МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління

КАРНИДАЛ Михайло Сергійович

Метод виявлення та класифікації транспортних засобів на основі комп'ютерного зору / Method for Detection and Classification of Vehicles Based on Computer Vision

спеціальність: 122 - Комп'ютерні науки
освітньо-професійна програма - Комп'ютерні науки

Кваліфікаційна робота

Виконав студент групи КНм-21
М.С. Карнидал
_____

Науковий керівник:
к.т.н., доцент Д.І. Загородня
_____

Кваліфікаційну роботу
допущено до захисту:
«___» _____ 20___ р.
В.о. завідувача кафедри
_____ Н.М. Васильків

ТЕРНОПІЛЬ – 2024

Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління
Освітній ступінь «магістр»
спеціальність: 122 – Комп'ютерні науки
освітньо-професійна програма – Комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ М.П. Комар
«____» _____20__р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Карнидал Михайло Сергійович
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи
Метод виявлення та класифікації транспортних засобів на основі комп'ютерного зору / Method for Detection and Classification of Vehicles Based on Computer Vision
керівник роботи к.т.н., доцент Д.І. Загородня
затверджені наказом по університету від 12 грудня 2023 року № 753.

2. Строк подання студентом закінченої кваліфікаційної роботи 4 грудня 2024 р.

3. Вихідні дані до кваліфікаційної роботи: завдання на кваліфікаційну роботу студента, наукові статті, технічна література.

4. Основні питання, які потрібно розробити
- огляд предметної області;
- аналіз алгоритмів для розв'язку задач класифікації;
- аналіз нейронних мереж для вирішення проблем задач класифікації;
- постановка задачі дослідження;
- дослідження алгоритму класифікації;
- запропонований алгоритм виявлення та класифікації даних;
- дослідження засобів реалізації запропонованого алгоритму;
- реалізація алгоритму класифікації даних на основі нейронних мереж;
- результати експериментальних досліджень.

5. Перелік графічного матеріалу у роботі
- структура згорткової нейронної мережі;
- архітектура YOLO;
- діаграми оцінки точності та швидкості роботи методу;
- результат роботи програми.

6. Консультанти розділів кваліфікаційної роботи

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

7. Дата видачі завдання 12 грудня 2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	Затвердження теми кваліфікаційної роботи, ознайомлення з літературними джерелами та складання плану роботи.	до 01.01. 2024 р.	
2	Написання 1 розділу кваліфікаційної роботи	до 01.03. 2024 р.	
3	Написання 2 розділу кваліфікаційної роботи	до 20.05.2024 р.	
4	Написання 3 розділу кваліфікаційної роботи	до 28.10. 2024 р.	
5	Представлення попереднього варіанту кваліфікаційної роботи, перевірка та внесення змін керівником	до 11.11.2024 р.	
6	Опрацювання зауважень та представлення завершеного варіанту кваліфікаційної роботи. Підготовка супроводжуючих документів.	до 25.11.2024 р.	
7	Перевірка кваліфікаційної роботи на оригінальність тексту.	до 30.11.2024 р.	
8	Оформлення кваліфікаційної роботи та отримання допуску до захисту.	до 04.12.2024 р.	
9	Подання кваліфікаційної роботи до захисту на засіданні атестаційної комісії.	до 14.12. 2024 р.	

Студент _____ М.С. Карнидал
підпис

Керівник роботи _____ к.т.н., доцент Д.І. Загородня
підпис

РЕЗЮМЕ

Кваліфікаційна робота на тему «Метод виявлення та класифікації транспортних засобів на основі комп'ютерного зору» виконана для здобуття освітнього ступеня «Магістр» за спеціальністю 122 «Комп’ютерні науки» в межах освітньо-професійної програми «Комп’ютерні науки» написана обсягом у 80 сторінок, містить 22 ілюстрації, 4 додатки та 38 використаних джерела.

Метою роботи є розробка методу для автоматизованої системи виявлення та класифікації транспортних засобів з використанням алгоритмів глибокого навчання. Така система здатна працювати в реальному часі та забезпечувати високу точність навіть у складних умовах дорожнього середовища..

Методи дослідження, використані у роботі, включають згорткові нейронні мережі, модифіковану модель YOLOv8 з модулем для покращеного виявлення транспортних засобів завдяки фокусуванню на важливих ознаках, традиційні методи обробки зображень для попередньої підготовки даних.

Результати дослідження: розроблено метод для інтегрованої системи моніторингу дорожнього руху, що використовує модифіковану архітектуру YOLOv8 із впровадженням модуля додаткової ували для підвищення точності детекції об'єктів. Проведено експериментальне дослідження та аналіз ефективності системи.

Практичне значення роботи полягає у створенні методу для системи, яка здатна автоматизувати процес моніторингу дорожнього руху в реальному часі.

Рекомендації для подальших досліджень включають вдосконалення алгоритмів роботи в умовах низького освітлення, розширення можливостей системи для аналізу складних дорожніх умов, інтеграцію з системами прогнозування заторів та розробку адаптивних моделей, які враховують погодні умови.

Ключові слова: ВИЯВЛЕННЯ ТА КЛАСИФІКАЦІЯ ТРАНСПОРТНИХ ЗАСОБІВ, ГЛИБОКЕ НАВЧАННЯ, YOLOV8, ОБРОБКА ЗОБРАЖЕНЬ, ВІДСТЕЖЕННЯ, КОМП'ЮТЕРНИЙ ЗІР, МОНІТОРИНГ ТРАНСПОРТУ.

ABSTRACT

Qualification work on the topic «Method for Detection and Classification of Vehicles Based on Computer Vision» for Master's degree on specialty 122 «Computer Science» educational and professional program «Computer Science» is written on 80 pages and it contains 22 figures, 4 annexes and 38 sources.

The aim of the work is to develop a method for an automated vehicle detection and classification system using deep learning algorithms. Such a system is capable of operating in real-time and ensuring high accuracy even in challenging road conditions.

The research methods used in the work include convolutional neural networks (CNN), a modified YOLOv8 model with an attention module for enhanced vehicle detection by focusing on significant features, and traditional image processing methods for data preprocessing.

Research results: A method for an integrated traffic monitoring system has been developed, utilizing the modified YOLOv8 architecture with an additional attention module to improve object detection accuracy. Experimental research and performance analysis of the system have been conducted.

The practical significance of the work lies in creating a method for a system capable of automating the traffic monitoring process in real-time.

Recommendations for further research include improving the algorithms' performance under low-light conditions, expanding the system's capabilities to analyze complex road environments, integrating with congestion prediction systems, and developing adaptive models that account for weather conditions.

Keywords: VEHICLE DETECTION AND CLASSIFICATION, DEEP LEARNING, YOLOV8, IMAGE PROCESSING, TRACKING, COMPUTER VISION, TRAFFIC MONITORING.

Зміст

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

CBAM – Модуль уваги згорткових блоків (Convolutional Block Attention Module)

CNN – Згорткові нейронні мережі (Convolutional Neural Networks)

COCO – Звичайні об'єкти у контексті (Common Objects in Context)

HOG – Гістограма орієнтованих градієнтів (Histogram of Oriented Gradients)

KNN – Метод найближчих сусідів (K-Nearest Neighbors)

MLP – Мережі прямого поширення (Multilayer Perceptron)

RNN – Рекурентні нейронні мережі (Recurrent Neural Networks)

SIFT – Масштабно-інваріантне перетворення ознак (Scale-Invariant Feature Transform)

SVM – Метод опорних векторів (Support Vector Machines)

YOLO – You Only Look Once

ВСТУП

Транспортна система є важливою складовою сучасної інфраструктури, яка забезпечує ефективне функціонування економіки, зручність для громадян та безпеку перевезень. Одним із ключових завдань сучасного транспорту є моніторинг і управління дорожнім рухом, що дозволяє знижувати рівень заторів, покращувати екологічну ситуацію та підвищувати безпеку на дорогах. Виявлення та класифікація транспортних засобів є основою для вирішення цих завдань, адже вони дозволяють зібрати необхідні дані для аналізу та прийняття управлінських рішень.

Актуальність теми дослідження зумовлена потребою у створенні точних і ефективних систем моніторингу дорожнього руху, які можуть автоматично визначати типи транспортних засобів, відстежувати їхні переміщення та забезпечувати роботу в реальному часі. Традиційні методи аналізу, засновані на обробці зображень, мають обмеження, зокрема в умовах низького освітлення або складного дорожнього середовища. Сучасні технології глибокого навчання, такі як згорткові нейронні мережі, пропонують нові можливості для підвищення точності та надійності таких систем.

Метою даного дослідження є розробка методу виявлення та класифікації транспортних засобів на основі модифікованої моделі YOLOv8, здатного працювати в реальному часі з підвищеною точністю завдяки використанню вдосконаленої архітектури, зокрема інтеграції модуля для фокусування на важливих ознаках об'єктів навіть у складних умовах.

Завданнями дослідження є:

– Аналіз сучасних методів і підходів до виявлення та класифікації транспортних засобів у відеопотоках.

– Розробка методології модифікації моделі YOLOv8 шляхом інтеграції модуля для покращення точності виявлення дрібних та перекритих об'єктів.

– Реалізація методу виявлення та класифікації транспортних засобів із використанням модифікованої архітектури.

– Проведення порівняльного аналізу продуктивності традиційних методів обробки зображень і сучасних глибоких нейронних мереж.

– Оцінка ефективності запропонованої системи у різних умовах експлуатації, включно з низьким освітленням, високою щільністю трафіку та динамічними змінами на дорозі.

Об’єктом дослідження є процес виявлення та класифікації транспортних засобів у відеопотоках. Предметом дослідження є модифіковані алгоритми та методи, що застосовуються для автоматизації моніторингу дорожнього руху з підвищеною точністю та швидкістю обробки.

Для досягнення поставленої мети у роботі використано такі методи:

– Згорткові нейронні мережі (CNN) для класифікації об’єктів на основі вхідних зображень.

– Модифікована модель YOLOv8 із інтегрованим модулем для виявлення, локалізації та покращеної обробки ознак транспортних засобів.

– Методи обробки зображень, такі як попереднє масштабування, колірні перетворення та аугментація даних для підвищення стійкості системи.

– Алгоритми трекінгу для відстеження об’єктів у відеопотоці та уникнення дублікатів детекцій між кадрами.

– Експериментальні дослідження та тестування в умовах реальних дорожніх сценаріїв.

Для досягнення поставленої мети у роботі використано такі методи: згорткові нейронні мережі для класифікації об’єктів, алгоритм YOLO для виявлення і відстеження об’єктів, методи традиційної обробки зображень, а також засоби тестування в умовах реальних дорожніх сценаріїв.

Наукова новизна дослідження полягає у розробці нової методу, що використовує модифіковану архітектуру YOLOv8 з модулем для підвищення точності виявлення та класифікації транспортних засобів. Завдяки впровадженню механізму уваги запропонований модифікований метод здатен краще фокусуватися на важливих просторових та канальних ознаках, що забезпечує високу

продуктивність навіть у складних умовах низького освітлення та великої кількості об’єктів на зображенні.

Практичне значення роботи полягає у можливості застосування розробленого методу для системи автоматизованого моніторингу дорожнього руху. Система дозволяє ефективно виявляти та класифікувати транспортні засоби, що дає змогу підвищити рівень безпеки на дорогах, оптимізувати роботу міських транспортних систем та забезпечити аналіз дорожньої ситуації у режимі реального часу. Такі інтелектуальні транспортні системи, використовують для міського відеоспостереження, а також можуть стати інструментом для планування та управління дорожнім рухом.

Апробація результатів дослідження здійснювалася шляхом представлення основних теоретичних положень і практичних результатів на Всеукраїнській науково-практичній конференції студентів, аспірантів та молодих вчених «Інтелектуальні комп’ютерні системи та мережі» (м. Тернопіль, Україна), а також на Міжнародній науково-практичній інтернет-конференції «Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення» (м.Тернопіль, Україна).

1 ТЕОРЕТИКО-МЕТОДИЧНІ ЗАСАДИ ВИЯВЛЕННЯ ТА КЛАСИФІКАЦІЇ ТРАНСПОРТНИХ ЗАСОБІВ

1.1 Аналіз предметної області та актуальність дослідження

Актуальність дослідження за тематикою комп’ютерного зору, а саме виявлення транспортних засобів полягає в необхідності такого роду технологічно-програмних рішень [1]. Важливою складовою в питанні актуальності дослідження в сфері комп’ютерного зору в даному випадку актуальність питання виявлення транспортних засобів на зображеннях, відео та відеопотоках є питання використання технології виявлення транспортних засобів в автономних транспортних засобах для забезпечення виконання ряду необхідних функцій для забезпечення безпечного перевезення транспортного вантажу або пасажирів. На даний момент сфера безпілотних транспортних засобів доволі стрімко розвивається що в свою чергу зумовлює розвиток технології виявлення транспортних засобів.

Основною проблемою виявлення транспортних засобів в даному випадку на транспортних засобах, які використовують технологію безпілотності які як можливо припустити обладнані системою комп’ютерного зору, а саме системою виявлення транспортних засобів полягає в тому що не завжди коректно відбувається виявлення транспортних. Наслідком цього некоректного виявлення транспортних засобів, а транспортного засобу є аварійні ситуації або ж навіть дорожньо-транспортні пригоди.

Актуальність розвитку алгоритмів та систем виявлення транспортних засобів покликана мінімізувати відсоток хибних виявлень в системах виявлення транспортних засобів з використанням комп’ютерного зору що наступним чином дозволить зменшенню нещасних випадків на дорогах з участю автономних транспортних засобів. Також слід зазначити, що висока енерговитратність алгоримтів та систем виявлення транспортних засобів запобігає більшому розвитку систем виявлення транспортних засобів.

Важливим фактором в актуальності теми дослідження алгоритмів виявлення транспортних засобів є використання алгоритмів виявлення транспортних засобів

в системах моніторингу дорожнього руху. Системи моніторингу дорожнього руху з використанням технології комп'ютерного зору та алгоритмів виявлення транспортних засобів за допомогою комп'ютерного зору забезпечують більш оптимальне управління за дорожнім рухом в тих місцях де вони використовується [2, 3]. Також слід зазнати що використання в системах моніторингу дорожнього руху алгоритмів виявлення транспортних засобів відкриває можливість оптимізації транспортних потоків. Підвищення оптимізованості транспортних потоків позитивно впливає на загальний стан транспортної інфраструктури та допомагає в зменшенні відсотків утворення транспортних засобів на дорогах що позитивно впливає на різні аспекти людського життя [4]. Також слід зазначити що актуальність дослідження в сфері комп'ютерного зору а даному випадку в алгоритмах виявлення транспортних засобів зумовлене необхідністю підвищити загальний стан безпеки. При впровадженні систем моніторингу дорожнього руху також інколи страждають від деякого відсотку некоректно виявлених транспортних засобів що в деяких випадках спричиняє аварійні ситуації або ж дорожньо-транспортні пригоди. Для того, щоб зменшити відсоток некоректно виявлених транспортних засобів необхідно покращення в деяких проблематичних місцях в алгоритмах виявлення транспортних засобів за допомогою комп'ютерного зору, а саме в енергоефективності даних алгоритмів адже алгоритми виявлення транспортних засобів потребують великих обчислювальних потужностей для біль коректного виявлення транспортних засобів особливо в режимі реального часу тобто у відеопотоці. Алгоритми які на даний час є високоефективними в плані використання обчислювальних ресурсів мають нижчий відсотковий показник коректно виявлених транспортних засобів, це зумовлене в певному ступені недосконалості сучасних алгоритмів виявлення транспортних засобів. На противагу низько ефективним в плані відсоткового коректно виявлених транспортних засобів існують високоефективні за критерієм коректно виявлених транспортних засобів проте дані алгоритми є низько ефективними в плані використання ресурсів.

Актуальність дослідження в сфері комп’ютерного зору, а саме в сфері виявлення транспортних засобів також зумовлене використанням алгоритмів виявлення транспортних засобів в системах безпеки. Серед систем безпеки де використовується алгоритми виявлення транспортних засобів це системи відеоспостереження що в свою чергу впливає на показники забезпечення безпеки на підприємствах також в громадських місцях і в приватних володіннях [5-10].

Ці системи дають змогу виявляти транспортні засоби різних типів від легкових автомобілів до вантажівок. Також слід зазначити, що актуальність теми дослідження в сфері комп’ютерного зору а саме в сфері виявлення транспортних засобів зумовлено розвитком логістики та транспорту. Оптимізація логістичних процесів та управління транспортними потоками позитивно впливає на багато секторів економіки. Необхідно зазначити що вирішення завдання проблеми виявлення транспортних засобів також необхідно визначити адже при виконанні логістичних завдань необхідна точність виявлення транспортних засобів та надійність тобто коректність виявлення транспортних засобів на фото відео та відеопотоках адже при некоректному визначенні транспортних засобів з’являється неочікувані проблеми такі як не оптимальність маршрутів, згаяний час в простої транспортних засобів які перевозлять необхідні вантажі. Ці чинники які виникають при некоректному виявленні транспортних засобів тягнуть за собою неефективно витрачені ті чи інші ресурси що можна охарактеризувати як збитки для тих чи інших компаній або окремих осіб. Для вирішення проблеми потрібно зменшити енерговитратність алгоритмів виявлення транспортних засобів про щоб це в свою чергу не спричиняло зменшення ефективності в плані точності виявлення транспортних засобів що є основною метою даних алгоритмів [11].

Актуальність дослідження в сфері комп’ютерного зору, а саме в сфері виявлення транспортних засобів зумовлено необхідністю моніторингу забруднення довкілля. Виявлення транспортних засобів дає змогу оцінити рівні забруднення довкілля що в свою чергу дає змогу розробити засоби для подолання проблеми забруднення довкілля. Необхідно зазначити що кожен транспортний засіб в результаті своєї роботи як залишковий продукт тобто відходи викидає в атмосферу

певну кількість шкідливих для довкілля хімічних елементів за винятком транспортних засобів які працюють за принципом використання електроенергії проте таких транспортних засобів на даний час не дуже великий відсоток від загальної кількості транспортних засобів, більшість з транспортних засобів складають транспортні засоби на двигунах внутрішнього згоряння що використовують бензин в якості потрібного енергоресурсу для виконання своїх функцій. Проблематикою для систем спостереження за транспортними засобами які використовуються для моніторингом стану забруднення через викиди небезпечних елементів в атмосферу є недостатня кількість та висока вартість таких систем що зумовлене низькою енергоефективністю алгоритмів виявлення транспортних засобів. Вирішенням цієї проблеми в системах моніторингу стану забруднення довкілля є збільшення енергоефективності алгоритмів виявлення транспортних засобів проте необхідно не допустити падіння точності виявлення транспортних засобів адже некоректне виявлення транспортних засобів стає причиною хибних вихідних в неякому відсотку залежно від відсотку некоректно виявлених транспортних засобів даних.

Також необхідно зазначити що актуальність теми дослідження сфери виявлення транспортних засобів зумовлене також і необхідністю управління паркувальними системами що зумовить зменшення забруднення атмосфери та довкілля. Проте некоректне виявлення транспортних засобів спричинить помилковість в деякому відсотку вихідних даних в цих системах [12].

Також необхідно зазначити що алгоритми виявлення транспортних засобів також використовуються в системах відеоспостереження у військових цілях і розвиток цієї галузі також зумовлено актуальність дослідження в тематиці комп’ютерного зору а саме в сфері виявлення транспортних засобів. При некоректному виявленні транспортних засобів в використанні в військових цілях результатом може стати негативні результати в тому чи іншому. Вирішенням цієї проблеми є загальне покращення алгоритмів виявлення транспортних засобів шляхом зменшення використання енергоресурсів проте що б не приводило до зменшення відсотку виявлення транспортних засобів, це дасть результат таким

чином що дасть змогу зменшити використання ресурсів для використання та впровадження та покращення систем виявлення транспортних засобів.

Розвиток IoT систем також позитивно впливає на актуальність дослідження в темі комп'ютерного зору а саме в сфері виявлення транспортних засобів. Для забезпечення впровадження так званого розумного міста необхідно забезпечити в першу чергу розумне автоматизоване управленні транспортним трафіком для мінімізації впливу людських помилок та зменшить відсоток дорожньо-транспортних пригод. Проблемою в впровадженні розумного міста є висока вартість та висока не енергоефективність алгоритмів комп'ютерного зору, а саме алгоритмів виявлення транспортних засобів. Вирішенням проблеми є вдосконалення алгоритмів виявлення транспортних засобів шляхом збільшення відсотку виявлення транспортних засобів та зменшенням енергоефективності алгоритму. В поєднанні цих двох вдосконалень та загальним розвитком систем комп'ютерного зору повинно стати зменшення вартості впровадження даних систем розумного міста що позитивно впливатиме на сектори економіки.

1.2 Аналіз методів виявлення та класифікації транспортних засобів

Існують безліч методик, моделей та методів в існуючому методичному та методологічному забезпечення в сфері виявлення об'єктів а саме транспортних засобів. HOG (Histogram of Oriented Gradients) це є один з традиційних методів в комп'ютерному зорі що використовується в завданнях виявлення транспортних засобів. Метод HOG дає змогу витягувати інформацію про контур та форму об'єктів [13].

Принцип роботи HOG можна поділити на декілька етапів. На першому етапі виконується дії такі, для початку дані трансформуються в сітку клітин далі для кожної клітини вираховується градієнт інтенсивності тобто вектор напрямку і величина де інтенсивність змінюється найінтенсивніше тобто виконується обчислення градієнтів. Наступним кроком відбувається створення гістограм тобто

під кожну клітину розробляється гістограма яка забезпечує орієнтацію градієнтів. Кожен елемент в даній гістограмі має відповідати деякому діапазону орієнтації градієнтів. Далі наступним чином так отримується вектор ознак, що описує розподіл орієнтації градієнтів в кожній певній клітині. Наступним етапом відбувається нормалізації тобто з метою покращити інваріантності у відповідності до змін освітленості і контрасту вектори ознак нормалізуються за підтримки блокової нормалізації, тобто це мається на увазі що вектори ознак з клітинами які є сусідами відбувається об'єднання в так звані блоки і для кожного блоку виконується обчислення норми. Наступним етапом принципу роботи HOG є формування дескриптора тобто відбувається злиття нормалізованих векторів ознак для усіх блоків що в свою чергу створює завершальний дескриптор HOG для даних.

Ефективність методу Histogram of Oriented Gradients (HOG) полягає у його здатності забезпечувати інваріантність до змін освітлення завдяки нормалізації. Цей процес робить HOG-дескриптор менш чутливим до варіацій освітлення. Крім того, HOG демонструє інваріантність до геометричних спотворень, що забезпечує стійкість до змін масштабу, поворотів або незначних деформацій даних.

Висока ефективність HOG також обумовлена обчислювальною простотою алгоритму. Метод є досить нескладним і може бути ефективно реалізований на сучасних комп'ютерах, що робить його доступним для широкого використання. Окрім цього, HOG володіє високою дискримінаційною здатністю, тобто дозволяє ефективно розпізнавати об'єкти багатьох класів [14].

Серед основних переваг HOG можна виділити простоту розуміння та реалізації, а також ефективність у різноманітних задачах, зокрема у виявленні транспортних засобів. Крім того, перевагою методу є його інваріантність до багатьох типів змін у даних, що забезпечує стійкість при обробці реальних зображень.

Проте метод HOG має й певні недоліки. Зокрема, він є чутливим до шуму у вхідних даних, що може значно вплинути на його продуктивність у випадку низькоякісних або зашумлених зображень. Ще одним недоліком HOG є його

неефективність при роботі з об’єктами зі складними текстурами, де дескриптор може не забезпечити необхідну точність.

Таким чином, HOG залишається потужним та ефективним методом для задач виявлення об’єктів завдяки своїй простоті, обчислювальній ефективності та стійкості до багатьох типів змін, проте його використання потребує врахування чутливості до шуму та обмежень у роботі з текстурованими об'єктами.

Серед методів виявлення транспортних засобів популярністю користується метод Scale-Invariant Feature Transform (SIFT). Це алгоритм, розроблений для задач виявлення та опису локальних ознак на зображеннях. Основною особливістю SIFT є його інваріантність до масштабів, обертання, змін освітленості та певних типів афінних трансформацій [15-17].

Принцип роботи SIFT полягає у кількох ключових етапах. Першим етапом є виявлення екстремумів у масштабно-інваріантній піраміді, де зображення аналізується на різних масштабах. Для цього застосовується оператор різниці гаусових фільтрів (DoG), який виявляє локальні екстремуми у просторі та масштабі. Далі виконується локалізація екстремумів – точки уточнюються за допомогою інтерполяції, а екстремуми з низьким контрастом або поганою локалізацією відкидаються.

Наступним етапом є обчислення орієнтації для кожної ключової точки. У цьому процесі будується градієнтна гістограма в локальній околиці точки, а домінантна орієнтація визначається як пік гістограми. Завершальним етапом є формування дескриптора. У локальній околиці ключової точки обчислюється вектор ознак (дескриптор), який описує розподіл градієнтів. Дескриптор вирівнюється відносно домінантної орієнтації, що забезпечує його інваріантність до обертання.

До основних переваг SIFT можна віднести його інваріантність до масштабу, обертання та змін освітлення, а також високий рівень повторюваності ключових точок. Це означає, що точки, знайдені на різних зображеннях одного об’єкта, мають подібні дескриптори. Ще однією перевагою є висока ефективність, оскільки алгоритм SIFT підтримує паралелізованість обчислень.

Проте метод SIFT має і свої недоліки. По-перше, це обчислювальна складність – для обробки великих обсягів даних або роботи в реальному часі потрібні значні обчислювальні ресурси. По-друге, алгоритм є чутливим до шуму, що може призвести до появи помилкових ключових точок на зашумлених зображеннях. Ще одним недоліком є те, що SIFT має обмежену стійкість до великих деформацій об'єктів, через що ефективність його роботи з об’єктами, які зазнали значних негативних трансформацій, знижується.

Таким чином, SIFT є надійним методом для задач виявлення та опису локальних ознак завдяки своїй стійкості до масштабів, обертання та освітлення. Проте його застосування вимагає врахування обчислювальних обмежень і чутливості до шумів у складних умовах.

Серед методів виявлення об’єктів широко відомо про алгоритм під назвою згорткові нейронні мережі. Згорткові нейронні мережі це є спеціальний вид штучних нейронних мереж, що має значні успіхи в сфері комп’ютерного зору тобто в виявленні об’єктів а саме в даному випадку в виявлені транспортних засобів. Згорткові нейронні мережі були натхненними біологічними процесами зору і непогано можуть підійти для оброблення даних які мають сітчасту структуру таких як зображення, відео або відеопотік.

Принцип роботи згорткових нейронних мереж або CNN наступний, згорткові нейронні мережі складаються з декількох різних шарів серед яких є згорткові шари тобто це являє собою основний будівельний блок згорткових нейронних мереж ці згорткові шари використовують фільтри тобто ядра для вхідних даних забираючи з даних характерні ознаки серед яких краї, кути текстури. Наступними шарами йдуть щари підвибірки тобто ці шари зменшують розмірність яке має зображення зберігаючи найважливішу інформацію. Наступним етапом є повні з’єднані шари тобто класифікуються забрані знаки видається кінцеви й результат [18].

Згорткові нейронні мережі (Convolutional Neural Networks, CNN) є потужним інструментом для розпізнавання і класифікації зображень, і вони чудово підходять для задач, пов’язаних із виявленням транспортних засобів на зображеннях. Наша

мета — класифікувати зображення з транспортними засобами, такими як автомобілі, вантажівки, фургони або велосипеди.

Процес починається з того (рисунок 1.1), що на вхід мережі подається зображення, наприклад, зображення автомобіля. У термінах комп'ютера це просто набір пікселів, що складають картинку. Зображення передається через кілька етапів обробки.

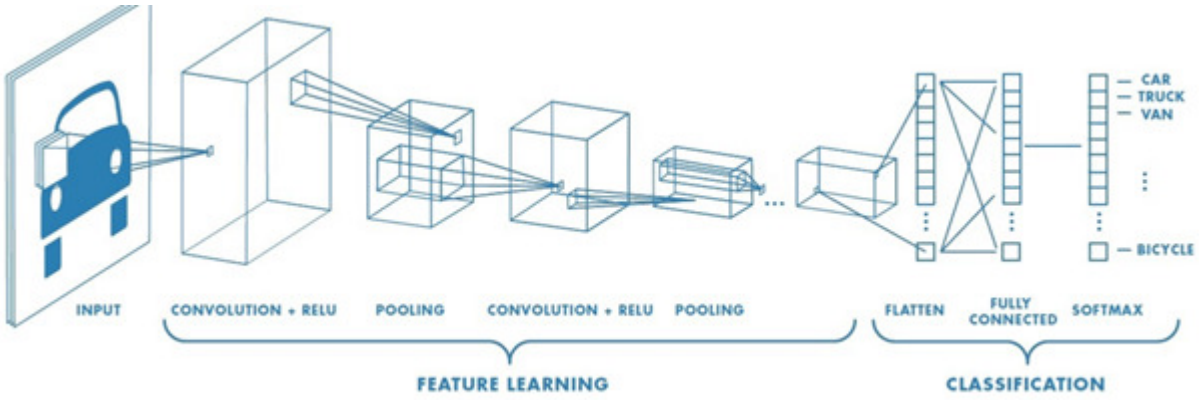


Рисунок 1.1 – Структура згорткової нейронної мережі

На першому етапі мережа застосовує згортку (convolution), де вона намагається виявити базові ознаки зображення, такі як контури, текстури чи кольори. Цей процес дозволяє мережі «побачити» основні елементи, що складають об'єкт, наприклад, контур кузова або форму колеса. Після цього застосовується функція ReLU (Rectified Linear Unit), яка допомагає активувати позитивні значення, відкидаючи негативні. Це сприяє більш швидкому навчанню та покращенню ефективності мережі.

Далі застосовується етап пулінгу (pooling), який зменшує розмір зображення. Це дозволяє зберігати лише важливі ознаки, одночасно зменшуючи обчислювальні витрати. Наприклад, мережа може зберегти тільки важливі характеристики об'єкта, такі як форма кузова чи кількість коліс, а інші деталі ігнорує.

Ці кроки повторюються кілька разів, і кожен раз мережа витягує все складніші ознаки. Якщо на першому етапі вона бачить лише базові контури, то на наступних етапах мережа може почати виявляти більш конкретні елементи, такі як колеса або двері автомобіля.

Коли зображення оброблено, наступним кроком є площинення (flatten), тобто перетворення даних у вектор. Цей вектор подається в повнозв'язаний шар (fully connected layer), де мережа комбінує всі отримані ознаки, щоб зробити остаточний прогноз.

Останній етап — це застосування функції Softmax, яка перетворює отримані дані в ймовірності для кожного з класів, наприклад, для класів «автомобіль», «вантажівка», «фургон», «велосипед» тощо. Мережа вибирає клас з найвищою ймовірністю, і це стає її фінальним прогнозом.

Наприклад, якщо мережа отримала зображення автомобіля, вона спочатку виявить базові ознаки, такі як контури кузова та колеса. Потім на більш глибоких етапах вона навчиться розпізнавати унікальні деталі, які характерні саме для автомобіля. В кінці, після всіх етапів обробки, мережа дасть високу ймовірність того, що це автомобіль, і саме цей клас буде обраний як результат.

Такий підхід дозволяє нейронним мережам не лише ефективно класифікувати транспортні засоби, але й бути стійкими до різних умов, таких як змінене освітлення чи деформації об'єкта на зображенні. Застосування таких методів є ключовим для систем, що займаються автоматизованим моніторингом дорожнього руху, де важливо точно виявляти та класифікувати різні типи транспортних засобів.

Ефективність згорткових нейронних мереж мереж підтверджується тим, що згорткові нейронні мережі інваріативні до зсуву тобто, згорткові нейронні мережі мають можливість виявляти ознаки об’єктів незалежно від розташування в даних. Також ефективність в даних таких як зображення та відео підтверджується тим, що згорткові нейронні мережі використовують виявлення локальних ознак тобто фільтри CNN дають змогу виявити локальні ознаки, що є важливим етапом розпізнавання для прикладу транспортних засобів. Також ефективність згорткових нейронних мереж підтверджується тим фатктом, що в згорткових нейронних мережаж виконується ієрархічна обробка тобто дані обробляються по черзі в різних шарах алгоритму від простих ознак до в свою чергу до більш абстрактних представлень [19].

Також перевагою згорткових нейронних мереж є висока точність, тобто згорткові нейронні мережі мають можливість показувати великі результати в метриці точності в багатьох завданнях комп'ютерного зору в даному випадку в виявленні об'єктів, а саме транспортних засобів різного типу.

YOLO – це алгоритм який побудований за принципом глибокого навчання, що був спеціально розроблений для виявлення об'єктів на зображеннях та у відеопотоці. Принцип роботи YOLO полягає в наступному: для початку дані розподіляються на сітку яка є рівномірною і кожна певна клітина має зону відповідальності за прогнозування об'єктів, центри яких є в середині клітинки. Далі виконується процес під назвою передбачення граничних рамок де для кожної окремої клітини мережа прогнозує декілька граничних рамок [20-21].

Серед переваг YOLO можливо зазначити те що YOLO швидкий алгоритм також алгоритм YOLO простий в використанні тобто архітектура YOLO відносно проста для того щоб її зрозуміти та реалізувати, також перевагою YOLO є те що YOLO має цілісне розуміння даних тобто YOLO розглядає вхідні дані як щось ціле що в свою чергу дозволяє алгоритму YOLO враховувати контакт та виявити об'єкти, які прикриваються. Проте в алгоритму YOLO є і недоліки це менша точність для малих об'єктів та чутливість до аспектрального співвідношення.

1.3 Аналіз вимог та постановка задачі дослідження

Об'єкт дослідження – система виявлення та класифікації транспортних засобів, що працює на основі алгоритмів глибокого навчання, зокрема згорткових нейронних мереж. У межах дослідження транспортні засоби включають автомобілі, автобуси, вантажівки, мотоцикли та інші категорії об'єктів, що підлягають детекції на основі отриманих зображень. Основний акцент робиться на ефективності системи в умовах різного типу дорожнього руху, включаючи інтенсивний трафік, різні ракурси зйомки та складні дорожні умови.

Дослідження базується на використанні модифікованого алгоритму YOLOv8, що був доопрацьований для підвищення точності та ефективності виявлення об'єктів. Ця модель поєднує класичні підходи YOLO зі вдосконаленнями архітектури, зокрема впровадженням модулів підвищеної уваги для покращення фокусування на важливих ознаках об’єктів.

Зважаючи на те, що якість результатів роботи алгоритмів глибокого навчання сильно залежить від умов освітлення та характеристик вхідних зображень, дослідження передбачає аналіз продуктивності моделі у різних умовах, включно з денним освітленням, низькою видимістю, погодними факторами та перекриттям об’єктів. Окремий акцент зроблено на сценаріях, що відповідають реальним умовам зйомки із вуличних камер [23].

Основні вимоги до системи включають кілька критичних аспектів. Точність виявлення є ключовою вимогою, адже система повинна надійно розпізнавати транспортні засоби різних типів, мінімізуючи кількість помилок, таких як хибні позитивні або негативні спрацювання. Додатково, система повинна демонструвати стійкість до зовнішніх факторів, таких як зміна умов освітлення, часткове перекриття об'єктів, різні ракурси зйомки та зміни погодних умов.

Основним завданням дослідження є розробка вдосконаленої моделі YOLOv8, яка поєднує високу швидкість обробки із високою точністю виявлення та класифікації транспортних засобів. Для досягнення поставленої мети модель було треновано на кількох датасетах, таких як "Vehicle Detection: 8 Classes" із платформи Kaggle, що містить анотовані зображення різних типів транспортних засобів та умов зйомки. Використання різноманітних наборів даних дозволяє моделі навчитися адаптуватися до реальних сценаріїв, включно зі складними умовами освітлення та високою щільністю об’єктів [23, 24].

Для перевірки узагальнюючої здатності моделі проводилось тестування на незалежних наборах даних, що дозволило оцінити її точність, стійкість та продуктивність у різних умовах експлуатації. Експериментальні дослідження включають оптимізацію метрик, для детальної оцінки ефективності системи.

Таким чином, метою даного дослідження є створення високоточного та енергоефективного алгоритму для виявлення та класифікації транспортних засобів на основі глибоких нейронних мереж. Запропонована система поєднує швидкість, адаптивність та точність, що робить її ефективною для моніторингу дорожнього руху в реальних умовах експлуатації.

Висновки до розділу 1

1. Проведено аналіз предметної області, сучасних методів та підходів до вирішення задач виявлення і класифікації транспортних засобів на основі комп'ютерного зору та алгоритмів глибокого навчання.

2. Проведено аналіз методів виявлення та класифікації транспортних засобів. Розглянуто основні принципи роботи згорткових нейронних мереж.

3. Визначено ключові вимоги до системи виявлення транспортних засобів: точність детекції, здатність до роботи в реальному часі та стійкість до зовнішніх факторів, таких як змінне освітлення, погодні умови та часткове перекриття об'єктів.

4. Сформульовано задачу дослідження, яка полягає у створенні високопродуктивного методу виявлення і класифікації транспортних засобів.

2 МЕТОД ВИЯВЛЕННЯ І КЛАСИФІКАЦІЇ ТРАНСПОРТНИХ ЗАСОБІВ

2.1 Опис інформації та підходів до виявлення та класифікації

Для роботи методу виявлення і класифікації транспортних засобів необхідно підготувати якісний набір даних, що забезпечить можливість ефективного навчання та тестування алгоритмів. Ці дані мають відображати різні умови дорожнього руху, враховувати різноманітність типів транспортних засобів, варіації освітлення та складні сценарії руху.

Основним джерелом інформації є відеозаписи дорожнього руху, які можна отримати з камер спостереження, встановлених над дорогами. Камери зазвичай розміщують під різними кутами, щоб забезпечити як вигляд зверху («пташиний зір»), так і горизонтальну перспективу, характерну для типових установок на перехрестях. Такі відеоматеріали охоплюють широкий спектр сценаріїв: від прямих автомагістралей до складних перехресть із багатосмуговим рухом.

Важливим аспектом є збір даних із різними умовами освітлення. Для цього використовують відео, зняті вдень, у сутінках і вночі. Для нічних умов особливу увагу слід приділити таким джерелам світла, як фари транспортних засобів, які можуть слугувати маркерами для алгоритмів обробки зображень. Додатково до цього потрібно врахувати відео, зняті за складних погодних умов, таких як дощ, туман або сніг.

Окрім відеозаписів з конкретних камер, можна використовувати відкриті набори даних, наприклад COCO (Common Objects in Context) чи KITTI Vision Benchmark Suite. Ці набори вже містять розмітку об’єктів, зокрема транспортних засобів, з чіткими координатами граничних рамок і класами об’єктів (автомобіль, вантажівка, автобус, мотоцикл тощо). Такі дані значно спрощують процес попереднього навчання моделей і дозволяють забезпечити високу якість класифікації [25].

Підготовлені дані повинні включати зображення та відео у високій роздільній здатності, щоб забезпечити деталізацію об’єктів. Особливу увагу слід приділити розмітці даних: для кожного транспортного засобу мають бути визначені його клас

і координати в кадрі. Це дозволить використовувати ці дані для тренування моделей глибокого навчання, таких як YOLO, і забезпечити високу точність розпізнавання.

Існує два основних підходи до виявлення і класифікації об’єктів. Перший — це підхід, заснований на нейронних мережах (NN), який використовує методи зворотного поширення помилок (backpropagation) у техніці "end-to-end". Найбільш поширеним прикладом є згорткові нейронні мережі (CNN). Другий підхід — це методи, що не використовують нейронні мережі, які базуються виключно на обробці зображень або застосовують прості процеси машинного навчання, наприклад, опорні вектори (SVM).

Розглядаючи історичний розвиток систем виявлення і класифікації транспортних засобів, перші успіхи були досягнуті у сфері підрахунку транспортних засобів. Прості системи, засновані на комп’ютерному зорі, використовували відокремлення фону від переднього плану, на якому знаходилися транспортні засоби. Найуспішніші підходи залежали від якості зйомки, яка дозволяла отримати достатню кількість пікселів для зображення транспортних засобів. Багато методів базувалися на оптичному потоці та використанні віртуальної лінії перетину для підрахунку кількості транспортних засобів. Однак великі транспортні засоби, такі як вантажівки з причепами, часто створювали викривлення в підрахунках, оскільки багато підходів не могли забезпечити точність.

Ще однією значною проблемою було те, що майже всі методи демонстрували низьку точність підрахунку за умов поганого освітлення, таких як сутінки, похмура погода або ніч. У цьому контексті було небагато успішних підходів, однак загальновизнано, що точність підрахунку та класифікації суттєво погіршується за умов низької освітленості, особливо вночі. Перші підходи також страждали від нездатності обробляти дані в режимі реального часу. Це було серйозною проблемою, оскільки відсутність обробки в реальному часі позбавляла такі системи практичного застосування для моніторингу доріг.

До 2010 року жодне дослідження із середніми обчислювальними можливостями не могло забезпечити класифікацію транспортних засобів у реальному часі. Проте з появою нейронних мереж сьогодні досягнуто високої точності класифікації транспортних засобів і їхнього підрахунку в реальному часі навіть при середніх обчислювальних ресурсах. Ключовою вимогою для таких результатів є якісний вид транспорту спереду з достатньою кількістю пікселів, що легко досягається при встановленні камер над дорогою. Однак ця можливість швидко погіршується за умов недостатнього освітлення, таких як сутінки, дощ чи похмура погода. Для моніторингу трафіку вночі нейронні мережі не використовуються, але інші методи, засновані на виявленні фар, успішно підраховують транспортні засоби. Точна класифікація вночі може бути забезпечена, якщо ділянка дороги буде добре освітлена. Такі умови легко створити, наприклад, у тунелях, де камери можуть бути розміщені над смугами руху для виявлення і класифікації транспортних засобів та за умов хорошої якості зйомки, системи здатні зчитувати номерні знаки транспортних засобів на в’їзді та виїзді з добре освітлених ділянок дороги.

2.2 Метод виявлення і класифікації об’єктів

Виявлення об’єктів за допомогою обробки вітео та зображень може бути виконане кількома способами. Основна стратегія полягає у виділенні об’єктів за допомогою масок, визначенні їх контурів і побудові граничних рамок. Цей підхід базується на експериментах з оптимізацією масок і врахуванням умов навколишнього середовища, у якому знаходяться об’єкти.

В даному методі виявлення транспортних засобів виділяються об’єкти що рухаються (камери залишалися стаціонарними). Об’єкти відповідного розміру маскуються у вигляді "областей" (blobs) із заданим центроїдом і граничною рамкою. Також можна здійснювати і підрахунок таких виявлених об’єктів шляхом реєстрації моменту, коли центроїд об’єкта перетинав визначену лінію. Для

підвищення обчислювальної ефективності визначалась область інтересу (ROI), що обмежувала ділянку зображення, яка аналізувалася. ROI була візуалізована у вигляді тонкого прямокутника (Рискнок 2.1).

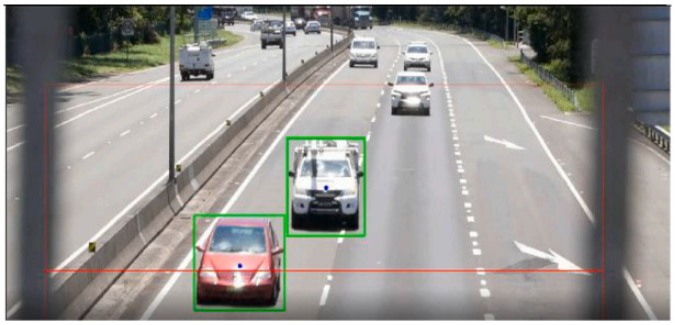


Рисунок 2.1 – Виділення області інтересу

Альтернативні методи, такі як k-ближчих сусідів, також можуть використовуватися для виділення об’єктів. Однак вони мають значно вищі обчислювальні витрати, що унеможливлює їх використання для виявлення транспортних засобів у реальному часі.

Оптимальним вибором для забезпечення балансу між швидкістю та точністю роботи системи є метод MOG2. Ключовим етапом у цьому підході є процес віднімання фону (background subtraction, BS), який дозволяє створювати маску переднього плану, ізолюючи об’єкти, що рухаються. Для цього застосовувався метод "Суміші Гауссіанів" (Mixture of Gaussians), який є одним із найшвидших і найточніших для задач цього типу. Маска переднього плану, отримана за допомогою цього методу, представлена на рисунку 2.2.

Рисунок 2.2 – Маска видалення фону.

На жаль, методи виявлення об’єктів, які не базуються на нейронних мережах, мають низку проблем, що стали очевидними під час дослідження. Оскільки процес виключно інтерпретує рух, визначаючи відмінності від фону та розмір об’єктів, проблеми виникають, коли транспортні засоби накладаються один на одного. У таких випадках програма розпізнає перекриті транспортні засоби як один об'єкт замість двох окремих. Це особливо помітно, коли більший транспортний засіб, наприклад вантажівка, закриває автомобіль, що знаходиться позаду [22].

Вирішенням цієї проблеми є використання підвищеного горизонтального ракурсу зйомки або повного виду зверху (bird’s eye view). Однак це значно обмежує практичність таких підходів, адже ідеальною умовою є незалежність від ракурсу камери. Додатково, система стикається з іншими обмеженнями: вона найкраще працює для двосмугових доріг із прямими ділянками. Ідеальна система мала б враховувати перехрестя, багатосмугові дороги та криві, залишаючись незалежною від кута зйомки. Проблему накладання транспортних засобів ілюструє рисунок 2.3.

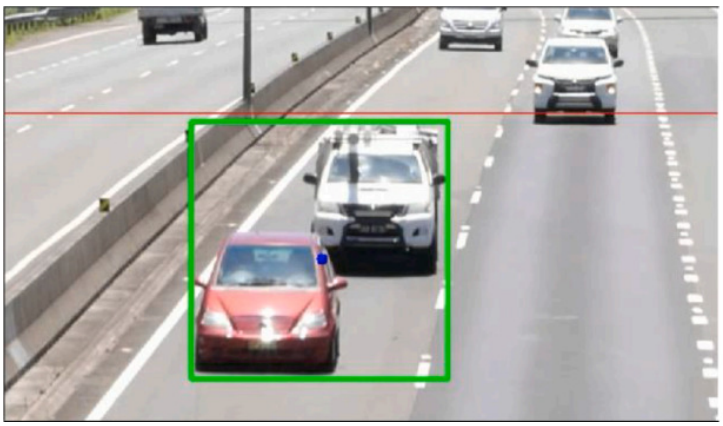


Рисунок 2.3 – Маска з накладанням.

Ще однією значною проблемою є умови низького освітлення. У таких ситуаціях виникає значний рівень шуму, через що процес віднімання фону стає неефективним і система не працює. Аналогічно, у випадку надмірної експозиції, коли камера не може коректно інтерпретувати передній план через відбите світло, виникають ті самі труднощі.

Остання проблема стосується зміни площі пікселів об’єкта у випадку наближення транспортного засобу до камери. При цьому розмір об’єкта збільшується, що суттєво змінює позицію його центроїда. Це призводить до неточних підрахунків, коли, наприклад, великі транспортні засоби, такі як вантажівки, займають дуже велику площу пікселів. Такий сценарій показано на рисунку 2.4.

Рисунок 2.4 – Приклад причини появи неточних результатів.

Якщо говорити про результати, цей базовий метод є досить точним, але із тривалістю роботи програми похибки накопичуються через проблему накладання та зміни розміру об’єктів.

2.3 Алгоритми комп’ютерного зору для аналізу зображень

Сьогодні для виявлення та класифікації об’єктів у реальному часі широко використовується алгоритм під назвою YOLO (You Only Look Once). До розробки YOLO у 2015 році Редмоном найбільш поширеним методом для виявлення об’єктів були R-CNN, які, проте, були занадто повільними для роботи в реальному часі [25]. Завдяки своїй швидкості та точності YOLO став успішним інструментом для таких завдань, як виявлення автомобілів, розпізнавання тварин та моніторинг порушень

безпеки, як показано на рисунку 2.5, де система виявлення об’єктів YOLO змінює розмір вхідного зображення, після чого пропускає його через одну згорткову нейронну мережу, а потім фільтрує результати виявлення за рівнем довіри моделі.

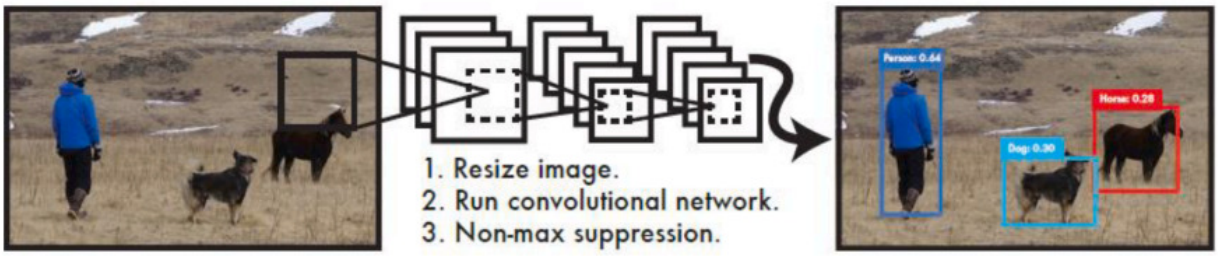


Рисунок 2.5 – Система виявлення об’єктів YOLO.

Основний принцип роботи YOLO полягає в тому, що алгоритм використовує вхідне зображення, розбиває його на декілька ковзаючих вікон різного розміру та пропускає їх через згорткову нейронну мережу, яка навчена виявляти певні об’єкти (наприклад, автомобілі, велосипеди, людей, фрукти тощо). Процес включає такі етапи:

Кожне ковзаюче вікно проходить згортку через CNN, яка обчислює ймовірності для об’єктів.

Якщо об’єкт знаходиться, визначаються його координати.

Усі ці дії виконуються в одному конвеєрі, без повторного перегляду зображення. Звідси і назва алгоритму: You Only Look Once.

Архітектура нейронної мережі YOLO, що демонструє зв’язки згорткових шарів від входу до виходу, представлена на рисунку 2.6, де мережа виявлення складається з 24 згорткових шарів, за якими йдуть 2 повнозв’язані шари. Почергові згорткові шари зменшують простір ознак, отриманих з попередніх шарів. Згорткові шари попередньо навчені на завданні класифікації ImageNet з половинною роздільною здатністю (вхідне зображення 224x224), після чого роздільна здатність подвоюється для виконання завдання виявлення [26].

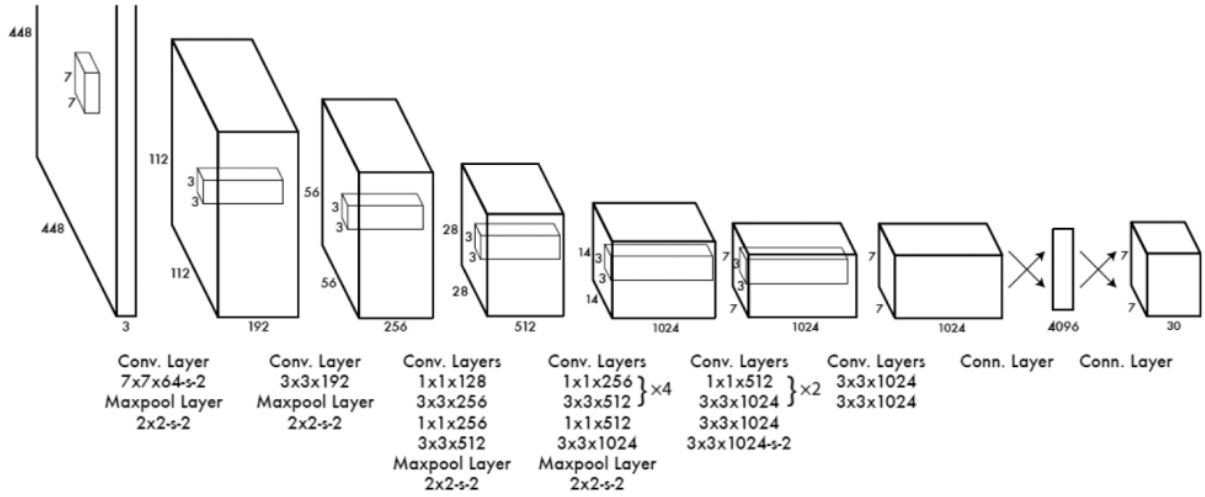


Рисунок 2.6 – Архітектура YOLO

Моделі YOLO постійно вдосконалювалися, стаючи швидшими та точнішими, що суттєво вплинуло на ефективність обробки зображень у реальному часі. З моменту створення першої версії у 2015 році YOLO отримала численні оновлення, кожне з яких впроваджувало ключові покращення для підвищення продуктивності.

YOLOv1 була першою версією, яка об'єднала завдання виявлення об'єктів у єдину нейронну мережу. Вона розділяла зображення на сітку та передбачала граничні рамки (bounding boxes) та класи об'єктів одночасно. Основною перевагою YOLOv1 була її швидкість, проте вона мала обмежену точність при виявленні дрібних об'єктів.

YOLOv2, також відома як YOLO9000, запропонувала кілька важливих нововведень. Основним стало використання мультимасштабних входів, що дозволило моделі обробляти зображення різного розміру. Початкова модель YOLO використовувала фіксовані вікна розміром 224x224 пікселів, тоді як YOLOv2 стала конкурентоспроможною з Faster R-CNN завдяки адаптації до масштабів об'єктів. Це значно покращило якість детекції об'єктів різного розміру та дозволило оптимізувати граничні рамки транспортних засобів за допомогою згорткової нейронної мережі.

YOLOv3 продовжила вдосконалення архітектури. У цій версії було впроваджено кластеризацію методом k-середніх для оптимізації розмірів «якорів»

(anchor boxes). Також злиття ознак (feature fusion) на різних рівнях мережі дозволило моделі краще працювати із дрібними об'єктами. Параметри згорткових і пулінг-шарів були оптимізовані для підвищення точності та зменшення розмірів моделі.

YOLOv4 стала інноваційною завдяки численним покращенням, зокрема впровадженню активації mish, агрегації ознак (PANet), а також оптимізаціям для тренування глибоких нейронних мереж. Ця версія досягла значного балансу між точністю та швидкістю, що дозволило ефективно використовувати її для задач реального часу.

YOLOv5 стала важливим кроком у розвитку, хоча вона не є офіційною версією від автора оригінального YOLO. У YOLOv5 було оптимізовано швидкість навчання та інтерфейс для застосування в різних умовах. Модель підтримувала можливість гнучкої адаптації до мобільних пристроїв та GPU завдяки меншій кількості параметрів.

YOLOv6 і YOLOv7 були орієнтовані на ефективність і продуктивність, пропонуючи ще кращі оптимізації для використання в умовах обмежених ресурсів. YOLOv7 стала однією з найшвидших і найточніших моделей на момент свого виходу, обганяючи конкурентів у багатьох завданнях детекції.

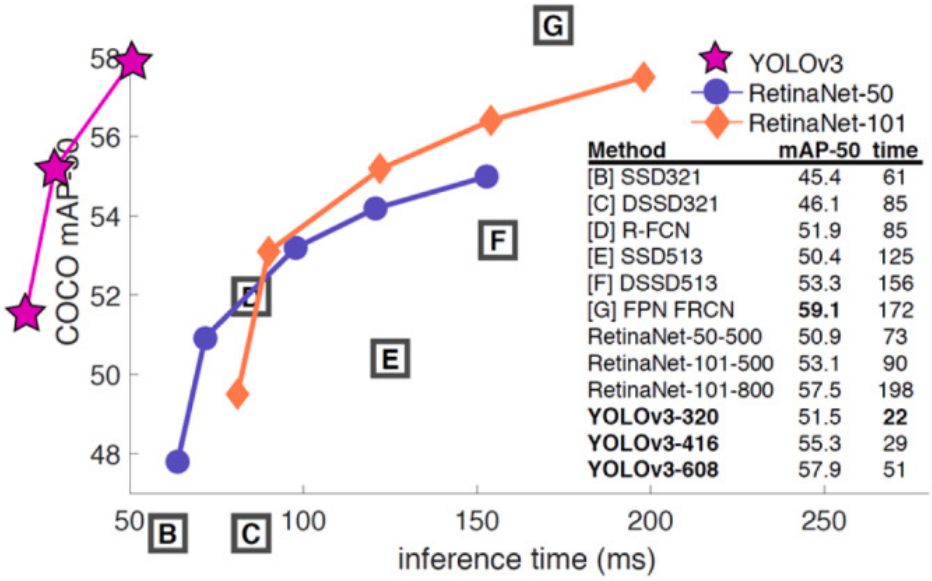
YOLOv8 представила значне оновлення у точності завдяки впровадженню нових архітектурних модулів, зокрема вдосконаленої уваги до ознак та адаптації до різних обчислювальних платформ. Ця версія забезпечила чудову точність детекції у задачах класифікації, сегментації та виявлення об'єктів у реальному часі.

YOLOv9 і YOLOv10 продовжили шлях оптимізації за рахунок нових механізмів уваги, що забезпечують ще більшу продуктивність на великих наборах даних, таких як MS COCO. Основний фокус був на покращенні здатності до обробки складних умов, включно з низьким освітленням і великою кількістю об'єктів.

На сьогоднішній день YOLOv11 є однією з найсучасніших версій, що демонструє видатні результати на наборі даних MS COCO для виявлення об'єктів. У порівнянні з попередніми версіями, YOLOv11 поєднує покращену деталізацію та

адаптивністю до різних умов. Модель забезпечує state-of-the-art показники продуктивності.

На рисунках 2.8 та 2.9 наведено порівняння попередніх та сучасних версій YOLO з іншими моделями виявлення об'єктів на основі нейронних мереж. Порівняльний аналіз показує, що YOLOv11 працює значно швидше, ніж більшість альтернативних методів детекції, одночасно забезпечуючи високу продуктивність у складних сценаріях [27].



Method	mAP-50	time
[B] SSD321	45.4	61
[C] DSSD321	46.1	85
[D] R-FCN	51.9	85
[E] SSD513	50.4	125
[F] DSSD513	53.3	156
[G] FPN FRCN	**59.1**	172
RetinaNet-50-500	50.9	73
RetinaNet-101-500	53.1	90
RetinaNet-101-800	57.5	198
YOLOv3-320	51.5	**22**
YOLOv3-416	55.3	29
YOLOv3-608	57.9	51

Рисунок 2.8 – Попередні версії YOLO

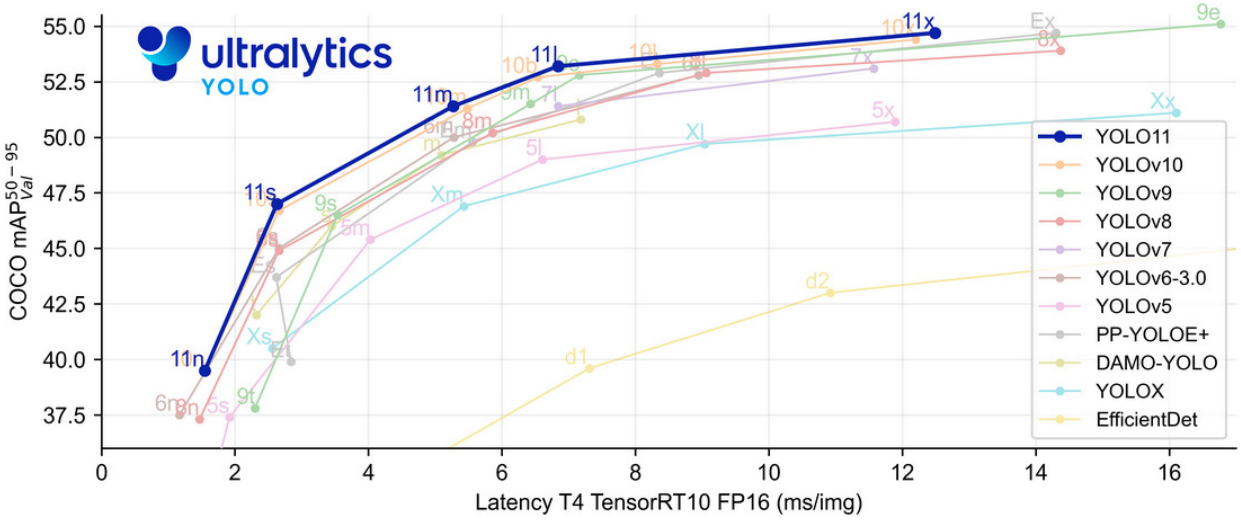


Рисунок 2.9 – Сучасні версії YOLO.

2.4 Аналіз процесу виявлення і класифікації

Перед застосуванням алгоритму необхідно провести навчання моделі. Для успішного навчання моделі YOLO на основі наявного датасету слід виконати кілька важливих кроків, починаючи від підготовки даних і закінчуючи налаштуванням параметрів тренування.

Спочатку необхідно підготувати середовище для роботи. Встановлюються всі необхідні програмні інструменти, включаючи Python та відповідні бібліотеки, як-от Ultralytics YOLO, PyTorch, OpenCV, NumPy та Matplotlib. Інструмент YOLO (наприклад, YOLOv8) завантажується за допомогою Ultralytics, що забезпечує зручний інтерфейс для роботи з моделями глибокого навчання [28-30].

Далі слід підготувати дані для навчання моделі. Датасет, який містить зображення та анотації (у форматі COCO або YOLO), завантажується із платформи, наприклад, Kaggle. Анотації повинні відповідати стандартному формату: для кожного зображення вказуються координати обмежувальних рамок (Xmin, Ymin, Xmax, Ymax) та клас об'єкта. Для кращого результату зображення проходять аугментацію (масштабування, обертання, зміна яскравості), що допомагає моделі краще узагальнювати дані на різних сценах.

Після підготовки даних створюється файл конфігурації навчання. У ньому визначаються основні параметри тренування, зокрема:

1. Датасет – шлях до навчальних, валідаційних і тестових даних.

2. Класи об’єктів – перелік класів, які модель має розпізнавати (наприклад, Car, Truck, Bus тощо).

3. Параметри навчання – кількість епох (наприклад, 10–100), розмір батчу, поріг впевненості, а також початкове значення швидкості навчання (learning rate).

4. Архітектура моделі – вибір моделі YOLO, наприклад, yolov8m.pt або yolov8l.pt.

Після налаштування конфігурації запускається процес навчання. Навчання моделі виконується за допомогою команди:

yolo train model=yolov8m.pt data=dataset.yaml epochs=50 batch=16 imgsz=640

Ця команда вказує моделі YOLO завантажити попередньо навчену архітектуру (наприклад, yolov8m.pt) та навчати її на заданому датасеті з певною кількістю епох. Під час навчання модель поступово оптимізує свої параметри, зменшуючи втрати (losses) та покращуючи показники точності детекції.

Важливим етапом є моніторинг процесу навчання. Під час кожної епохи обчислюються втрати, такі як box_loss (втрати на координатах рамок), cls_loss (втрати на класифікації об'єктів) і dfl_loss (втрати, пов'язані з позицією рамок). Також оцінюються метрики Precision, Recall та mAP (mean Average Precision) на валідаційних даних. Прогрес навчання візуалізується у вигляді графіків, що дозволяє відстежувати стабільність процесу та уникати перенавчання [31].

Після завершення навчання модель автоматично зберігається разом із результатами, включно з графіками втрат, матрицею помилок, кривими Precision-Recall та іншими метриками, які використовуються для оцінки продуктивності моделі.

Процес виявлення і класифікації транспортних засобів за допомогою YOLO можна поділити на такі основні етапи виявлення і класифікації.

На етапі виявлення YOLO використовується для ідентифікації транспортних засобів у кожному кадрі відеопотоку. Алгоритм визначає граничні рамки для кожного об’єкта і класифікує його за типом (наприклад, легковий автомобіль, вантажівка тощо).

На етапі класифікації для кожного виявленого транспортного засобу створюється унікальний ідентифікатор, що дозволяє відстежувати його переміщення між кадрами. Це особливо важливо для уникнення подвійного підрахунку.

Використання YOLO для виявлення і класифікації транспортних засобів показало значні переваги, зокрема високу точність завдяки попередньому навчанню, швидкість обробки кадрів у реальному часі та гнучкість у застосуванні

до інших задач класифікації. Ці фактори зробили YOLO одним із найпопулярніших інструментів для обробки зображень у реальному часі.

Для виявлення і класифікації транспортних засобів у реальному часі важливо не лише виявляти об’єкти, але й відстежувати їх між кадрами відеопотоку. Це дозволяє уникнути дублювання підрахунків і забезпечити точність класифікації. Однак, відстеження об’єктів може бути складним завданням, особливо у випадках перекриття транспортних засобів або при роботі з інтенсивним трафіком.

2.5 Результати дослідження методів класифікації

Для дослідження методів класифікації необхідно провести оцінку продуктивності моделей для виявлення та розпізнавання транспортних засобів на зображеннях. Аналіз включав кількісні та якісні показники ефективності алгоритмів, зокрема точність класифікації, повноту, середню точність (mAP) та здатність до розпізнавання об'єктів у різних умовах [32, 33].

Основним показником якості роботи моделей є Precision (точність) та Recall (повнота), які демонструють, наскільки коректно моделі визначають об'єкти та наскільки повно вони здатні їх виявити.

Точність визначає частку правильно передбачених об’єктів серед усіх, які модель класифікувала як позитивні:

$$\text{Precision} = \frac{TP}{TP + FP}$$

де:

TP — кількість правильно передбачених позитивних об’єктів (True Positives),

FP — кількість хибно передбачених позитивних об’єктів (False Positives).

Recall (Повнота) показує, яку частку всіх реальних позитивних об’єктів модель правильно виявила:

$$\text{Recall} = \frac{TP}{TP + FN}$$

де:

TP — кількість правильно передбачених позитивних об’єктів, *FN* — кількість об’єктів, які модель не виявила (False Negatives).

Метрика F1-Score є гармонійним середнім між Precision та Recall. Ця метрика допомагає знайти баланс між точністю та повнотою:

$$F1\text{-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Середня точність (Mean Average Precision, mAP) використовується для оцінки якості детекції об'єктів на різних порогах Intersection over Union (IoU):

$$\text{mAP} = \frac{1}{N} \sum_{i=1}^{N} AP_i$$

де:

AP*i* — Average Precision для кожного класу *i*, *N* — загальна кількість класів.

Значення mAP (mean Average Precision), обчислене для різних порогів IoU, слугує загальним показником точності виявлення для всіх класів об'єктів.

Перетин Intersection over Union (IoU) є метрикою, що вимірює ступінь перетину між передбаченою рамкою та реальною рамкою (ground truth).

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

де:

- Area of Overlap — площа спільного перетину передбаченої та справжньої рамки,

- Area of Union — площа об’єднання передбаченої та справжньої рамки. IoU зазвичай використовується для оцінки правильності детекції на заданому порозі (наприклад, IoU > 0.5).

Для навчання моделей класифікації та детекції використовуються функції втрат (Loss Functions):

$$\mathrm{L_{box}} = \mathrm{SmoothL1}(x_{\mathrm{pred}} - x_{\mathrm{true}})$$

де:

x_{pred} та x_{true} — координати передбачених та справжніх рамок.

Для визначення втрат на класах використовується Cross-Entropy Loss:

$$\mathrm{L_{cls}} = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)$$

де:

C — кількість класів,

y_i — справжнє значення,

$\hat{y}_i$ — прогнозована ймовірність належності до класу *i*.

Ці формули є ключовими для оцінки якості роботи методів класифікації та детекції об’єктів у дослідженні. Precision, Recall і F1-Score дозволяють оцінити продуктивність моделей на рівні класів, тоді як mAP і IoU допомагають оцінити загальну ефективність детекції об'єктів.

Дослідження показало, що точність класифікації значною мірою залежить від кількості об’єктів кожного класу у навчальному наборі даних. Класи, які мають більше прикладів для тренування, демонструють вищі показники Precision та Recall. Наприклад, об'єкти, які часто трапляються у кадрах, мають кращі показники детекції, тоді як рідкісні або дрібні об'єкти часто залишаються невиявленими або неправильно класифікованими.

Було виявлено, що моделі демонструють найкращі результати для класів із чітко окресленими ознаками, наприклад, легкові автомобілі. Натомість складнощі

виникають при розпізнаванні об’єктів, що перекривають один одного, або у випадках слабко виражених ознак, як-от автобуси чи мотоцикли. Крім того, в умовах низького освітлення та великої кількості шуму продуктивність моделей може знижуватися, що призводить до зменшення точності класифікації та збільшення хибних позитивних виявлень.

Аналіз Precision-Recall та F1-Confidence показав залежність якості класифікації від порога впевненості моделі. При високих значеннях порога зменшується кількість помилкових позитивних передбачень, однак при цьому модель може пропускати певні об’єкти, що знижує Recall. При низькому порозі модель демонструє високий рівень Recall, але точність зменшується через збільшення хибних спрацьовувань.

Також було відзначено, що розмір обмежувальних рамок (bounding boxes) та їх положення на зображенні впливають на якість класифікації. Дрібні об'єкти або об'єкти на краях кадру частіше класифікуються з помилками, оскільки їх ознаки можуть бути недостатньо вираженими для коректного розпізнавання.

Результати дослідження показали, що методи класифікації транспортних засобів ефективно справляються із завданнями виявлення та розпізнавання об'єктів у реальних умовах. Продуктивність моделей залежить від якості навчального набору даних, умов освітлення, розмірів об'єктів та складності сцени. Для подальшого покращення результатів необхідно враховувати баланс класів у датасеті, виконувати аугментацію даних та оптимізувати параметри навчання моделі.

Висновки до розділу 2

1. Здійснено опис інформації та підходів до виявлення та класифікації транспортних засобів.

2. Проведено детальний аналіз методів і алгоритмів для виявлення та класифікації транспортних засобів. Особлива увага приділена моделям серії YOLO та їхньому еволюційному розвитку.

3. Проведено порівняльний аналіз моделей YOLO з іншими сучасними підходами до детекції об'єктів, такими як Faster R-CNN та SSD.

4. Розглянуто основні метрики оцінки ефективності алгоритмів, зокрема що використовуються для кількісної оцінки якості детекції та класифікації об'єктів.

3 ПРОГРАМНО-ТЕХНОЛОГІЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1 Опис інструментів, програмних засобів та бібліотек

Реалізація методу була розділена на кілька ключових етапів, кожен з відповідним набором інструментів і погромних засобів та бібліотек, що дозволило створити інтегровану систему для автоматичного моніторингу дорожнього руху.

1. Підготовка та навчання моделі. На початковому етапі здійснюється навчання моделі YOLO на спеціально підготовленому наборі даних. Використовується датасет, який містить зображення транспортних засобів різних класів з відповідними анотаціями. Під час навчання модель оптимізує свої параметри, щоб досягти максимальної точності виявлення та класифікації об'єктів. Після завершення навчання модель зберігається для подальшого використання в системі.

2. Попередня обробка відеопотоку. На цьому етапі здійснюється зчитування відеопотоку за допомогою бібліотеки OpenCV. Кожен кадр перетворюється з формату BGR у RGB, що є необхідним для коректного функціонування моделі YOLO. Далі кадр масштабується до потрібного розміру (наприклад, 640×640 пікселів), що дозволяє моделі працювати оптимально, зберігаючи баланс між швидкістю обробки та точністю.

3. Виявлення транспортних засобів. Використовується попередньо навчена модель YOLO. Вхідні кадри передаються в модель, яка генерує результати у вигляді граничних рамок (bounding boxes) для кожного виявленого об’єкта, визначає класи (наприклад, "автомобіль", "вантажівка", "автобус") та рівні довіри (confidence scores). Це дозволяє системі фільтрувати об’єкти з низькою ймовірністю та зосереджуватися на найбільш релевантних.

4. Відстеження об’єктів. Для забезпечення безперервного відстеження транспортних засобів між кадрами використовується алгоритм трекінгу. Кожному виявленому об’єкту призначається унікальний ідентифікатор, що дозволяє відстежувати його рух у часі. Алгоритм враховує просторові координати рамок, що

повертаються моделлю YOLO, для прогнозування позицій об’єктів у наступних кадрах.

5. Класифікація транспортних засобів. Класифікація виконується на основі даних, згенерованих YOLO. Завдяки навчанню моделі на великому наборі даних, система здатна розпізнавати декілька типів об’єктів, таких як автомобілі, автобуси, вантажівки, мотоцикли тощо. Кожному класу об’єктів присвоюються відповідні кольори для візуалізації, що полегшує ідентифікацію об’єктів на екрані.

6. Візуалізація результатів. Після виявлення та класифікації всі результати візуалізуються безпосередньо на відеопотоці. Об'єкти обрамлюються граничними рамками, позначаються центроїди та мітки класів. Це дозволяє користувачу в реальному часі бачити, які транспортні засоби були розпізнані, їхнє місцезнаходження та класи. Така візуалізація є критичною для систем моніторингу та аналізу дорожнього руху.

Основою роботи є алгоритм YOLO, зокрема YOLO8 і YOLO11, що представлє собою передові рішення для задач комп’ютерного зору. YOLO8 демонструє високу швидкість обробки завдяки оптимізованій архітектурі, що робить його ідеальним для задач реального часу, таких як моніторинг трафіку або відеоспостереження. З іншого боку, YOLO11 зосереджений на досягненні максимальної точності та деталізації, що дозволяє виявляти навіть дрібні об’єкти та забезпечує стабільну роботу у складних умовах, наприклад, при поганій видимості чи щільному трафіку [34].

Задача виконувалась на комп'ютері з процесором AMD Ryzen 7 6850, 32 ГБ оперативної пам'яті та твердотільним диском об’ємом 1 ТБ під управлінням операційної системи Linux Ubuntu 24.04 LTS. Дана конфігурація забезпечила стабільну роботу моделей та дозволила виконати необхідні обчислення з високою продуктивністю.

Для програмної реалізації використовувалась мова програмування Python, яка забезпечує гнучкість і широкий набір інструментів для задач комп’ютерного зору. Основною бібліотекою для роботи з YOLO-моделями стала Ultralytics, що дозволяє легко завантажувати готові моделі, адаптувати їх до конкретних завдань і

отримувати результати детекції об’єктів на зображеннях. Для реалізації основних операцій глибокого навчання було використано фреймворк PyTorch, що є основою для навчання й тестування моделей YOLO8 та YOLO11. Обробка зображень і відеокадрів виконувалась за допомогою бібліотеки OpenCV, яка забезпечує можливість попередньої обробки даних, таких як масштабування та нормалізація, а також візуалізації результатів детекції [35, 36].

Дані, які використовувались для задачі, включають зображення, отримане із відеокадру огляду об’їзної дороги м. Тернопіль. Це зображення надає реалістичний приклад дорожньої ситуації, що містить різні типи транспортних засобів, зокрема легкові автомобілі та вантажівки. Завдяки цьому було створено умови для тестування моделей у середовищі, наближеному до реального. Для навчання моделей було використано датасет "Vehicle Detection: 8 Classes" з платформи Kaggle, що містить анотовані зображення транспортних засобів восьми різних класів. Цей набір даних дозволив ефективно навчити та перевірити моделі на задачах виявлення й класифікації.

Результати роботи моделей зберігались у текстових файлах, які містять таблицю з основними параметрами кожного виявленого об’єкта. До них належать координати рамок (Xmin, Ymin, Xmax, Ymax), клас об’єкта та ймовірність його виявлення. Для обробки отриманих даних використовувались бібліотеки Pandas та NumPy, що дозволили виконати аналіз результатів і підготувати дані для подальшої оцінки. Візуалізація отриманих результатів була реалізована за допомогою бібліотеки Matplotlib, яка забезпечує відображення зображень із нанесеними рамками та відповідними мітками об’єктів [37].

3.2 Реалізація навчання та валідації моделі

Для побудови та навчання моделі з інтегрованим CBAM (Convolutional Block Attention Module) потрібно використати нейромережу для задачі виявлення та

класифікації об'єктів на основі базової архітектури YOLO8. Навчання моделі включає модифікацію архітектури, підготовку даних та оптимізацію параметрів.

Першим етапом є мпортування необхідних бібліотек. Тут використовуються бібліотеки PyTorch, Ultralytics, OpenCV, NumPy та інші для роботи з глибоким навчанням, обробки зображень та аналізу результатів.

Для реалізації моделі з інтегрованим одулем створюється клас CBAM, що додає Channel Attention та Spatial Attention до нейромережі. CBAM допомагає виділяти найважливіші просторові області та канали, що покращує якість детекції.

```
class ModifiedYOLO8(nn.Module):
    def __init__(self, original_model):
        super(ModifiedYOLO8, self).__init__()
        self.backbone = original_model.model[0]  # Backbone YOLO8
        self.cbam = CBAM(channels=256)   # Додамо CBAM на 256
каналах
        self.head = original_model.model[1:]   # Залишок моделі
YOLO8

    def forward(self, x):
        x = self.backbone(x)  # Обробка ознак у Backbone
        x = self.cbam(x)  # Додавання CBAM
        for module in self.head:
            x = module(x)  # Подальша обробка
        return x
```

Додається CBAM у backbone моделі YOLO8, для покращення ефективності виявлення об'єктів, зокрема покращує обробку ознак на ранніх етапах мережі.

```
class ModifiedYOLO8(nn.Module):
    def __init__(self, original_model):
        super(ModifiedYOLO8, self).__init__()
        self.backbone = original_model.model[0]  # Backbone YOLO8
        self.cbam = CBAM(channels=256)   # Додамо CBAM на 256
каналах
        self.head = original_model.model[1:]   # Залишок моделі
YOLO8
```

```
    def forward(self, x):
        x = self.backbone(x)  # Обробка ознак у Backbone
        x = self.cbam(x)  # Додавання CBAM
        for module in self.head:
            x = module(x)  # Подальша обробка
        return x
```

CBAM додає два типи уваги: Channel Attention (увага до каналів) та Spatial Attention (просторова увага), що дозволяє моделі фокусуватися на найбільш релевантних особливостях у зображенні. CBAM працює у два етапи: Channel Attention - Модуль визначає, які канали (ознаки) є найбільш важливими для даного зображення. Для цього застосовуються операції глобального усереднення та максимізації на канальному рівні, що дозволяє виділити найбільш релевантні ознаки об'єктів. Spatial Attention - на наступному етапі CBAM фокусується на просторовому розташуванні об'єктів. Модуль генерує карту уваги, яка підкреслює області, де ймовірно знаходяться об'єкти, відсіюючи менш важливі частини зображення. Додавання CBAM до архітектури YOLO8 дозволяє моделі краще виділяти ключові області та важливі ознаки, що особливо корисно для виявлення дрібних об'єктів або об'єктів у складних умовах.

Проводиться підготовка даних для навчання, що включає завантаження датасету "Vehicle Detection: 8 Classes" з Kaggle або інші дані для задач детекції. Далі проводиться аугментацію даних (масштабування, обрізання, зміна яскравості) для покращення узагальнюючої здатності моделі.

```
from ultralytics import YOLO
from torchvision import transforms

train_transform = transforms.Compose([
    transforms.Resize((640, 640)),
    transforms.ColorJitter(brightness=0.2, contrast=0.2),
    transforms.RandomHorizontalFlip(),
    transforms.ToTensor(),
])
```

Для навчання завантажується модифікована модель, налаштовується оптимізатор, функцію втрат та запускається процес навчання.

```
# Завантаження YOLO8
base_model = YOLO("yolov8m.pt")
model = ModifiedYOLO8(base_model)

# Параметри навчання
optimizer = torch.optim.Adam(model.parameters(), lr=0.001)
criterion = nn.CrossEntropyLoss()  # Функція втрат

# Цикл навчання
for epoch in range(20):  # Кількість епох
    for images, targets in train_loader:     # Даталоадер тренувальних даних
        optimizer.zero_grad()
        outputs = model(images)  # Передбачення
        loss = criterion(outputs, targets)  # Розрахунок втрат
        loss.backward()
        optimizer.step()
    print(f"Epoch [{epoch+1}/20], Loss: {loss.item():.4f}")
```

Після навчання модель тестується на валідаційному наборі. Для оцінки якості детекції використовуються метрики Precision, Recall, F1-Score та mAP.

```
# Оцінка на тестовому наборі
test_results = model.predict(source="test_images/", save=True)
print("Testing completed. Results saved.")
```

Візуалізація результатів здійснюється шляхом детекції об'єктів на тестових зображеннях із нанесеними рамками, класами та ймовірностями.

```
for result in test_results:
    result.show()  # Відображення результатів
```

На рисунку 3.1. представлені графіки результатів навчання та валідації, що свідчать про стабільний процес тренування моделі. Втрати під час навчання (box_loss, cls_loss, dfl_loss) зменшуються з кожною епохою як на тренувальних, так і на валідаційних даних. Поступове зменшення box_loss показує покращення визначення координат обмежувальних рамок, тоді як cls_loss вказує на підвищення точності класифікації об’єктів. Метрики точності (Precision) та повноти (Recall) демонструють позитивну динаміку, досягаючи понад 0.7 на останніх епохах. Підвищення значень mAP50 та mAP50-95 підтверджує покращення середньої точності моделі на різних порогах IoU, що є важливим показником для задачі детекції об’єктів.

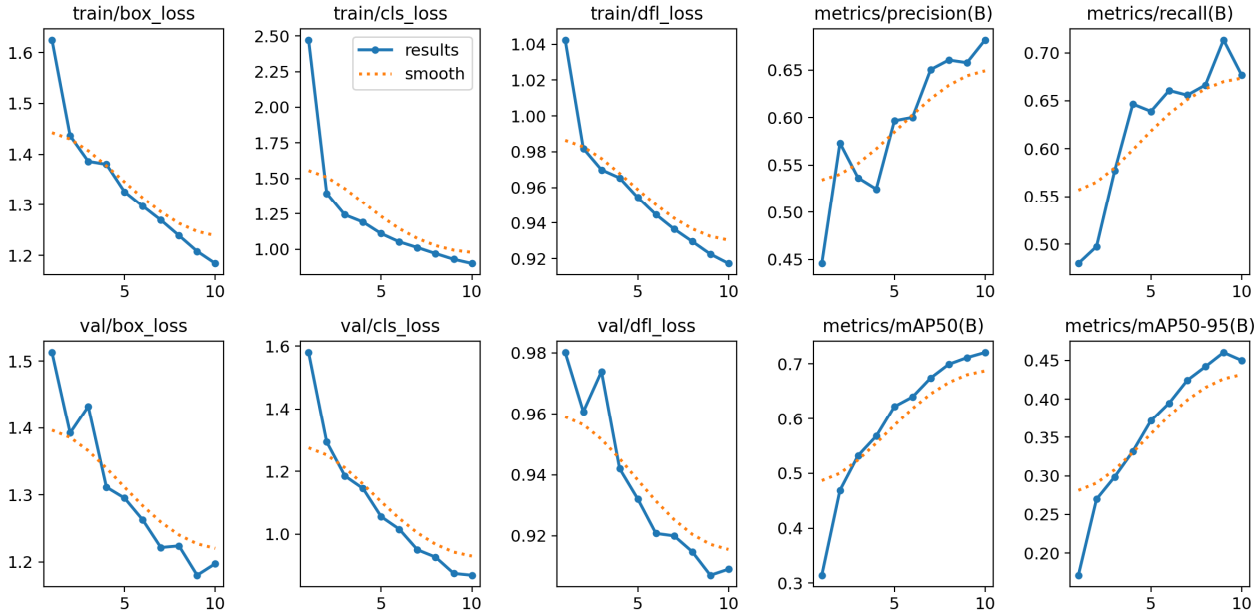


Рисунок 3.1 – Графіки результатів навчання та валідації.

На рисунку 3.2 представлена матриця помилок. Вона демонструє розподіл правильних і помилкових передбачень для кожного класу. Найбільша кількість правильних передбачень належить класу Car (1570 випадків), що вказує на високу точність розпізнавання цього класу. Такі класи, як LCV (Light Motor Vehicle) та Multi-Axle, також показують хороші результати, але інколи плутаються з іншими типами об'єктів. Клас Bus має найнижчу кількість правильних передбачень, що свідчить про складність розпізнавання цього об'єкта.

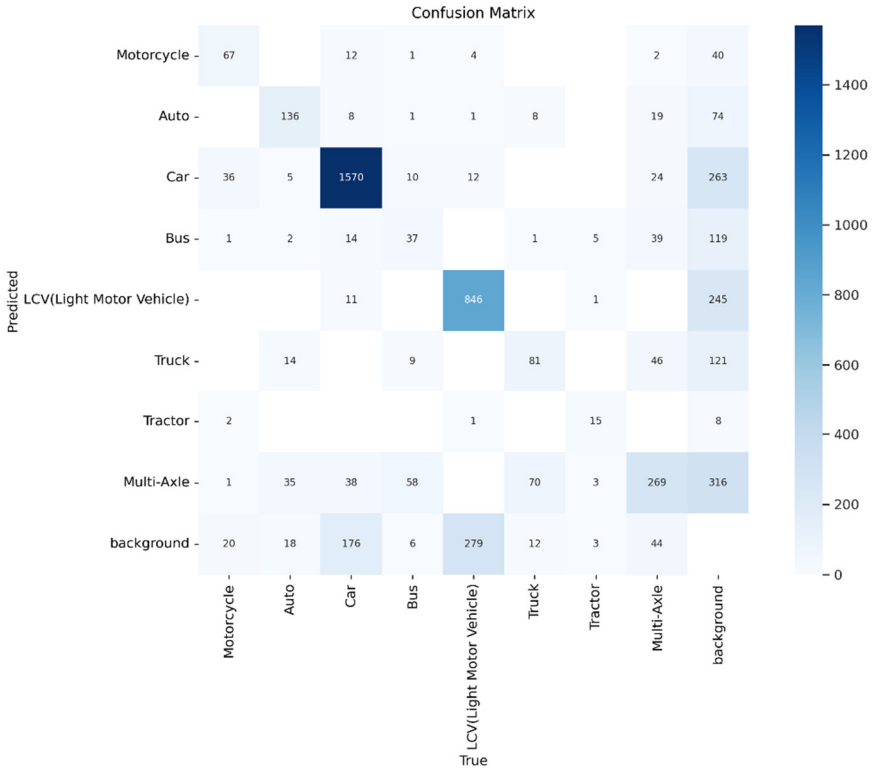


Рисунок 3.2 – Матриця помилок.

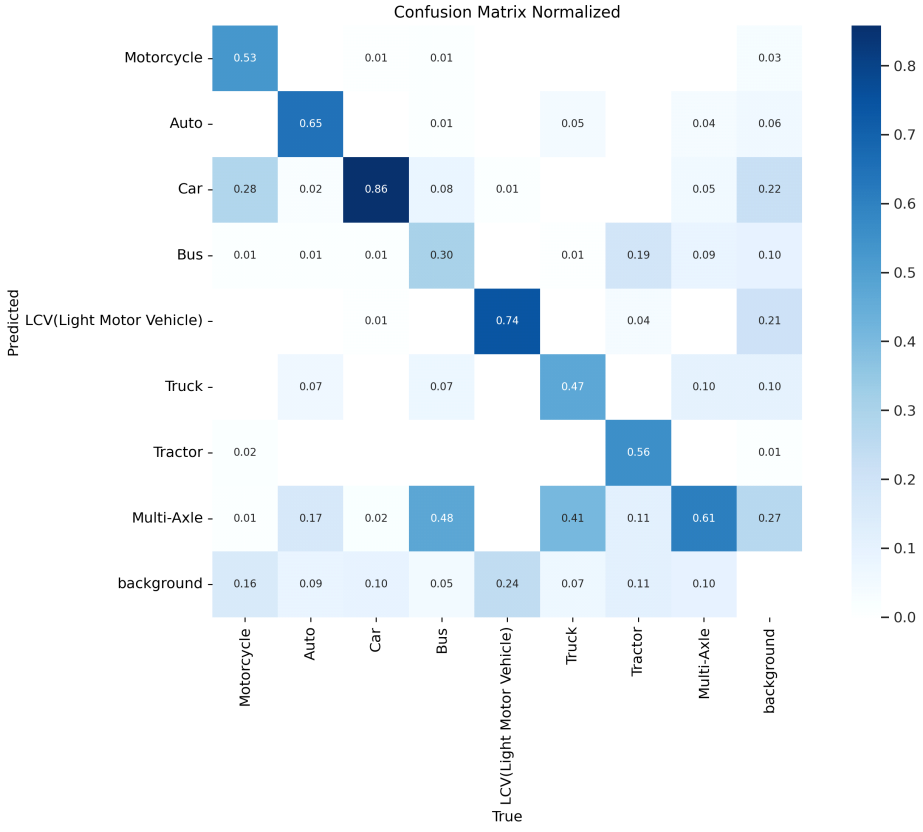


Рисунок 3.3 – Нормалізована матриця помилок.

На рисунку 3.3 представлена нормалізована матриця помилок. Вона спостереження, де клас Car досягає 86% точності, тоді як Bus демонструє лише 30%. Клас Motorcycle має 53% точності, проте часто помилково визначається як Auto.

На рисунку 3.4 представлений графік кривої F1-Confidence, що показує баланс між Precision та Recall при різних порогах впевненості. Максимальне значення F1 для всіх класів становить 0.68 при порозі 0.289. Найкраща продуктивність досягається для класів Car, Auto та LCV, тоді як класи Bus та Motorcycle мають нижчі результати, що свідчить про їх складність для моделі.

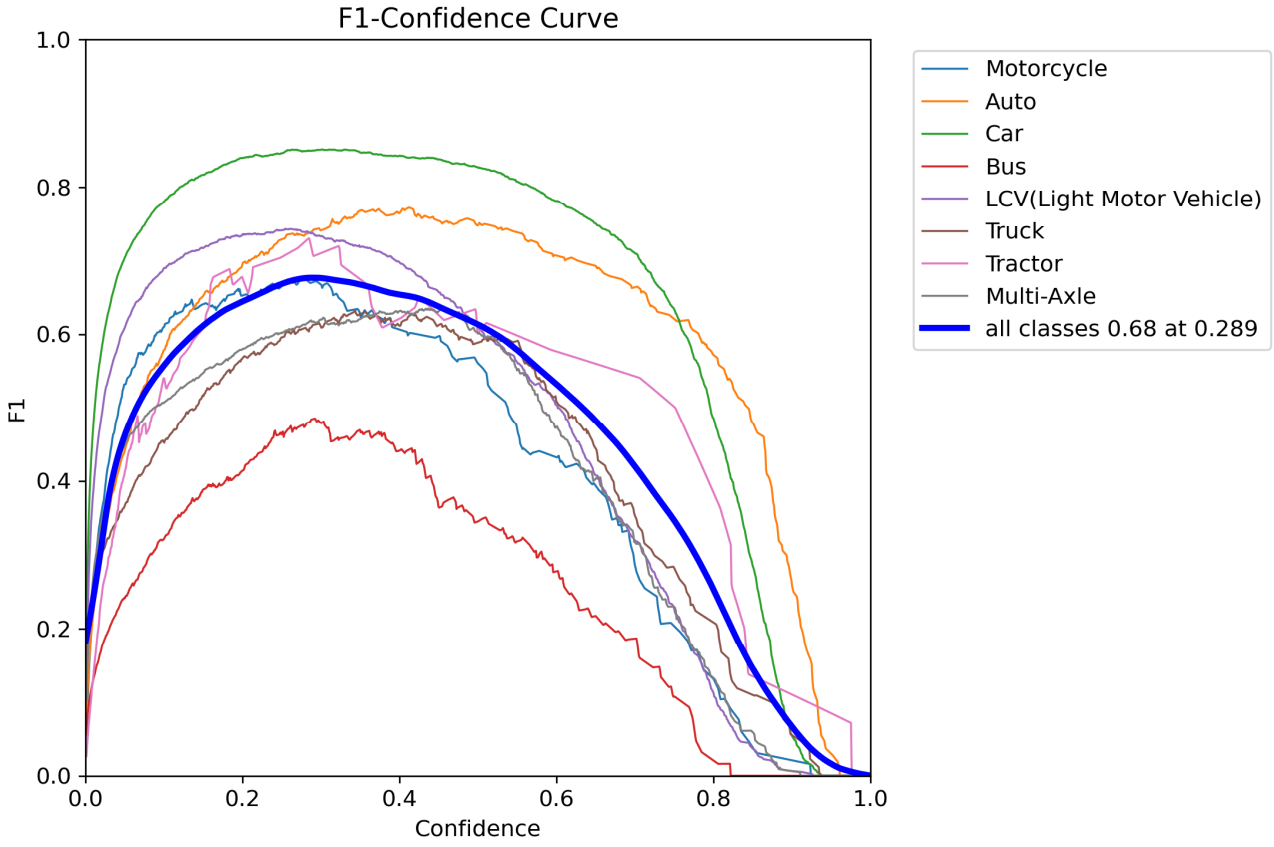


Рисунок 3.4 – Баланс між Precision та Recall при різних порогах впевненості.

Рисунок 3.5 та рисунок 3.6 показують Графіки розподіл уі кореляції позицій, ширини та висоти обмежувальних рамок. Аналіз міток (розподілу об'єктів) показує, що найбільша кількість об'єктів належить до класу Car, що пояснює високу продуктивність моделі для цього класу. Спостерігається концентрація рамок у

центральній частині зображень, що відповідає типовому розташуванню об’єктів у дорожніх сценах. Більшість обмежувальних рамок мають невеликі значення ширини та висоти, що вказує на виклики, пов'язані з виявленням дрібних об'єктів.

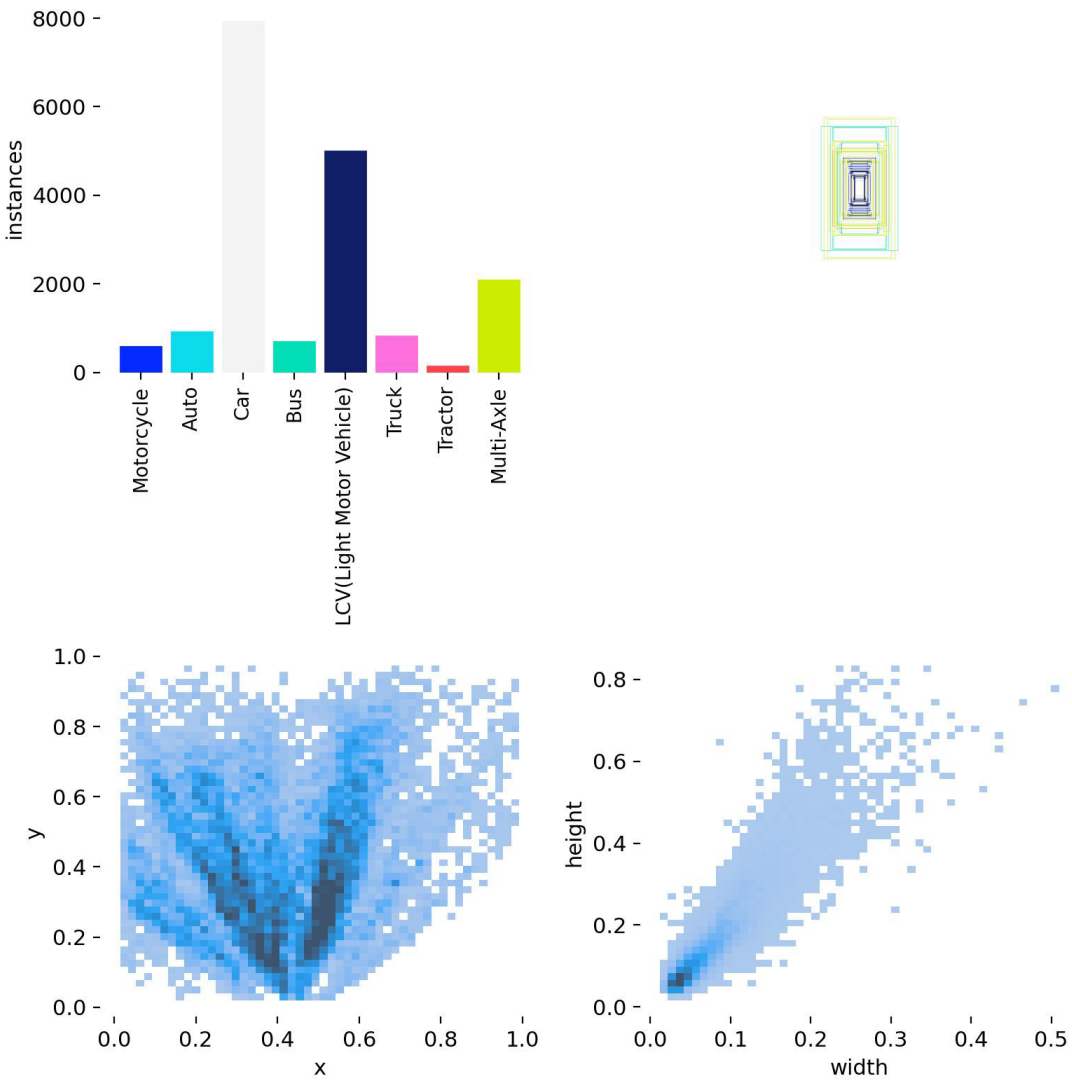


Рисунок 3.5 – Розподіл позицій, ширини та висоти обмежувальних рамок.

Крива Precision-Confidence (рисунок 3.7) показує, як точність змінюється при різних значеннях порогу впевненості. Модель досягає Precision = 1.0 для високих порогів впевненості, що свідчить про надійність при строгих критеріях. Найвищі значення Precision демонструють класи Car та LCV, тоді як Bus має найнижчу продуктивність.

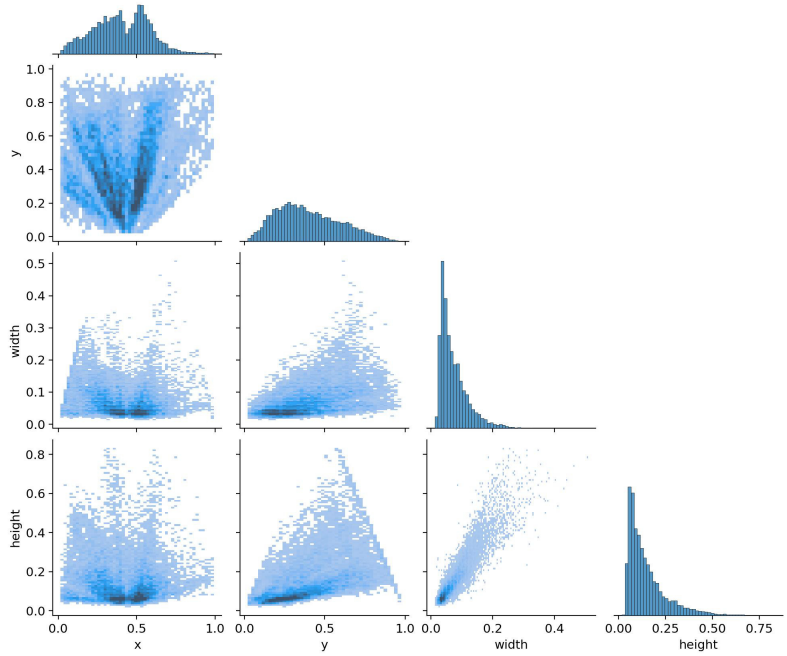


Рисунок 3.6 – Кореляцію позицій, ширини та висоти обмежувальних рамок

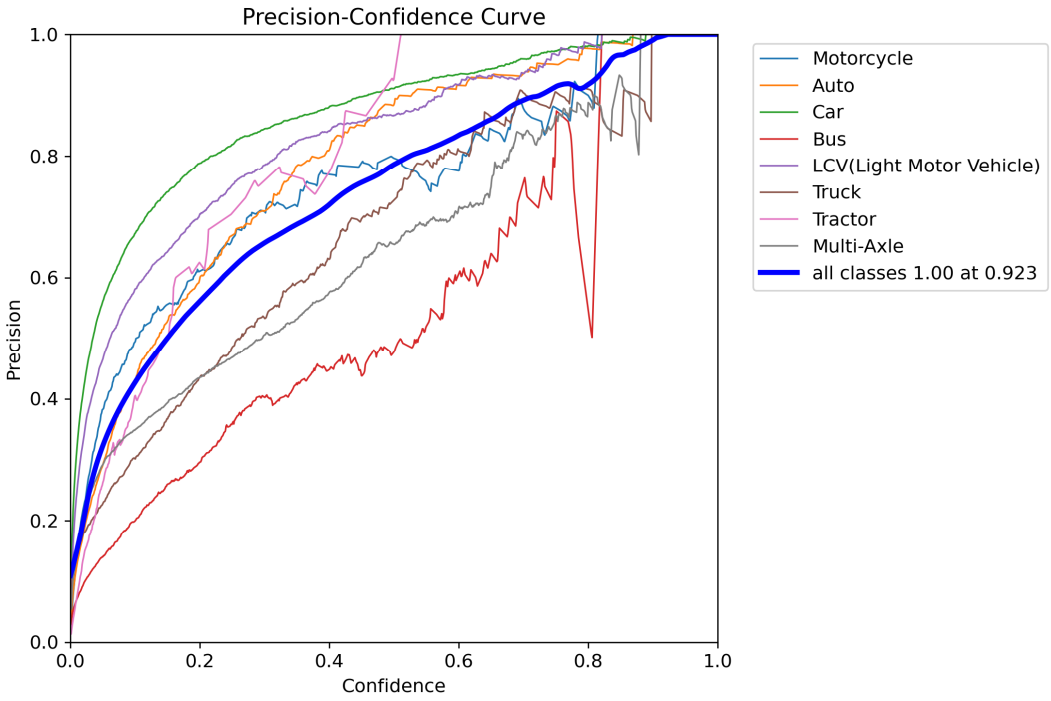


Рисунок 3.7 – Крива що показує як Precision змінюється залежно від порогу впевненості моделі.

Графік Precision-Recall (рисунок 3.8) демонструє баланс між цими двома метриками, де клас Car досягає найвищих значень (0.902), тоді як Bus залишається на рівні 0.421. Загальне значення складає 0.711, що є хорошим результатом для детекції об’єктів.

На кривій Recall-Confidence (рисунок 3.9) показано, як змінюється Recall при різних порогах впевненості. Максимальне значення Recall для всіх класів складає 0.97, що свідчить про здатність моделі виявляти майже всі об'єкти при низьких порогах впевненості. Найкращі результати досягаються для класів Car та Auto, тоді як Bus демонструє нижчий рівень Recall.

Загалом модель показала високу продуктивність для класів **Car**, **Auto** та **LCV**, але потребує подальшого доопрацювання для класів **Bus** та **Motorcycle**, які є складнішими для розпізнавання. Загальні метрики 0.711 і максимальний **F1** = 0.68 підтверджують ефективність моделі для задач виявлення транспортних засобів, проте для покращення результатів необхідно збалансувати датасет та додатково налаштувати модель.

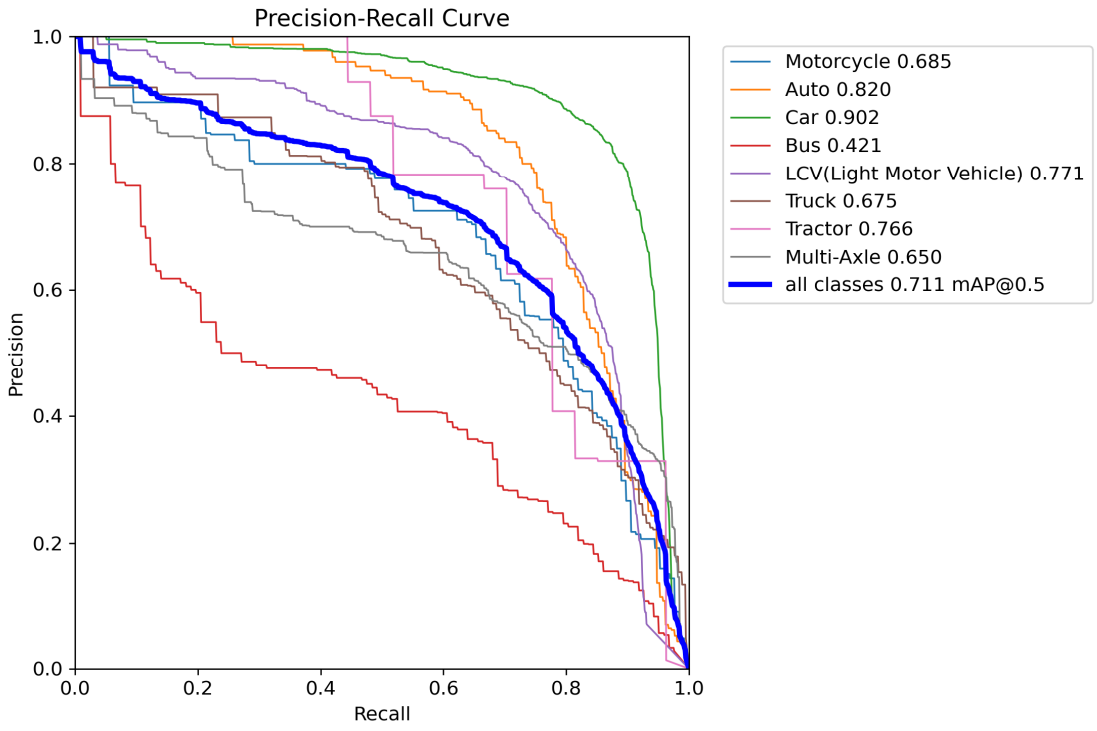


Рисунок 3.8 – Крива Precision-Recall показує, як модель балансує між Precision та Recall

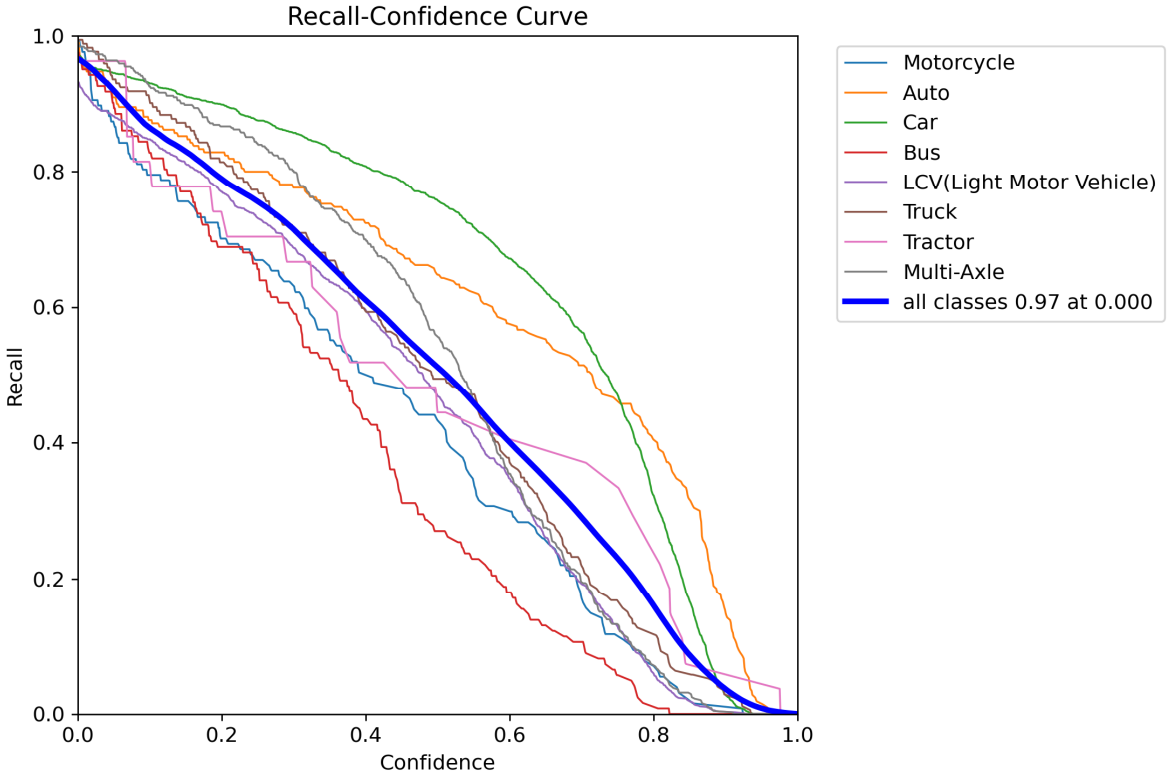


Рисунок 3.9 – Крива як Recall змінюється залежно від порогу впевненості

3.3 Результати експериментальних досліджень

Для аналізу було використано зображення, отримане із відеоматеріалу, опублікованого на платформі YouTube, що демонструє об’їзну дорогу у місті Тернопіль (Рисунок 3.10). Обране зображення є статичним кадром, який забезпечує репрезентативність реальних умов дорожнього середовища. Кадр містить характерні елементи типової транспортної ситуації, що дозволяють оцінити ефективність алгоритмів виявлення та класифікації об'єктів.

На зображенні представлені різні типи транспортних засобів, зокрема легкові автомобілі та вантажівки, що дозволяє провести перевірку здатності алгоритмів класифікувати об'єкти за класами. Щільність дорожнього руху, представлена у вигляді частково перекритих транспортних засобів, створює умови для оцінки алгоритмів виявлення об’єктів у складних сценаріях. Така ситуація є типовою

проблемою для задач детекції, оскільки перекриття може ускладнювати визначення меж об’єктів.

Рисунок 3.10 - Відеокадр з огляду об’їзної дороги, м.Тернопіль

Додаткову складність вносить наявність реальних умов освітлення та навколишнього середовища, характерних для місця зйомки. Природне освітлення, тіні, а також елементи інфраструктури на фоні зображення забезпечують випробування алгоритмів на стійкість до зовнішніх факторів, що є важливим аспектом для оцінки їх продуктивності у реальних умовах експлуатації. Таким чином, обраний кадр дозволяє всебічно перевірити можливості алгоритмів виявлення та класифікації транспортних засобів на прикладі реального дорожнього середовища, враховуючи складні умови, такі як щільний трафік, часткове перекриття об'єктів та природні умови освітлення.

На рисунку 3.11 представлено результати виявлення і класифікації об’єктів за допомогою за допомогою алгоритму YOLO8. На рисунку 3.12 представлено результати виявлення і класифікації об’єктів за допомогою модифікованого алгоритму YOLO8

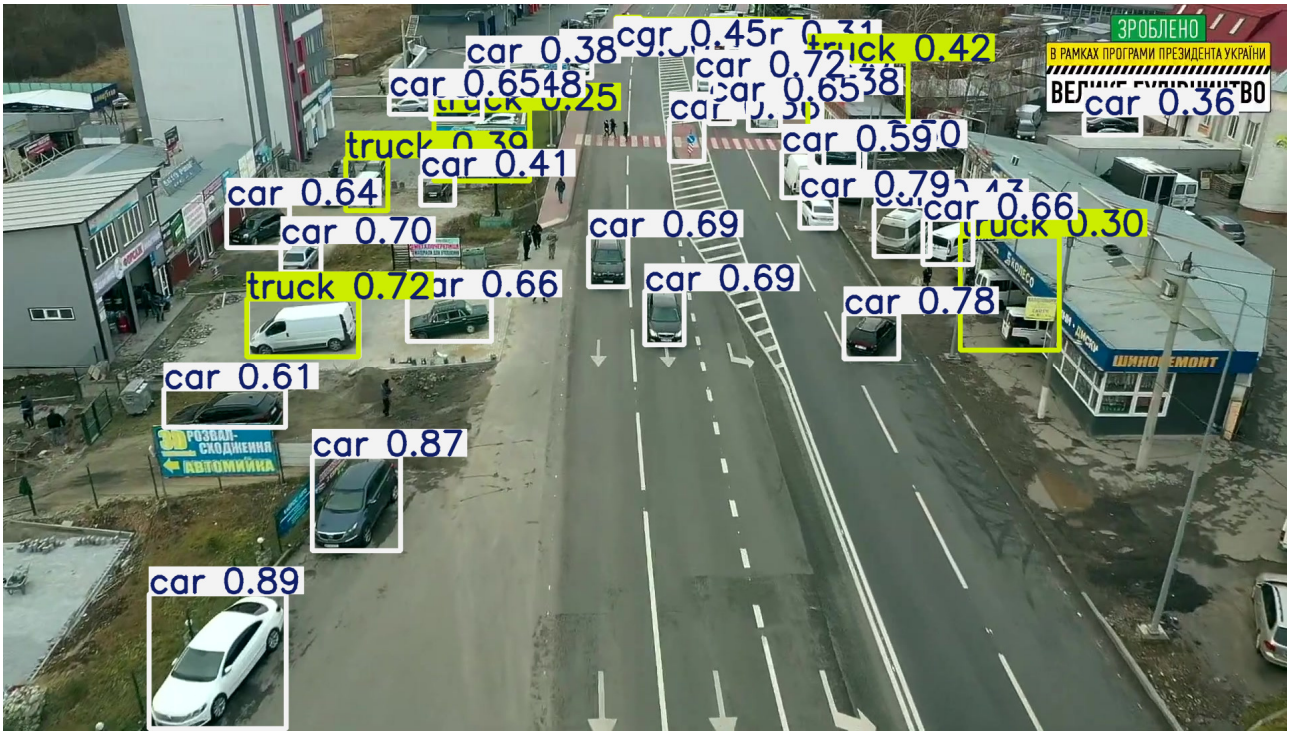


Рисунок 3.11 – Виявлення і класифікація об’єктів за допомогою алгоритму YOLO8

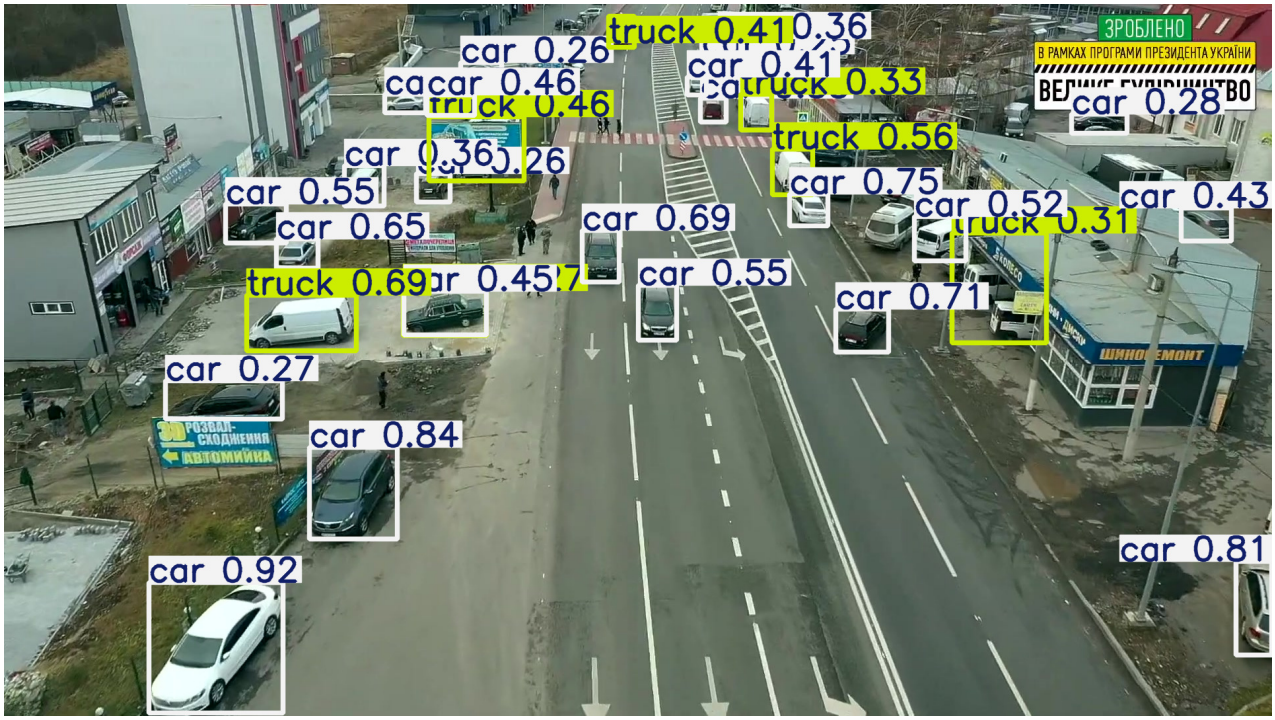


Рисунок 3.12 – Виявлення і класифікація об’єктів за допомогою модифікованого алгоритму YOLO8

Для навчання та оцінки продуктивності моделей YOLO8 та модифікованої YOLO8 використовувався датасет "Vehicle Detection: 8 Classes", доступний на платформі Kaggle. Датасет містить анотовані зображення транспортних засобів, розділені на 8 класів, що дозволяє ефективно тренувати й тестувати моделі для задач виявлення та класифікації об'єктів на зображеннях.

Інформація про виділені об’єкти зберігається у файли з наступною табличною структурою (Додаток Г):

```
Index,Class,Probability,Xmin,Ymin,Xmax,Ymax
1,2,0.89,218.98,876.36,423.64,1076.58
2,2,0.87,462.36,672.52,594.21,812.83
3,2,0.79,1191.21,286.72,1245.54,332.98
4,2,0.78,1257.80,459.34,1339.51,527.00
5,2,0.72,1034.20,108.74,1065.05,137.52
6,7,0.72,364.13,437.99,532.00,523.90
7,2,0.70,415.04,355.58,475.24,397.06
8,2,0.69,875.66,345.43,934.15,419.07
                          * * *
14,2,0.64,334.77,298.20,428.51,361.17
15,2,0.61,240.40,572.26,422.22,629.41
16,2,0.59,1164.34,218.76,1227.63,287.96
17,2,0.52,365.29,437.99,532.36,524.00
```

Кожен такий файл починається з заголовка, де описані ключові параметри для кожного виявленого об'єкта. У заголовку визначено шість колонок: Index — номер виділеного об'єкта, який починається з 1 і зростає для кожного нового запису. Class — ідентифікатор класу об'єкта. Клас 2 відповідає легковому автомобілю (car), а клас 7 — вантажному автомобілю (truck). Probability — це ймовірність, з якою алгоритм ідентифікував об'єкт, представлена у вигляді числа від 0 до 1, де 1 означає максимальну впевненість. Xmin та Ymin — координати верхнього лівого кута прямокутника, що окреслює об'єкт. Xmax та Ymax — координати нижнього правого кута цього прямокутника.

Для аналізу порівнювались результати виявлення та класифікації об'єктів, отримані за допомогою базового YOLO8 та модифікованого YOLO8.

Обидва алгоритми продемонстрували відмінності у кількості виявлених об'єктів, класифікації, ймовірностях та координатах граничних рамок. Базовий YOLO8 виявив 35 об'єктів, тоді як модифікований YOLO8 знайшов 31 об'єкт. Ця різниця свідчить про більшу чутливість базового YOLO8 до деталей на зображенні, що може призводити до хибних спрацювань. Натомість модифікований YOLO8 показав меншу кількість детекцій, але з вищою впевненістю для деяких об'єктів.

За класифікацією, більшість об'єктів в обох алгоритмах належать до класу 2 (легкові автомобілі). Базовий YOLO8 зафіксував 29 об'єктів класу 2 та 6 об'єктів класу 7 (вантажні автомобілі). Модифікований YOLO8 виявив 23 об'єкти класу 2 та 8 об'єктів класу 7. Перевага модифікованого YOLO8 у кількості вантажних автомобілів свідчить про його покращену здатність до класифікації складніших об'єктів.

Щодо ймовірностей, найвищий показник для базового YOLO8 становив 0.89, тоді як у модифікованого алгоритму максимальна впевненість досягла 0.92. Середнє значення ймовірностей для YOLO8 склало 0.525, а для модифікованого YOLO8 – 0.52. Незважаючи на трохи нижчу середню ймовірність, модифікований YOLO8 продемонстрував вищу точність для об'єктів класу 7, що підтверджується їхньою більшою кількістю у результатах.

Порівняння координат рамок (код для порівняння представлений в додатку Д) показало незначні відмінності між алгоритмами. Модифікований YOLO8 показав точнішу локалізацію для вантажних автомобілів, тоді як базовий YOLO8 частіше визначав рамки для легкових автомобілів, включаючи об'єкти з меншою впевненістю.

Час виконання для базового YOLO8 склав 0.82 секунди, тоді як модифікований YOLO8 витратив 1.04 секунди на обробку зображення. Збільшений час обробки модифікованої моделі пов'язаний із додатковими обчисленнями, спрямованими на покращення детекції складних об'єктів.

У підсумку базовий YOLO8 продемонстрував більшу кількість виявлень за менший час, що свідчить про його ефективність для загальних задач виявлення. Модифікований YOLO8, натомість, забезпечив вищу точність класифікації складніших об'єктів, таких як вантажні автомобілі, але за рахунок зменшення загальної кількості детекцій та збільшення часу обробки. Це робить модифіковану модель більш придатною для завдань, де пріоритетом є деталізація та точність виявлення. Подальший аналіз на великих наборах даних допоможе визначити вплив цих модифікацій у різних умовах, включно з низьким освітленням і щільним трафіком.

Висновки до розділу 3

1. Проведено опис інструментів, програмних засобів та бібліотек для реалізації системи виявлення та класифікації транспортних засобів на основі алгоритмів глибокого навчання та комп’ютерного зору.
2. Модифіковано архітектуру YOLOv8 з інтеграцією модуля уваги для покращення точності детекції об'єктів, особливо в складних умовах, таких як перекриття, різні ракурси зйомки та змінне освітлення.
3. Проведене навчання моделі на датасеті "Vehicle Detection: 8 Classes" дозволило адаптувати алгоритм до різних умов середовища та реалістичних сценаріїв.
4. Результати дослідження показали, що модифікація YOLOv8 забезпечила краще виявлення об'єктів при збереженні прийнятної швидкості обробки. Час виконання для модифікованої моделі дещо збільшився, проте це компенсувалося вищою точністю, що є критичним для задач моніторингу дорожнього руху.

ВИСНОВКИ

1. Проведено аналіз предметної області, сучасних методів та підходів до вирішення задач виявлення і класифікації транспортних засобів на основі комп'ютерного зору та алгоритмів глибокого навчання.

2. Проведено аналіз методів виявлення та класифікації транспортних засобів. Розглянуто основні принципи роботи згорткових нейронних мереж.

3. Визначено ключові вимоги до системи виявлення транспортних засобів: точність детекції, здатність до роботи в реальному часі та стійкість до зовнішніх факторів, таких як змінне освітлення, погодні умови та часткове перекриття об'єктів.

4. Сформульовано задачу дослідження, яка полягає у створенні високопродуктивного методу виявлення і класифікації транспортних засобів.

5. Здійснено опис інформації та підходів до виявлення та класифікації транспортних засобів.

6. Проведено детальний аналіз методів і алгоритмів для виявлення та класифікації транспортних засобів. Особлива увага приділена моделям серії YOLO та їхньому еволюційному розвитку.

7. Проведено порівняльний аналіз моделей YOLO з іншими сучасними підходами до детекції об'єктів, такими як Faster R-CNN та SSD.

8. Розглянуто основні метрики оцінки ефективності алгоритмів, зокрема що використовуються для кількісної оцінки якості детекції та класифікації об'єктів.

9. Проведено опис інструментів, програмних засобів та бібліотек для реалізації системи виявлення та класифікації транспортних засобів на основі алгоритмів глибокого навчання та комп’ютерного зору.

10. Модифіковано архітектуру YOLOv8 з інтеграцією модуля уваги для покращення точності детекції об'єктів, особливо в складних умовах, таких як перекриття, різні ракурси зйомки та змінне освітлення.

11. Проведене навчання моделі на датасеті "Vehicle Detection: 8 Classes" дозволило адаптувати алгоритм до різних умов середовища та реалістичних сценаріїв.

12. Результати дослідження показали, що модифікація YOLOv8 забезпечила краще виявлення об'єктів при збереженні прийнятної швидкості обробки. Час виконання для модифікованої моделі дещо збільшився, проте це компенсувалося вищою точністю, що є критичним для задач моніторингу дорожнього руху.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1.Ian Goodfellow, Yoshua Bengio, Aaron Courville. Deep Learning: Foundations and Concepts – Springer, 2023 – 649p.

2.Класифікація транспортних засобів за екологічними показниками (2023) [Електронний ресурс]. – Режим доступу: https://ir.nmu.org.ua/handle/123456789/160680

3.Вдосконалення методів класифікації транспортних засобів (2022) [Електронний ресурс]. – Режим доступу:- https://elib.lntu.edu.ua/sites/default/files/elib_upload/%D0%9A%D0%B8%D1%89%D1%83%D0%BD/page1.html

4.Перспективи розвитку класифікації транспортних засобів в Україні (2021) [Електронний ресурс]. – Режим доступу: https://elib.lntu.edu.ua/sites/default/files/elib_upload/%D0%9A%D0%B8%D1%89%D1%83%D0%BD/page1.html

5.Аналіз методів класифікації транспортних засобів (2020) [Електронний ресурс]. – Режим доступу: https://mbel.kaist.ac.kr/lab/publication.html

6.Сучасні тенденції класифікації транспортних засобів (2019) [Електронний ресурс]. – Режим доступу: https://elib.lntu.edu.ua/sites/default/files/elib_upload/%D0%9A%D0%B8%D1%89%D1%83%D0%BD/page1.html

7.Транспортні системи: Класифікація, проектування, експлуатація (2022) [Електронний ресурс]. – Режим доступу: https://nace.lursoft.lv/H/transport-i-skladirovanie

8.Класифікація та кодування транспортних засобів (2021) [Електронний ресурс]. – Режим доступу: https://www.kmu.gov.ua/npas/243939925

9.Транспортне право: Класифікація транспортних засобів (2020) [Електронний ресурс]. – Режим доступу: http://nubip.edu.ua/sites/default/files/u123/%D1%80%D0%BF%20%D1%82%D1%80

%D0%B0%D0%BD%D1%81%20%D0%BF%D1%80%D0%B0%D0%B2%D0%BE.pdf

10.Експлуатація та технічне обслуговування транспортних засобів: Класифікація (2019) [Електронний ресурс]. – Режим доступу: https://www.mdoffice.com.ua/ua/aMDOClassDic.html?dic_num=13

11.Огляд сучасних моделей виявлення об’єктів на основі глибокого навчання (2023) [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/2104.11892

12.Сучасні методи сегментації зображень у комп'ютерному зорі (2022) [Електронний ресурс]. – Режим доступу: http://arxiv.org/abs/2001.05566

13.SIMPLE ONLINE AND REALTIME TRACKING [Електронний ресурс]. – Режим доступу: https://arxiv.org/pdf/1602.00763.pdf

14.What Is Computer Vision? [Електронний ресурс]. – Режим доступу: https://builtin.com/machine-learning/computer-vision

15.Computer Vision. What is Computer Vision? [Електронний ресурс]. – Режим доступу: https://www.protex.ai/glossary/computer-vision

16.An Introduction to Computer Vision [Електронний ресурс]. – Режим доступу: https://www.baeldung.com/cs/computer-vision

17.Applications of Computer Vision in AI [Електронний ресурс]. – Режим доступу: https://www.analyticssteps.com/blogs/7-applications-computer-vision-ai

18.Глибоке навчання для комп'ютерного зору (2021) [Електронний ресурс]. – Режим доступу: https://www.amazon.com/Deep-Learning-Adaptive-Computation-Machine/dp/0262035618

19.Object Recognition and How Does it Work? [Електронний ресурс]. – Режим доступу до ресурсу: https://www.cprime.com/resources/blog/what-is-object-recognition-and-how-does-it-work/

20.YOLO Object Detection Explained [Електронний ресурс]. – Режим доступу: https://www.datacamp.com/blog/yolo-object-detection-explained

21.What Is YOLO Algorithm? [Електронний ресурс]. – Режим доступу: https://www.baeldung.com/cs/yolo-algorithm

22.TrafficVision [Електронний ресурс]. – Режим доступу: http://www.coralsales.com/products/its/camerasanddetection/trafficvision/

23.Object Tracking [Електронний ресурс]. – Режим доступу до ресурсу: https://encord.com/glossary/object-tracking-definition/

24.The Complete Guide to Object Tracking [Електронний ресурс]. – Режим доступу до ресурсу: https://encord.com/blog/object-tracking-guide/#h1

25.Introduction to the Top Object Tracking Techniques [Електронний ресурс]. – Режим доступу до ресурсу: https://www.tasq.ai/blog/object-tracking/

26.Комп'ютерний зір для автономних транспортних засобів: огляд (2019) [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/1704.05519

27.Ultralytics. Benchmark Mode — Ultralytics Documentation [Електронний ресурс]. – URL: https://docs.ultralytics.com/modes/benchmark/

28.OpenCV Tutorial | OpenCV using Python [Електронний ресурс]. – Режим доступу: https://www.javatpoint.com/opencv

29.Комп'ютерний зір: основи та алгоритми (2023) [Електронний ресурс]. – Режим доступу: http://szeliski.org/Book/

30.Комп'ютерний зір: сучасні методи та застосування (2022) [Електронний ресурс]. – Режим доступу: https://www.amazon.com/Computer-Vision-Modern-Approach-2nd/dp/013608592X

31.Azhad Zuraimi, Fadhlan Hafizhelmi Kamaru Zaman, "Vehicle Detection and Tracking using YOLO and DeepSORT", in 2021 IEEE 11th IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE) DOI:10.1109/ISCAIE51753.2021.9431784

32.M. Ravichandran, K. Laxmikant and A. Muthu, "Efficient Vehicle Detection and Classification using YOLO v8 for Real-Time Applications," 2023 Global Conference on Information Technologies and Communications (GCITC), Bangalore, India, 2023, pp. 1-5, doi: 10.1109/GCITC60406.2023.10426587.

33.J. Karangwa, J. Liu and Z. Zeng, "Vehicle Detection for Autonomous Driving: A Review of Algorithms and Datasets," in IEEE Transactions on Intelligent Transportation Systems, vol. 24, no. 11, pp. 11568-11594, Nov. 2023, doi: 10.1109/TITS.2023.3292278.

34.Michael Abebe Berwo, Asad Khan, Yong Fang, Hamza Fahim, Shumaila Javaid, Jabar Mahmood, Zain Ul Abideen, Syam M.S. «Deep Learning Techniques for Vehicle Detection and Classification from Images/Videos: A Survey», in Sensors 2023, 23(10), 4832; https://doi.org/10.3390/s23104832

35.Об'їзна дорога м. Тернопіль. Україна. Велике будівництво [Електронний ресурс] // YouTube. – URL: https://www.youtube.com/watch?v=OkFRvRb0ZNk

36.Карнидал М.С. Аналіз алгоритмів YOLO8 та YOLO11 в задачах виявлення і класифікації транспортних засобів / М.С. Карнидал // Матеріали Всеукраїнської науково-практичної конференції студентів, аспірантів та молодих вчених «Інтелектуальні Комп’ютерні Системи та Мережі» - Тернопіль: Західноукраїнський національний університет, кафедра комп’ютерної інженерії, 2024 р., С. 166-168.

37.Карнидал М.С. Структура інтелектуальної системи виявлення та класифікації транспортних засобів на основі комп'ютерного зору / М.С. Карнидал // Міжнародна науково-практична інтернет-конференція «Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення» [Електронний ресурс] – Режим доступу: http://www.konferenciaonline.org.ua/ua/article/id-2019/

38.Комар М.П., Саченко А.О., Васильків Н.М., Загородня Д.І. Методичні рекомендації до виконання кваліфікаційної роботи з освітньо-професійної програми «Комп’ютерні науки» спеціальності 122 «Комп’ютерні науки» за другим (магістерським) рівнем вищої освіти. – Тернопіль: ЗУНУ, 2024. – 32 с.

Додаток А

Копії публікацій

ЗАХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ

ВСЕУКРАЇНСЬКА НАУКОВО-ПРАКТИЧНА КОНФЕРЕНЦІЯ СТУДЕНТІВ, АСПІРАНТІВ ТА МОЛОДИХ ВЧЕНИХ «ІНТЕЛЕКТУАЛЬНІ КОМП'ЮТЕРНІ СИСТЕМИ ТА МЕРЕЖІ»

5 ЛИСТОПАДА 2024

KI.WUNU.EDU.UA/CONFERENCE/

ТЕРНОПІЛЬ

2024

Карнидал М.С.
магістрант 2 курсу ФКІТ ЗУНУ
Науковий керівник к.т.н., доцент Загородня Д.І., кафедра ЮСУ ЗУНУ

АНАЛІЗ АЛГОРИТМІВ YOLO8 ТА YOLO11 В ЗАДАЧАХ ВИЯВЛЕННЯ І КЛАСИФІКАЦІЇ ТРАНСПОРТНИХ ЗАСОБІВ

Вступ. Сучасні технології обробки зображень відкривають нові можливості для розвитку інтелектуальних транспортних систем. Алгоритм "You Only Look Once" (YOLO) зарекомендував себе як один із найефективніших інструментів для реального часу розпізнавання об'єктів, завдяки швидкості та високій точності. YOLO став важливим елементом для вирішення таких завдань, як виявлення транспортних засобів, розпізнавання дорожніх знаків і забезпечення безпеки пішоходів. У даній статті представлено огляд використання алгоритмів YOLO8 та YOLO11 у задачах виявлення та класифікації транспортних засобів на зображеннях [1]. Проведено порівняльний аналіз роботи обох алгоритмів, включаючи їх швидкість обробки, точність детекції та здатність до розпізнавання об'єктів різних класів.

Постановка задачі. У сучасних інтелектуальних транспортних системах, автономних автомобілях та системах відеоспостереження важливим є пошук оптимального балансу між швидкістю обробки даних і точністю виявлення об'єктів. Для цього актуально досліджувати та порівнювати можливості різних алгоритмів комп'ютерного зору, таких як YOLO8 та YOLO11, для розпізнавання транспортних засобів.

Метою даного дослідження є порівняння роботи алгоритмів YOLO8 та YOLO11 у задачах виявлення та класифікації транспортних засобів на основі їхньої швидкості, точності та здатності розпізнавати різні класи об'єктів. Зокрема, увагу зосереджено на таких аспектах: порівнянні точності виявлення легкових та вантажних автомобілів, оцінці продуктивності алгоритмів у реальному часі, аналізі переваг і недоліків.

Основний матеріал. Розпізнавання транспортних засобів є ключовим компонентом сучасних інтелектуальних систем керування дорожнім рухом, автономних автомобілів і розумного відеоспостереження. У цьому контексті алгоритми YOLO8 та YOLO11 пропонують два різні підходи до вирішення цих задач, кожен із яких має свої переваги й обмеження.

YOLO8 демонструє високу продуктивність завдяки швидкому часу обробки зображень. Його архітектура оптимізована для задач, які потребують роботи в реальному часі, таких як моніторинг дорожньої ситуації або автономне керування транспортом. Його головною перевагою є здатність швидко ідентифікувати більшість об'єктів на зображенні, зокрема легкові та вантажні автомобілі. Завдяки цьому YOLO8 є ефективним вибором для сценаріїв, де час реакції є критичним, наприклад, у системах активної безпеки.

YOLO11, у свою чергу, орієнтований на підвищену деталізацію та точність розпізнавання. Він вдосконалює архітектуру попередніх версій YOLO, забезпечуючи вищий рівень деталізації при класифікації транспортних засобів і визначенні їхніх координат. Це робить його особливо корисним у складних умовах, таких як низьке освітлення, висока щільність трафіку або наявність дрібних об'єктів на зображенні. YOLO11 демонструє більшу стійкість до таких викликів, що робить його придатним для завдань, де пріоритетом є точність.

Однією з ключових особливостей обох алгоритмів є їхня здатність класифікувати транспортні засоби за типами, зокрема виділяти легкові автомобілі, вантажівки, автобуси, мотоцикли тощо. Ця властивість полегшує аналіз дорожнього руху, оцінку його щільності та адаптацію роботи інтелектуальних світлофорів. Однак YOLO11 демонструє вищу, що свідчить про його адаптованість до розпізнавання специфічних типів транспортних засобів.

Ще однією перевагою YOLO11 є його універсальність у застосуванні на різних платформах. Завдяки оптимізації для високопродуктивних графічних процесорів (GPU) та edge-пристроїв, таких як NVIDIA Jetson, він може бути використаний у різних сценаріях — від великих міських транспортних систем до аналізу дорожньої ситуації у віддалених місцевостях за допомогою дронів.

YOLO8, хоча й поступається деталізацією YOLO11, є значно швидшим і легшим у використанні. Це робить його ідеальним для систем із обмеженими обчислювальними ресурсами або для завдань, де потрібна максимальна швидкість обробки. Обидва алгоритми підтримують можливість налаштування через відкритий код, що дозволяє адаптувати їх до специфічних потреб, таких як виявлення певних типів транспортних засобів або дорожньої інфраструктури.

Незважаючи на всі переваги, обидва алгоритми мають свої обмеження. YOLO8, хоча і швидкий, може мати меншу точність у розпізнаванні дрібних об'єктів, а YOLO11 вимагає більш потужних апаратних ресурсів для досягнення максимальних результатів. У будь-якому випадку, обидва алгоритми є потужними інструментами, які відповідають сучасним вимогам до розпізнавання транспортних засобів і забезпечують високу ефективність у своїх відповідних сферах застосування.

Практична реалізація. Для аналізу було використано зображення, отримане із відеоматеріалу, взятого з YouTube [2], що демонструє об'їзну дорогу м. Тернопіль (Рисунок 1). Зображення є статичним кадром, що було вибрано для забезпечення репрезентативності реальних дорожніх умов. Цей кадр зосереджує увагу на типовій ситуації руху транспорту, яка включає:різні типи транспортних засобів: легкові автомобілі, вантажівки, що створює умови для перевірки класифікаційних можливостей алгоритмів; щільність дорожнього руху: транспортні засоби можуть частково перекривати один одного, що є типовою проблемою для задач детекції; реальні умови освітлення та навколишнього середовища, характерні для місця зйомки, що додає складності алгоритмам розпізнавання.

Рисунок 1 - Відеокадр з огляду об'їзної дороги, м.Тернопіль

Виконання виявлення і розпізнавання здійснювалось на комп'ютері з процесором AMD Ryzen 7 6850, 32 ГБ оперативної пам'яті, твердотільним диском об'ємом 1 ТБ і операційною системою Linux Ubuntu 24.04 LTS. На рисунку 2 зображено

Рисунок 2 – Виявлення і класифікація об'єктів за допомогою за допомогою алгоритму YOLO8 (зліва) та YOLO11 (справа)

Для навчання та оцінки продуктивності моделей YOLO8 та YOLO11 використовувався датасет "Vehicle Detection: 8 Classes", доступний на платформі Kaggle. Датасет містить анотовані зображення транспортних засобів, розділені на 8 класів, що дозволяє ефективно тренувати й тестувати моделі для задач виявлення та класифікації об'єктів на зображеннях.

Інформація про виділені об'єкти зберігається у файли з наступною табличною структурою:

```
Index,Class,Probability,Xmin,Ymin,Xmax,Ymax
1,2,0.90,1276.91,850.37,1461.93,946.91
2,7,0.89,594.20,702.81,827.79,915.28
3,2,0.82,1064.12,873.93,1151.35,979.51
4,2,0.81,372.97,755.45,472.04,830.66
                    * * *
24,7,0.29,1084.07,406.83,1217.56,481.14
25,2,0.28,1283.68,418.28,1346.29,457.18
26,2,0.27,960.85,361.08,992.95,412.60
```

Кожен такий файл починається з заголовка, де описані ключові параметри для кожного виявленого об'єкта. У заголовку визначено шість колонок: Index — номер виділеного об'єкта, який починається з 1 і зростає для кожного нового запису. Class — ідентифікатор класу об'єкта. Клас 2 відповідає легковому автомобілю (car), а клас 7 — вантажному автомобілю (truck). Probability — це ймовірність, з якою алгоритм ідентифікував об'єкт, представлена у вигляді числа від 0 до 1, де 1 означає максимальну впевненість. Xmin та Ymin — координати верхнього лівого кута прямокутника, що окреслює об'єкт. Xmax та Ymax — координати нижнього правого кута цього прямокутника.

Аналіз результатів. Для аналізу результатів порівнюються файли що містять результати виявлення та класифікації кожного з алгоритмів. Результати роботи алгоритмів YOLO8 і YOLO11 показали певні відмінності у кількості виявлених об'єктів, класифікації, ймовірностях та координатах рамок. YOLO8 виявив 26 об'єктів, що на два більше, ніж YOLO11, який знайшов 24 об'єкти. Ця різниця свідчить про вищу чутливість YOLO8 до деталей на зображенні. Проте можливість хибних спрацьовувань для цих додаткових об'єктів потребує подальшого аналізу.

З точки зору класифікації, в обох алгоритмах переважна більшість об'єктів належить до класу 2 (легкові автомобілі). Однак YOLO8 визначив чотири об'єкти класу 7 (вантажні автомобілі), тоді як YOLO11 ідентифікував п'ять таких об'єктів. Це може свідчити про те, що YOLO11 краще справляється з розпізнаванням вантажних автомобілів.

Щодо ймовірностей, найвища впевненість у YOLO8 становила 0.90 для об'єкта класу 2, тоді як YOLO11 досяг максимального значення ймовірності 0.87. У цілому, YOLO8 продемонстрував дещо вищі ймовірності для деяких виявлень, що свідчить про його більшу впевненість. Проте необхідно перевірити, чи ці об'єкти є правильними.

Координати рамок в обох алгоритмах у більшості випадків збігаються. Проте для деяких об'єктів є відмінності, які можуть бути наслідком різних підходів до визначення меж об'єктів.

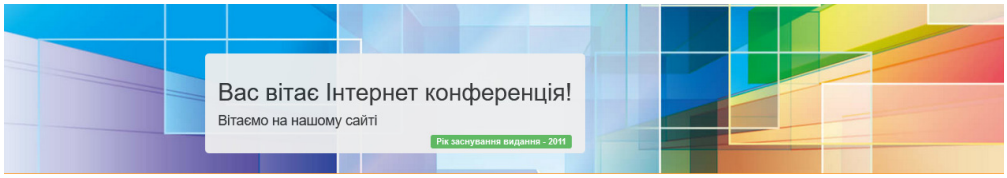
Час виконання також відрізняється: YOLO8 обробив зображення за 0.81 секунди, тоді як YOLO11 витратив 0.91 секунди. Це робить YOLO8 швидшим, що може бути перевагою в умовах реального часу.

Висновки. Результати роботи алгоритмів YOLO8 і YOLO11 показали певні відмінності у кількості виявлених об'єктів, класифікації, ймовірностях та координатах рамок. Різниця у кількості свідчить про вищу чутливість YOLO8 до деталей на зображенні, проте можливість хибних спрацьовувань для цих додаткових об'єктів потребує подальшого аналізу. У цілому, YOLO8 є швидшим, але менш точним у деталізації, тоді як YOLO11 працює трохи повільніше, але забезпечує вищу деталізацію та точність виявлення для певних класів об'єктів.

Список літератури

1. Ultralytics. Benchmark Mode — Ultralytics Documentation [Електронний ресурс]. – URL: https://docs.ultralytics.com/modes/benchmark/

2. Об'їзна дорога м. Тернопіль. Україна. Велике будівництво [Електронний ресурс] // YouTube. – URL: https://www.youtube.com/watch?v=OkFRvRb0ZNk



РОЗПОДІЛЕНЕ ГЛИБОКЕ НАВЧАННЯ ДЛЯ ОБРОБКИ ДАНИХ ІНТЕРНЕТУ РЕЧЕЙ

[1. Інформаційні системи і технології]

Автор: **Демчишин Владислав Володимирович, магістр, Західноукраїнський національний університет, м. Тернопіль**

Відкрити тези доповіді »

ІНТЕГРАЦІЯ GOOGLE SHEETS З PYTHON ТА MATLAB ДЛЯ СКЛАДНИХ ОБЧИСЛЕНЬ

[1. Інформаційні системи і технології]

Автор: **Замируєва О.В., кандидат фізико-математичних наук, доцент кафедри теоретичної та комп'ютерної фізики імені А. В. Свідзинського навчально-наукового фізико-технологічного інституту ВНУ імені Лесі Українки, м. Луцьк; Шваліковський А., студент 1-ого курсу навчально-наукового фізико-технологічного інституту ВНУ імені Лесі Українки, м. Луцьк; Мельничук А., студент 1-ого курсу навчально-наукового фізико-технологічного інституту ВНУ імені Лесі Українки, м. Луцьк**

Відкрити тези доповіді »

СТРУКТУРА ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ ВИЯВЛЕННЯ ТА КЛАСИФІКАЦІЇ ТРАНСПОРТНИХ ЗАСОБІВ НА ОСНОВІ КОМП'ЮТЕРНОГО ЗОРУ

[1. Інформаційні системи і технології]

Автор: **Карнидал Михайло Сергійович, студент, Західноукраїнський національний університет, м. Тернопіль**

Відкрити тези доповіді »

ОПТИМІЗАЦІЯ ПРОГРАМНОГО КОДУ НА АСЕМБЛЕРІ ДЛЯ СУЧАСНИХ ПРОЦЕСОРІВ

[1. Інформаційні системи і технології]

Автор: **Клименко Ілля Максимович, студент, Державний торгівельно-економічний університет, м.Київ**

Відкрити тези доповіді »

НАЛАШТУВАННЯ ПАРАМЕТРІВ ГЛИБОКИХ НЕЙРОННИХ МЕРЕЖ ЗА ДОПОМОГОЮ ЕВОЛЮЦІЙНИХ АЛГОРИТМІВ

[1. Інформаційні системи і технології]

Автор: **Книш Тетяна Олегівна, магістр, Західноукраїнський національний університет, м. Тернопіль**

Відкрити тези доповіді »

Карнидал Михайло Сергійович, *студент,*
Західноукраїнський національний університет, м. Тернопіль

СТРУКТУРА ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ ВИЯВЛЕННЯ ТА КЛАСИФІКАЦІЇ ТРАНСПОРТНИХ ЗАСОБІВ НА ОСНОВІ КОМП'ЮТЕРНОГО ЗОРУ

На даний час стрімко розвивається концепція «розумного міста» де зазвичай присутні інтелектуальні системи управління транспортом для оптимізації дорожнього руху. Це дає змогу зменшити затори та знизити рівень шкідливих викидів. Інтеграція виявлення транспортних засобів у такі системи забезпечить нові можливості для моніторингу, автоматичного збору статистики та покращення умов для роботи громадського транспорту.

Традиційні підходи виявлення та класифікації транспортних засобів, що базуються на ручному аналізі або застарілих технологіях, не здатні забезпечити належний рівень точності та швидкості обробки даних. Впровадження інтелектуальних систем виявлення та класифікації транспортних засобів вимагає створення структурованих та адаптивних рішень для автоматизованого виявлення транспортних засобів у реальному часі та прийнятті рішень. Побудова такої структури є важливим етапом для забезпечення ефективної взаємодії між елементами системи.

Початковим етапом системи виявлення транспортних засобів є (I) збір даних. Це може включати використання фізичних камер (захоплення відео зображень з камер, розташованих на дорогах) або підключення до відеопотоків з вже наявних джерел відеозапису, інформацію з відеореєстраторів, дронів. На цьому етапі відбувається перетворення відео в послідовність кадрів та фільтрація некоректних або низькоякісних зображень. Отримані відео чи зображення можуть потребувати (II) попередньої обробки для покращення якості, наприклад видалення шуму або вирівнювання яскравості зображень. Цей етап також може включати конвертацію зображень у потрібний формат, зменшення роздільної здатності зображень для

пришвидшення обробки або інші методи обробки зображень. Після цього проводиться (III) виявлення транспортних засобів та їхнє обведення межами для подальшої обробки. Найчастіше для виявлення транспортних засобів використовують наступні методи: SSD (Single Shot Multibox Detector), YOLO (You Only Look Once) та Faster R-CNN, які використовують техніки обробки зображень розпізнавання об'єктів за контуром та пошук об'єктів за кольором чи текстурою [1-2].

Також одним з важливих кроків є (IV) класифікація транспортних засобів за типами (наприклад, легкові автомобілі, вантажівки, автобуси, мотоцикл тощо) і розпізнавання номерних знаків. Це дозволяє отримати додаткову інформацію про транспортні засоби, визначити їх тип та ідентифікувати. Для цього використовують глибокі нейронні мережі (Convolutional Neural Networks), такі як ResNet чи MobileNet [3].

Якщо система працює з відео, то після виявлення транспортних засобів відбувається процес (V) відстеження, що дозволяє слідкувати за рухом об'єктів протягом часу та визначати траєкторію руху транспортних засобів. Це може бути досягнуто за допомогою алгоритмів слідкування за об'єктами такими як Kalman Filter, оптичний потік (Optical Flow), DeepSORT (Deep Simple Online and Realtime Tracking), які використовують історію траєкторій та інші характеристики для точного відстеження [4].

Наступними етапами є (VI) аналіз даних та (VII) прийняття рішень. З використанням аналізу траєкторії транспортного засобу можна розраховувати швидкість руху, аномальну поведінку (наприклад, автомобілі, які рухаються у забороненому напрямку). Це може бути корисною інформацією для оцінки трафіку та виявлення ділянок зі заторами. Система може відправляти сповіщення (наприклад, про аварії чи затори), генерувати звітів про трафік, управляти світлофорами в розумних містах, бути інтегрованою з системами безпеки чи контролю доступу.

Отримані дані про трафік, включаючи відстежені об'єкти, класифікацію, номерні знаки та швидкість, можуть бути зібрані та збережені для подальшого

використання. Це включає (VIII) зберігання даних у відповідному форматі або базі даних для подальшої обробки та аналізу. Остаточний етап включає візуалізацію отриманих даних про трафік у зручному форматі, наприклад, за допомогою графіків, діаграм чи карт. Крім того, аналітичні інструменти можуть бути використані для виявлення патернів, трендів та проблемних ділянок доріг.

Побудова системи виявлення та класифікації транспортних засобів обумовлена необхідністю забезпечення високої точності, швидкості та надійності в обробці транспортних потоків, а розробка такої системи є важливим кроком до створення ефективної, адаптивної та інтелектуальної транспортної інфраструктури, що сприяє підвищенню безпеки на дорогах та має позитивний економічний та соціальний ефект, оскільки сприяє зниженню витрат на обслуговування транспортної інфраструктури, покращенню умов руху на дорогах та зменшенню аварійності.

Література:

1. Azhad Zuraimi, Fadhlan Hafizhelmi Kamaru Zaman, "Vehicle Detection and Tracking using YOLO and DeepSORT", in 2021 IEEE 11th IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE) DOI:10.1109/ISCAIE51753.2021.9431784

2. M. Ravichandran, K. Laxmikant and A. Muthu, "Efficient Vehicle Detection and Classification using YOLO v8 for Real-Time Applications," 2023 Global Conference on Information Technologies and Communications (GCITC), Bangalore, India, 2023, pp. 1-5, doi: 10.1109/GCITC60406.2023.10426587.

3. J. Karangwa, J. Liu and Z. Zeng, "Vehicle Detection for Autonomous Driving: A Review of Algorithms and Datasets," in IEEE Transactions on Intelligent Transportation Systems, vol. 24, no. 11, pp. 11568-11594, Nov. 2023, doi: 10.1109/TITS.2023.3292278.

4. Michael Abebe Berwo, Asad Khan, Yong Fang, Hamza Fahim, Shumaila Javaid, Jabar Mahmood, Zain Ul Abideen, Syam M.S. «Deep Learning Techniques for Vehicle Detection and Classification from Images/Videos: A Survey», in Sensors 2023, 23(10), 4832; https://doi.org/10.3390/s23104832

Додаток Б

Код модифікованого методу

```
from ultralytics import YOLO
import time
import torch
import torch.nn as nn
import torch.nn.functional as F

# CBAM Module
class CBAM(nn.Module):
    def __init__(self, channels, reduction=16):
        super(CBAM, self).__init__()
        # Channel Attention
        self.channel_attention = nn.Sequential(
            nn.AdaptiveAvgPool2d(1),
            nn.Conv2d(channels, channels // reduction, 1,
bias=False),
            nn.ReLU(),
            nn.Conv2d(channels // reduction, channels, 1,
bias=False),
            nn.Sigmoid()
        )
        # Spatial Attention
        self.spatial_attention = nn.Sequential(
            nn.Conv2d(2, 1, 7, padding=3, bias=False),
            nn.Sigmoid()
        )

    def forward(self, x):
        # Channel Attention
        ca = self.channel_attention(x)
        x = x * ca
        # Spatial Attention
        sa = torch.cat([torch.mean(x, dim=1, keepdim=True),
torch.max(x, dim=1, keepdim=True)[0]], dim=1)
        sa = self.spatial_attention(sa)
        x = x * sa
        return x

# Модифікація моделі YOLO8 з CBAM
class ModifiedYOLO8(nn.Module):
    def __init__(self, original_model):
        super(ModifiedYOLO8, self).__init__()
        self.backbone = original_model.model[0]  # Backbone
моделі YOLO8
        self.cbam = CBAM(channels=256)  # Вибір кількості каналів
для CBAM (адаптуйте при потребі)
        self.head = original_model.model[1:]  # Залишок моделі
YOLO8

    def forward(self, x):
```

```
            x = self.backbone(x)
            x = self.cbam(x)
            for module in self.head:
                x = module(x)
            return x

    # Завантаження оригінальної моделі YOLO8
    original_model = YOLO("yolov8m.pt")
    yolo_model = ModifiedYOLO8(original_model)

    # Detect objects from classes
    classes = [1, 2, 3, 5, 7]

    # Вимірювання часу виконання
    start_time = time.time()

    # Виконання передбачення з модифікованою моделлю
    results = original_model.predict(source="output.jpg",
classes=classes)

    end_time = time.time()
    execution_time = end_time - start_time

    # Збереження результатів
    output_file = "detectionsYolo8_CBAM.txt"
    with open(output_file, "w") as f:
        f.write("Index,Class,Probability,Xmin,Ymin,Xmax,Ymax\n")   #
Заголовок CSV
        for i, (box, cls, conf) in
enumerate(zip(results[0].boxes.xyxy, results[0].boxes.cls,
results[0].boxes.conf)):
            xmin, ymin, xmax, ymax = box.tolist()
            class_id = int(cls)
            probability = float(conf)

f.write(f"{i+1},{class_id},{probability:.2f},{xmin:.2f},{ymin:.2f},{
xmax:.2f},{ymax:.2f}\n")

        f.write(f"\nExecution Time: {execution_time:.2f} seconds\n")

    print(f"Detections saved to {output_file}")
    results[0].show()
```

Додаток В

Інформація про виділені об’єкти

Результати роботи методу YOLOv8

```
Index,Class,Probability,Xmin,Ymin,Xmax,Ymax
1,2,0.89,218.98,876.36,423.64,1076.58
2,2,0.87,462.36,672.52,594.21,812.83
3,2,0.79,1191.21,286.72,1245.54,332.98
4,2,0.78,1257.80,459.34,1339.51,527.00
5,2,0.72,1034.20,108.74,1065.05,137.52
6,7,0.72,364.13,437.99,532.00,523.90
7,2,0.70,415.04,355.58,475.24,397.06
8,2,0.69,875.66,345.43,934.15,419.07
9,2,0.69,959.22,425.20,1018.53,505.36
10,2,0.66,604.85,434.00,730.77,500.16
11,2,0.66,1375.70,319.90,1450.80,386.34
12,2,0.65,577.47,136.09,640.15,163.56
13,2,0.65,1055.65,144.17,1093.79,177.74
14,2,0.64,334.77,298.20,428.51,361.17
15,2,0.61,240.40,572.26,422.22,629.41
16,2,0.59,1164.34,218.76,1227.63,287.96
17,2,0.52,365.29,437.99,532.36,524.00
18,2,0.48,639.46,138.69,716.10,170.83
19,2,0.45,916.15,21.70,953.50,65.82
20,2,0.43,1302.01,299.14,1386.11,374.05
21,7,0.42,1203.52,84.61,1353.07,214.37
22,2,0.41,624.97,255.80,675.01,298.21
23,7,0.39,512.65,228.83,575.99,306.73
24,2,0.38,692.89,87.87,754.14,113.15
25,2,0.38,1114.92,134.45,1162.03,187.33
26,2,0.36,996.68,170.65,1048.64,233.28
27,2,0.36,1616.59,164.62,1701.37,194.58
28,2,0.35,1100.84,100.08,1136.35,146.24
29,2,0.31,1082.71,63.61,1114.66,101.63
30,7,0.30,1431.30,344.72,1579.73,513.88
31,2,0.30,1206.79,211.58,1279.39,242.60
32,2,0.30,832.32,77.57,880.60,109.50
33,2,0.26,1210.22,213.54,1288.38,247.69
34,7,0.26,916.28,19.71,953.98,65.63
35,7,0.25,644.06,157.02,787.59,262.53

Execution Time: 0.82 seconds
```

Результати роботи модифікованого методу YOLOv8

```
Index,Class,Probability,Xmin,Ymin,Xmax,Ymax
1,2,0.92,219.00,877.37,422.43,1075.35
2,2,0.84,462.15,672.84,595.78,811.13
3,2,0.81,1865.73,845.16,1919.32,987.43
4,2,0.75,1189.68,287.32,1247.81,335.26
5,2,0.71,1259.34,461.01,1339.55,526.84
6,7,0.69,367.44,440.39,533.33,524.54
7,2,0.69,875.98,342.70,931.90,420.41
8,2,0.65,412.60,355.85,475.65,399.02
9,7,0.56,1163.31,219.39,1224.32,288.18
10,2,0.55,335.15,302.30,427.21,361.47
11,2,0.55,960.63,425.82,1018.31,508.94
12,2,0.52,1378.68,321.37,1456.28,388.41
13,2,0.46,640.37,137.64,712.04,168.36
14,7,0.46,643.33,170.29,787.03,267.26
15,2,0.45,605.44,434.24,730.88,500.92
16,2,0.43,1782.52,308.96,1858.55,358.93
17,2,0.41,1034.77,110.38,1063.67,137.22
18,7,0.41,915.68,19.00,954.38,65.90
19,2,0.38,576.54,136.33,638.80,164.59
20,2,0.36,515.43,244.46,575.80,305.71
21,2,0.36,1081.08,52.76,1115.74,101.00
22,7,0.33,1115.86,136.38,1162.48,190.89
23,2,0.33,1054.82,142.01,1093.77,178.87
24,2,0.31,1116.11,136.14,1162.27,190.62
25,7,0.31,1435.40,346.21,1578.06,512.04
26,2,0.28,1616.99,166.83,1700.09,195.48
27,7,0.27,604.28,433.81,730.42,501.50
28,2,0.27,244.08,571.93,421.71,629.94
29,2,0.26,691.44,88.63,753.15,113.94
30,2,0.26,624.20,256.16,675.15,299.44
31,2,0.25,1056.42,35.77,1109.98,73.01

Execution Time: 1.04 seconds
```

Додаток Г

Код для порівняння результатів

```
import csv
import math

def iou(box1, box2, delta=10):
    """
    Розрахунок Intersection over Union (IoU) для двох прямокутників із похибкою координат.
    delta – величина похибки (в пікселях) для +/- корекції координат.
    """
    x1 = max(box1[0] - delta, box2[0] - delta)
    y1 = max(box1[1] - delta, box2[1] - delta)
    x2 = min(box1[2] + delta, box2[2] + delta)
    y2 = min(box1[3] + delta, box2[3] + delta)

    inter_area = max(0, x2 - x1) * max(0, y2 - y1)
    box1_area = (box1[2] - box1[0]) * (box1[3] - box1[1])
    box2_area = (box2[2] - box2[0]) * (box2[3] - box2[1])

    union_area = box1_area + box2_area - inter_area
    return inter_area / union_area if union_area > 0 else 0

def read_detections(file_path):
    """Зчитування детекцій із файлу detections.txt."""
    detections = []
    with open(file_path, "r") as f:
        reader = csv.reader(f)
        next(reader)  # Пропустити заголовок
        for row in reader:
            if len(row) < 7:  # Пропустити рядки без даних
                continue
            index = int(row[0])
            class_id = int(row[1])
            probability = float(row[2])
            xmin, ymin, xmax, ymax = map(float, row[3:7])
            detections.append({"index": index, "class": class_id, "probability": probability, "box": (xmin, ymin, xmax, ymax)})
    return detections
```

```
    def compare_detections(file1, file2, iou_threshold=0.5,
prob_threshold=0.1, delta=10):
        """Порівняння детекцій між двома файлами з урахуванням похибки в
координатах."""
        detections1 = read_detections(file1)
        detections2 = read_detections(file2)

        matches = []
        for det1 in detections1:
            for det2 in detections2:
                iou_score = iou(det1["box"], det2["box"], delta)
                prob_diff = abs(det1["probability"] - det2["probability"])
                if det1["class"] == det2["class"] and iou_score >
iou_threshold and prob_diff <= prob_threshold:
                    matches.append({
                        "file1_index": det1["index"],
                        "file2_index": det2["index"],
                        "class": det1["class"],
                        "iou": iou_score,
                        "prob_diff": prob_diff
                    })

        return matches

    def save_matches_to_file(matches, output_file):
        """Зберігає результати збігів у файл."""
        with open(output_file, "w", newline="") as f:
            writer = csv.writer(f)
            # Заголовок файлу
            writer.writerow(["File1_Index", "File2_Index", "Class", "IoU",
"Probability_Difference"])
            # Запис результатів
            for match in matches:
                writer.writerow([
                    match["file1_index"],
                    match["file2_index"],
                    match["class"],
                    f"{match['iou']:.2f}",
```

```
                    f"{match['prob_diff']:.2f}"
                ])

    """
    # Шляхи до файлів
    file1 = "detectionsYolo8File0.txt"
    file2 = "detectionsYolo8mFile0.txt"
    output_file = "detectionsYolo8AndYolo8mFile0Out.csv"

    """

    # Шляхи до файлів
    file1 = "detectionsYolo8File4.txt"
    file2 = "detectionsYolo8mFile4.txt"
    output_file = "detectionsYolo8AndYolo8mFile4Out.csv"

    # Порівняння
    matches = compare_detections(file1, file2, delta=10)

    # Вивід результатів
    print(f"Знайдено {len(matches)} збігів:")
    for match in matches:
        print(f"File1 Index: {match['file1_index']}, File2 Index:
{match['file2_index']}, "
              f"Class: {match['class']}, IoU: {match['iou']:.2f}, Prob
Diff: {match['prob_diff']:.2f}")

    # Збереження результатів у файл
    save_matches_to_file(matches, output_file)
    print(f"Результати збережено у файл: {output_file}")
```