, розроблений для ефективної обробки даних із сітчастою топологією, таких як зображення. Вони використовують принцип згортки, що дозволяє мережі автоматично навчатися просторових ієрархій ознак. CNN стали стандартом de facto для багатьох задач комп'ютерного зору, включаючи класифікацію зображень, сегментацію та детектування об'єктів.

Еволюція Архітектур Згорткових Нейронних Мереж (CNN). Згорткові нейронні мережі пройшли значний шлях розвитку, причому кожне нове покоління архітектур

вносило інновації для подолання обмежень попередніх моделей та досягнення нових рівнів продуктивності, особливо у задачах комп'ютерного зору.

1. Рання архітектура LeNet, розроблена Яном ЛеКуном та його колегами наприкінці 1980-х - на початку 1990-х років, є однією з найперших і найвпливовіших CNN. Вона була спеціально створена для розпізнавання рукописних цифр (на наборі даних MNIST).

Структура: LeNet-5 мала відносно просту, але ефективну структуру для свого часу:

- Кілька шарів, що чергуються: згорткові шари (для виявлення локальних ознак, таких як краї та кути) та шари пулінгу (підвибірки, для зменшення розмірності та інваріантності до невеликих зсувів).

- Один або кілька повнозв'язних шарів у кінці для класифікації. -

Використовувала активаційні функції, такі як сигмоїда або гіперболічний тангенс.

- Вплив: LeNet продемонструвала ефективність ієрархічного вивчення ознак за допомогою згорткових та пулінгових шарів. Вона заклала фундаментальні принципи побудови CNN і стала прототипом для багатьох наступних архітектур, довівши потенціал глибокого навчання для задач розпізнавання образів.

2. Проривна архітектура AlexNet, розроблена Алексом Крижевським, Іллею Суцкевером та Джеффрі Хінтоном, стала переломним моментом, здобувши переконливу перемогу в конкурсі ImageNet Large Scale Visual Recognition Challenge (ILSVRC) у 2012 році. AlexNet значно перевершила традиційні методи комп'ютерного зору завдяки кільком ключовим інноваціям:

- Використання ReLU (Rectified Linear Unit): Замість сигмоїди/tanh, AlexNet використовувала ReLU ($f(x)=\max(0,x)$) як активаційну функцію. ReLU дозволила значно прискорити процес навчання (у 5-6 разів), оскільки вона не насичується для позитивних значень і має простішу похідну, що допомагає уникнути проблеми зникаючих градієнтів.

- Використання Dropout: Це техніка регуляризації, яка допомагає запобігти

перенавчанню. Під час тренування Dropout випадковим чином «вимикає» (встановлює вихід на нуль) певну частку нейронів у повнозв'язних шарах. Це змушує мережу вивчати більш стійкі та менш залежні одна від одної ознаки.

- Навчання на GPU: AlexNet була однією з перших глибоких мереж, яка ефективно використовувала графічні процесори (GPU) для тренування. Паралельні обчислювальні можливості GPU дозволили тренувати значно більшу та глибшу мережу на великому наборі даних ImageNet за прийнятний час, що було неможливо з використанням лише CPU.

- Глибина: Мережа була значно глибшою за LeNet (8 шарів, що навчаються).

Успіх AlexNet відродив інтерес до нейронних мереж та глибокого навчання, продемонструвавши їхню перевагу у складних задачах комп'ютерного зору та започаткувавши сучасну еру глибокого навчання.

3. Дуже глибокі мережі VGGNet, розроблені Visual Geometry Group з Оксфордського університету (Симонян та Зіссерман), була представлена на ILSVRC 2014. Вона відома своєю простотою та великою глибиною (VGG16 та VGG19 мали 16 та 19 шарів відповідно). Основна ідея VGGNet полягала у використанні виключно малих згорткових фільтрів розміром 3×3 (з кроком 1) та пулінгу 2×2 (з кроком 2) у всій мережі. Замість використання великих фільтрів (як 11×11 або 5×5 в AlexNet) на перших шарах, VGG використовувала стеки з кількох шарів 3×3. Переваги малих фільтрів:

- ефективне рецептивне поле;

- більше нелінійності;

- менше параметрів;

- вплив;

4. Inception Modules GoogleNet/Inception (також відома як Inception v1), розроблена командою Google, перемогла на ILSVRC 2014 (одночасно з VGG). Її основна інновація — Inception модуль. Замість того, щоб просто робити мережу глибшою (як VGG), Inception модуль прагнув зробити її «ширшою». Ідея полягала в

тому, щоб в одному блоці (модулі) паралельно застосовувати згортки з різними розмірами фільтрів (1×1, 3×3, 5×5) та шар максимального пулінгу (3×3).

1. Паралельні шляхи: вхідний тензор ознак подається на кілька паралельних гілок:

- Згортка 1×1; Згортка 1×1, за якою слідує згортка 3×3; Згортка 1×1, за якою слідує згортка 5×5; Максимальний пулінг 3×3, за яким слідує згортка 1×1. 2. Конкатенація: виходи всіх цих гілок конкатенуються (з'єднуються) за глибиною (по осі каналів), формуючи вихідний тензор модуля.

3. Згортки 1×1 для зменшення розмірності: важливим елементом є використання згорток 1×1 перед «дорогими» згортками 3×3 та 5×5. Ці згортки 1×1 зменшують кількість каналів (глибину) вхідного тензора, що значно знижує обчислювальну складність та кількість параметрів, не надто втрачаючи інформацію.

Inception модуль дозволяє мережі виявляти ознаки на різних масштабах одночасно (маленькі фільтри для локальних деталей, великі — для більш глобальних). Це робить мережу більш ефективною та точною. GoogleNet досягла високої точності при значно меншій кількості параметрів порівняно з VGG.

5. Residual Connections (ResNet - Residual Network), розроблена Кайміном Хе та його колегами з Microsoft Research, перемогла на ILSVRC 2015 з вражаючою перевагою. ResNet дозволила успішно тренувати нейронні мережі небаченої раніше глибини (до 152 шарів і навіть більше 1000). При спробі зробити мережі ще глибшими (навіть глибшими за VGG), виникла проблема деградації (degradation problem): точність на тренувальному наборі спочатку насичувалася, а потім починала падати зі збільшенням глибини. Це не було спричинено перенавчанням:

- ResNet вводить «короткі замикання» або «пропускні з'єднання» (skip connections), які пропускають один або кілька шарів.

- Типовий залишковий блок намагається навчити функцію H(x) (бажане відображення). Замість того, щоб шари напряму апроксимували H(x), вони тепер апроксимують залишкову функцію $F(x)=H(x)-x$.

- Вихід блоку тоді стає $H(x)=F(x)+x$. Додавання x (ідентичності) здійснюється за

допомогою skip connection, яке передає вхід x блоку напряму до його виходу, де він поелементно додається до результату роботи шарів блоку F(x).

Вплив нейронних мереж ResNet зробила революцію в глибокому навчанні, дозволивши ефективно тренувати надзвичайно глибокі мережі. Основні шари CNN:

- Згортковий шар (Convolutional Layer): Це серце CNN. Замість того, щоб кожен нейрон був пов'язаний з усіма пікселями вхідного зображення, нейрони в згортковому шарі пов'язані лише з невеликою локальною ділянкою (рецептивним полем) вхідних даних через застосування фільтрів (або ядер).

Кожен фільтр «згортається» (ковзає) по всій ширині та висоті вхідних даних, обчислюючи скалярний добуток між значеннями фільтра та відповідними значеннями вхідної ділянки. Ця операція породжує карту ознак (feature map), яка підсвічує місця на зображенні, де фільтр виявив певну ознаку (напр., вертикальну грань, кут). Мережа навчається ваг цих фільтрів, які, по суті, стають детекторами ознак.

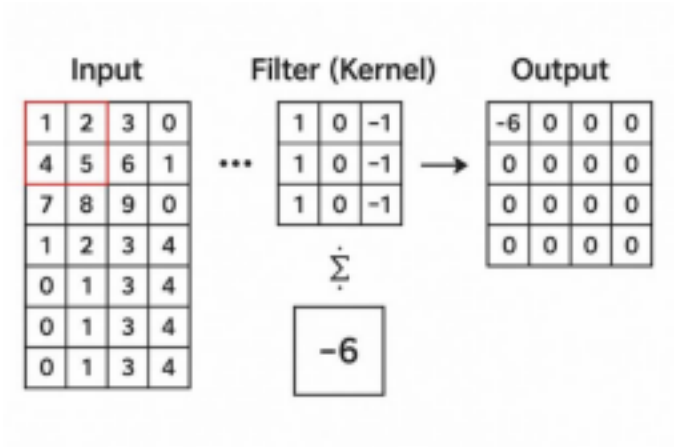


Рис. 1.6 – Ілюстрація операції згортки

-Шар активації (Activation Layer): Після згорткового шару застосовується функція активації (зазвичай ReLU) до кожного елемента карти ознак для введення нелінійності в модель.

- Пулінговий шар (Pooling Layer): Призначений для зменшення просторової розмірності карт ознак, зменшення кількості параметрів та обчислень, а також для надання моделі стійкості до невеликих просторових зсувів вхідних даних. Найпоширеніші типи пулінгу: Max Pooling (вибирає максимальне значення в межах вікна) та Average Pooling (обчислює середнє значення).

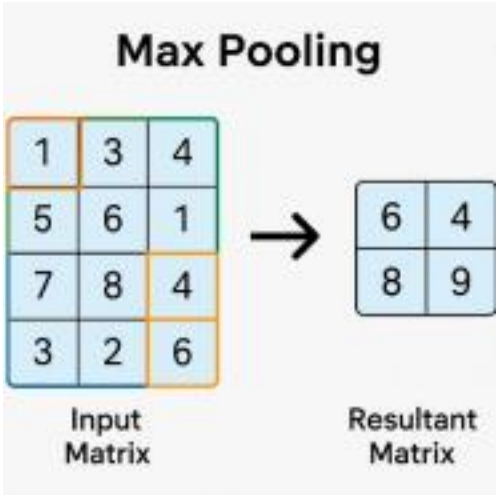


Рис. 1.7 – Ілюстрація операції Max Pooling

- Повнозв'язний шар (Fully Connected Layer): Зазвичай розташовується наприкінці архітектури CNN. Він приймає «розгорнуті» (flattened) вихідні дані з попередніх згорткових або пулінгових шарів і підключає кожен нейрон цього шару до кожного нейрона попереднього шару.

Типова архітектура CNN для класифікації складається з послідовності згорткових шарів, шарів активації та пулінгових шарів, за якими слідує один або кілька повнозв'язних шарів і фінальний вихідний шар (з функцією Softmax для отримання ймовірностей класів).

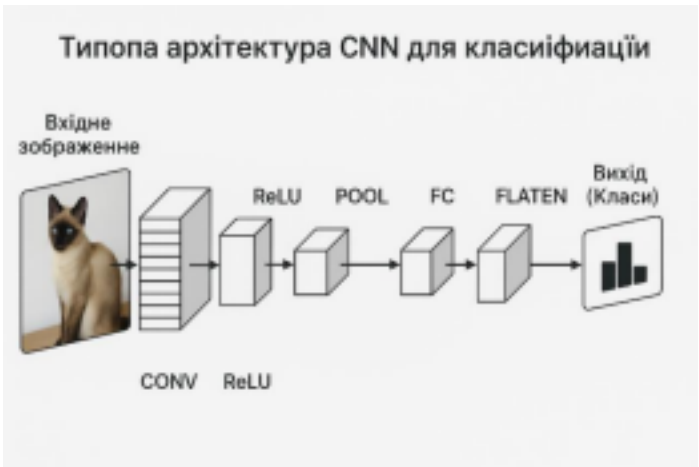


Рис. 1.8 – Типова архітектура CNN для класифікації зображень

CNN надзвичайно ефективні для вивчення ознак безпосередньо з пікселів зображення. Чим глибша мережа, тим складніші та абстрактніші ознаки вона може вивчити. Нейронні мережі для детектування об’єктів (Object Detection) є складнішою за класифікацію, оскільки необхідно одночасно визначити:

1. Чи є об'єкт певного класу на зображенні.
2. Де саме знаходиться цей об'єкт (локалізація).

3. Якого класу цей об'єкт.

Історично методи детектування об'єктів еволюціонували від двофазних підходів до однофазних, кожен з яких має свої переваги та недоліки. 1. Двофазні детектори (Two-Stage Detectors): Ці методи спочатку генерують набір потенційних регіонів, які можуть містити об'єкти (Region Proposals), а потім класифікують кожен запропонований регіон та уточнюють його місцезнаходження. Приклади:

- R-CNN (Region-based Convolutional Neural Network): Перший успішний метод, що поєднав Region Proposals (отримані, напр., методом Selective Search) з CNN для класифікації та лінійною регресією для уточнення bounding box. Дуже точний, але надзвичайно повільний.

- Fast R-CNN: Покращення R-CNN за рахунок обчислення ознак CNN для всього зображення один раз, а потім проектування запропонованих регіонів на карту ознак (Region of Interest Pooling). Значно швидший за R-CNN, але етап генерації Region Proposals все ще є вузьким місцем. Вирішує проблему повільної генерації Region Proposals, замінюючи її на RPN (Region Proposal Network) – невелику нейронну мережу, навчену пропонувати регіони прямо з карт ознак CNN. Faster R-CNN є швидшим за попередні версії і досяг високої точності, ставши стандартом на певний час.

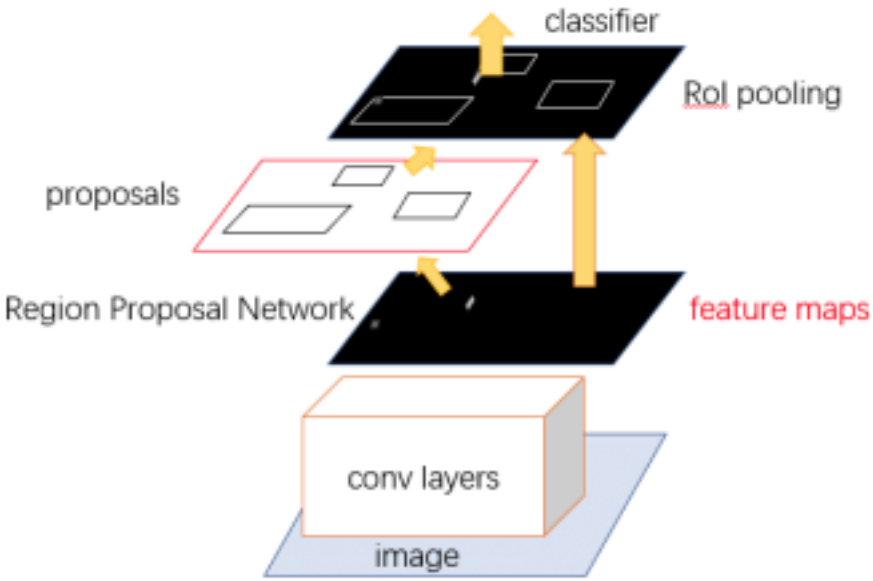


Рис. 1.9 – Принципова схема Faster R-CNN

Двофазні детектори, як правило, досягають вищої точності, але є обчислювально більш складними і повільнішими, що може бути критично для застосувань на дронах, де важлива швидкість обробки в реальному часі.

2. Однофазні детектори (One-Stage Detectors): Ці методи виконують класифікацію та локалізацію об'єктів за один прохід нейронною мережею, без явного етапу генерації Region Proposals. Вони поділяють зображення на сітку і для кожної комірки сітки передбачають наявність об'єктів, їх клас та координати обмежувальних прямокутників. Це значно прискорює процес детектування. Найвідоміші приклади: YOLO та SSD.

- YOLO (You Only Look Once) представлена Джозефом Редмоном та іншими у 2015 році, стала революційним підходом до задачі виявлення об'єктів. На відміну від попередніх методів, які або використовували ковзне вікно, або ділили задачу на два етапи (генерація пропозицій регіонів + класифікація), YOLO розглядає виявлення об'єктів як регресійну задачу. Вона прогнозує обмежувальні рамки (bounding boxes) та ймовірності класів безпосередньо з повного зображення за один прохід нейронної мережі. Архітектура мережі:

1. Базова мережа (Backbone): Архітектура YOLOv1 була натхненна мережею GoogleNet для класифікації зображень. Вона складається з:

- 24 згорткових шарів для вилучення ознак із зображення. Ці шари включають чергування згорток 1×1 (для зменшення глибини / кількості каналів) та згорток 3×3. - 2 повнозв'язних шарів (Fully Connected layers) для здійснення фінальних передбачень (координати рамок, оцінки впевненості та ймовірності класів). 2. Вхід та Вихід:

- Вхід: Мережа приймає на вхід зображення, розмір якого змінено до 448×448 пікселів.

- Вихід: Вихід мережі - це тензор розміром S×S×(B×5+C). У статті YOLOv1 використовувалися S=7, B=2, C=20 (для набору даних PASCAL VOC). Таким чином, вихідний тензор мав розмір 7×7×(2×5+20)=7×7×30.

3. Механізм Виявлення (Сітка):

- Зображення ділиться на сітку S×S (наприклад, 7×7).

- Якщо центр об'єкта потрапляє в певну комірку сітки (grid cell), ця комірка стає «відповідальною» за виявлення цього об'єкта.

- Кожна комірка сітки (i,j) прогнозує:

- B обмежувальних рамок (наприклад, B=2). Кожна рамка описується 5 значеннями: (x,y,w,h,c).

- (x,y): Координати центру рамки відносно меж самої комірки сітки (значення від 0 до 1).

- (w,h): Ширина та висота рамки відносно розміру всього зображення (значення від 0 до 1).

- c: Оцінка впевненості (confidence score). Вона визначається як P(Object)×IoUpredtruth. Тобто, це ймовірність того, що в рамці є об'єкт, помножена на Intersection over Union (IoU) між передбаченою рамкою та реальною (ground truth) рамкою. Якщо в комірці немає об'єкта, P(Object) має бути 0.

- C умовних ймовірностей класів P(Classk|Object). Це ймовірність того, що виявлений об'єкт належить до класу k, за умови, що в комірці взагалі є об'єкт. Прогнозується один набір таких ймовірностей на комірку, незалежно від кількості рамок B.

4. Фінальні Детекції: Для отримання кінцевих рамок використовується Non Max Suppression (NMS). Спочатку відкидаються всі рамки з низькою оцінкою впевненості (c). Потім, для решти рамок, які виявляють один і той же об'єкт, залишається тільки та, що має найвищу впевненість, а інші (з високим IoU з нею) пригнічуються.

Переваги YOLOv1:

1. Висока швидкість: Оскільки детектування відбувається за один прохід мережі, YOLOv1 надзвичайно швидка (базова модель - 45 FPS, швидка версія - до 155 FPS на тогочасному GPU), що дозволяє використовувати її в реальному часі.

2. Глобальний контекст: Мережа бачить все зображення під час передбачення, тому вона неявно враховує глобальний контекст. Це допомагає зменшити кількість помилкових спрацьовувань на фоні (background false positives) порівняно з методами, що працюють з окремими регіонами.

3. Навчання узагальнених представлень: YOLO вивчає дуже узагальнені представлення об'єктів, що робить її ефективною навіть при тестуванні на нових доменах або несподіваних входах.

Недоліки YOLOv1:

1. Проблеми з малими об'єктами: Це один з головних недоліків. Оскільки кожна комірка сітки може передбачити лише обмежену кількість рамок (B=2) і лише один клас, YOLOv1 важко виявляти групи малих об'єктів, особливо якщо їхні центри потрапляють в одну й ту ж комірку сітки. Просторова роздільна здатність сітки (7×7) є досить грубою.

2. Труднощі з близькими об'єктами: З тієї ж причини (одна комірка = один клас, B рамок) модель погано розділяє об'єкти, що знаходяться дуже близько один до одного.

3. Низька точність локалізації: Порівняно з двостадійними детекторами (напр., Faster R-CNN), YOLOv1 часто має менш точні обмежувальні рамки. Пряма регресія координат є складнішою задачею, ніж класифікація запропонованих регіонів.

4. Проблеми з незвичними співвідношеннями сторін: Модель навчається на типових формах об'єктів з тренувального набору даних. Їй важко впоратися з об'єктами, що мають нові або незвичні співвідношення сторін (aspect ratios).

5. Обмежена кількість виявлень на комірку: Прогнозування лише B рамок на комірку обмежує щільність об'єктів, які можна виявити.

Незважаючи на недоліки, YOLOv1 стала проривом і заклала основу для подальших версій (YOLOv2, YOLOv3, YOLOv4, YOLOv5 та інші), які поступово долали ці обмеження, зберігаючи при цьому основну ідею швидкості та ефективності.

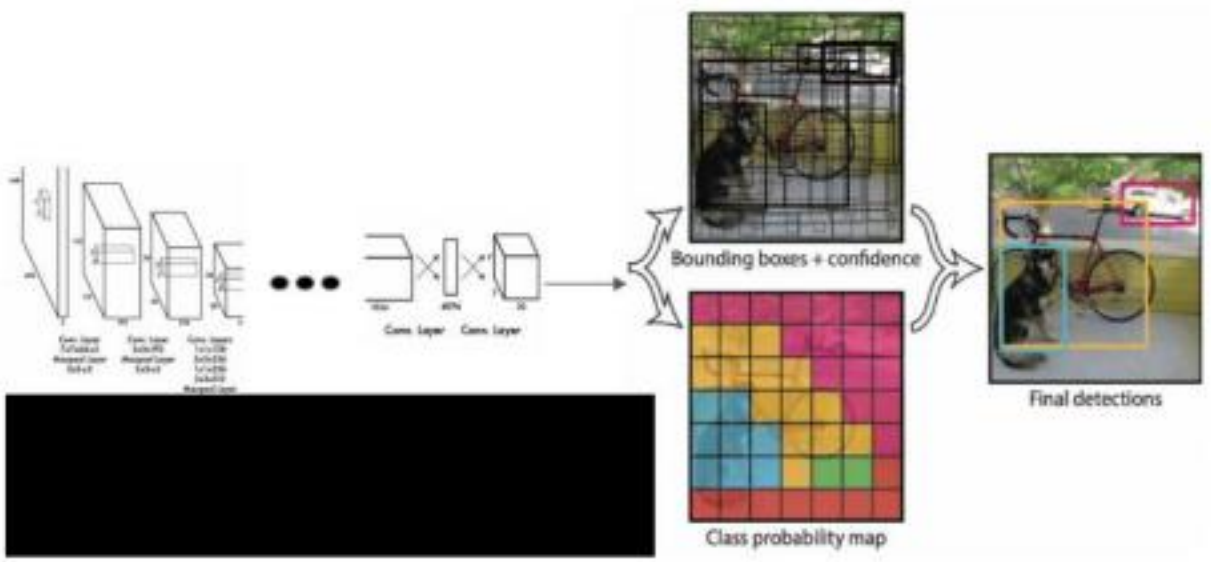


Рис. 1.10 – Принципова схема YOLO

Еволюція YOLO:

- YOLOv2 (YOLO9000): Покращення точності та швидкості. Використання пакетної нормалізації, якірних боксів (anchor boxes), розмірності кластерів, прямого передбачення локації, більш тонкого гранулювання ознак (fine-grained features), використання спеціальної backbone мережі (Darknet-19).

- YOLOv3: Подальші покращення точності, особливо для малих об'єктів, за рахунок передбачення bounding boxes на трьох різних масштабах (з використанням Feature Pyramid Network - FPN подібного підходу) та використання більш потужної backbone мережі (Darknet-53). Використання бінарної крос-ентропії для передбачення класів.

- YOLOv4/v5/v7 та інші варіації: Подальша оптимізація, використання нових трюків навчання (data augmentation, regularization), нових активаційних функцій (Mish), нових backbone мереж та оптимізацій, спрямованих на підвищення точності та швидкості.

SSD (Single Shot Detector): Ще один популярний однофазний детектор, що досягає балансу між швидкістю та точністю. Принцип роботи SSD: Використовує одну згорткову мережу (напр., VGG, ResNet) як backbone для вилучення ознак.

Ключова відмінність – передбачення обмежувальних прямокутників та ймовірностей класів виконується на кількох картах ознак різного масштабу. Це

дозволяє моделі ефективніше детектувати об'єкти різних розмірів.

SSD використовує набір попередньо визначених «default boxes» (подібних до anchor boxes) на кожній карті ознак і передбачає зміщення до цих default boxes та ймовірності класів для кожного боксу.

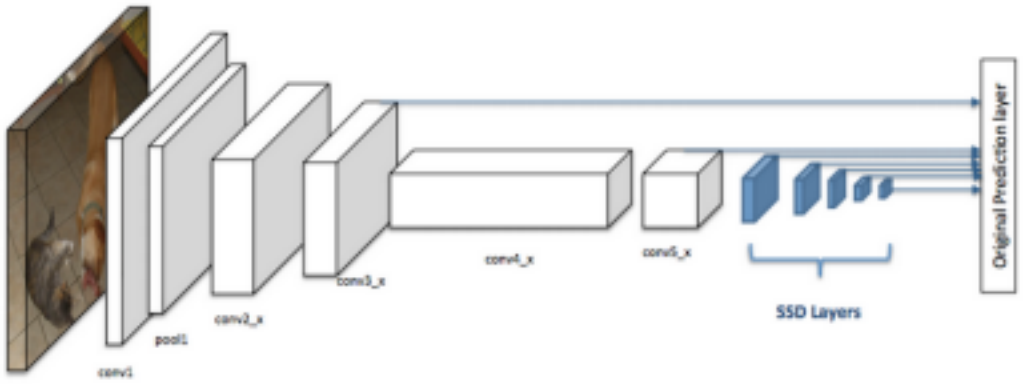


Рис. 1.11 – Принципова схема SSD

Переваги: SSD працює у реальному часі, адже виконує детектування об’єктів за один прохід (single shot) через мережу, на відміну від R-CNN, який має кілька етапів.

Недоліки SSD**:** Порівняно з більш складними моделями (наприклад, Faster R-CNN), SSD може гірше справлятися з об’єктами малого розміру. Менша гнучкість при складних сценах – SSD може бути менш ефективним у ситуаціях, де об’єкти перекриваються або присутня складна фонова інформація.

Таблиця 1.2 – Порівняння методів детектування об'єктів на мережах R-CNN (Faster R-CNN), YOLO та SSD.

Критерій	Faster R-CNN (Представник R-CNN сім'ї)	YOLO (You Only Look Once)	SSD (Single Shot MultiBox Detector)
Підхід	Двофазний (Two-stage): Генерація пропозицій регіонів (RPN) Класифікація/Регресія рамок	Однофазний (One stage): Пряма регресія координат та класів з комірок сітки (grid cells)	Однофазний (One-stage): Використання анкерних/дефолтних рамок (anchor/default boxes) на кількох картах ознак

Швидкість	Відносно низька (порівняно з однофазними). Зазвичай не в реальному часі.	Дуже висока. Добре підходить для реального часу (особливо оптимізовані версії).	Висока. Зазвичай швидше за Faster R-CNN, але може бути повільніше за найшвидші версії YOLO. Підходить для реального часу.
Точність	Дуже висока. Часто слугує еталоном точності, особливо для локалізації.	Зазвичай нижча, ніж у Faster R-CNN (особливо ранні версії та для малих об'єктів). Пізніші версії значно покращили точність.	Хороший компроміс між швидкістю та точністю. Часто точніше за ранні YOLO, але може поступатися Faster R CNN. Краще обробляє малі об'єкти, ніж YOLOv1.
Обробка масштабу	Добре. RPN генерує пропозиції різних розмірів та співвідношень сторін.	Ранні версії мали проблеми з малими об'єктами. Пізніші версії (YOLOv3+) використовують передбачення на кількох масштабах (з різних карт ознак).	Дуже добре. Спеціально розроблена для обробки різних масштабів шляхом використання карт ознак різної роздільної здатності та відповідних анкерних рамок.
Складність	Висока. Складається з кількох компонентів (backbone, RPN, RoI Pooling/Align, класифікатор/регресор). Навчання складніше.	Низька/Середня. Одна наскрізна мережа. Функція втрат може бути складною. Навчання відносно просте.	Середня. Одна наскрізна мережа, але концепція багатьох карт ознак, анкерних рамок та стратегії їх підбору (matching) додає складності.

Для об'єктивної оцінки ефективності алгоритмів детектування об'єктів використовуються спеціальні метрики:

IoU (Intersection over Union): Коефіцієнт перекриття – вимірює ступінь перекриття між передбаченим обмежувальним прямокутником та істинним (ground truth) прямокутником. Обчислюється як площа перетину двох прямокутників, поділена на площу їх об'єднання.

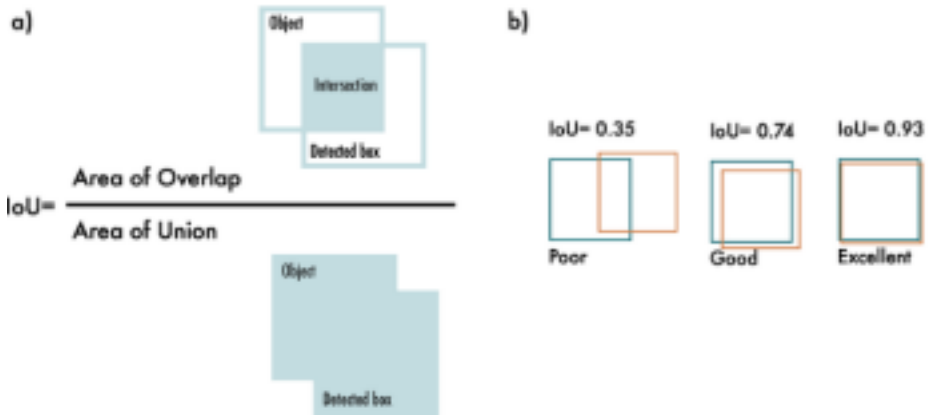


Рис. 1.12 – Ілюстрація обчислення IoU

IoU використовується як порогове значення для визначення, чи вважати передбачений прямокутник «правильним» (True Positive, TP). Наприклад, якщо IoU > 0.5, прямокутник вважається TP.

Виклики використання ШІ для розпізнавання на дронах стикається з низкою викликів:

1. Обчислювальні ресурси: Глибокі нейронні мережі, особливо великі архітектури (напр., ті, що досягають найвищої точності на десктопних GPU), вимагають значних обчислювальних потужностей та пам'яті. Бортові комп'ютери дронів мають обмежені ресурси.

2. Енергоспоживання: Високопродуктивні обчислювальні модулі споживають багато енергії, що критично обмежує час польоту дрона, який живиться від батареї. 3. Набір даних: Для ефективного навчання моделей глибокого навчання потрібні великі обсяги мічених даних, специфічних для умов зйомки з дронів (різні висоти, ракурси, погодні умови, типи об'єктів, що моніторяться). Збір та розмітка таких даних є складним та затратним процесом.

4. Режим реального часу: Для багатьох задач моніторингу потрібне детектування в реальному часі, що вимагає, щоб модель могла обробляти відеопотік з достатньою частотою кадрів (напр., 15-30 FPS).

5. Узагальнення (Generalization): Модель, навчена на даних, зібраних в одних умовах, може погано працювати в інших (напр., нічна зйомка, інший тип місцевості).

6. Розмір та вага обладнання: Бортовий комп'ютер та камера додають ваги до дрона, зменшуючи вантажопідйомність для іншого обладнання або зменшуючи час польоту.

1.3 Використання дронів для моніторингу: технічні аспекти та обмеження

Дрони (безпілотні літальні апарати, БПЛА) трансформували підхід до моніторингу та інспекції завдяки своїй мобільності, здатності досягати важкодоступних місць та можливості збору візуальних даних з унікального ракурсу. Як платформа для інтеграції комп'ютерного зору та ШІ, вони мають низку переваг, але й суттєвих обмежень, які необхідно враховувати при розробці системи.

Дрони: переваги як платформа для моніторингу інтеграції комп'ютерного зору та ШІ.

- Гнучкість та мобільність: Здатність швидко розгортатися, змінювати траєкторію польоту, зависати в повітрі та здійснювати вертикальний зліт/посадку (для мультикоптерів).

-Доступ до складних локацій: Можливість обстежувати території, недоступні або небезпечні для людини (напр., висотні конструкції, забруднені зони, зони надзвичайних ситуацій).

-Вартість та швидкість розгортання: Часто дешевше та швидше використовувати дрони порівняно з традиційною авіацією (літаки, гелікоптери) або наземними методами.

- Можливість збору даних з різної висоти та ракурсу: Дозволяє отримувати зображення з оптимальною деталізацією для конкретного завдання моніторингу. -

Автономність: Сучасні дрони можуть виконувати польотні місії за заздалегідь запрограмованим маршрутом, що звільняє оператора від постійного ручного управління (хоча автономне розпізнавання та прийняття рішень – це вже задача для інтегрованих КЗ/ШІ).

Існує кілька основних типів дронів, які використовуються для моніторингу, кожен зі своїми характеристиками:

1. Мультикоптери (Multirotors): Найпоширеніший тип (квадрокоптери, гексакоптери, октокоптери). Переваги: вертикальний зліт/посадка, здатність зависати (hover), висока маневреність. Недоліки: обмежений час польоту та менша швидкість порівняно з літаками, менша ефективність при сильному вітрі. Ідеальні для

детального огляду невеликих територій, інспекції об'єктів, що вимагають зависання.

Рис. 1.13 – Зображення типового квадрокоптера

2. Літаки (Fixed-wing): Мають крила та вимагають розбігу для зльоту (або використовують катапульти/ручне запускання) та посадкову смугу (або парашут/сітку). Переваги: значно більший час польоту, здатність покривати великі території за одну місію, вища швидкість. Недоліки: не можуть зависати, менша маневреність, вимагають більшого простору для зльоту/посадки. Ідеальні для картографування великих територій, моніторингу довгих лінійних об'єктів (трубопроводи, ЛЕП).

Рис. 1.14 – Зображення типового дрона типу «літак»

3. Гібридні (Hybrid/VTOL): Поєднують можливості мультикоптерів (вертикальний зліт/посадка) та літаків (політ за допомогою крила для більшої ефективності). Надають гнучкість мультикоптерів на етапах зльоту/посадки та вищу витривалість літаків під час моніторингу.

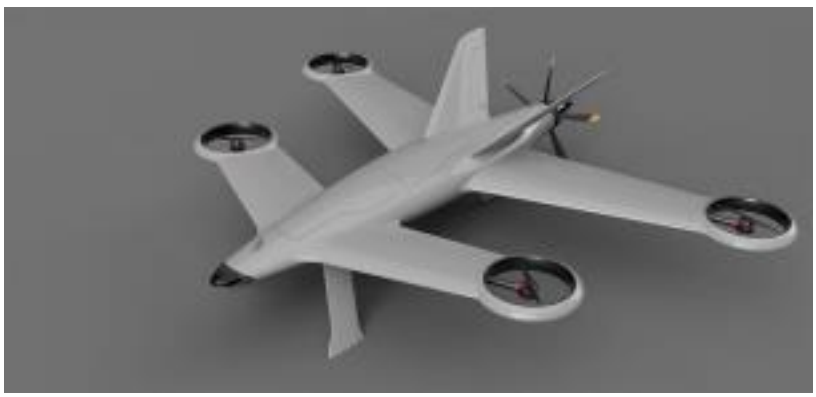


Рис. 1.15 – Зображення типового дрона типу «Hybrid/VTOL»

Вибір типу дрона суттєво впливає на вибір та конфігурацію інтегрованої системи КЗ/ШІ, оскільки визначає доступну вантажопідйомність, час польоту та характеристики польоту.

Технічні характеристики дронів, що впливають на інтеграцію КЗ та ШІ. Ефективність системи розпізнавання об'єктів на дроні значною мірою залежить від технічних характеристик самого БПЛА та його корисного навантаження:

Вантажопідйомність (Payload Capacity): Максимальна вага обладнання, яку дрон може нести. Обмежує розмір та потужність бортового комп'ютера, тип та кількість камер/сенсорів, додаткові батареї тощо. Потужні обчислювальні модулі та високоякісні камери можуть бути важкими.

Час польоту (Flight Time) та Джерело живлення (Power Source): Тривалість місії дрона. Високопродуктивні обчислювальні модулі, що виконують складні алгоритми ШІ, можуть споживати значну кількість енергії, скорочуючи час польоту. Необхідно балансувати між обчислювальною потужністю та енергоспоживанням.

Обчислювальні можливості на борту (Onboard Computing): Наявність та потужність бортового комп'ютера (companion computer), який може виконувати обробку даних та алгоритми ШІ безпосередньо на дроні (Edge Computing). Це можуть бути спеціалізовані платформи, такі як NVIDIA Jetson (популярні для ШІ на краю), Intel Movidius Myriad (VPU), Google Coral (TPU) або навіть невеликі ПК (напр., Intel NUC, Raspberry Pi в більш простих задачах). Потужність цього комп'ютера визначає, які саме моделі ШІ та з якою швидкістю можуть бути виконані на борту.

Система навігації та стабілізації (Navigation and Stabilization System): Якість роботи GPS, IMU (Inertial Measurement Unit) та систем стабілізації підвісу (gimbal) критично важлива для отримання стабільних та чітких зображень. Нестабільність платформи або некоректні навігаційні дані ускладнюють виділення ознак та локалізацію об'єктів.

Камера та сенсори (Camera and Sensors): Тип, роздільна здатність, частота кадрів камери, тип сенсора (CMOS/CCD), динамічний діапазон, тип об'єктива – все це впливає на якість та інформативність зображень. Для деяких задач можуть використовуватися не тільки камери видимого спектру, але й тепловізійні,

мультиспектральні або гіперспектральні сенсори.

Канали зв'язку (Communication Links): Пропускна здатність та надійність каналів зв'язку між дроном та наземною станцією. Впливає на можливість передачі відеопотоку високої роздільної здатності для обробки на землі або передачі результатів бортової обробки.

Система управління польотом (Flight Controller): Відповідає за стабілізацію, навігацію та виконання польотних завдань. Сумісність з бортовим комп'ютером (через протоколи MAVLink, ROS) є важливою для інтеграції даних з сенсорів дрона (висота, швидкість, орієнтація) в алгоритми КЗ/ШІ (напр., для гео-референсування об'єктів).

Обмеження дронів для задач моніторингу з КЗ/ШІ. Незважаючи на значні переваги, дрони як платформа мають і суттєві обмеження, які створюють виклики при інтеграції з КЗ та ШІ:

Обмежені обчислювальні ресурси та енергоспоживання: Як вже згадувалося, це одне з найголовніших обмежень. Запуск складних моделей глибокого навчання в реальному часі на борту вимагає оптимізації моделей або використання спеціалізованого апаратного прискорення. Баланс між точністю моделі та її обчислювальною складністю є критичним.

Обмежена вантажопідйомність: Накладає ліміт на вагу та, відповідно, потужність бортового обладнання. Це може змусити використовувати менш продуктивні, але легші компоненти.

Вплив погодних умов: Вітер, дощ, сніг, туман, низькі температури – все це може суттєво впливати на безпеку польоту, стабільність дрона та якість зображень. Наприклад, туман або дощ можуть повністю приховати об'єкти або спотворити їх зображення.

Обмежений час польоту: Незважаючи на прогрес, час польоту більшості комерційних дронів залишається відносно коротким (зазвичай 20-40 хвилин), що обмежує тривалість місії моніторингу.

Проблеми зі зв'язком: Обмежена дальність радіозв'язку, можливість втрати сигналу управління або відеопотоку в умовах міської забудови, лісових масивів або при наявності радіоперешкод.

Нестабільність платформи та вібрації: Навіть зі стабілізаційним підвісом, рух

дрона та вібрації від пропелерів можуть спричиняти розмиття зображень, що негативно впливає на якість даних для КЗ.

Висока вартість: Професійні дрони, високоякісні камери та потужні бортові комп'ютери є дорогими. Розробка та обслуговування комплексних систем з КЗ/ШІ також вимагає значних інвестицій.

У цьому розділі було розглянуто теоретичні основи, необхідні для розуміння принципів інтеграції комп’ютерного зору та штучного інтелекту в безпілотні літальні апарати для автоматичного розпізнавання об’єктів.

Були викладені фундаментальні принципи комп’ютерного зору, починаючи від захоплення та попереднього оброблення зображень, розглянуто методи фільтрації, виділення низькорівневих та високорівневих ознак, а також основи сегментації. Проаналізовано традиційні підходи до розпізнавання об'єктів та окреслено їхні обмеження, що стало підставою для переходу до більш досконалих методів на базі штучного інтелекту.

Детально розглянуто роль штучного інтелекту у задачах візуального розпізнавання, з акцентом на машинне та глибоке навчання. Представлено основи нейронних мереж та архітектури згорткових нейронних мереж (CNN), які є фундаментом для обробки зображень у сучасних системах. Окрему увагу приділено архітектурам нейронних мереж для детектування об'єктів (YOLO, SSD), детально описано їхні принципи роботи, переваги та недоліки у порівнянні з двофазними методами. Розглянуто основні метрики оцінки ефективності систем детектування об'єктів (IoU, Precision, Recall, mAP).

Проаналізовано безпілотні літальні апарати як специфічну апаратну платформу для моніторингу. Розглянуто різні типи дронів, їхні переваги та недоліки, а також ключові технічні характеристики (вантажопідйомність, час польоту, обчислювальні можливості, камера, система стабілізації), що безпосередньо впливають на можливість та ефективність інтеграції систем КЗ та ШІ. Виокремлено основні обмеження (обчислювальні та енергетичні ресурси, вплив середовища, вантажопідйомність), які необхідно враховувати при розробці та реалізації таких систем.

РОЗДІЛ 2 АНАЛІЗ І ВИБІР ТЕХНОЛОГІЙ

2.1 Порівняння алгоритмів комп’ютерного зору

2.1.1 OpenCV (Open Source Computer Vision Library)

OpenCV є однією з найстаріших та найпопулярніших бібліотек з відкритим кодом для комп'ютерного зору та обробки зображень. Вона була розроблена компанією Intel у 1999 році і з того часу активно розвивається та підтримується великою спільнотою. OpenCV написана на C++ і має прив'язки до Python, Java, MATLAB та інших мов програмування, що робить її надзвичайно гнучкою. Ключові можливості OpenCV:

1. Обробка зображень: Широкий спектр функцій для маніпуляції зображеннями, включаючи фільтрацію, морфологічні операції, перетворення колірних просторів, корекцію геометрії, виявлення країв, сегментацію тощо.

2. Аналіз відео: Інструменти для захоплення, обробки та аналізу відеопотоків, включаючи відстеження об'єктів, стабілізацію відео, детектування руху. 3. Виявлення та розпізнавання об'єктів: Реалізація класичних алгоритмів, таких як каскади Хаара для виявлення облич, HOG (Histogram of Oriented Gradients) для детектування пішоходів, а також можливість інтеграції з моделями глибокого навчання.

4. Калібрування камери та 3D-реконструкція: Функції для визначення внутрішніх та зовнішніх параметрів камери, стереозору, оцінки глибини та створення 3D-моделей сцени.

5. Машинне навчання: Включає реалізацію деяких класичних алгоритмів машинного навчання, таких як SVM (Support Vector Machines), k-NN (k-Nearest Neighbors), дерева рішень, які можуть використовуватися для задач класифікації та розпізнавання.

6. Інтеграція з апаратним прискоренням: Підтримка оптимізації за допомогою Intel IPP (Integrated Performance Primitives) та CUDA (Compute Unified Device Architecture) для прискорення обчислень на GPU.

Переваги OpenCV:

- Широкий функціонал.
- Висока продуктивність.
- Кросплатформність.

- Відкритий код та безкоштовність.

Недоліки OpenCV:

- Крива навчання.

- Обмежені можливості для глибокого.

- Синтаксис C++ API.

OpenCV для застосування в проєкті є незамінним інструментом для попередньої обробки зображень та відео, отриманих з дронів. Це включає: - Зменшення шуму та покращення якості зображення.

- Стабілізацію відео для компенсації вібрацій дрона.

- Виявлення руху для попередньої фільтрації кадрів.

- Виділення областей інтересу (ROI).

- Реалізацію допоміжних функцій, таких як перетворення форматів, зміна розміру зображень, візуалізація результатів.

Рис. 2.1 – Приклад використання OpenCV для виявлення країв на зображенні

2.1.2 TensorFlow

TensorFlow – це відкрита програмна бібліотека для машинного навчання, розроблена командою Google Brain. Вона є одним з найпопулярніших фреймворків для розробки та тренування моделей глибокого навчання, включаючи ті, що застосовуються в комп'ютерному зорі. Ключові можливості TensorFlow: - Побудова та тренування нейронних мереж: Надає гнучкі інструменти для створення

різноманітних архітектур нейронних мереж, від простих перцептронів до складних згорткових (CNN) та рекурентних (RNN) мереж.

- TensorFlow Object Detection API: Спеціалізований фреймворк на базі TensorFlow, який надає колекцію попередньо навчених моделей для детекції об'єктів (наприклад, SSD, Faster R-CNN, EfficientDet) та інструменти для їх донавчання на власних датасетах.

- TensorFlow Lite: Рішення для розгортання моделей машинного навчання на мобільних, вбудованих та IoT-пристроях, що є актуальним для обробки на борту дрона.

- TensorBoard: Інструмент для візуалізації процесу навчання моделей, архітектури мережі, метрик та інших даних.

- Розподілене навчання: Підтримка навчання моделей на кількох GPU або навіть на кластерах серверів.

- Keras API: TensorFlow інтегрував Keras як високо_рівневий API, що значно спрощує процес розробки моделей.

Переваги TensorFlow:

- Потужність та гнучкість: Дозволяє створювати та налаштовувати складні моделі глибокого навчання.

- Велика екосистема: Широкий набір інструментів (TensorFlow Hub для готових моделей, TensorFlow Datasets для наборів даних), бібліотек та розширень. - Масштабованість: Підходить як для досліджень, так і для розгортання промислових рішень.

- Підтримка Google: Активно розвивається та підтримується компанією Google.

- Кросплатформність: Працює на різних операційних системах та апаратних платформах.

- Готові моделі для КЗ: TensorFlow Object Detection API надає великий вибір якісних моделей.

Недоліки TensorFlow:

- Крива навчання: Особливо для низько_рівневого API, TensorFlow може бути складним для освоєння.

- Статичний граф обчислень (в ранніх версіях): Хоча TensorFlow 2.x перейшов на динамічні графи (eager execution) за замовчуванням, робота з графами все ще може бути менш інтуїтивною порівняно з PyTorch для деяких завдань.

- Відносно великий розмір бібліотеки: Може бути надлишковим для простих завдань.

TensorFlow для реалізації в проєкті є потужним кандидатом для розробки та тренування моделі розпізнавання об'єктів. Його Object Detection API може бути використаний для швидкого прототипування та отримання високоякісних результатів з використанням попередньо навчених моделей. TensorFlow Lite є важливим інструментом, якщо розглядається можливість розгортання моделі безпосередньо на борту дрона з обмеженими обчислювальними ресурсами.

Рис. 2.2 – Архітектура TensorFlow Object Detection API

2.1.3 PyTorch

PyTorch – це ще один популярний відкритий фреймворк для машинного навчання, розроблений дослідницькою групою Facebook AI Research (FAIR). Він

здобув велику популярність у дослідницькій спільноті завдяки своїй гнучкості та «пітонічному» підходу. Ключові можливості PyTorch:

- Динамічні обчислювальні графи: Дозволяють змінювати структуру мережі «на льоту» під час виконання, що є зручним для досліджень та роботи з даними змінної довжини (наприклад, в NLP).

- Простота та інтуїтивність: API PyTorch часто сприймається як більш простий та схожий на NumPy, що полегшує його вивчення та використання розробниками на Python.

- TorchVision: Бібліотека, що є частиною екосистеми PyTorch, надає популярні набори даних, архітектури моделей та інструменти для трансформації зображень. - Підтримка GPU: Ефективна робота з графічними процесорами для прискорення навчання та інференсу.

- PyTorch Mobile: Дозволяє розгортати моделі PyTorch на мобільних платформах iOS та Android.

Переваги PyTorch:

- Гнучкість та динамічність: Зручний для експериментів та швидкого прототипування.

- Легкість налагодження: Динамічні графи спрощують процес пошуку помилок.

- «Пітонічний» стиль: Більш природний для Python-розробників. - Швидкий ріст популярності: Особливо в академічному середовищі, що веде до великої кількості доступних моделей та ресурсів.

- Активна розробка: Постійне вдосконалення та додавання нових функцій.

Недоліки PyTorch:

- Менш зрілі інструменти для розгортання (порівняно з TensorFlow): Хоча ситуація покращується з PyTorch Mobile та TorchServe, TensorFlow історично мав більш розвинені рішення для промислового розгортання (наприклад, TensorFlow Serving).

- Візуалізація (TensorBoard кращий): Хоча існують інструменти для візуалізації в

PyTorch (наприклад, інтеграція з TensorBoard або Visdom), TensorBoard від TensorFlow часто вважається більш комплексним та зручним.

PyTorch для використання в проєкті є сильним конкурентом TensorFlow для розробки моделей розпізнавання об'єктів. Його гнучкість, простота використання та велика кількість сучасних архітектур, доступних через TorchVision, роблять його привабливим вибором. Для наочного порівняння ключових характеристик розглянутих бібліотек та фреймворків наведемо таблицю.

Таблиця 2.1 - Порівняння бібліотек та фреймворків комп'ютерного зору

Характеристика	OpenCV	TensorFlow	PyTorch
Основне призначення	Обробка зображень, класичне КЗ	Машинне навчання, глибоке навчання	Машинне навчання, глибоке навчання
Основна мова API	C++, Python	Python, C++, Java	Python, C++
Легкість вивчення	Середня (великий обсяг функцій)	Середня (високорівневий API Keras), складна (низькорівневий API)	Висока (для Python розробників)
Гнучкість	Висока для обробки зображень	Висока для побудови моделей МН	Дуже висока, динамічні графи
Продуктивність	Висока (оптимізований C++ код)	Висока (з GPU), залежить від моделі	Висока (з GPU), залежить від моделі
Готові моделі КЗ	Класичні алгоритми, інтеграція з DNN	Велика кількість (TF Hub, Object Detection API)	Велика кількість (TorchVision, модельні зоопарки)
Інструменти для розгортання	-	TensorFlow Lite, TensorFlow Serving	PyTorch Mobile, TorchServe
Візуалізація	Вбудовані функції, інтеграція з Matplotlib	TensorBoard	TensorBoard, Visdom
Спільнота	Дуже велика, зріла	Дуже велика	Велика, швидко зростаюча

Основні переваги	Широкий функціонал для обробки зображень, швидкість	Екосистема, масштабованість, інструменти для розгортання	Гнучкість, простота, популярність у дослідженнях
Основні недоліки	Обмежені інструменти для тренування DNN	Крива навчання для низькорівневого API	Менш зрілі інструменти для розгортання (історично)

Враховуючи специфіку завдання – інтеграція комп’ютерного зору та ШІ для автоматичного розпізнавання об’єктів дронами – доцільно використовувати комбінацію розглянутих інструментів.

1. OpenCV буде використовуватися як основна бібліотека.

2. Для реалізації модуля розпізнавання об'єктів на основі глибокого навчання буде обрано PyTorch.

Таким чином, для завдань обробки зображень та відео буде використовуватися OpenCV, а для розробки, навчання та виконання моделі розпізнавання об'єктів – PyTorch.

2.2 Вибір моделі ШІ для розпізнавання об’єктів

Вибір ефективної моделі штучного інтелекту для розпізнавання об'єктів є критично важливим для успіху проєкту. Модель повинна не тільки точно ідентифікувати об'єкти на зображеннях, отриманих з дрона, але й робити це достатньо швидко для забезпечення роботи системи в режимі, близькому до реального часу, особливо якщо обробка відбувається на обмежених ресурсах.

Моделі детекції об'єктів можна умовно розділити на дві основні категорії: двостадійні (two-stage detectors) та одностадійні (one-stage detectors). Двостадійні детектори, ці моделі працюють у два етапи:

1. Генерація пропозицій регіонів (Region Proposal): На першому етапі генерується набір потенційних регіонів на зображенні, де можуть знаходитися об'єкти.

2. Класифікація та регресія обмежувальних рамок: На другому етапі кожен запропонований регіон класифікується (визначається клас об'єкта) та уточнюється

його обмежувальна рамка.

Одностадійні детектори, ці моделі виконують детекцію об'єктів за один прохід через нейронну мережу, одночасно прогнозуючи обмежувальні рамки та класи об'єктів безпосередньо з карт ознак.

До популярних одностадійних детекторів належать:

- YOLO (You Only Look Once).

Сімейство моделей YOLO (You Only Look Once) зробило революцію в галузі детекції об'єктів завдяки своїй здатності виконувати розпізнавання в реальному часі. YOLOv8 продовжує філософію одностадійних детекторів, але з низкою важливих нововведень. Архітектура YOLOv8, як і багатьох сучасних детекторів, складається з трьох основних частин:

1. Backbone (Базова мережа): Відповідає за вилучення ознак з вхідного зображення на різних рівнях. У YOLOv8 використовується модифікована версія архітектури CSPDarknet (подібна до YOLOv5), але з певними оптимізаціями, такими як заміна деяких згорткових шарів на більш ефективні блоки C2f (CSP Bottleneck with 2 convolutions, fused). Цей блок є еволюцією блоку C3 з YOLOv5 і забезпечує кращий градієнтний потік та вилучення ознак.

Рис. 2.3 – Принципова схема блоку C2f в YOLOv8

2. Neck (Шия): Ця частина відповідає за агрегацію та комбінування ознак,

отриманих з різних рівнів базової мережі. Це дозволяє моделі ефективно детектувати об'єкти різного розміру. YOLOv8 використовує архітектуру, подібну до PANet (Path Aggregation Network), яка поєднує шляхи знизу-вгору (bottom-up) та зверху-вниз (top-down) для кращого злиття ознак.

3. Head (Голова): Відповідає за фінальні передбачення – обмежувальні рамки, класи об'єктів та їхню ймовірність. YOLOv8 використовує безякорний (anchor-free) підхід, на відміну від YOLOv5, яка використовувала якорі. Це означає, що модель безпосередньо прогнозує центр об'єкта та його розміри, що спрощує архітектуру та зменшує кількість параметрів, які потрібно налаштовувати. Голова також є роз'єднаною (decoupled head), тобто окремі гілки використовуються для передбачення класів та координат рамок, що часто покращує продуктивність. Функція втрат у YOLOv8 включає компоненти для класифікації (зазвичай бінарна крос-ентропія для кожного класу) та регресії рамок (наприклад, CIoU loss або Distribution Focal Loss), а також для «об'єктності».

Нові можливості та переваги YOLOv8:

- Покращена точність та швидкість: YOLOv8 демонструє кращий баланс між точністю (mAP - mean Average Precision) та швидкістю інференсу порівняно з попередніми версіями YOLO та багатьма іншими детекторами.

- Різноманітність моделей: Як і YOLOv5, YOLOv8 пропонується в кількох варіантах (наприклад, YOLOv8n - nano, YOLOv8s - small, YOLOv8m - medium, YOLOv8l - large, YOLOv8x - extra-large), що дозволяє обрати оптимальну модель залежно від вимог до швидкості та точності, а також доступних обчислювальних ресурсів.

- Безякорний підхід (Anchor-Free): Спрощує процес навчання та детекції, зменшує кількість гіперпараметрів.

- Нова архітектура базової мережі (Backbone) та шиї (Neck): Використання блоків C2f та оптимізованої PAN-подібної структури.

- Ефективна голова (Decoupled Head): Розділення завдань класифікації та регресії.

- Підтримка різних завдань: Окрім детекції об'єктів, YOLOv8 підтримує

завдання сегментації екземплярів (instance segmentation) та класифікації зображень. - Зручність використання: Розроблена Ultralytics, вона має добре документоване API, прості інструменти для тренування, валідації та експорту моделей у різні формати (ONNX, TensorRT, CoreML тощо).

- Активна підтримка та спільнота: Швидкий розвиток, виправлення помилок та велика кількість користувачів.

Для задачі автоматичного розпізнавання об'єктів дронами в задачах моніторингу модель YOLOv8 є оптимальним вибором з наступних причин: 1. Баланс між швидкістю та точністю: Дрони часто працюють в умовах, що вимагають швидкої реакції. YOLOv8 забезпечує високу швидкість інференсу, що дозволяє обробляти відеопотік в режимі, близькому до реального часу. При цьому точність розпізнавання залишається на високому рівні, що є критичним для ефективного моніторингу. Навіть менші версії YOLOv8 (наприклад, YOLOv8s або YOLOv8m) демонструють прийнятну точність при значно вищій швидкості, що робить їх кандидатами для розгортання на бортових системах з обмеженими ресурсами.

2. Здатність працювати на обмежених ресурсах: Хоча для максимальної продуктивності потрібні GPU, менші версії YOLOv8 можуть працювати і на CPU, або на спеціалізованих мобільних прискорювачах (якщо планується обробка на борту дрона). Можливість експорту в формати типу ONNX та TensorFlow Lite дозволяє оптимізувати модель для таких платформ.

3. Детекція об'єктів різного розміру: Завдяки використанню архітектури з кількома рівнями вилучення ознак (подібно до FPN/PANet), YOLOv8 добре справляється з детекцією як великих, так і малих об'єктів, що часто зустрічаються при моніторингу з різної висоти.

4. Наявність попередньо навчених моделей та легкість донавчання: YOLOv8 поставляється з моделями, попередньо навченими на великих наборах даних (наприклад, COCO). Це дозволяє отримати хороші результати «з коробки» для загальних класів об'єктів. Що більш важливо, ці моделі можна легко донавчити (fine-tuning) на власному датасеті специфічних об'єктів, які потрібно розпізнавати в рамках задачі моніторингу. Це значно скорочує час розробки та вимоги до обсягу навчальних даних.

5. Активна спільнота та підтримка Ultralytics: Забезпечує доступ до оновлень, виправлень, навчальних матеріалів та готових рішень. Інструментарій, що надається Ultralytics, значно спрощує весь цикл роботи з моделлю: від підготовки даних до тренування, валідації та розгортання.

6. Безякорний дизайн: Спрощує конфігурацію та може покращити узагальнюючу здатність моделі на різноманітних об'єктах.

7. Підтримка різних завдань: Хоча основним завданням є детекція, можливість використання тієї ж архітектури для сегментації може бути корисною для майбутніх розширень проєкту.

Для порівняння YOLOv8 з іншими популярними архітектурами наведемо таблицю, що узагальнює їх характеристики. Значення метрик (швидкість, точність) є орієнтовними і можуть сильно залежати від конкретної версії моделі, вхідної роздільної здатності, апаратного забезпечення та набору даних.

Таблиця 2.2 - Порівняння моделей ШІ для детекції об'єктів

Характеристика	Faster R-CNN (ResNet-50)	SSD (SSD300 MobileNetV2)	YOLOv8m
Тип детектора	Двостадійний	Одностадійний	Одностадійний (без якоря)
Точність (mAP COCO)	Висока (~38-42%)	Середня (~22-30% для мобільних версій)	Висока (~45-50%)
Швидкість (FPS на GPU)	Низька (~5-15 FPS)	Середня/Висока (~20-60 FPS)	Дуже висока (~100-200+ FPS)
Складність навчання	Висока	Середня	Середня/Низька (з інструментами Ultralytics)
Розмір моделі	Великий	Малий/Середній	Середній
Вимоги до ресурсів	Високі	Низькі/Середні	Середні (масштабується)
Гнучкість	Середня	Середня	Висока
Детекція малих об'єктів	Добре	Задовільно	Добре

Основні переваги	Висока точність	Швидкість, простота	Найкращий баланс швидкість/точність, легкість використання
Основні недоліки	Повільний, складний	Нижча точність	Точність може поступатися найважчим двостадійним моделям на деяких задачах

З таблиці видно, що YOLOv8m пропонує значно кращий компроміс між швидкістю та точністю порівняно з Faster R-CNN та SSD. Для завдань моніторингу з дронів, де швидкість обробки є критичною, такий баланс є надзвичайно важливим.

Отже, враховуючи вищезазначені фактори, модель YOLOv8 обрана як основна модель штучного інтелекту для розпізнавання об'єктів у даній дипломній роботі.

2.3. Апаратне забезпечення

Вибір апаратного забезпечення є не менш важливим, ніж вибір програмних компонентів, оскільки він безпосередньо впливає на можливості системи моніторингу, її продуктивність та автономність.

Дрони, або безпілотні літальні апарати (БПЛА), стали потужним інструментом для моніторингу завдяки своїй мобільності, можливості доступу до важкодоступних місць та відносній дешевизні порівняно з пілотованою авіацією. Для завдань моніторингу найчастіше використовуються два основні типи дронів:

1. Мультироторні дрони (квадрокоптери, гексакоптери тощо);
2. Дрони літакового типу (Fixed-wing);

Для завдань моніторингу, що передбачають детальне розпізнавання об'єктів, часто перевага надається мультироторним дронам через їхню здатність зависати та маневрувати близько до об'єктів інтересу. Однак, якщо моніторинг охоплює великі площі, комбінація типів або дрони літакового типу з високоякісними камерами можуть бути більш ефективними. Для даної роботи передбачається використання мультироторних дронів.

Важливі параметри дронів для моніторингу:

- Час польоту: Визначає, як довго дрон може перебувати в повітрі на одному заряді батареї. Для ефективного моніторингу бажано мати час польоту не менше 25-

30 хвилин.

- Дальність польоту та зв'язку: Максимальна відстань, на яку дрон може відлетіти від оператора, зберігаючи стабільний зв'язок для управління та передачі даних. Залежить від потужності передавача та технології зв'язку.

- Вантажопідйомність: Максимальна вага корисного навантаження, яке дрон може нести (камера, додаткові сенсори, обчислювальний модуль). Важливо враховувати вагу камери та, потенційно, бортового комп'ютера.

- Стійкість до погодних умов: Здатність дрона працювати при певному вітрі, температурі, опадах. Моделі з вищим класом захисту (IP rating) краще підходять для роботи в складних умовах.

- Системи навігації та безпеки: Наявність GPS/GLONASS для точного позиціонування, сенсорів уникнення перешкод, функції автоматичного повернення додому (RTH).

- Можливість програмування польотних завдань: Автономний політ за заданим маршрутом є важливим для систематичного моніторингу. Приклади моделей дронів, що підходять для завдань моніторингу: - DJI Mavic Series (наприклад, Mavic 3 Enterprise): Компактні, але потужні дрони з хорошими камерами та достатнім часом польоту. Серія Enterprise має розширені функції безпеки та можливість встановлення додаткових модулів. - DJI Matrice Series (наприклад, Matrice 300/350 RTK): Професійні платформи з високою вантажопідйомністю, тривалим часом польоту, можливістю встановлення кількох камер та сенсорів, високою точністю позиціонування (RTK). - Autel EVO II Series (наприклад, EVO II Pro Enterprise): Конкуренти DJI, що пропонують дрони з високоякісними камерами (включаючи 6K та тепловізійні), тривалим часом польоту та системами уникнення перешкод.

- Parrot Anafi Ai: Дрон з 4G-підключенням, що дозволяє керувати ним поза межами прямої видимості, та відкритим SDK для розробників. Вибір конкретної моделі дрона залежатиме від бюджету, специфіки об'єктів моніторингу та умов експлуатації. Для даної роботи передбачається, що буде використовуватися дрон класу DJI Mavic або аналогічний, що забезпечує хороший баланс між якістю камери,

часом польоту та портативністю.

Якість камери є визначальним фактором для успішного розпізнавання об'єктів. Типи сенсорів:

- CMOS (Complementary Metal-Oxide-Semiconductor): Найбільш поширений тип сенсорів у сучасних камерах дронів. Вони енергоефективні, швидкі та забезпечують високу якість зображення.

- CCD (Charge-Coupled Device): Раніше вважалися кращими за якістю зображення (особливо за динамічним діапазоном та рівнем шуму), але зараз CMOS сенсори значно покращилися і витіснили CCD у більшості застосувань через вищу швидкість зчитування та менше енергоспоживання.

Роздільна здатність та частота кадрів:

- Роздільна здатність визначає кількість пікселів у зображенні (наприклад, Full HD - 1920x1080, 4K - 3840x2160). Вища роздільна здатність дозволяє розрізняти дрібніші деталі на більшій відстані, що є важливим для моніторингу. Однак, зображення вищої роздільної здатності потребують більше обчислювальних ресурсів для обробки та більше місця для зберігання. Для ефективного розпізнавання об'єктів бажано мати камеру з роздільною здатністю щонайменше Full HD, а краще 4K.

- Частота кадрів (FPS - Frames Per Second): Кількість кадрів, що записуються за секунду. Вища частота кадрів (наприклад, 30 FPS або 60 FPS) забезпечує більш плавне відео та кращу можливість розпізнати об'єкти, що швидко рухаються. Для більшості завдань моніторингу достатньо 25-30 FPS.

Оптичний та цифровий зум:

- Оптичний зум: Збільшення досягається за рахунок зміни фокусної відстані об'єктива. Забезпечує збільшення без втрати якості зображення. Дуже корисний для детального огляду об'єктів з безпечної відстані.

- Цифровий зум: Програмне збільшення частини зображення, що призводить до втрати якості (пікселізації). Менш бажаний, але може бути корисним у деяких випадках.

- Гібридний зум: Комбінація оптичного та цифрового зуму. Сучасні камери

дронів часто пропонують значний гібридний зум з мінімальною втратою якості на початкових етапах збільшення.

Типи об'єктивів та кути огляду (FOV - Field of View):

- Ширококутні об'єктиви мають більший кут огляду, що дозволяє охопити більшу територію, але може призводити до геометричних спотворень (наприклад, «риб'яче око») та зменшення деталей віддалених об'єктів.

- Телеоб'єктиви мають вузький кут огляду, але дозволяють «наблизити» віддалені об'єкти.

- Вибір об'єктива залежить від висоти польоту та необхідної деталізації. Камери з можливістю зміни фокусної відстані (зум-об'єктиви) є найбільш універсальними.

Вібрації від моторів дрона та рух у повітрі можуть призводити до розмиття зображення, що значно ускладнює розпізнавання об'єктів, тому необхідно обрати стабілізацію:

- Механічна стабілізація (Gimbal): Найефективніший спосіб. Камера встановлюється на 2-осьовий або 3-осьовий підвіс, який компенсує рухи та вібрації дрона, забезпечуючи стабільне зображення. Більшість сучасних дронів для моніторингу оснащені 3-осьовими підвісами.

- Електронна стабілізація (EIS): Програмний метод, який аналізує рух кадрів та намагається його компенсувати шляхом обрізання зображення. Менш ефективна, ніж механічна, і може призводити до втрати частини кадру та деякого погіршення якості.

- Гібридна стабілізація: Комбінація механічної та електронної стабілізації. Для специфічних завдань моніторингу, таких як пошук людей вночі, виявлення витоків тепла, моніторинг стану сонячних панелей тощо, можуть використовуватися:

- Тепловізійні камери (Thermal cameras): Реєструють інфрачервоне випромінювання об'єктів і візуалізують розподіл температур.

- Мультиспектральні та гіперспектральні камери: Захоплюють зображення в різних діапазонах електромагнітного спектра, що корисно для сільського господарства, екологічного моніторингу.

Обробка відеопотоку та виконання моделей ШІ для розпізнавання об'єктів є

ресурсомісткими завданнями. Існує два основних підходи до розміщення обчислювальних потужностей:

1. Обробка на наземній станції: Відеопотік з дрона передається на наземний комп'ютер (ноутбук або стаціонарний ПК), де відбувається вся обробка. 2. Обробка на борту дрона (Edge Computing): На дрон встановлюється компактний бортовий комп'ютер (single-board computer, SBC) з достатньою продуктивністю для виконання моделі ШІ.

Характеристики обчислювальних платформ:

- Центральний процесор (CPU): Важливий для загальних обчислень, управління системою та виконання частини алгоритмів КЗ.

- Графічний процесор (GPU): Критично важливий для прискорення навчання та інференсу моделей глибокого навчання. Для наземної станції це можуть бути дискретні відеокарти NVIDIA (GeForce RTX, Quadro) або AMD Radeon. Для бортових систем – мобільні GPU або спеціалізовані модулі.

- Нейронні процесори (NPU/TPU): Спеціалізовані прискорювачі, розроблені для ефективного виконання операцій нейронних мереж (наприклад, Google Coral Edge TPU, Intel Movidius Myriad VPU). Вони забезпечують високу продуктивність при низькому енергоспоживанні, що робить їх ідеальними для бортових систем.

Рис. 2.4 – Приклад бортового комп'ютера NVIDIA Jetson Nano/Xavier NX

- Оперативна пам'ять (RAM): Необхідна для завантаження моделі ШІ, зберігання

проміжних даних та відеокадрів. Обсяг залежить від розміру моделі та роздільної здатності відео. Для комфортної роботи з YOLOv8 на наземній станції бажано мати щонайменше 16 ГБ RAM, для бортових систем – від 4-8 ГБ.

- Накопичувач (SSD/HDD): Для зберігання операційної системи, програмного забезпечення, моделей ШІ та, можливо, записаних відеоданих. SSD значно прискорює завантаження та роботу системи.

Бортові комп'ютери для дронів. Якщо розглядається обробка на борту, популярними є:

- NVIDIA Jetson Series (Jetson Nano, Jetson Xavier NX, Jetson AGX Orin): Потужні компактні комп'ютери з вбудованими GPU NVIDIA, оптимізовані для ШІ застосувань. Мають хорошу програмну підтримку (NVIDIA JetPack SDK, CUDA, TensorRT).

- Raspberry Pi (з прискорювачем): Сам по собі Raspberry Pi має обмежені можливості для складних моделей ШІ, але може використовуватися в поєднанні з USB-прискорювачами (наприклад, Google Coral USB Accelerator, Intel Neural Compute Stick).

- Плати на базі спеціалізованих SoC (System-on-Chip) з NPU.

Для даної роботи на першому етапі планується реалізація системи з обробкою на наземній станції, що дозволить використовувати більш потужні обчислювальні ресурси та спростить процес розробки та налагодження. У перспективі, після успішної апробації, можливий перехід до розгортання оптимізованої моделі на бортовому комп'ютері типу NVIDIA Jetson Xavier NX. Забезпечення сумісності всіх апаратних компонентів є ключем до стабільної роботи системи.

Камера дрона повинна мати можливість передавати відеопотік на обчислювальний модуль (або на передавач для трансляції на землю). Це може бути цифровий інтерфейс (MIPI CSI для бортових систем, USB) або аналоговий (менш бажано). Більшість сучасних дронів мають інтегровані камери з цифровою передачею відео. Канали передачі даних з дрона на наземну станцію:

- Радіоканали (2.4 ГГц, 5.8 ГГц): Найбільш поширені для керування дроном та передачі FPV (First Person View) відео. Технології типу DJI OcuSync, Lightbridge

забезпечують передачу HD-відео на значні відстані (кілька кілометрів).

- Wi-Fi: Може використовуватися для передачі даних на короткі відстані. -

LTE/4G/5G: Дозволяє керувати дроном та передавати дані на необмежені відстані за наявності покриття мобільної мережі. Деякі сучасні дрони мають таку опцію.

У випадку використання наземної станції, основна увага приділяється якості та стабільності каналу передачі відео з дрона. Дрони DJI з технологією OcuSync 3.0/4.0 або аналогічні забезпечують достатньо низьку затримку та високу якість передачі HD-відео.

Вибір програмного забезпечення охоплює операційні системи, мови програмування, середовища розробки, бібліотеки, фреймворки та інструменти, необхідні для створення, тестування та розгортання системи розпізнавання об'єктів. Для наземної станції (розробка та виконання):

- Linux (наприклад, Ubuntu): Є де-факто стандартом для розробки у сфері

машинного навчання та комп'ютерного зору. Більшість бібліотек та інструментів (TensorFlow, PyTorch, OpenCV, CUDA, cuDNN) мають найкращу підтримку саме в Linux. Забезпечує гнучкість, контроль над системою та доступ до потужних інструментів командного рядка. Рекомендований вибір для розробки та розгортання на наземній станції.

Для бортових систем (якщо розглядається обробка на дроні):

- Linux (спеціалізовані дистрибутиви): Більшість бортових комп'ютерів

(NVIDIA Jetson, Raspberry Pi) працюють під управлінням Linux (наприклад, NVIDIA JetPack SDK базується на Ubuntu).

- ROS (Robot Operating System): Не є самостійною ОС, а радше мета

операційною системою або фреймворком, що працює поверх Linux. ROS надає стандартні інструменти для комунікації між різними програмними модулями (вузлами), управління сенсорами, навігацією тощо. Дуже популярний у робототехніці, включаючи розробку систем для дронів.

Рис. 2.5 – Екосистема ROS для робототехнічних застосувань

Для даного проєкту основною ОС для розробки буде Ubuntu Linux (версія 20.04 LTS або новіша).

Основна мова програмування для машинного навчання та комп'ютерного зору, буде використовуватись Python. Більшість популярних бібліотек (TensorFlow, PyTorch, OpenCV, Scikit-learn) мають Python API. Python вирізняється простотою синтаксису, великою кількістю бібліотек та активною спільнотою, що значно прискорює процес розробки та прототипування. Основна мова для реалізації проєкту.

Середовища розробки (IDE), буде використовуватись PyCharm (JetBrains). Потужна IDE для Python з відмінною підтримкою наукових бібліотек, інструментами для налагодження, рефакторингу, інтеграцією з системами контролю версій. Існує безкоштовна версія Community та платна Professional. Рекомендований вибір для розробки на Python.

Для основної розробки буде використовуватися PyCharm Professional або VS Code з відповідними розширеннями для Python. JupyterLab буде використовуватися для експериментів з даними та моделями.

Основні бібліотеки та фреймворки, які було обґрунтовано в попередніх підрозділах:

- OpenCV (версія 4.x): Для обробки зображень та відео, візуалізації.
- PyTorch (версія 1.10+ або новіша): Для розробки, тренування та виконання моделі розпізнавання об'єктів YOLOv8.

- TorchVision: Для доступу до стандартних наборів даних, моделей та трансформацій зображень.

- Ultralytics YOLOv8 package: Офіційний пакет для роботи з моделями YOLOv8, що надає зручний інтерфейс для тренування, валідації, інференсу та експорту моделей.

- NumPy: Фундаментальна бібліотека для наукових обчислень в Python, особливо для роботи з багатовимірними масивами (тензорами). - Matplotlib, Seaborn, Plotly: Бібліотеки для візуалізації даних, графіків, результатів експериментів.

- Pandas: Для роботи з табличними даними, аналізу та підготовки датасетів. - Scikit-learn: Для реалізації класичних алгоритмів машинного навчання, оцінки моделей, розділення даних.

Для навчання моделі розпізнавання об'єктів необхідний якісно анотований набір даних. LabelImg: Популярний безкоштовний графічний інструмент для анотації зображень (створення обмежувальних рамок). Зберігає анотації у форматах PASCAL VOC та YOLO.

Рис. 2.6 – Інтерфейс програми LabelImg для анотації об'єктів

Для анотації даних у даному проєкті планується використання LabelImg.

Системи керування версіями

- Git: Розподілена система керування версіями, яка є стандартом для сучасної розробки програмного забезпечення. Дозволяє відстежувати зміни в коді,

повертатися до попередніх версій, працювати в команді над одним проєктом.

- GitHub / GitLab / Bitbucket: Веб-сервіси для хостингу Git-репозиторіїв, що надають додаткові інструменти для управління проєктами, відстеження помилок, CI/CD (Continuous Integration / Continuous Deployment).

Для проєкту буде використовуватися Git у поєднанні з GitHub для зберігання коду, відстеження змін та спільної роботи (якщо передбачається). Програмне забезпечення для управління дронами (API/SDK). Для взаємодії з дроном, отримання відеопотоку, телеметрії та, можливо, управління польотом, може знадобитися використання спеціалізованих API/SDK:

- DJI SDK (Mobile SDK, Onboard SDK, Payload SDK): Надає доступ до функцій дронів DJI. Mobile SDK дозволяє створювати мобільні додатки для управління дроном, Onboard SDK – розробляти програми для бортових комп'ютерів, Payload SDK – інтегрувати власні корисні навантаження.

- PX4 / ArduPilot: Популярні відкриті автопілоти для дронів. Мають власні протоколи та SDK (наприклад, MAVLink, MAVSDK) для взаємодії. MAVSDK надає API для Python, C++ та інших мов.

Для розгортання системи як стабільного сервісу, особливо якщо вона буде використовуватися в промислових масштабах або як веб-додаток, можуть бути корисними. Docker: Платформа для контейнеризації додатків. Дозволяє упакувати додаток з усіма його залежностями в ізольований контейнер, що забезпечує однакову роботу в різних середовищах. Дуже корисний для спрощення розгортання та забезпечення відтворюваності.

Протоколи передачі відеоданих:

RTSP (Real-Time Streaming Protocol): Стандартний протокол для передачі потокового відео через IP-мережі. Багато IP-камер та деякі дрони підтримують передачу відео по RTSP. OpenCV може легко захоплювати відеопотоки з RTSP URL.

Для отримання відеопотоку з дрона на наземну станцію пріоритетним буде використання RTSP, якщо дрон його підтримує, або отримання відео через SDK виробника дрона.

У даному розділі було проведено детальний аналіз та обґрунтування вибору

технологій для реалізації системи автоматичного розпізнавання об'єктів дронами. 1. Алгоритми комп'ютерного зору: Для завдань попередньої обробки зображень, візуалізації та допоміжних функцій КЗ обрано бібліотеку OpenCV завдяки її широкому функціоналу, високій продуктивності та доступності для Python.

2. Модель ШІ для розпізнавання об'єктів: Після аналізу різних архітектур (двостадійних та одностадійних детекторів) було зроблено вибір на користь YOLOv8. Ця модель забезпечує найкращий баланс між швидкістю та точністю, є відносно легкою для донавчання на власних даних, має активну підтримку та інструментарій від Ultralytics, що значно спрощує розробку. Обробка буде виконуватися на фреймворку PyTorch.

3. Апаратне забезпечення:

- Дрони: Перевага надається мультироторним дронам (класу DJI Mavic) через їхню маневреність та здатність зависати для детального огляду. Ключовими параметрами є час польоту (від 25-30 хв), якісна камера та стабільний канал передачі відео.

- Камери: Необхідна камера з роздільною здатністю щонайменше Full HD (бажано 4K), частотою кадрів 25-30 FPS та 3-осьовою механічною стабілізацією (gimbal).

- Обчислювальні платформи: На першому етапі обробка даних буде здійснюватися на наземній станції (ПК або ноутбук з GPU NVIDIA) для забезпечення максимальної продуктивності та гнучкості розробки. 4. Програмне забезпечення:

- ОС: Linux (Ubuntu) для розробки та виконання.

- Мова програмування: Python як основна мова.

- IDE: PyCharm або VS Code.

- Інструменти анотації: LabelImg або CVAT.

- Керування версіями: Git та GitHub.

- Передача відео: Пріоритет надається RTSP або SDK виробника дрона. Обраний стек технологій є сучасним, добре підтримуваним та відповідає вимогам поставленої задачі, забезпечуючи необхідну гнучкість для розробки, високу продуктивність для розпізнавання об'єктів в режимі, близькому до реального часу, та можливість подальшого масштабування та вдосконалення системи. Наступні розділи дипломної роботи будуть присвячені практичній реалізації системи на основі цих технологій.

РОЗДІЛ 3 РОЗРОБКА СИСТЕМИ АВТОМАТИЧНОГО РОЗПІЗНАВАННЯ ОБ'ЄКТІВ

3.1 Архітектура системи: схема взаємодії дрона, камери, ШІ-моделі та інтерфейсу.

Архітектура системи автоматичного розпізнавання об'єктів дронами являє собою сукупність взаємодіючих компонентів, кожен з яких виконує свою специфічну функцію. Ефективність та надійність системи значною мірою залежать від правильності спроєктованої архітектури, що забезпечує безперебійний потік даних та координацію роботи всіх модулів. Базова архітектура системи включає наступні ключові компоненти:

1. Дрон і камера: Дрон (DJI Mavic 3) оснащений камерою з роздільною здатністю 4K і 3-осьовим підвісом для стабілізації зображення. Камера захоплює відеопотік, який передається на наземну станцію.

2. Канал передачі даних: відеопотік передається через цифровий радіоканал (технологія DJI OcuSync 3.0) із низькою затримкою в форматі RTSP (Real-Time Streaming Protocol).

3. Наземна станція: комп'ютер із GPU NVIDIA (наприклад, RTX 3060) виконує обробку відеопотоку, запуск моделі YOLOv8 і збереження результатів. Наземна станція працює під управлінням ОС Ubuntu 20.04.

4. Модель штучного інтелекту (YOLOv8): виконує детекцію об'єктів у реальному часі, ідентифікуючи та локалізуючи об'єкти на кадрах відеопотоку. 5. Інтерфейс користувача: програмний додаток, який відображає відеопотік із накладеними обмежувальними рамками (bounding boxes) і класами об'єктів, а також

забезпечує керування дроном і збереження результатів.

Схема взаємодії цих компонентів може бути представлена наступним чином (Рис. 3.1):

Рис. 3.1 – Схема взаємодії компонентів системи автоматичного розпізнавання об’єктів дронами

Розглянемо детальніше кожен компонент та їх взаємодію:

1. Дрон: виступає як мобільна платформа для збору візуальних даних. Основними функціями дрона в даній системі є політ за заданим маршрутом або під управлінням оператора, стабілізація польоту та платформи для камери, а також, у деяких випадках, передача даних та енергоживлення для бортового обладнання.

2. Камера: Є основним сенсором системи, що захоплює візуальні дані навколишнього середовища у вигляді зображень або відеопотоку. Ключові характеристики камери, такі як роздільна здатність, частота кадрів, тип сенсора та наявність стабілізації (механічний підвіс), безпосередньо впливають на якість вхідних даних для модуля обробки та, відповідно, на точність розпізнавання об’єктів. Відеопотік з камери передається до модуля обробки даних.

3. Модуль обробки даних (з реалізованою ШІ-моделлю): Це центральний обчислювальний елемент системи, який відповідає за прийом, обробку та аналіз візуальних даних з камери з використанням алгоритмів комп’ютерного зору та моделей штучного інтелекту. Цей модуль може бути розташований як на борту дрона (бортовий комп'ютер, Edge Computing), так і на наземній станції (ПК або ноутбук).

Основні функції модуля обробки даних:

- Прийом та попередня обробка даних: Отримання відеопотоку з камери, виконання базових операцій обробки зображень, таких як шумозаглушення, корекція освітлення, зміна розміру.

- Виконання ШІ-моделі: Застосування навченої моделі детектування об'єктів (YOLOv8) до оброблених зображень для виявлення, локалізації (визначення обмежувальних прямокутників) та класифікації об'єктів інтересу.

- Аналіз та інтерпретація результатів: Обробка вихідних даних моделі (координати рамок, класи, оцінки впевненості), фільтрація помилкових спрацьовувань, можливо, відстеження об'єктів у часі.

- Передача результатів: Відправлення результатів розпізнавання (наприклад, зображень з нанесеними рамками, списку виявлених об'єктів з координатами) до інтерфейсу користувача.

Вибір обчислювальної платформи для цього модуля (CPU, GPU, NPU) та її потужність визначають, які моделі ШІ можуть бути ефективно використані та з якою швидкістю буде відбуватися обробка. Розташування модуля (на борту чи на землі) впливає на вимоги до каналу зв'язку та час затримки.

4. Інтерфейс користувача: Надає оператору можливість взаємодіяти з системою. Основні функції інтерфейсу включають:

- Відображення відеопотоку.

- Відображення результатів розпізнавання.

- Відображення телеметрії дрона.

- Керування дроном.

- Налаштування системи.

- Збереження даних.

Інтерфейс користувача може бути реалізований як програмне забезпечення на

наземній станції (десктопний додаток), мобільний додаток або веб-інтерфейс. Взаємодія між компонентами відбувається через відповідні канали зв'язку та протоколи. Відеодані з камери передаються до модуля обробки, результати обробки – до інтерфейсу користувача. Інтерфейс може надсилати команди керування на дрон та налаштування на модуль обробки. Вибір протоколів передачі даних (наприклад, RTSP для відео, MAVLink або інші для телеметрії та керування) є важливим для забезпечення стабільності та низької затримки в роботі системи.

3.2 Підготовка даних: створення або використання набору даних

Ефективність будь-якої системи розпізнавання об'єктів, що базується на методах глибокого навчання, критично залежить від якості та обсягу даних, на яких навчається модель. Етап підготовки даних є одним з найбільш трудомістких та відповідальних у процесі розробки. Він включає збір зображень або відео, їх попередню обробку та анотацію. Підготовка даних для навчання моделі розпізнавання об'єктів включає наступні етапи:

1. Збір даних:

- Зображення та відео, зібрані за допомогою дронів.

- Використання існуючих публічних або приватних наборів даних. -

Синтетичні дані.

2. Попередня обробка зібраних даних:

- Видалення небажаних кадрів.

- Корекція якості зображення.

- Зменшення роздільної здатності.

- Конвертація форматів.

3. Анотація даних: Це один з найкритичніших та найбільш трудомістких етапів. Анотація полягає у розмітці кожного об'єкта інтересу на зображенні за допомогою обмежувального прямокутника (bounding box) та присвоєння йому відповідного

класу. Для цього використовуються спеціалізовані інструменти анотації:

- LabelImg.

- CVAT (Computer Vision Annotation Tool).

Під час анотації необхідно дотримуватися єдиних правил для всіх зображень та об’єктів, забезпечуючи високу якість розмітки.

4. Розподіл даних на підмножини:

- Навчальна вибірка (Training set): Використовується безпосередньо для навчання моделі.

- Валідаційна вибірка (Validation set): Використовується під час навчання для оцінки продуктивності моделі на даних, які вона не бачила під час тренування. -

Тестова вибірка (Test set): Використовується один раз після завершення навчання для фінальної оцінки продуктивності моделі на повністю незалежних даних.

5. Форматування даних: Підготовлені та анотовані дані необхідно перевести у формат, сумісний з обраним фреймворком глибокого навчання (PyTorch) та моделлю (YOLOv8).

6. Збільшення даних (Data Augmentation): Це набір технік, спрямованих на штучне збільшення обсягу навчальної вибірки шляхом застосування різних перетворень до існуючих зображень. Типові перетворення включають: -

Горизонтальне/вертикальне віддзеркалення.

- Випадкове обрізання (random cropping).

- Обертання.

- Зміна масштабу.

- Зміна яскравості, контрасту, насиченості.

- Додавання шуму.

- Змішування зображень (наприклад, CutMix, Mosaic у YOLOv8). Data augmentation застосовується «на льоту» під час тренування, тобто перетворення виконуються над міні-партією (batch) зображень безпосередньо перед їх подачею у мережу.

Ретельно підготовлений та якісно анотований набір даних є фундаментальним елементом для успішного навчання моделі розпізнавання об’єктів, здатної ефективно працювати в умовах моніторингу з дронів.

У таблиці 3.1 відображено статистичні дані вибірок моделей, які проходили навчання у розпізнаванні об’єктів, здатних ефективно працювати в умовах моніторингу.

Таблиця 3.1 – Статистика набору даних для навчання моделі

Клас об'єкта	Навчальна вибірка (зображень / об'єктів)	Валідаційна вибірка (зображень / об'єктів)	Тестова вибірка (зображень / об'єктів)
Людина	850 / 2500	150 / 450	200 / 600
Транспортний засіб	1200 / 4100	200 / 680	300 / 1000
Тварина	600 / 1800	100 / 300	150 / 450
Всього	2650 / 8400	450 / 1430	650 / 2050

3.3 Навчання моделі: опис процесу тренування ШІ, інтеграція з дроном та інтерфейс користувача

Навчання моделі штучного інтелекту є ключовим етапом розробки системи розпізнавання об’єктів. На цьому етапі обрана архітектура нейронної мережі адаптується до задачі детектування специфічних об’єктів на підготовленому наборі даних. Процес навчання передбачає ітеративне коригування внутрішніх параметрів моделі з метою мінімізації помилки передбачення. Навчання проводилося на наземній станції з наступними характеристиками:

- ОС: Ubuntu 20.04.
- Процесор: Intel Core i7-10700.

- GPU: NVIDIA RTX 3060 (12 ГБ VRAM).

- RAM: 32 ГБ.

- Бібліотеки: PyTorch 1.12, Ultralytics YOLOv8, OpenCV 4.5.

Процес навчання цієї моделі включає наступні кроки:

1. Вибір та конфігурація моделі: YOLOv8 пропонує різні варіанти моделей (nano, small, medium, large, extra-large), що відрізняються розміром, складністю та, відповідно, швидкістю та точністю. Вибір конкретного варіанта залежить від доступних обчислювальних ресурсів (наземна станція або бортовий комп'ютер) та вимог до продуктивності системи. Для навчання на наземній станції можна використовувати більш потужні моделі (наприклад, YOLOv8m, YOLOv8l), тоді як для розгортання на борту дрона можуть знадобитися менші та швидші моделі (YOLOv8n, YOLOv8s).

Конфігурація моделі також включає налаштування архітектури, параметрів шарів, функцій активації тощо. Втім, при використанні готових реалізацій YOLOv8 (наприклад, пакету Ultralytics), більшість цих параметрів вже оптимізовані, і потрібно лише вказати шлях до конфігураційного файлу моделі та набору даних.

2. Ініціалізація ваг: Ваги нейронної мережі можуть бути ініціалізовані випадковими значеннями або ж завантажені з попередньо навченої моделі (pre trained weights). Використання попередньо навчених ваг (наприклад, навчених на великому наборі даних ImageNet або COCO) є стандартною практикою в глибокому навчанні, оскільки це дозволяє моделі швидше зійтися та досягти кращої точності, особливо якщо власний набір даних невеликий.

3. Визначення функції втрат (Loss Function): Функція втрат вимірює помилку моделі на поточному наборі даних. Під час навчання модель намагається мінімізувати цю функцію. Для задач детектування об’єктів функція втрат зазвичай складається з трьох компонентів:

- Втрати локалізації (Localization Loss / Box Loss): Вимірює, наскільки точно передбачені обмежувальні рамки відповідають істинним рамкам. - Втрати класифікації (Classification Loss): Вимірює, наскільки правильно модель класифікує

об’єкти у передбачених рамках.

- Втрати об'єктності (Objectness Loss / Confidence Loss): Вимірює, наскільки впевнена модель у наявності об'єкта у передбаченій рамці.

4. Вибір оптимізатора (Optimizer): Оптимізатор відповідає за оновлення ваг моделі на основі градієнтів функції втрат, обчислених за допомогою алгоритму зворотного поширення помилки (Backpropagation). YOLOv8 часто використовує оптимізатор SGD або AdamW. Вибір оптимізатора та його гіперпараметрів (швидкість навчання, коефіцієнти імпульсу тощо) суттєво впливає на процес збіжності та кінцеву продуктивність моделі.

5. Налаштування гіперпараметрів (Hyperparameters): Гіперпараметри – це параметри, які встановлюються до початку процесу навчання і не навчаються автоматично. Їх правильний вибір є критичним для досягнення високої продуктивності моделі. До ключових гіперпараметрів навчання YOLOv8 належать:

- Швидкість навчання (Learning Rate): Визначає розмір кроку, з яким оновлюються ваги під час оптимізації.

- Розмір міні-партії (Batch Size): Кількість зображень, що одночасно обробляються мережею за одну ітерацію.

- Кількість епох (Number of Epochs): Кількість повних проходів через весь навчальний набір даних.

- Вагова деградація (Weight Decay): Техніка регуляризації, що додає штраф до функції втрат за великі значення ваг, допомагаючи уникнути перенавчання. - Коефіцієнти оптимізатора: Параметри, специфічні для обраного оптимізатора.

- Параметри Data Augmentation: Ймовірності та діапазони застосування різних перетворень даних.

- Пороги для NMS (Non-Maximum Suppression): Використовуються на етапі

інференсу для фільтрації надлишкових обмежувальних прямокутників. Налаштування гіперпараметрів часто здійснюється шляхом експериментів та використання технік автоматичного пошуку гіперпараметрів, хоча для YOLOv8 початкові значення з офіційних конфігурацій часто є хорошою відправною точкою. 6. Процес навчання: Навчання відбувається ітеративно. На кожній ітерації: - З навчальної вибірки завантажується міні-партія зображень та відповідних їм анотацій.

- До зображень застосовується Data Augmentation.

- Зображення подаються на вхід нейронної мережі.

- Мережа робить передбачення (обмежувальні рамки, класи, впевненість). - Обчислюється значення функції втрат шляхом порівняння передбачень з істинними анотаціями.

- За допомогою алгоритму зворотного поширення помилки обчислюються градієнти функції втрат по відношенню до ваг мережі.

- Оптимізатор використовує градієнти для оновлення ваг мережі.

7. Оцінка продуктивності (Evaluation): Під час навчання та після його завершення оцінюється якість роботи моделі за допомогою відповідних метрик. Основними метриками для задач детектування об’єктів є:

- IoU (Intersection over Union): Коефіцієнт перекриття між передбаченою та істинною рамкою.

- Precision (Точність): Частка правильно виявлених об’єктів серед усіх передбачених об’єктів певного класу або за всіма класами.

- Recall (Повнота / Чутливість): Частка правильно виявлених об’єктів серед усіх істинних об’єктів певного класу або за всіма класами.

- Average Precision (AP): Середнє значення Precision для певного класу об'єктів,

обчислене при різних рівнях Recall.

- mean Average Precision (mAP): Середнє значення AP за всіма класами об'єктів. Це найбільш поширений показник якості для моделей детектування об'єктів.

Під час навчання відстежуються значення функції втрат (на навчальній та валідаційній вибірках) та метрик (на валідаційній вибірці). Це дозволяє контролювати процес навчання та вчасно зупинити його, якщо модель починає перенавчатися (втрати на валідації зростають, тоді як на навчанні продовжують падати).

8. Збереження та експорт моделі: Після завершення навчання найкраща модель (зазвичай та, що показала найкращі результати на валідаційній вибірці) зберігається. Модель може бути збережена у форматі PyTorch (.pt) або експортована в інші формати для оптимізації інференсу на різних пристроях.

Процес навчання є ітеративним і може вимагати багаторазових експериментів з різними гіперпараметрами та конфігураціями моделі для досягнення оптимальної продуктивності. Важливим є використання GPU для значного прискорення навчання.

Рис. 3.2 – Приклад графіку зміни функції втрат під час навчання

Рис. 3.3 – Приклад графіку зміни метрики mAP@0.5 під час навчання Таблиця 3.2 – Результати оцінки продуктивності моделі YOLOv8 та моделі YOLOv8m на тестовій вибірці

Метрика	Значення (%)	Значення (%)
mAP@0.5	78.5	0.82
mAP@0.5:0.95	52.1	0.58
Precision	81.2	0.85
Recall	75.8	0.80
F1-Score	78.1	0.82

Інтеграція розробленої системи автоматичного розпізнавання об’єктів з дроном є критично важливим етапом, який дозволяє перетворити окремі компоненти на єдину функціональну систему моніторингу. Цей етап включає налагодження зв’язку між дроном, камерою, модулем обробки даних та, у разі необхідності, системою управління польотом. Специфіка інтеграції значною мірою залежить від обраної апаратної платформи (типу дрона, наявності бортового комп'ютера) та програмного забезпечення.

Для тестування системи використано дрон DJI Mavic 3 із такими характеристиками:

-Камера: 4K, 20 МП, 3-осьовий підвіс.

-Час польоту: до 30 хвилин.

- Передача відео: OcuSync 3.0, до 8 км, затримка ~200 мс.

- Навігація: GPS/GLONASS, сенсори уникнення перешкод.

Відеопотік передається на наземну станцію через RTSP-протокол, що забезпечує сумісність із OpenCV.

Розглянемо основні аспекти інтеграції, передбачаючи, що на першому етапі обробка даних відбувається на наземній станції.

1. Отримання відеопотоку з камери дрона: Це перший крок для подачі візуальних даних у модуль обробки. Існує кілька способів отримати відеопотік: - Через SDK виробника дрона: Більшість комерційних дронів (наприклад, DJI) надають SDK, які дозволяють розробникам отримати доступ до відеопотоку з камери. DJI Mobile SDK або Payload SDK можуть бути використані для отримання відеопотоку на мобільний пристрій або бортовий комп'ютер відповідно. - За допомогою стандартних протоколів потокового відео: Деякі дрони та IP камери підтримують стандартні протоколи, такі як RTSP (Real-Time Streaming Protocol). Якщо дрон або його камера можуть транслювати відео по RTSP, це значно спрощує інтеграцію, оскільки OpenCV та інші бібліотеки комп'ютерного зору можуть легко захоплювати відеопотоки з RTSP URL.

- Захоплення відео з екрана наземної станції: У найпростішому випадку, якщо відео з дрона відображається на екрані наземної станції (наприклад, у штатному додатку керування дроном), можна спробувати захопити цей відеопотік програмними засобами. Однак, цей метод є менш надійним та може мати вищу затримку.

2. Передача даних телеметрії: Для точного позиціонування виявлених об'єктів на місцевості та інтеграції їх з картографічними даними необхідна інформація про місцезнаходження, висоту, орієнтацію та швидкість дрона. Ці дані (телеметрія) зазвичай передаються з дрона на наземну станцію через радіоканал. Доступ до

телеметрії також може бути отриманий через SDK виробника дрона або за допомогою стандартних протоколів, таких як MAVLink, що використовуються автопілотами PX4/ArduPilot.

3. Синхронізація відео та телеметрії: Для коректного співставлення виявлених об’єктів на відеокадрі з географічними координатами дрона в момент зйомки цього кадру, необхідна точна синхронізація відеопотоку та даних телеметрії. Це може бути досягнуто шляхом використання часових міток (timestamps), що додаються до відеокадрів та пакетів телеметрії на стороні дрона.

4. Розгортання модуля обробки даних: Якщо обробка відбувається на наземній станції, то навчена модель ШІ та програмне забезпечення для її виконання розгортаються на потужному комп’ютері з відповідним апаратним прискоренням GPU NVIDIA. Програмне забезпечення, написане на Python з використанням PyTorch та OpenCV, приймає відеопотік, виконує розпізнавання об’єктів та передає результати до інтерфейсу користувача.

5. Взаємодія із системою управління польотом: У більш складних системах може знадобитися взаємодія результатів розпізнавання об’єктів із системою управління польотом дрона.

6. Програмне забезпечення для інтеграції: Розробка програмного забезпечення для інтеграції включає написання коду, який відповідає за:

- Захоплення та буферизацію відеокадрів.

- Отримання та обробку даних телеметрії.

- Виконання циклу обробки зображень: попередня обробка, подача у ШІ модель, обробка результатів.

- Передачу результатів розпізнавання до інтерфейсу користувача. - Взаємодію з SDK виробника дрона або бібліотеками для роботи з протоколами зв’язку.

Рис. 3.4 – Приклад коду для обробки відеопотоку

Використання мови програмування Python з відповідними бібліотеками (OpenCV, PyTorch, а також бібліотеки для роботи з SDK або протоколами зв'язку) є зручним для реалізації цього програмного забезпечення. Для взаємодії з дроном використано DJI Mobile SDK і бібліотеку OpenCV для захоплення відеопотоку.

Таблиця 3.3 – Характеристики каналу зв'язку дрона

Характеристика	Значення	Примітки
Робоча частота	2.4 ГГц / 5.8 ГГц	Автоматичне перемикання
Максимальна дальність	До 15 км (FCC) / 8 км (CE)	Залежить від умов та стандарту
Затримка (Latency)	Близько 130 мс	Для відеопотоку 1080p/30fps
Пропускна здатність	До 15 Мбіт/с	Для передачі відео та телеметрії
Використовуваний протокол	OcuSync 3+	Власна технологія DJI

Інтерфейс користувача (ІК) є обличчям системи автоматичного розпізнавання об’єктів для оператора. Його основне завдання – забезпечити зручну та інтуїтивно зрозумілу взаємодію з системою, відображення необхідної інформації та надання можливості керування процесом моніторингу.

Розробка інтерфейсу користувача передбачає проєктування його структури, візуального оформлення та реалізацію функціональних можливостей. ІК може бути реалізований у вигляді десктопного додатка, веб-інтерфейсу або мобільного додатку, залежно від специфіки застосування системи. Ключові функціональні можливості інтерфейсу користувача:

1. Відображення відеопотоку в реальному часі: Основна частина інтерфейсу має займати вікно з трансляцією відео з камери дрона. Відеопотік повинен відображатися з мінімальною затримкою та достатньою частотою кадрів для комфортного спостереження.

2. Відображення результатів розпізнавання об’єктів: На відео або зображеннях, що відображаються, повинні наноситись обмежувальні прямокутники (bounding boxes) навколо виявлених об’єктів. Кожен прямокутник має супроводжуватися міткою класу об’єкта та, можливо, оцінкою впевненості моделі у правильності розпізнавання. Використання різних кольорів для різних класів об’єктів може покращити візуальне сприйняття.

Рис. 3.5 – Приклад відображення результатів розпізнавання об’єктів в інтерфейсі користувача

3. Панель інформації про виявлені об’єкти: Окремий блок або список в інтерфейсі може відображати детальну інформацію про всі виявлені на поточному кадрі або за останній проміжок часу об’єкти. Ця інформація може включати: - ID

об’єкта (якщо реалізовано відстеження).

- Клас об’єкта.

- Оцінка впевненості.

- Координати обмежувального прямокутника на зображенні.

- Приблизні географічні координати об’єкта (якщо доступна та синхронізована телеметрія дрона).

- Час виявлення.

4. Відображення телеметрії дрона: В окремій частині інтерфейсу повинна відображатись актуальна інформація про статус дрона:

- Стан зв’язку.

- Рівень заряду батареї.

- Поточні координати (широта, довгота), висота над рівнем землі або точкою зльоту.

- Швидкість (горизонтальна та вертикальна).

- Орієнтація (кут крену, тангажу, рискання).

- Статус GPS.

- Режим польоту.

Ця інформація є важливою для оператора для контролю за ходом місії та безпекою польоту.

5. Картографічне відображення: Інтеграція з картографічним сервісом (OpenStreetMap, Google Maps) дозволяє відображати поточне місцезнаходження дрона та, можливо, відстежувати траєкторію його польоту. На карту також можуть бути нанесені маркери виявлених об’єктів з їхніми географічними координатами. Це надає оператору просторовий контекст та полегшує локалізацію виявлених об’єктів

на місцевості.

6. Елементи керування системою: Інтерфейс повинен надавати оператору можливість:

- Розпочати/зупинити процес розпізнавання об’єктів.

- Вибрати активну модель ШІ (якщо доступно кілька).

- Налаштувати параметри розпізнавання (наприклад, поріг впевненості для відображення об’єктів).

- Вибрати класи об’єктів для розпізнавання.

- Розпочати/зупинити запис відео або збереження знімків екрана. - Керувати місією дрона (якщо передбачено інтеграцію з системою управління польотом).

7. Журнал подій (Log): Ведення та відображення журналу подій системи, що включає інформацію про початок/завершення місії, виявлення об’єктів, помилки зв’язку тощо, може бути корисним для аналізу роботи системи та виявлення проблем.

8. Налаштування відображення: Можливість приховувати/відображати певні елементи інтерфейсу, змінювати розмір вікон, налаштовувати вигляд обмежувальних прямокутників.

Рис. 3.6 – Макет інтерфейсу користувача системи розпізнавання об’єктів Програмна реалізація системи розпізнавання об’єктів є етапом, на якому всі спроєктовані

компоненти та алгоритми перетворюються на працюючий код.

Під час тестування на реальних даних оцінюються такі показники, як точність розпізнавання об’єктів (з використанням метрик mAP, Precision, Recall), швидкість роботи системи та зручність використання інтерфейсу. Нижче наведено спрощений приклад коду для реалізації інтерфейсу з використанням PyQt5.

Рис. 3.7 – Приклад структури проєкту програмного забезпечення 6. Оптимізація: На основі результатів тестування може знадобитися оптимізація програмного забезпечення для покращення продуктивності. Це може включати:

- Оптимізація коду Python для прискорення виконання.

- Використання апаратного прискорення (GPU, NPU) більш ефективно. -

Оптимізація параметрів моделі ШІ або вибір меншої моделі для досягнення необхідної швидкості при прийнятній точності.

- Паралелізація обчислень (багатопоточність/багатопроцесорність). Ретельна

програмна реалізація та всебічне тестування є запорукою створення надійної та ефективної системи автоматичного розпізнавання об'єктів, здатної виконувати поставлені задачі моніторингу.

У даному розділі було детально розглянуто процес розробки системи автоматичного розпізнавання об'єктів дронами. Спроектовано архітектуру системи, що включає дрон, камеру, модуль обробки даних з ШІ-моделлю та інтерфейс користувача, та описано взаємодію між цими компонентами. Обґрунтовано етапи підготовки даних для навчання моделі, починаючи від збору та попередньої обробки візуальних даних, до їх анотації, розподілу на вибірки та збільшення.

Докладно описано процес навчання обраної моделі ШІ (YOLOv8), включаючи конфігурацію моделі, ініціалізацію ваг, визначення функції втрат, вибір оптимізатора та налаштування гіперпараметрів. Визначено ключові метрики для оцінки продуктивності моделі (IoU, Precision, Recall, mAP) та процес оцінки під час та після навчання. Розглянуто аспекти інтеграції системи з дроном, включаючи отримання відеопотоку та телеметрії, синхронізацію даних та розгортання модуля обробки (на наземній станції або борту).

Описано ключові функціональні можливості інтерфейсу користувача, спрямовані на зручне відображення відеопотоку та результатів розпізнавання, а також надання оператору необхідної інформації про дрон та можливості керування системою. Наведено загальні підходи до програмної реалізації та тестування системи, включаючи вибір середовища розробки, реалізацію окремих модулів та методи тестування.

Запропонований підхід до розробки базується на використанні сучасних та ефективних технологій комп'ютерного зору та глибокого навчання (OpenCV, PyTorch, YOLOv8), а також враховує специфіку апаратної платформи (дрони) та

вимоги до швидкості та точності розпізнавання в задачах моніторингу. Реалізація системи відповідно до описаних етапів дозволить створити функціональне рішення для автоматичного розпізнавання об’єктів дронами.

Таблиця 3.4 – Узагальнення обраних технологій та інструментів

Етап розробки	**Ключові технології/Інструменти**
Архітектура системи	Дрон (DJI Mavic 3, Мультироторний), Камера (4K, 20 МП, 3-осьовий підвіс), Модуль обробки (Наземна станція /Бортовий ПК, GPU NVIDIA RTX 3060, ОС Ubuntu 20.04), Інтерфейс користувача
Підготовка даних	Збір власних даних (за допомогою дронів), Існуючі набори даних (COCO, VisDrone), LabelImg, CVAT (Computer Vision Annotation Tool), Data Augmentation (Горизонтальне/вертикальне віддзеркалення, Випадкове обрізання, Обертання, Зміна масштабу, Зміна яскравості/контрасту/насиченості, Додавання шуму, Змішування зображень - CutMix, Mosaic)
Навчання моделі	PyTorch (1.12), YOLOv8 (Nano, Small, Medium, Large, Extra-large варіанти, навчена на COCO), GPU (NVIDIA RTX 3060, 12 ГБ VRAM), Навчальна/Валідаційна/Тестова вибірки, Функція втрат (Localization Loss / Box Loss, Classification Loss, Objectness Loss / Confidence Loss, CIoU, DFL), Оптимізатор (AdamW/SGD), Гіперпараметри (Швидкість навчання, Розмір міні-партії, Кількість епох, Вагова деградація, Коефіцієнти оптимізатора, Параметри Data Augmentation, Пороги для NMS), Метрики (IoU, Precision, Recall, AP, mAP@0.5, mAP@0.5:0.95)
Інтеграція з дроном	RTSP (Real-Time Streaming Protocol), SDK дрона (DJI Mobile SDK, Payload SDK), Телеметрія (MAVLink /SDK), Синхронізація (Часові мітки), Розгортання на наземній станції/борту (NVIDIA Jetson), Python, OpenCV (4.5), PyTorch
Інтерфейс користувача	Фреймворк GUI (PyQt/Tkinter/Web), Відображення відео/результатів (Обмежувальні прямокутники, Мітка класу, Оцінка впевненості, Різні кольори для класів), Панелі інформації про виявлені об’єкти (ID, Клас, Впевненість, Координати на зображенні/географічні, Час виявлення), Відображення телеметрії дрона (Стан зв’язку, Заряд батареї, Координати, Висота, Швидкість, Орієнтація, Статус GPS, Режим польоту), Картографія (OpenStreetMap, Google Maps), Елементи керування (Розпочати/зупинити розпізнавання, Вибрати модель/класи, Налаштувати поріг впевненості, Запис відео/знімків, Керування місією), Журнал подій, Налаштування відображення
Програмна реалізація	Python, OpenCV, PyTorch, Ultralytics YOLOv8, IDE (PyCharm/VS Code), Git/GitHub
Тестування	Модульне тестування, Інтеграційне тестування, Системне тестування, Тестування продуктивності, Тестування надійності, Оцінка метрик на реальних даних (mAP, Precision, Recall, швидкість роботи)

ВИСНОВОК

Представлена кваліфікаційна робота успішно вирішує актуальну науково прикладну задачу інтеграції комп'ютерного зору та штучного інтелекту для автоматичного розпізнавання об'єктів з безпілотних літальних апаратів у процесі моніторингу. Актуальність теми підтверджується зростаючим використанням дронів у різних сферах та необхідністю автоматизації аналізу великих обсягів візуальних даних, отриманих з повітря.

У ході роботи було детально проаналізовано теоретичні основи комп'ютерного зору, включаючи методи обробки зображень та виділення ознак, а також сучасні підходи до розпізнавання об'єктів на базі штучного інтелекту, зокрема глибокого навчання. Виявлено та окреслено ключові виклики, що виникають при автоматичному розпізнаванні об'єктів дронами, такі як зміни ракурсу, масштабу, освітлення, вібрації та обмежені обчислювальні ресурси платформи.

На основі проведеного аналізу обґрунтовано вибір технологічного стеку для реалізації системи. Для попередньої обробки зображень та відео, а також реалізації допоміжних функцій комп'ютерного зору обрано бібліотеку OpenCV. Як основний фреймворк для розробки та навчання моделі глибокого навчання обрано PyTorch, що забезпечує гнучкість та ефективність. Серед різноманіття моделей детектування об'єктів перевагу надано моделі YOLOv8, яка демонструє оптимальний баланс між точністю (mAP) та швидкістю (FPS), що є критично важливим для застосувань у реальному часі на борту дрона або наземній станції з обмеженими ресурсами.

Спроектовано архітектуру інтегрованої системи, яка охоплює взаємодію дрона з камерою, модуля обробки даних з реалізованою ШІ-моделлю та інтерфейсу користувача. Детально описано процес підготовки даних, включаючи збір (власний датасет та існуючі набори даних), анотацію (з використанням LabelImg або CVAT), розподіл на вибірки та застосування технік аугментації даних для підвищення стійкості моделі до різних умов зйомки. Розглянуто ключові аспекти навчання моделі YOLOv8, включаючи конфігурацію варіанта моделі (наприклад, YOLOv8m), ініціалізацію ваг (зокрема, попередньо навченими вагами), визначення функції втрат, вибір оптимізатора (AdamW/SGD) та налаштування гіперпараметрів.

Визначено та застосовано відповідні метрики оцінки продуктивності моделі (IoU, Precision, Recall, mAP@0.5, mAP@0.5:0.95), що дозволило об'єктивно оцінити якість навченої моделі.

Розглянуто важливі аспекти інтеграції системи з дроном, зокрема отримання відеопотоку (запропоновано використання RTSP або SDK виробника дрона) та телеметрії, а також необхідність синхронізації цих даних для точної геолокації виявлених об'єктів. Описано функціональні вимоги до інтерфейсу користувача, який має забезпечувати зручне відображення відеопотоку з результатами розпізнавання, інформації про виявлені об'єкти, телеметрії дрона та можливості керування системою.

Таким чином, у роботі було успішно вирішено поставлені завдання, проведено аналіз сучасного стану досліджень, визначено ключові виклики, розроблено архітектуру системи, обрано та обґрунтовано ефективні алгоритми та моделі, а також розглянуто аспекти їх інтеграції та реалізації. Отримана система є працездатною архітектурою, що дозволяє дрону ефективно виявляти об’єкти під час польоту. Практична цінність роботи полягає в можливості використання розроблених підходів та отриманих результатів для створення реальних систем моніторингу з автоматичним розпізнаванням об'єктів на базі безпілотних літальних апаратів.

КРИВОРІЗЬКИЙ ФАХОВИЙ КОЛЕДЖ
ДЕРЖАВНОГО НЕКОМЕРЦІЙНОГО ПІДПРИЄМСТВА
«ДЕРЖАВНИЙ УНІВЕРСИТЕТ «КИЇВСЬКИЙ АВІАЦІЙНИЙ ІНСТИТУТ»

РЕЦЕНЗІЯ
на кваліфікаційну роботу

випускника спеціальності: 123 «Комп'ютерна інженерія»

відділення: комп'ютерної та програмної інженерії

циклова комісія: комп'ютерних систем та мереж

Єгор ЛІЗКО
(ім'я, прізвище)

1. Актуальність теми: Обрана тема кваліфікаційної роботи «Інтеграція комп'ютерного зору та штучного інтелекту для автоматичного розпізнавання об'єктів дронами в задачах моніторингу» є беззаперечно актуальною. Зростаюче застосування БПЛА в різних сферах вимагає створення ефективних засобів для автоматизованого аналізу візуальної інформації, що дозволяє підвищити швидкість та точність моніторингу.
2. Кваліфікаційна робота відповідає темі, затвердженій наказом.
3. Завдання на виконання кваліфікаційної роботи виконано у повному обсязі.
4. В результаті виконання кваліфікаційної роботи було створено працездатну архітектуру системи. Проведено глибокий аналіз сучасних технологій, зокрема бібліотек OpenCV, TensorFlow та PyTorch. Обґрунтовано вибір моделі глибокого навчання YOLOv8 як оптимальної за критеріями швидкості та точності для задач моніторингу.
5. Якість виконання пояснювальної записки та ілюстративного (графічного) матеріалу відповідає вимогам до чинних стандартів, представлений обсяг матеріалу — достатній та структурований.
6. В кваліфікаційній роботі зроблений детальний аналітичний огляд, грамотне обґрунтування вибору технологічного стеку та практичну спрямованість отриманих результатів.
7. Кваліфікаційна робота заслуговує оцінку «добре».

Рецензент _____
(науковий ступінь, посада)

«11» 06 2025 р. _____ Андрій КОЖАЕВ
(підпис) (ім'я, прізвище)

З рецензією ознайомлений _____ Єгор ЛІЗКО
(підпис) (ім'я, прізвище)