Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра обчислювальної техніки

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Комп'ютерна система розпізнавання дорожньої розмітки та виявлення пошкоджень дорожнього покриття»

08-54.БДР.020.00.000 ПЗ

Виконав: студент 4 курсу, групи 1СП-21Б
спеціальності 123 – «Комп'ютерна інженерія»
освітня програма – «Системне програмування»
_____ Шелестун І.С.
(прізвище та ініціали)
Керівник: к.т.н. доц. каф. ОТ
_____ Кожем'яко А. В.
(прізвище та ініціали)
Рецензент: к.т.н. доц. каф. ПЗ
_____ Майданюк В.П.
(прізвище та ініціали)

Допущено до захисту
Завідувач кафедри ОТ
д.т.н., проф. Азаров О. Д.
«19» 06 2025 р.

Вінниця ВНТУ – 2025 рік

Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра обчислювальної техніки

Освітньо-кваліфікаційний рівень бакалавр

Галузь знань 12 – Інформаційні технології

Спеціальність 123 – «Комп'ютерна інженерія»

Освітня програма «Комп'ютерна інженерія»

ЗАТВЕРДЖУЮ
Завідувач кафедри ОТ проф., д.т.н.
_____ Азаров О. Д.
«21» березня 2025 року

ЗАВДАННЯ

НА БАКАЛАВРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Шелестуну Іллі Сергійовичу

1 Тема роботи : Комп'ютерна система розпізнавання дорожньої розмітки та виявлення пошкоджень дорожнього покриття, керівник роботи Кожем'яко Андрій Вікторович, к.т.н., доцент кафедри ОТ, затверджено наказом вищого навчального закладу від «20» березня 2025 року №97.

2 Строк подання студентом роботи 13.05.2025

3 Вихідні дані до роботи: створення програми для розпізнавання дорожньої розмітки та виявлення пошкоджень дорожнього покриття з використанням мови програмування Python, бібліотек TensorFlow та OpenCV, а також платформ Roboflow і Ultralytics.

4 Зміст розрахунково-пояснювальної записки: вступ, теоретичні основи нейронних мереж для розпізнавання зображень, тестування моделей та програмного засобу, вибір технологій та процес тренування моделей, опис реалізація програмної частини, функціональний аналіз програмного продукту, висновки, перелік використаних джерел, додатки.

5 Перелік графічного матеріалу : блок – схема головного алгоритму обробк інструкцій, структурна схема програми, скріншоти з результатами роботи програми.

6 Консультанти розділу роботи приведені в таблиці 1.

Таблиця 1 – Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1-4	к.т.н., доц. каф. ОТ Кожем'яко. В.	21.03.25	13.06.25
Нормоконтроль	ас. каф. ОТ Швець С. І.	14.05.2025	30.05.2025

7 Дата видачі завдання 21.03.2025 р.

8 Календарний план кваліфікаційної роботи приведений в таблиці 2

Таблиця 2 – Календарний план

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання	Підпис
1	Постановка задачі розробки системи розпізнавання дорожньої розмітки та виявлення пошкоджень дорожнього покриття	21.03.25	
2	Аналіз літературних джерел. Попередня розробка основних розділів	21.03.25— 30.03.25	
3	Аналіз технологій для тренування нейронних моделей	21.03.25— 30.03.25	
4	Розробка програмного засобу	31.03.25— 13.04.25	
5	Тестування реалізованого застосунку	28.04.25— 11.05.25	
6	Оформлення пояснювальної записки та додатків	13.05.25	
7	Перевірка якості виконання бакалаврської кваліфікаційної роботи та усунення недоліків	13.05.25	

Студент _____ Шелестун І.С.

Керівник роботи _____ Кожем'яко А.В.

АНОТАЦІЯ

УДК 004.20

Шелестун І.С. Система розпізнавання дорожньої розмітки та виявлення пошкоджень дорожнього покриття. Бакалаврська кваліфікаційна робота зі спеціальності 123 – Комп'ютерна інженерія, освітня програма – Системне програмування. Вінниця: ВНТУ, 2025, 86 с.

На укр. мові. Бібліогр.: 26 назв; рис.: 40; табл. 6.

У бакалаврській кваліфікаційній роботі розроблено систему для розпізнавання дорожньої розмітки та виявлення пошкоджень дорожнього покриття із застосуванням згорткових нейронних мереж. Виконано аналіз сучасних архітектур штучних нейронних мереж для комп'ютерного зору, зокрема CNN, YOLO, R-CNN, Fast R-CNN, Faster R-CNN, RetinaNet, SSD та EfficientDet. Створено програмне забезпечення з використанням мови програмування Python та бібліотек TensorFlow, OpenCV, а також платформ Roboflow і Ultralytics. Виконано підготовку та аугментацію наборів даних, навчання моделей та оцінювання їхньої ефективності. Проведено функціонально-вартісний та техніко-економічний аналіз варіантів реалізації системи. Результати тестування підтверджують доцільність використання розробленого програмного забезпечення для моніторингу транспортної інфраструктури.

Ключові слова: розпізнавання об'єктів, нейронні мережі, згорткові нейронні мережі, YOLO, R-CNN, TensorFlow, OpenCV, CVAT, Roboflow, Ultralytics, дорожня розмітка, дефекти дорожнього покриття, комп'ютерний зір.

ABSTRACT

Shelestun I.S. Road marking recognition system and road surface damage detection. Bachelor's thesis in the specialty 123 – Computer Engineering, educational program – System Programming. Vinnytsia: VNTU, 2025, 86 p.

In Ukrainian. Bibliography: 26 titles; Fig.: 40; Table 6.

In the bachelor's thesis, a system for road marking recognition and road surface damage detection using convolutional neural networks was developed. An analysis of modern architectures of artificial neural networks for computer vision was performed, in particular CNN, YOLO, R-CNN, Fast R-CNN, Faster R-CNN, RetinaNet, SSD and EfficientDet. Software was created using the Python programming language and the TensorFlow, OpenCV, as well as the Roboflow and Ultralytics platforms.

Data sets were prepared and augmented, models were trained and their effectiveness was assessed. Functional-cost and technical-economic analysis of system implementation options was conducted. Testing results confirm the feasibility of using the developed software for monitoring transport infrastructure.

Keywords: object recognition, neural networks, convolutional neural networks, YOLO, R-CNN, TensorFlow, OpenCV, CVAT, Roboflow, Ultralytics, road markings, road surface defects, computer vision.

ЗМІСТ

ВСТУП

У сучасному світі технологічний прогрес істотно впливає на різні сфери людської діяльності, зокрема на автоматизацію процесів у транспортній галузі, промисловості та міському господарстві. Одним із важливих напрямків розвитку є створення систем, що здатні автоматично аналізувати навколишнє середовище за допомогою методів комп’ютерного зору та машинного навчання.

Розпізнавання об’єктів — це технологія, яка дає змогу комп’ютерам та іншим пристроям самостійно виявляти та класифікувати об’єкти на основі зображень чи відеопотоку. Особливу актуальність ця технологія має у транспортних системах, де вона використовується для розпізнавання дорожньої розмітки та виявлення пошкоджень дорожнього покриття, що сприяє підвищенню безпеки руху та ефективності обслуговування інфраструктури.

Важливу роль у розвитку систем комп'ютерного зору відіграє використання нейронних мереж, особливо методів глибокого навчання, які дозволяють обробляти великі обсяги даних і виявляти складні закономірності. Завдяки цьому стало можливим створення високоточних систем аналізу стану дорожньої інфраструктури, придатних для реального застосування.

Таким чином, дана бакалаврська робота спрямована на подальший розвиток технологій комп’ютерного зору для транспортної галузі, що сприятиме підвищенню безпеки дорожнього руху та якості дорожнього покриття.

Об'єкт дослідження — процес розпізнавання дорожньої розмітки та виявлення пошкоджень дорожнього покриття.

Предмет дослідження — методи та технології використання нейронних мереж для автоматизованого розпізнавання дорожньої розмітки та виявлення пошкоджень покриття.

Метою роботи є розробка та вдосконалення комп’ютерної системи для розпізнавання дорожньої розмітки та виявлення пошкоджень з використанням сучасних нейронних мереж і засобів візуалізації процесу навчання.

Задачі дослідження бакалаврської роботи:

— проаналізувати сучасні методи та алгоритми розпізнавання дорожньої розмітки та пошкоджень дорожнього покриття з використанням нейронних мереж;

— обґрунтувати вибір технологій та архітектур моделей на основі їх переваг і недоліків;

— визначити та підготувати набір даних для навчання і тестування моделей;

— здійснити навчання нейронних мереж для розпізнавання дорожньої розмітки та пошкоджень;

— розробити програмне забезпечення для інтеграції моделей у комп'ютерну систему з використанням архітектури YOLO із використанням бібліотек TensorFlow, OpenCV та платформ Roboflow і Ultralytics;

— провести тестування створеної системи в умовах, наближених до реальних.

Публікації за темою роботи: За результатами роботи опубліковані тези доповіді: Богомолов, С., Кожем'яко, А., & Шелестун І. (2025). Система розпізнавання дорожньої розмітки та виявлення пошкоджень дорожнього покриття в ВНТКП ВНТУ . Факультет інформаційних технологій та комп'ютерної інженерії. Отримано з : https://conferences.vntu.edu.ua/index.php/all-fitki/all-fitki-2025/paper/view/24198

1 ТЕОРЕТИЧНІ ОСНОВИ НЕЙРОННИХ МЕРЕЖ ДЛЯ РОЗПІЗНАВАННЯ ЗОБРАЖЕНЬ

1.1 Штучні нейронні мережі

Штучні нейронні мережі за останні десятиліття стали об’єктом активного дослідження та знайшли широке застосування у різноманітних сферах. Вони представляють собою математичні моделі, розроблені на основі принципів, що імітують функціонування біологічних нейронних мереж людського мозку. Ці системи складаються з численних взаємопов’язаних елементів – нейронів (рис.1.1), які координаційно працюють, щоб вирішувати складні завдання [1]. Завдяки здатності до навчання та адаптації, нейронні мережі успішно застосовуються для вирішення таких задач, як розпізнавання образів, прогнозування, класифікація та багато інших.

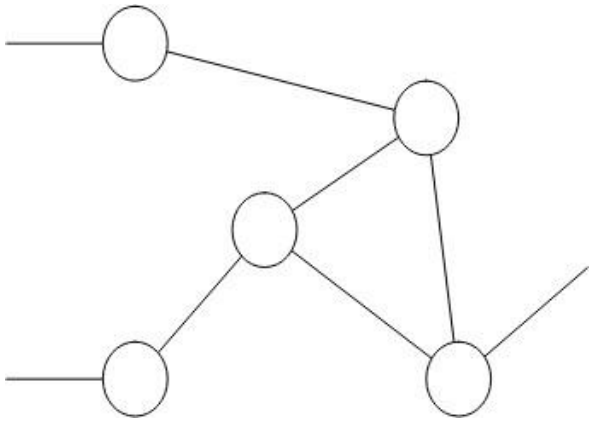

Рисунок 1.1 — Приклад ШНМ з довільною структурою

Нейрон отримує вхідні значення, виконує їх зважене сумування, а потім передає результат через обрану функцію активації. Вихід нейрона може бути поданий як вхід до наступного шару або вважатися кінцевим результатом моделі залежно від її структури та завдань.

Однією з найпоширеніших архітектур штучних нейронних мереж є багатошаровий перцептрон (MLP). Він складається з вхідного шару, одного або кількох прихованих шарів і вихідного шару нейронів. У цій архітектурі кожен нейрон одного шару з'єднаний із усіма нейронами наступного.

MLP активно використовуються для виконання задач класифікації та регресії, знаходячи застосування, наприклад, у розпізнаванні образів та прогнозуванні даних.

Згорткові нейронні мережі (CNN) показують високу ефективність у роботі з зображеннями та відео. Вони включають згорткові шари, завдяки яким виділяються ключові особливості вхідних даних, такі як контури, текстури та інші деталі. Це робить CNN незамінними для таких сфер, як розпізнавання облич, аналіз медичних зображень і автономне керування транспортними засобами.

Рекурентні нейронні мережі (RNN) використовуються для роботи з послідовними даними, такими як текст або часові ряди. Завдяки здатності зберігати інформацію про попередні кроки обчислення, вони ефективно застосовуються у задачах прогнозування, аналізу часових рядів і обробки природної мови. Наприклад, у машинному перекладі RNN враховують контекст попередніх слів, що дозволяє досягти більш точного результату.

Існують також інші архітектури штучних нейронних мереж, розроблені для специфічних завдань. Зокрема, довготривала короткочасна пам'ять (LSTM) та мережі з ґейтованими рекурентними блоками (GRU) є покращеними варіантами рекурентних мереж (RNN), які ефективніше зберігають інформацію протягом тривалих часових інтервалів. Автокодери застосовуються для зменшення розмірності даних і вилучення ключових характеристик, тоді як генеративно-змагальні мережі (GAN) здатні генерувати нові, реалістичні дані, такі як зображення чи звуки.

1.2 Згорткові нейронні мережі

Згорткові нейронні мережі (CNN) [2,3] стали одними з найпоширеніших і найефективніших архітектур для аналізу зображень та інших структурованих даних. Такий підхід автоматично виділяє релевантні ознаки з вхідних даних, що значно підвищує продуктивність моделей у задачах комп'ютерного зору, як-от класифікація, виявлення чи сегментація об'єктів. У цьому розділі детальніше розглядаються ключові компоненти, архітектура та функції активації CNN.

Загальна структура згорткової нейронної мережі, призначеної для виконання задачі класифікації, наведена на рисунку 1.2

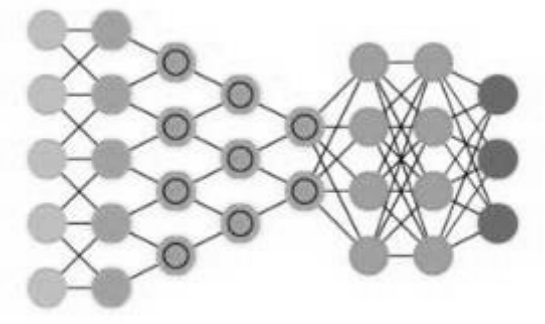

Рисунок 1.2 — Схематичне представлення згорткових нейронних мереж

До головних компонентів згорткової нейронної мережі (Convolutional Neural Network, CNN) входять: згортковий шар, підсумковий (або об'єднувальний) шар та повнозв’язний шар.

Згортковий шар є ключовим елементом CNN, який здійснює операцію згортки над вхідними даними (рис. 1.3). Завдяки цій операції стає можливим витягування локальних патернів із зображення, що особливо важливо для ідентифікації та розпізнавання структурних елементів. Згортковий набір складається з набору фільтрів із вагами, які навчаються для аналізу різних ділянок зображення. У випадку двовимірних даних, таких як зображення, згортка описується за певною формулою (1.1).

$$(F * K)(i, j) = \sum_m \sum_n F(i - m, j - n) \cdot K(m, n) \quad (1.1)$$

де F — вхідне зображення,

K — фільтр.

Підсумкові шари (pooling layers) використовуються для зменшення розмірності даних, зберігаючи при цьому ключову інформацію. Це здійснюється шляхом об’єднання (підсумовування) значень у визначених областях. Найпоширенішим типом таких шарів є максимальне підсумовування (max pooling), яке вибирає найбільше значення у кожному вікні (формула 1.2).

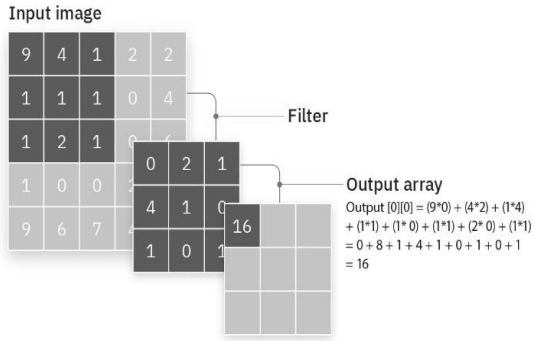

Рисунок 1.3 — Застосування згортки одного фільтру до сегменту зображення

$$y[i, j] = \max\{x[m, n] | (m, n) \in Region(i, j)\} \quad (1.2)$$

де Region (i, j) – область, з якої береться максимальне значення.

Повнозв'язні шари (FC) застосовуються на фінальному етапі згорткових нейронних мереж (CNN) для класифікації отриманих ознак. Вихід попереднього шару перетворюється у вектор, який подається на вхід FC-шару. Цей шар виконує обчислення лінійної комбінації вхідних даних відповідно до формули (1.3).

$$y = Wx + b, \quad (1.3)$$

де W – матриця ваги,

x – вхідні дані,

b – вектор зміщень.

Стандартна архітектура згорткової нейронної мережі (CNN) включає послідовне чергування згорткових і підсумкових (пулінгових) шарів, після яких йдуть один або кілька повнозв’язних шарів. Розгляньмо детальніше основні компоненти типової архітектури CNN.

Згорткові шари відіграють ключову роль завдяки використанню ядра (фільтра), яке переміщується по зображенню. Ядро обчислює зважену суму пікселів у межах свого вікна, формуючи карту ознак (feature map), яка містить важливу інформацію про просторові характеристики зображення.

Підсумкові шари виконують функцію зменшення розміру карт ознак. Водночас вони зберігають базову інформацію, необхідну для подальшої обробки.

Наприклад, max pooling із вікном розміром 2×2 і кроком 2 зменшує лінійні розміри карти ознак удвічі. Це спрощує обчислення, скорочує витрати ресурсів і запобігає перенавчанню, залишаючи тільки найбільш суттєві риси зображення.

На фінальному етапі CNN застосовуються повнозв'язні (Fully Connected, FC) шари для остаточної класифікації. Вихід останнього згорткового або підсумкового шару трансформується у вектор, який подається на вхід до FC-шару. Наприклад, якщо потрібно класифікувати об'єкт на 10 класів, вихід останнього FC-шару буде представлений вектором із 10 елементів. Кожен елемент такого вектора вкаже ймовірність належності до відповідного класу.

Для забезпечення нелінійності між шарами застосовуються функції активації. Найпоширенішою з них у згорткових нейронних мережах (CNN) є ReLU (1.4), оскільки вона додає моделі нелінійність і допомагає виявляти більш складні шаблони. Інші функції активації, такі як sigmoid (1.5) або tanh (1.6), також можуть використовуватися, проте ReLU здобула популярність завдяки своїй ефективності у вирішенні проблеми зникнення градієнта [4].

$$f(x) = \max(0, x), \tag{1.4}$$

де x — вхідне значення (активація нейрона до застосування функції),

f(x) — вихідне значення після активації,

max (0,x) означає, що якщо x більше нуля, він залишається незмінним, а якщо xx менше або дорівнює нулю, вихід буде нульовим.

$$f(x) = \frac{1}{1+e^{-x}}, \tag{1.5}$$

де x — вхідне значення (сигнал нейрона),

e— експоненційна функція (приблизно 2.718),

f(x) — вихід, який буде в діапазоні (0,1), оскільки ця функція стискає значення до цього інтервалу.

$$f(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}. \quad (1.6)$$

де x — вхідне значення,

e^{x} та e^{-x} — експоненційні вирази,

f(x) — вихід, який лежить у діапазоні (-1,1), що дозволяє симетричне відображення негативних і позитивних значень.

Для підвищення ефективності роботи згорткових нейронних мереж (CNN) [3] застосовують різноманітні методи та техніки:

1) регуляризація, яка спрямована на запобігання перенавчанню моделі та має основні підходи до регуляризації;

— Dropout — випадкове вимикання нейронів під час навчання, що знижує ризик перенавчання;

— L2-регуляризацію — додавання штрафу за надмірно великі ваги в функцію втрат, що сприяє їх нормалізації.

2) аугментація даних, яка передбачає створення нових тренувальних прикладів шляхом модифікації наявних даних, наприклад, за допомогою обертання, зсуву, зміни яскравості тощо, це дозволяє моделі краще адаптуватися до різноманітності вхідних даних.

Згорткові нейронні мережі представляють собою потужний інструмент для обробки зображень та інших типів структурованих даних, значною мірою завдяки їх здатності автоматично виділяти важливі ознаки з вхідних даних. Завдяки використанню згорткових і підсумкових шарів вдається зменшувати розмірність даних, одночасно зберігаючи ключову інформацію. Це робить CNN надзвичайно ефективними для вирішення широкого круга завдань, включно з класифікацією зображень, детекцією об'єктів та сегментацією.

1.3 Метрики ефективності моделей

Є кілька ключових метрик, що застосовуються для оцінювання продуктивності моделей згорткових нейронних мереж (CNN). Серед найважливіших виділяють частоту кадрів за секунду (FPS) і рівень точності. Окрім цього, для оцінки обчислювальних ресурсів враховують такі параметри, як число операцій з плаваючою комою за секунду (FLOPs) та обсяги параметрів моделі [6]

Кількість параметрів моделі визначає обсяг змінних, які потрібно налаштувати під час її навчання. До цих параметрів належать ваги та зсуви, що використовуються в нейронних мережах. Рівень складності моделі напряму залежить від кількості параметрів, яка також є показником об’єму пам’яті, необхідного для її зберігання. Чим більше параметрів у моделі, тим більше пам’яті потрібно для її навчання та збереження.

Показник FLOPs характеризує обчислювальну складність моделі, демонструючи кількість операцій з плаваючою точкою, що виконуються за секунду. Це важливий критерій, який дозволяє порівнювати ефективність різних моделей. Чим вищий показник FLOPs, тим складнішою є модель і тим більше обчислювальних ресурсів вона потребує для своєї роботи.

FPS відображає кількість кадрів, які модель може обробляти за секунду під час виконання завдань, наприклад, виявлення об’єктів на зображеннях. Цей параметр критично важливий у системах реального часу, таких як виявлення транспортних засобів на дорозі. Для забезпечення стабільної роботи системи оптимальний рівень FPS має становити 35-40 кадрів на секунду, що забезпечує достатню плавність відтворення відео. Досягнення ще вищого показника FPS є бажаним, однак це потребує оптимізації моделі та використання більш продуктивного обладнання.

Під час вибору і налаштування моделей CNN необхідно враховувати кілька важливих метрик, таких як точність, FPS, FLOPs і кількість параметрів моделі. Ці показники дозволяють оцінити продуктивність моделі та її вимоги до обчислювальних ресурсів. Аналіз цих параметрів допомагає досягти оптимального

співвідношення між точністю та продуктивністю, що сприяє ефективному застосуванню моделей у практичних завданнях.

Однією з ключових метрик для оцінки точності розпізнавання є середня точність, або mAP. Цей показник визначається як усереднене значення точності розпізнавання (AP) для всіх класів, які може ідентифікувати модель. Точність для кожного окремого класу (AP) обчислюється на основі коректності роботи моделі стосовно відповідного класу (1.7).

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i, \tag{1.7}$$

де N – загальна кількість класів, які розпізнаються моделлю,

AP_i – показник AP для класу i.

Показник mAP є ключовим інструментом для оцінки ефективності моделей розпізнавання об'єктів, оскільки він враховує точність визначення для всіх класів, присутніх у наборі даних. Це дозволяє отримати більш збалансоване та об'єктивне уявлення про результати роботи моделі, на відміну від оцінок, які базуються лише на точності для окремих класів. Для розрахунку Average Precision (AP) необхідно використовувати такі метрики, як IoU, precision, recall та PR-крива.

IoU [8] є метрикою, що дозволяє оцінити точність об'єктного детектора на заданому наборі даних. Вона вимірює співвідношення площі перетину між передбаченим обмежуючим прямокутником і фактичним прямокутником до площі їх об'єднання (1.8).

$$IoU = \frac{B_1 \cap B_2}{B_1 \cup B_2}, \tag{1.8}$$

де B_1, B_2 — відповідно передбачуваний і фактичний обмежуючий прямокутник.

Значення IoU коливається в межах від 0 до 1: 0 свідчить про відсутність збігу, тоді як 1 означає повний збіг між передбаченою та реальною областями. Зазвичай для оцінки точності моделі використовується поріг α = 0.5 (рис. 1.4).

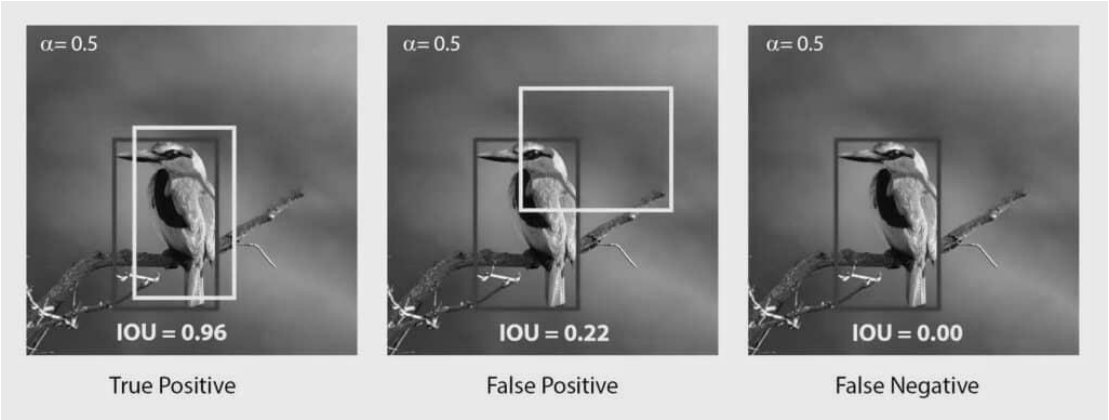

Рисунок 1.4 — Точність детекції об’єктів на основі IoU

Якщо показник IoU перевищує заданий поріг, прогноз вважається істинно позитивним (True Positive, TP). Якщо ж IoU менше порога або передбачено невірний клас, прогноз розглядається як хибно позитивний (False Positive, FP). У разі, коли модель не генерує прогноз для конкретного об'єкта, це класифікується як хибно негативний (False Negative, FN). Значення TP, FP і FN використовуються для розрахунку метрик точності та повноти згідно зі стандартними формулами, при цьому TN (True Negative) не враховується. Точність і повнота є базисом для побудови PR-кривої, яка наочно ілюструє здатність моделі до якісної класифікації позитивних випадків із одночасним дотриманням високої точності.

Для обчислення AP визначають площу під PR-кривою для кожного класу[7]. Приклад такої кривої наведено на рисунку 1.5. Цей підхід застосовується в метриці PASCAL VOC 2010. Існує кілька способів розрахунку AP, які відрізняються методологіями та використовують специфічні набори даних для оцінки ефективності моделей. Ось основні набори даних, які застосовуються для оцінювання якості моделей:

— PASCAL VOC 2005 використовував площу під ROC-кривою замість PR-кривої;

— PASCAL VOC 2007 впровадив одинадцятиточкову інтерполяцію для мінімізації впливу варіацій PR-кривої на результат;

— MS COCO використовує 101-бальну інтерполяційну версію AP.

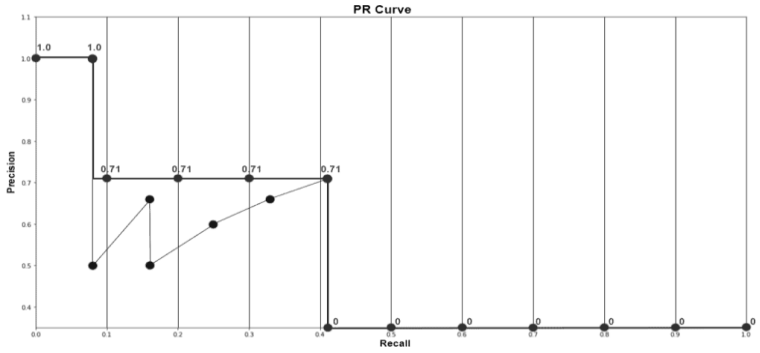

Рисунок 1.5 — Приклад PR-кривої та методу точкової інтерполяції

1.4 Моделі розпізнавання об’єктів

Існує безліч моделей, призначених для розпізнавання об’єктів у відео, причому кожна з них має свої особливості та підходи до вирішення завдань комп’ютерного зору. Розглянемо кілька ключових моделей.

YOLO (You Only Look Once) [9]:

— модель використовує єдину конволюційну нейронну мережу (CNN), яка здатна прогнозувати обмежувальні рамки та ймовірності класів для кількох об'єктів одночасно за один прохід через мережу;

— висока швидкість обробки є однією з головних переваг YOLO, оскільки вона аналізує все зображення як цілісну одиницю, що дозволяє працювати в режимі реального часу;

— хоча модель може поступатися окремим рішенням у точності розпізнавання дрібних об'єктів або складних сцен, її продуктивність залишається достатньо високою для більшості прикладних завдань.

Faster R-CNN (Region-based Convolutional Neural Networks) [10]:

— архітектура включає два ключові компоненти: регіональну пропозиційну мережу (RPN), яка генерує регіональні пропозиції, та мережу для класифікації і регресії обмежувальних рамок (рис. 1.6);

— Faster R-CNN забезпечує високу точність виявлення об'єктів завдяки ретельному аналізу кожної запропонованої області;

— в порівнянні з YOLO, Faster R-CNN працює повільніше, що може створювати труднощі для застосувань, яким необхідна обробка в режимі реального часу.

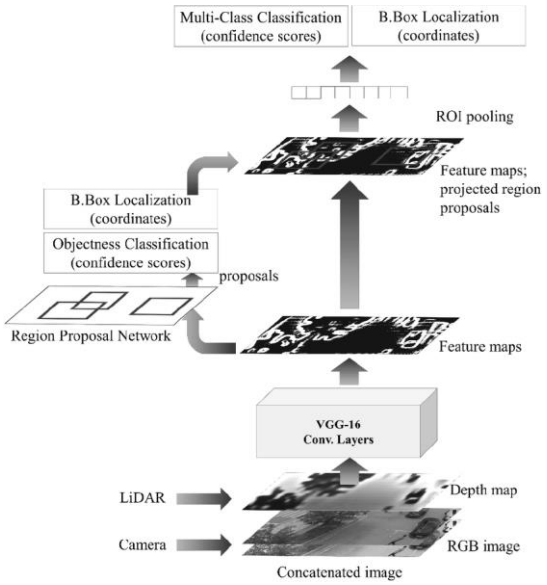

Рисунок 1.6 — Візуалізація архітектури R-CNN

SSD (Single-Shot Detector) [11] :

— архітектура використовує єдиний прохід через згорткову нейронну мережу (CNN) для створення кількох обмежувальних рамок та їх класифікації, що забезпечує оптимальний баланс між продуктивністю та точністю (рис. 1.7);

— SSD демонструє високу точність, особливо у випадках розпізнавання великих об'єктів;

— SSD працює швидше, ніж Faster R-CNN, однак може поступатися YOLO за умов обмежених обчислювальних можливостей.

RetinaNet [12]:

— архітектура (рис. 1.8) використовує Focal Loss для вирішення проблеми дисбалансу класів, що значно підвищує точність виявлення об'єктів, особливо рідкісних або дрібних;

— завдяки використанню складних методів навчання, RetinaNet демонструє високу ефективність;

— хоча швидкість обробки нижча порівняно з YOLO та SSD, модель залишається практичною для сценаріїв, де точність має вирішальне значення.

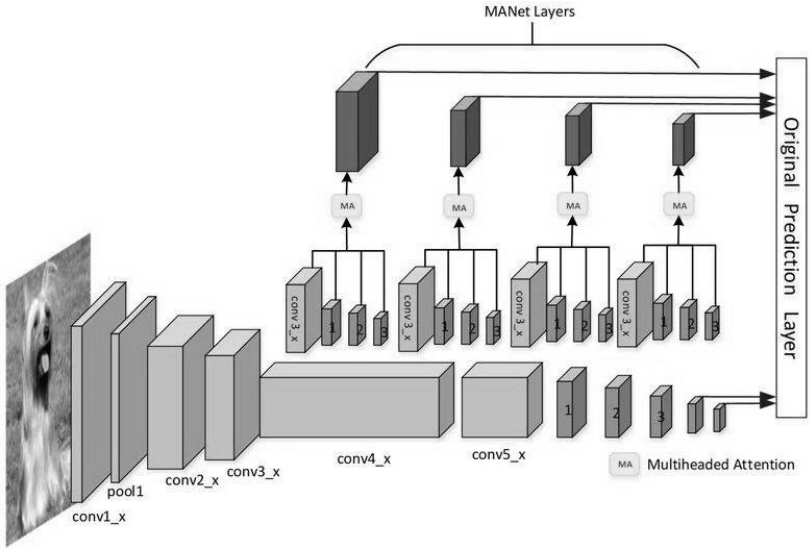

Рисунок 1.7 — Демонстрація архітектури SSD

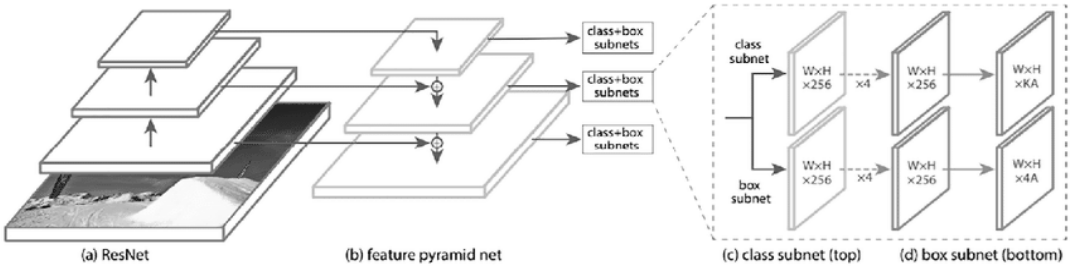

Рисунок 1.8 — Візуалізація архітектури RetinaNet

EfficientDet [13]:

— архітектура (рис. 1.9) базується на оптимізованій EfficientNet, яка виступає як основна мережа, забезпечуючи високу ефективність при мінімальних обчислювальних витратах;

— завдяки складним технікам масштабування EfficientDet досягає високої точності розпізнавання;

— модель демонструє високу швидкість обробки, що робить її придатною для використання на мобільних та вбудованих платформах.

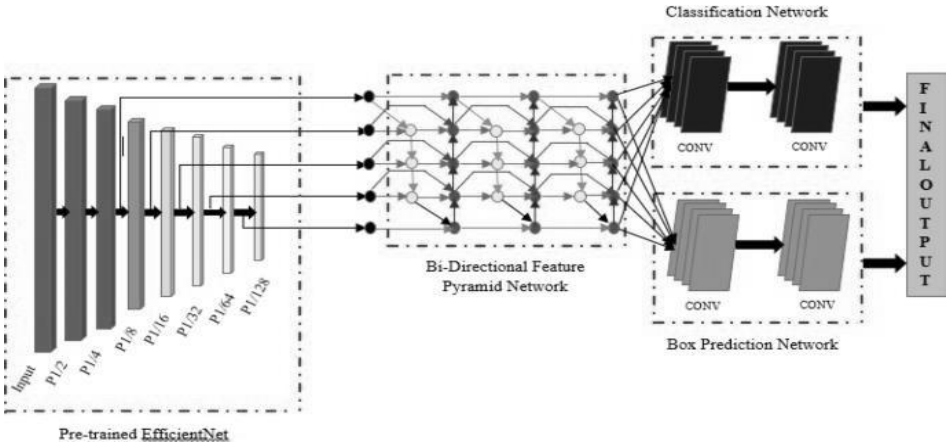

Рисунок 1.9 — Візуалізація архітектури EfficientDet

У першому розділі було розглянуто теоретичні основи нейронних мереж та їх застосування в комп'ютерному зорі. Починаючи з пояснення базових концепцій нейронних мереж, зокрема їх будови. Також було детально описано технологію згорткових нейронних мереж: застосування, будову, способи регуляризації для оптимізації навчання. Було описано постановку задачі розпізнавання, а також перелічено основні сучасні моделі для розпізнавання об’єктів.

Окрему увагу було приділено питанням навчання моделей на основі великих обсягів даних, методам покращення узагальнюючої здатності мереж, а також технікам боротьби з перенавчанням. Розглянуто переваги застосування глибокого навчання порівняно з класичними алгоритмами комп’ютерного зору, що дозволяє досягати високих показників точності навіть за умов складного фону, часткового перекриття об’єктів чи недостатньої якості зображень.

Таким чином, перший розділ надає ґрунтовне теоретичне підґрунтя для розуміння сучасних методів аналізу зображень за допомогою нейронних мереж, акцентуючи увагу на їх практичному застосуванні. Викладені концепції та методи є основою для розробки систем, які можуть автоматично і з високою точністю виявляти дефекти на дорожньому покритті та розпізнавати дорожню розмітку, що дозволяє комплексно оцінювати стан дорожньої інфраструктури. Здобуті теоретичні знання будуть використані в подальших етапах дослідження для обґрунтованого вибору моделі, побудови ефективного програмного рішення та досягнення високого рівня достовірності результатів розпізнавання.

2 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ АНАЛІЗУ ДОРОЖНЬОГО ПОКРИТТЯ ТА РОЗМІТКИ

2.1 Актуальність розробки системи автоматичного розпізнавання дорожньої розмітки та дефектів дорожнього покриття

Вирішальною причиною вибору цієї теми дипломної роботи є значний практичний потенціал розробки систем автоматичного розпізнавання дорожньої розмітки та виявлення дефектів дорожнього покриття. Така система здатна підвищити ефективність управління дорожньою інфраструктурою, оскільки дозволяє одночасно аналізувати як якість дорожнього полотна, так і стан дорожньої розмітки, що є важливими факторами безпеки дорожнього руху.

Впровадження технологій розпізнавання та класифікації стану дорожнього полотна й дорожньої розмітки допомагає ефективніше розподіляти ресурси, оскільки, на відміну від традиційних методів інспекцій, автоматичний підхід забезпечує оперативне, точне та комплексне виявлення як пошкоджених ділянок, так і зношених чи відсутніх елементів розмітки.

Автоматизація процесів розпізнавання дефектів і контролю якості розмітки дозволяє зменшити людський фактор, підвищуючи точність, швидкість та об’єктивність оцінки стану доріг. Це сприяє своєчасному реагуванню на проблеми, оптимізації витрат на ремонт, а також зниженню ризику дорожньо-транспортних пригод, спричинених як поганим станом покриття, так і відсутністю належної розмітки.

Впровадження такої системи дає можливість вирішувати кілька важливих завдань одночасно:

— автоматизоване виявлення дефектів дорожнього покриття та недоліків дорожньої розмітки для підвищення безпеки дорожнього руху;

— контроль якості ремонтних робіт та відновлення розмітки;

— урахування стану дороги та розмітки при плануванні маршрутів логістичними компаніями.

Таким чином, напрямок дипломної роботи пов’язаний із створенням системи, яка забезпечує комплексне розпізнавання як дефектів дорожнього покриття, так і елементів дорожньої розмітки, що дозволяє застосовувати її для різних сценаріїв використання, описаних вище.

2.2 Огляд існуючих методів аналізу стану дорожнього покриття та дорожньої розмітки

Аналіз стану дорожнього покриття та дорожньої розмітки є важливими завданнями управління дорожньою інфраструктурою, що безпосередньо впливають на безпеку руху та ефективність експлуатації доріг. Існують різні методи, що застосовуються для виявлення дефектів дорожнього полотна та оцінки якості дорожньої розмітки. Умовно їх можна поділити на традиційні та автоматизовані методи. Далі розглянемо їхні особливості, приклади, переваги та недоліки.

2.2.1 Традиційні методи

Традиційні методи аналізу стану дорожнього покриття та оцінки розмітки включають візуальний огляд, застосування спеціалізованих транспортних засобів та лабораторні дослідження

Візуальний огляд — один з найстаріших та найпоширеніших методів контролю стану дорожнього полотна і розмітки [13]. Інспектори оглядають дорогу пішки або з транспортного засобу, фіксуючи дефекти, такі як тріщини, ями, вибоїни, стерту або пошкоджену розмітку (рис.2.1) Цей метод є простим та відносно дешевим, проте має суттєві обмеження: потребує значних людських ресурсів, піддається суб'єктивним оцінкам та не завжди дозволяє виявити приховані дефекти чи слабко помітну розмітку.

Використання спеціалізованих транспортних засобів — застосування дорожніх машин, обладнаних датчиками та приладами для вимірювання характеристик дороги та оцінки стану розмітки. Такі машини можуть вимірювати рівність покриття, фіксувати нерівності, записувати колір та яскравість розмітки.

Наприклад, профілографи допомагають точно визначити нерівності дорожнього полотна, а спеціальні камери можуть фіксувати стан дорожньої розмітки для подальшої оцінки [14].

Перелічимо основні переваги традиційних методів:

— простота і доступність;

— економічність;

— гнучкість.

Водночас варто врахувати недоліки традиційних методів:

— суб'єктивність;

— часозатратність;

— низька точність;

— небезпека.

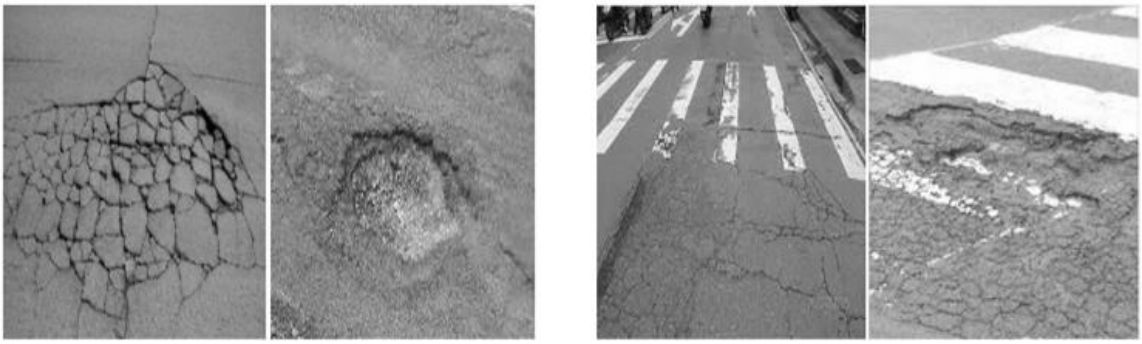

Рисунок 2.1 — Приклади пошкодженого дорожнього покриття та розмітки

2.2.2 Автоматизовані методи

З розвитком технологій з'явилися нові, автоматизовані методи аналізу стану дорожнього покриття та розпізнавання дорожньої розмітки. Вони дозволяють значно підвищити ефективність та точність виявлення дефектів і оцінки стану розмітки, мінімізуючи людський фактор.

Технології машинного зору та штучного інтелекту активно впроваджуються в аналіз стану дорожнього покриття та розмітки [14]. Камери, встановлені на транспортних засобах або дронах, здійснюють відеозйомку або фотографування дорожньої поверхні. Отримані дані обробляються за допомогою алгоритмів машинного навчання, які можуть автоматично розпізнавати дефекти покриття (тріщини, ями, вибоїни) та елементи дорожньої розмітки (лінії, пішохідні переходи, стрілки тощо). Це дає змогу не лише своєчасно виявляти пошкодження, а й контролювати стан розмітки, оцінюючи її зношеність та потребу в оновленні. Цей метод дозволяє швидко і точно аналізувати великі ділянки доріг, знижуючи витрати на інспекцію та обслуговування.

Використання лазерних сканерів (рис.2.2) є ще одним автоматизованим методом [15]. Лазерні сканери створюють тривимірні моделі дорожнього полотна, що дозволяє детально аналізувати як стан покриття, так і рельєф нанесеної розмітки. [16]. Вони здатні виявляти навіть незначні нерівності, деформації та втрати контрастності розмітки, що робить цей метод високоточним і ефективним. Проте, як і інші високотехнологічні рішення, лазерне сканування вимагає значних фінансових витрат на обладнання та його обслуговування.

Основні переваги автоматизованих методів:

— точність та об'єктивність;
— швидкість;
— безпека;
— детальність даних.

Основні недоліки автоматизованих методів:

— висока вартість;
— складність у використанні;
— технічні ризики;
— обмеження в адаптивності.

Рисунок 2.2 — Приклад використання лазерних сканерів

2.3 Технології комп’ютерного зору

Комп'ютерний зір є однією з ключових складових штучного інтелекту, що забезпечує автоматизацію аналізу та інтерпретації візуальної інформації. Головна його мета — надати комп'ютерам здатність «бачити» та взаємодіяти із візуальним середовищем: розпізнавати об'єкти, аналізувати сцени та приймати рішення на основі отриманих даних, подібно до того, як це робить людина.

Системи комп'ютерного зору вирішують широкий спектр завдань, серед яких основними є класифікація зображень, виявлення та локалізація об'єктів, сегментація, відстеження у відео, покращення якості зображень і навіть генерація нових. Для реалізації таких функцій активно використовуються сучасні методи машинного та глибокого навчання, зокрема згорткові нейронні мережі (CNN), рекурентні нейронні мережі (RNN, LSTM), трансформери, автокодувальники та генеративно-змагальні мережі (GAN).

Кожен напрямок комп'ютерного зору ґрунтується на спеціальних алгоритмах і підходах, які забезпечують оптимальну ефективність для конкретних завдань. Завдяки цьому технології комп'ютерного зору широко застосовуються у багатьох галузях, таких як автономний транспорт, системи безпеки, медицина, промислове виробництво, сільське господарство, а також у моніторингу інфраструктури.

Для кожного ключового напрямку комп’ютерного зору існують відповідні методи їх реалізації, що розглядаються нижче:

— класифікація зображень полягає у віднесенні зображення до певного класу за допомогою моделей, тренованих на маркованих даних, таких як CNN або Vision Transformer [17];

— детекція об'єктів передбачає одночасну класифікацію та локалізацію об'єктів на зображенні за допомогою моделей на кшталт R-CNN, YOLO або SSD[18];

— сегментація зображень полягає в класифікації кожного пікселя зображення, що дозволяє точно виділяти об'єкти навіть із подібними характеристиками [19];

— відстеження об'єктів забезпечує визначення положення об'єктів на кожному кадрі відео з урахуванням їх попередніх позицій, використовуючи Kalman Filters, RNN або LSTM [20];

— відновлення зображень спрямоване на покращення якості або реконструкцію пошкоджених зображень за допомогою автокодувальників чи генеративних моделей, таких як GAN;

— генерація зображень передбачає створення нових зображень на основі навчання, з використанням моделей GAN або VAE для досягнення високої якості згенерованих результатів [21, 22].

Нижче описано методи та їх реалізації, які відносяться до комп’ютерного зору:

— метод детекції та класифікації зображень (рис. 2.3) полягає у визначенні належності зображення до певного класу;

— метод сегментації (рис. 2.4) передбачає визначення класу кожного пікселя, що дає змогу точно виділяти об'єкти, навіть якщо вони мають подібні характеристики;

— метод відстеження об'єктів (рис. 2.5) дозволяє визначати координати об'єктів на відео з урахуванням їхнього руху в часі;

— метод відновлення зображень (рис. 2.6) має на меті підвищення якості або реконструкцію пошкоджених зображень;

— метод генерації зображень (рис. 2.7) передбачає створення нових візуальних зразків на основі навчання моделі.

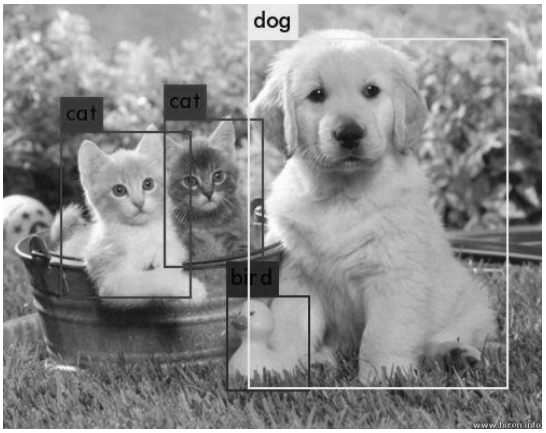

Рисунок 2.3 — Приклад детекції та класифікації зображень

Рисунок 2.4 — Приклад сегментації зображень

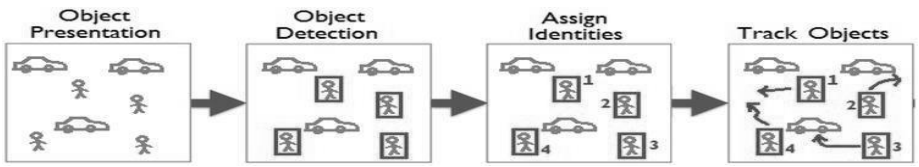

Рисунок 2.5 — Ілюстрація роботи відстеження об’єктів за рахунок знаходження об’єктів в сцені та положення в наступному кадрі

Рисунок 2.6 — Демонстрація відновленого зображення за допомогою GAN

Рисунок 2.7 — Фотографії людей та згенеровані зображення

2.4 Розпізнавання зображень

Основним завданням аналізу кадру з відео є визначення класів ознак зображення, які його описують. До таких класів належать:

— геометричні ознаки, що характеризують форму, розміри та пропорції об'єкта;

— текстурні ознаки, що відображають властивості поверхні,

наприклад, шорсткість або гладкість;

— колірні ознаки, які визначають колірні особливості об'єкта чи його фрагментів.

Для кожного класу ознак існують певні методи їх визначення, що наведені нижче.

Методи, базовані на зовнішньому вигляді. Метод на основі частин передбачає поділ об'єкта на окремі складові, які аналізуються індивідуально. Наприклад, транспортний засіб можна розбити на передню, задню та бокові частини. Метод на основі ознак використовує характерні параметри об'єкта, такі як форма та колір, для його унікальної характеристики.

Методи, залежні від руху. Метод різниці кадрів визначає зміни між послідовними кадрами для ідентифікації рухомих об'єктів. Відмінності у пікселях сигналізують про присутність руху. Метод віднімання фону спрямований на розділення фонових і передніх об'єктів шляхом усереднення ряду кадрів для створення стабільного фону. Він добре працює за умови статичного оточення, але може давати похибки в разі динамічних змін фону. Нейронні мережі, особливо згорткові та їх модифікації, продемонстрували високу ефективність у задачах розпізнавання зображень. Вони здатні обробляти великі масиви даних і забезпечувати високу точність результатів, що робить їх оптимальним вибором для систем реального часу. Однак варто зазначити, що такі методи вимагають значних обчислювальних ресурсів.

2.5 Проблематика обраної теми

Автоматизація процесу розпізнавання дефектів дорожнього покриття та оцінки стану дорожньої розмітки є важливим і актуальним напрямом, але водночас супроводжується низкою проблем і викликів, які необхідно враховувати при розробці та впровадженні таких технологій.

По-перше, існує різноманіття типів дорожніх покриттів і розмітки, кожна з яких має свої унікальні особливості, типові проблеми та характерні способи

нанесення. Матеріали, такі як асфальт, бетон, гравій, а також фарби й інші матеріали для розмітки, потребують різних підходів до аналізу.

Це ускладнює розробку універсальної системи, здатної ефективно працювати за різних умов і для різних видів доріг та розмітки.

По-друге, значним викликом є інтеграція автоматизованих систем у сферу управління дорожньою інфраструктурою, що є особливо складним у реаліях високого рівня корупції. Корупційні ризики можуть перешкоджати впровадженню таких інновацій через загрозу інтересам груп, які заробляють на застарілих та ручних процесах контролю доріг і розмітки.

Важливою задачею також є визначення найефективніших підходів і технологій побудови системи розпізнавання дефектів і оцінки якості дорожньої розмітки. Сучасні рішення пропонують широкий вибір методів комп'ютерного зору, таких як згорткові нейронні мережі (ResNet, VGG), детектори об’єктів (YOLO, SSD), моделі сегментації (U-Net, Mask R-CNN) і системи для виявлення аномалій (автоенкодери, GAN). Кожна архітектура підходить для окремих завдань: виявлення тріщин, ям, нерівностей, стертої чи пошкодженої розмітки тощо.

Ще одним важливим аспектом є збирання й обробка даних для навчання системи. Необхідно створити великий датасет, що охоплюватиме різні види дефектів дорожнього покриття та всі можливі варіації станів розмітки. Це потребує значних зусиль як від дослідників, так і від дорожніх служб, а також забезпечення точного маркування цих даних задля досягнення максимальної ефективності системи.

За нинішніх умов війни оптимізація бюджетних витрат набуває ще більшого значення. Впровадження автоматизованої системи дозволить суттєво зменшити витрати на обслуговування доріг завдяки своєчасному й точному виявленню проблемних ділянок. Це сприятиме ефективнішому плануванню ремонтних робіт, оперативному оновленню розмітки й оптимізації ресурсів.

Попри всі виклики, успішна реалізація подібної системи може значно підвищити ефективність управління дорожньою інфраструктурою. Вона не лише знизить витрати й покращить безпеку руху за рахунок якісної розмітки та

своєчасного ремонту, але й стане соціально значущим проектом. Це особливо актуально в умовах обмежених ресурсів і складної економічної ситуації.

2.6 Постановка задачі

Основною задачею є проєктування та створення програмного забезпечення, яке дозволить оптимізувати один із процесів дорожньо-будівельної галузі. Зокрема, акцент зроблено на розпізнаванні об’єктів і їх класифікації, зокрема ідентифікації дефектів дорожнього покриття та аналізі дорожньої розмітки. У центрі уваги — впровадження ефективних методів класифікації.

Після аналізу різних підходів обрано використання нейронних мереж для детекції та розпізнавання об’єктів. Для навчання моделі застосовується підхід із використанням навчання зі вчителем. Дані для тренування формуються за допомогою попередньо підготовленого набору фотографій із промаркованими об’єктами. До кожного промаркованого елемента, включаючи дефекти дорожнього покриття та елементи дорожньої розмітки (лінії, знаки, символи), буде прив’язана відповідна мітка для подальшої обробки.

Результати навчання моделі зберігатимуться локально для подальших аналізів і паралельно передаватимуться на хмарну платформу ClearML. Це дозволить відслідковувати й аналізувати додаткові аспекти процесу.

На початковому етапі планується розроблення програми, здатної розпізнавати об’єкти незалежно від стану дорожнього покриття. Завданням програми буде класифікація об’єктів щодо оцінки стану дороги, зокрема визначення рівня зношення покриття та стану дорожньої розмітки. Це рішення дасть змогу точно оцінити залишковий ресурс дороги, якість видимості та відповідність існуючої розмітки стандартам. На основі отриманих даних можна буде приймати обґрунтовані рішення щодо потреби в інвестиціях на ремонт дорожнього полотна, оновлення розмітки чи обслуговування іншої дорожньої інфраструктури.

У другому розділі було ґрунтовно проаналізовано теоретичні та прикладні аспекти, пов’язані з автоматичним розпізнаванням дефектів дорожнього покриття та елементів дорожньої розмітки за допомогою методів комп’ютерного зору. Основну увагу приділено вивченню актуальності впровадження подібних систем в інфраструктуру сучасних міст, з огляду на необхідність оперативного моніторингу стану доріг, своєчасного виявлення пошкоджень та оптимізації витрат на їхнє обслуговування. Встановлено, що автоматизовані системи на базі штучного інтелекту здатні істотно підвищити ефективність управління дорожнім господарством, забезпечити покращення якості дорожнього покриття та рівень безпеки учасників дорожнього руху.

У межах розділу також було сформульовано проблематику дослідження, зокрема визначено основні технічні та алгоритмічні виклики, пов’язані з розпізнаванням об’єктів на зображеннях за умов складного фону, зношеності дорожнього покриття та неоднорідного освітлення. Визначено потребу в розробці універсального підходу, здатного одночасно виявляти як дефекти покриття (тріщини, вибоїни, ями), так і елементи дорожньої розмітки, що зникають або погано читаються.

Таким чином, другий розділ заклав міцне теоретичне підґрунтя для реалізації системи автоматичного аналізу стану доріг, поєднавши сучасні досягнення у сфері комп’ютерного зору, глибинного навчання та обробки зображень. Проведений аналіз дозволив обґрунтувати вибір архітектури майбутньої системи, визначити критерії її ефективності та сформувати постановку задачі для подальшої розробки програмної реалізації. Також було враховано специфіку дорожніх умов, різноманітність типів покриття та розмітки, що потребує адаптивного підходу до навчання моделей. Визначено, що важливою умовою успішної реалізації проєкту є наявність якісного датасету з достовірною анотацією, який дозволить моделі навчитися виявляти як типові, так і нетипові пошкодження та елементи розмітки.
Крім того, окреслено можливі шляхи подальшого вдосконалення системи, зокрема за рахунок інтеграції геоприв’язки, хмарного зберігання даних та використання мобільних пристроїв для збору зображень у польових умовах.

3 ОПИС РЕАЛІЗАЦІЇ ПРОГРАМНОЇ ЧАСТИНИ

3.1 Аналіз вхідних наборів даних та підготовка до обробки нейронною мережею

Основною проблемою під час аналізу вхідних наборів даних виявилася велика різноманітність типів доріг та дефектів. Окрім фотографій, зроблених в Україні, було залучено додаткові дані з міжнародних джерел. Це надало змогу сформувати багатий і різноплановий набір даних, що включає різні види дорожніх покриттів і дефектів з усього світу.

Загальний обсяг набору даних становить 7000 зображень для навчання та близько 300 зображень для перевірки. Для забезпечення високої якості й розмаїття даних використовувалися наведені нижче джерела:

— набір даних вибоїн від платформи Roboflow, який містить великий масив попередньо розмічених зображень із дефектами доріг, включно з ямами, тріщинами та іншими пошкодженнями;

— дані, отримані з публікацій дослідницьких робіт;

— фото, отримані з відеозаписів на YouTube, Google Maps (рисунки 3.1, 3.2) і розмічені вручну;

— набір даних RDD2022 — велика колекція зображень доріг із різних країн.

Рисунок 3.1 — Приклад фотографії з тренувального набору даних зробленої в Китаї

Рисунок 3.2 — Приклад фотографії взятої з Google Maps

Одним із важливих аспектів стало визначення відповідного інструмента для розмітки даних. Серед численних варіантів був обраний CVAT.

CVAT (Computer Vision Annotation Tool) є програмним забезпеченням з відкритим вихідним кодом [23], створеним компанією Intel.

Він пропонує зручний та функціональний інтерфейс для виконання задач із розмітки даних (рис. 3.3). Нижче наведено ключові переваги цього інструменту:

— підтримка різноманітних типів анотацій, включаючи полігони, прямокутники, точки та лінії;

— можливість організації спільної роботи над проєктами;

— проста інтеграція з іншими програмними засобами та легке налаштування.

Далі наведено причини вибору CVAT:

— архітектура CVAT підтримує розмітку різних типів об'єктів, таких як полігони, точки, лінії та прямокутники;

— інструмент працює з багатьма форматами зображень і відео, що значно спрощує імпорт інформації з різних джерел та для різних типів доріг;

— зручний дизайн інтерфейсу сприяє швидкому навчанню нових розмітників та забезпечує ефективний процес роботи навіть із великими масивами даних;

— можливість одночасної роботи кількох користувачів над одним проєктом значно пришвидшує процес розмітки.

—

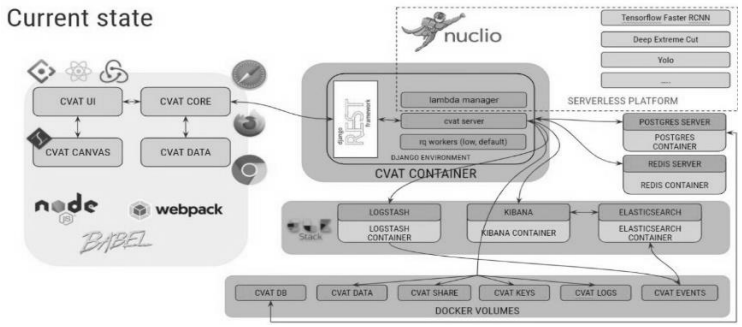

Рисунок 3.3 — Візуалізація архітектури роботи CVAT

Алгоритм аналізу даних виглядав так:

— зображення і відео доріг були зібрані з різних джерел, таких як урядові бази даних, дослідницькі проєкти та громадські ініціативи;

— усі зібрані матеріали були завантажені до CVAT і впорядковані в проєкти, що відображали конкретні типи доріг або регіони для аналізу;

— для визначення меж дефектів неправильної форми використовувалися полігони, тоді як прямокутники використовувалися для окреслення областей інтересу;

— на фінальному етапі дані проходили перевірку для забезпечення високої якості.

3.2 Опис обраної архітектури нейронної мережі

Для виконання даного завдання, серед перелічених у підрозділі 2.4 методів, було обрано модель YOLO. Ця архітектура має багато версій, кожна з яких має свої

особливості. У межах цієї роботи використано одну з найсучасніших версій — YOLOv8 (рисунок 3.4).

Основні причини вибору моделі YOLO такі:

— модель спроектована таким чином, щоб функціонувати на пристроях із обмеженими апаратними ресурсами;

— вона забезпечує баланс між високою швидкістю обробки та достатнім рівнем точності;

— легко адаптується до різноманітних завдань і може застосовуватися для розпізнавання об'єктів у різних сценаріях і на різних наборах даних;

— має широку спільноту користувачів та розробників, що забезпечує доступ до численних матеріалів, документації та практичних прикладів.

Серед переваг YOLOv8 у порівнянні з попередніми версіями можна виділити впровадження найсучасніших технологій та оптимізацій, які забезпечують високу точність і швидкість роботи. Крім того, модель адаптована для ефективного функціонування на різноманітних апаратних платформах і в різних умовах використання.

У січні 2023 року компанія Ultralytics представила YOLOv8, яка на момент випуску була однією з найпрогресивніших моделей. З точки зору архітектури, вона частково базується на YOLOv5, але має деякі вдосконалення. Зокрема, в YOLOv8 застосовано модуль c2f замість CSPNet, який є покращеною версією останнього із двома послідовними згортковими шарами. Модель також створена без використання якорів, що відображає її засади на базі YOLOv5. Одним із ключових оновлень YOLOv8 [24] стало впровадження розділених головних блоків для незалежної обробки задач визначення об'єктності, класифікації та прогнозування межових рамок.

У процесі прогнозування вона використовує сигмоїдний шар для оцінки об'єктності та softmax для класифікаційних задач, що додає гнучкості й точності аналізу.

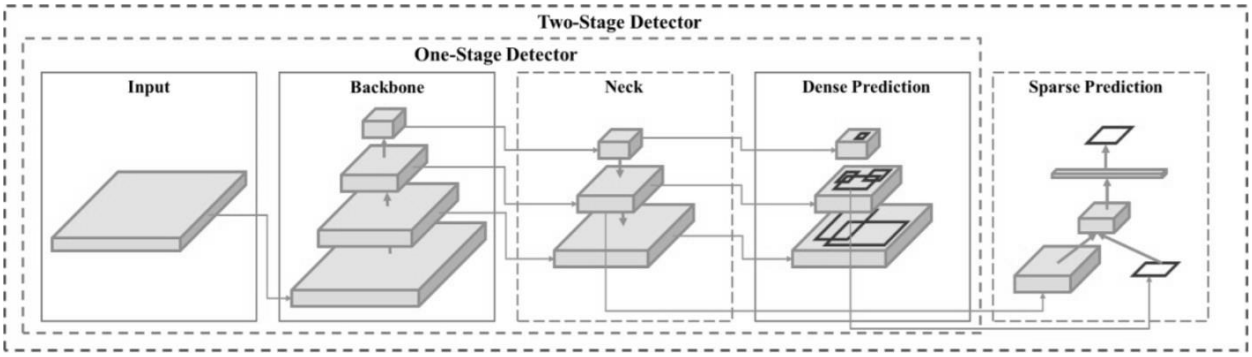

Рисунок 3.4 — Загальна архітектура YOLOv8

YOLOv8 використовує функції втрат CIoU та DFL для обчислення втрат межових рамок, а також бінарну крос-ентропію для втрат класифікації. Завдяки цим функціям втрат було досягнуто покращення продуктивності виявлення об'єктів, особливо при роботі з об'єктами малого розміру. Крім того, представлено модель YOLOv8-Seg для семантичної сегментації, що була впроваджена з мінімальними змінами в оригінальній архітектурі моделі.

Моделі YOLO Nano, YOLO Small та YOLO Middle представляють різні масштаби архітектури YOLO, адаптовані для різних завдань. У контексті визначення стану дорожнього покриття оптимальним вибором є YOLO Nano через його збалансоване співвідношення між продуктивністю та вимогами до обчислювальних ресурсів. Ця модель ефективно працює на пристроях з обмеженими ресурсами, забезпечуючи швидке і точне виявлення дефектів дорожнього покриття у режимі реального часу.

3.3 Опис використаних інструментів розробки

Для створення програмного забезпечення, яке вирішує поставлену задачу, було обрано мову програмування Python. Основними перевагами цієї мови є висока продуктивність, широкий спектр доступних бібліотек та простота синтаксису.

Python дозволяє швидко писати код, забезпечуючи його легкість у читанні та підтримці, що є ключовим фактором для якісної розробки та подальшого супроводу програмного забезпечення. Крім того, значною перевагою Python є активна спільнота розробників та доступність численних ресурсів.

Для роботи з Python було створено віртуальне середовище (рисунок 3.5). Використання віртуального середовища має кілька важливих переваг.

По-перше, воно дає змогу ізолювати залежності проекту, що запобігає конфліктам між бібліотеками з відмінними версіями.

По-друге, це сприяє відтворюваності проектів, адже всі залежності можна зафіксувати в одному місці та легко відновити при необхідності.

По-третє, створення віртуальних середовищ значно спрощує управління проектами, дозволяючи використовувати різні версії Python та бібліотек для окремих проектів. Це також підвищує рівень безпеки, оскільки ізольовані середовища зводять до мінімуму ризик випадкових змін системних бібліотек.

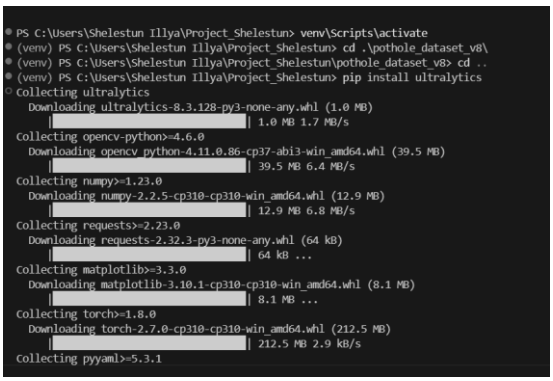

Рисунок 3.5 — Створення віртуального середовища та встановлення Ultralytics

Задля оптимізації роботи та отримання ряду значних переваг було прийнято рішення впровадити YOLOv8 разом із пакетом Ultralytics [25]. Такий вибір обумовлений суттєвими покращеннями у точності, швидкості та ефективності обробки даних, які забезпечує найновіша версія YOLO.

Ultralytics — це розробник інструментів для комп'ютерного зору, головним продуктом якого є YOLO (You Only Look Once). Використання YOLOv8 із цим пакетом надає низку переваг, що робить його чудовим рішенням для широкого спектра задач у галузі комп'ютерного зору.

Нижче розглянуто ключові аспекти, які роблять цей продукт незамінним:

— інтегрована екосистема, яка забезпечує зручне маркування та організацію даних через Ultralytics Hub і Roboflow, що значно спрощує управління наборами для тренування моделей;

— розширені інструменти навчання дозволяють ефективно працювати з моделями, зокрема через ноутбуки Paperspace, що додає гнучкості у процесі розробки;

— системи логування та моніторингу включають Comet і ClearML, що дає змогу детально аналізувати кожен етап навчання та відстежувати ключові параметри моделі;

— легке впровадження моделей на різних платформах, зокрема на Neural Magic, полегшує інтеграцію в прикладні рішення;

— підтримка численних форматів експорту моделей, таких як TensorFlow, ONNX, CoreML тощо, забезпечує сумісність із різними технологічними екосистемами;

— висока продуктивність YOLOv8, яка вирізняється високою швидкістю роботи разом із точністю та адаптивністю до різноманітних апаратних платформ — від серверів до мобільних пристроїв;

— розвинена документація, яка передбачає наявність детальних технічних матеріалів і підтримка активної спільноти сприяють оперативному вирішенню проблем і доступу до найактуальніших оновлень.

Для моніторингу та ефективного менеджменту процесів тренування було застосовано ClearML. Цей інструмент виявився справді незамінним завдяки таким перевагам:

— система ClearML дозволяє в реальному часі відслідковувати ключові показники, як-от точність, втрати та швидкість навчання, а також надає їхню візуалізацію для ефективного аналізу;

— інструмент дає змогу організовувати та порівнювати

експерименти, зберігати конфігурації моделей і гіперпараметрів, що значно полегшує пошук оптимальних налаштувань;

— використання API ClearML дозволяє автоматизувати процеси моніторингу й запускати експерименти за допомогою попередньо створених скриптів;

— інтеграція з API ClearML надає можливість розробляти скрипти для автоматичного запуску експериментів, збору даних і створення звітів;

— експерименти легко запускати на віддалених серверах чи кластерах із повним контролем над процесом виконання;

— автоматичне збереження конфігурацій середовища, включно з версіями бібліотек і залежностей, забезпечує повну повторюваність досліджень;

— система версій дозволяє повертатися до попередніх ітерацій для аналізу та порівняння результатів.

Інтеграція ClearML із YOLOv8 та пакетом Ultralytics суттєво оптимізувала процес тренування моделей завдяки можливостям реального моніторингу, управління експериментами та автоматизації рутинних завдань. Це забезпечило досягнення високого рівня ефективності й точності при реалізації проекту.

3.4 Розробка комп’ютерної системи розпізнавання дорожньої розмітки та виявлення пошкоджень дорожнього покриття

У цьому розділі розглянуто реалізацію комп’ютерної системи, що виконує автоматизоване розпізнавання елементів дорожньої інфраструктури, зокрема пошкоджень дорожнього покриття та елементів дорожньої розмітки. Розроблена система ґрунтується на методах комп’ютерного зору та нейронних мереж, що дає змогу досягти високої точності в умовах реального середовища.

Архітектура системи побудована за модульним принципом, де кожна з підсистем відповідає за виконання окремого завдання — виявлення пошкоджень або розпізнавання розмітки. Вхідними даними для обох модулів є зображення

дороги, отримані з відеопотоку або фотозйомки з мобільної платформи. Результати обробки зберігаються в зручному форматі для подальшого аналізу або візуалізації.

3.4.1 Розробка підсистеми виявлення пошкоджень дорожнього покриття

Підсистема виявлення пошкоджень дорожнього покриття реалізована на базі моделі YOLO, навченої на наборі зображень, що містять різні типи дефектів — тріщини, вибоїни, ями тощо. Метою підсистеми є ідентифікація пошкоджених ділянок на зображеннях дороги, локалізація цих ділянок та їх візуальне виділення.

Для навчання використовувався розмічений датасет із реальними зображеннями дорожніх покриттів. Під час підготовки даних були застосовані методи аугментації: зміна масштабу, поворот, зміна яскравості та контрастності. Це дозволило суттєво покращити узагальнюючу здатність моделі.

Результатом роботи є побудова обмежувальних рамок навколо пошкоджених ділянок дороги, класифікація ступеня небезпеки пошкодження за площею, а також виведення відповідного кольорового маркування.

Приклад роботи підсистеми наведено у додатку Г (див. рис. Г.1 – ГЗ), де видно процес виявлення вибоїн на реальних зображеннях дороги.

На основі результатів навчання моделі можна зробити аналіз графіків, представлених у додатку Г (див. рис. Г.4 – Г.7):

— протягом 50 епох спостерігалося послідовне зниження функції втрат на тренувальному наборі;

— показники метрик точності та повноти зростали, що вказує на поліпшення здатності моделі виявляти дефекти;

— зростання mAP50(B) і mAP50-95(B) свідчить про здатність моделі до точного виявлення об’єктів різного розміру;

— на валідаційному наборі також спостерігається зменшення помилок, хоч і повільніше, що може вказувати на наявність варіативності в даних.

Нижче наведено приклад функції, що реалізує обробку зображень для виявлення пошкоджень дорожнього покриття за допомогою моделі YOLO (див. лістинг 3.1).

Лістинг 3.1 — Функція обробки зображень для виявлення пошкоджень дорожнього покриття за допомогою моделі YOLO

```
import cv2
from ultralytics import YOLO
def process_image(input_path, output_path, model_path):
    image = cv2.imread(input_path)
    if image is None:
        print("Не вдалося завантажити зображення.")
        return
    model = YOLO(model_path)
    threshold = 0.5
    area_threshold_yellow = 5000
    area_threshold_red = 20000

    def classify_rectangle(area):
        if area < area_threshold_yellow:
            return (0, 255, 0), "ЗЕЛЕНИЙ"
        elif area < area_threshold_red:
            return (0, 255, 255), "ЖОВТИЙ"
        else:
            return (0, 0, 255), "ЧЕРВОНИЙ"

    results = model(image)[0]
    for box in results.boxes:
        x1, y1, x2, y2 = map(int, box.xyxy[0])
        area = (x2 - x1) * (y2 - y1)
        color, label = classify_rectangle(area)
        cv2.rectangle(image, (x1, y1), (x2, y2), color, 2)
        cv2.putText(image, label, (x1, y1 - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.6, color, 2)
    cv2.imwrite(output_path, image)
    print("Обробка завершена. Результат збережено до", output_path)
```

3.4.2 Розробка підсистеми розпізнавання дорожньої розмітки

Підсистема розпізнавання дорожньої розмітки реалізована з використанням модифікованої моделі YOLO, адаптованої для виявлення вузьких об'єктів розмітки — суцільних та пунктирних ліній, стоп-ліній, а також пошкоджених сегментів розмітки.

Для навчання було використано спеціалізований датасет із зображеннями вуличної інфраструктури з маркованими елементами розмітки. Попередня обробка

включала корекцію освітлення, фільтрацію шуму та масштабування, що дозволило підвищити якість розпізнавання в різних умовах освітлення та за різної якості зображення.

Приклад роботи даної підсистеми наведено у додатку Г (див. рис. Г.8), де видно процес розпізнавання розмітки на дорозі.

На основі аналізу результатів навчання (див. рис. Г.9, Г.10) можна зробити такі висновки:

— протягом 50 епох спостерігається стабільне зниження функції втрат, що свідчить про ефективність оптимізації;

— метрики точності та повноти демонструють покращення здатності моделі точно розпізнавати елементи дорожньої розмітки навіть на складному тлі та при різному освітленні;

— зростання показників mAP50(B) та mAP50-95(B) підтверджує ефективність моделі при виявленні об’єктів різного розміру та форми;

— оцінка на валідаційних даних засвідчує, що модель здатна до узагальнення та демонструє стійкість до частково затертих або зруйнованих фрагментів розмітки.

Нижче наведено приклад функції обробки зображень для розпізнавання дорожньої розмітки (див. лістинг 3.2).

Лістинг 3.2 — Функція обробки зображень для розпізнавання дорожньої розмітки за допомогою моделі YOLO

```
import cv2
from ultralytics import YOLO
def detect_damaged_marking_simple(image_path, output_path, model_path):
    frame = cv2.imread(image_path)
    if frame is None:
        print("Не вдалося завантажити зображення.")
        return
    model = YOLO(model_path)
    threshold = 0.5
    results = model(frame)[0]
    for box in results.boxes.data.tolist():
        x1, y1, x2, y2, score, class_id = box
        if score > threshold:
            class_name = results.names[int(class_id)]
```

```
            if class_name.lower() in ['damaged_road_marking', 'road_line_damaged', 'пошкоджена_розмітка']:
                color = (255, 0, 0)
                label = "РОЗМІТКА-ПОШКОДЖЕНА"

                cv2.rectangle(frame, (int(x1), int(y1)), (int(x2), int(y2)), color, 4)
                text = f"{class_name.upper()} {score:.2f} {label}"
                cv2.putText(frame, text, (int(x1), int(y1 - 10)),
                            cv2.FONT_HERSHEY_SIMPLEX, 1.3, color, 3, cv2.LINE_AA)
                coords = f"({int(x1)}, {int(y1)}), ({int(x2)}, {int(y2)})"
                cv2.putText(frame, coords, (int(x1), int(y2) + 30),
                            cv2.FONT_HERSHEY_SIMPLEX, 0.8, (255, 255, 255), 2, cv2.LINE_AA)

    cv2.imwrite(output_path, frame)
    print("Обробка завершена. Результат збережено до", output_path)
```

Розпізнавання дефектів на асфальтному покритті показало найкращі результати. Це обумовлено кількома ключовими факторами. По-перше, більша кількість зображень була використана для тренування моделі, що забезпечило широкий спектр даних для навчання нейронної мережі. По- друге, під час відеозйомки відсутні коливання камери, що значно покращило якість зображень і зменшило можливість помилок при розпізнаванні. По- третє, загальна висока якість зображень сприяла точнішому визначенню дефектів асфальтного покриття.

Детекція дефектів на бетонному покритті показала трохи гірші результати. Основною причиною цього є менша кількість даних, доступних для тренування моделі. Обмеженість набору даних зменшує можливості нейронної мережі ефективно вивчати різноманітні типи дефектів, характерних для бетонних доріг.

Незважаючи на складність детекції дефектів на ґрунтових дорогах, модель показала достойні результати. Цей тип доріг має свою специфіку, і хоча кількість та якість зображень для тренування була меншою, модель змогла адекватно адаптуватися до різних умов і забезпечити задовільний рівень точності. Це свідчить про потенціал нейронних мереж у розпізнаванні дефектів на різних типах дорожнього покриття навіть за наявності певних обмежень у даних.

Після детального аналізу графіків, ми можемо сформулювати висновки щодо чотирьох ключових метрик на рисунку Г.7. На графіках тренувань видно, що значення втрат значно знижуються протягом перших 10 ітерацій, після чого стабілізуються, що свідчить про ефективне навчання моделі. На графіку валідації

видно, що значення mAP50 та mAP50-95 поступово покращуються, тоді як показники precision і recall залишаються відносно стабільними після початкового покращення. Моніторинг GPU показує стабільне використання пам'яті та високий рівень завантаження (приблизно 80%), що вказує на інтенсивне використання графічного процесора під час тренувань. Моніторинг системних ресурсів показує стабільне використання процесора

(CPU) і пам'яті (RAM) протягом всього тренування, із невеликими коливаннями у використанні мережевих ресурсів та дискових операцій.

Модель для виявлення дорожньої розмітки також показала стабільні результати. На рисунку Г.8 показано, що система успішно розпізнає дорожню розмітку на асфальтовому покритті за різних умов освітлення. Незважаючи на наявність перешкод, таких як тіні або зміни фактури дорожнього покриття, алгоритм коректно визначає ключові елементи. Виявлені ділянки розмітки позначені синіми, зеленими та жовтими рамками, що свідчить про високу точність локалізації та класифікації.

У третьому розділі було детально розглянуто процес реалізації програмної частини системи автоматизованого розпізнавання дефектів дорожнього покриття та елементів дорожньої розмітки із застосуванням сучасних технологій комп’ютерного зору та глибинного навчання. Основну увагу було приділено аналізу вхідних даних, їх попередній обробці та формуванню комбінованого датасету, який включає зображення як з різних відкритих джерел, так і зібрані самостійно. Цей підхід дозволив забезпечити більшу різноманітність прикладів для навчання моделі, що позитивно вплинуло на точність і надійність системи.

У якості основної архітектури для вирішення завдань розпізнавання було обрано сучасну нейронну мережу YOLOv8, яка забезпечує високу швидкість обробки зображень в режимі реального часу та високу точність локалізації об’єктів.

Таким чином, результати, отримані на етапі реалізації програмної частини, свідчать про доцільність застосування моделі YOLOv8 для вирішення завдань розпізнавання дорожньої розмітки та дефектів дорожнього покриття.

4 ФУНКЦІОНАЛЬНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

У цьому розділі проведено оцінювання функціональних можливостей майбутнього програмного забезпечення, яке призначене для класифікації та розпізнавання дефектів дорожнього покриття та пошкоджень дорожньої розмітки з використанням технологій глибинного навчання та нейронних мереж. Запропоноване рішення має на меті підвищити рівень безпеки дорожнього руху, сприяти ефективній роботі комунальних і дорожніх служб, а також покращити контроль за якістю виконаних ремонтних робіт і нанесенням розмітки.

Особливу увагу приділено аналізу функцій, що відповідають за розпізнавання порушень горизонтальної дорожньої розмітки, оскільки її зношення або пошкодження безпосередньо впливають на безпеку учасників дорожнього руху, особливо за умов поганої видимості або в нічний час. Реалізація функцій моніторингу стану розмітки дозволяє своєчасно виявляти ділянки, що потребують оновлення, та забезпечувати відповідність дорожнім стандартам.

Програмний продукт побудовано на модульній архітектурі, яка дозволяє реалізувати декілька ключових функціональних підсистем: обробку вхідних зображень, попередню фільтрацію, виділення ознак, класифікацію об'єктів і виведення результатів. Це забезпечує гнучкість налаштування, масштабованість системи та можливість її подальшої інтеграції з іншими компонентами дорожньої інфраструктури.

Під час аналізу було розглянуто декілька підходів до реалізації функціональних вимог, зокрема вибір методів глибинного навчання, моделей комп’ютерного зору, способів представлення результатів для користувача тощо. Обґрунтований вибір реалізованих функцій базується на оцінці їх відповідності цілям і задачам програмного продукту, а також здатності до подальшої адаптації під змінні умови експлуатації.

4.1 Постановка задачі проектування

У роботі використовується метод функціонального аналізу (ФА) для дослідження ключових функцій програмного продукту, що призначений для

прогнозування стабільності фінансових показників підприємства. Основна мета функціонального аналізу полягає у виявленні, оцінці та структуризації функцій системи з метою забезпечення її повної відповідності поставленим завданням.

З огляду на те, що кожна окрема підсистема взаємодіє з іншими компонентами системи, функціональні вимоги до них мають бути узгодженими і спрямованими на досягнення загальної мети. У процесі аналізу враховано потребу в інтеграції модулів збору, обробки та аналізу даних, що дозволяє забезпечити цілісність і послідовність у функціонуванні всієї системи.

Основні функціональні вимоги до програмного забезпечення передбачають:

— підтримку багатоплатформності для забезпечення роботи програми на різних операційних системах;

— модульність та гнучкість архітектури для зручного масштабування і оновлення компонентів;

— наявність зручного та інтуїтивно зрозумілого інтерфейсу користувача;

— забезпечення високої продуктивності при виконанні основних функцій;

— можливість інтеграції з іншими інформаційними системами для обміну даними.

Таким чином, функціональний аналіз дозволяє сформувати чітке уявлення про функціональні характеристики майбутнього програмного продукту, що є основою для його ефективного проєктування та реалізації.

4.2 Обґрунтування функцій програмного продукту

Головна задача $F0$ – розробка ймовірного програмного виробу, який застосовує способи класифікації та розпізнавання вад дорожнього покриття з використанням нейронних мереж.

Взявши за основу цю задачу,можна виділити наступні:

— $F1$ – вибір мови програмування;

— $F2$ – вибір моделей класифікації та розпізнавання;

— $F3$ – вибір середовища розробки.

Кожна з цих функцій має декілька варіантів реалізації:

— функція $F1$: Python, C++.

— функція $F2$: вибір готових переднавчених моделей класифікації та розпізнавання, власна розробка моделей.

— функція $F3$: Visual Studio Code, Sublime Text.

Варіанти реалізації головних задач наведені у морфологічній карті системи (рис. 4.1).

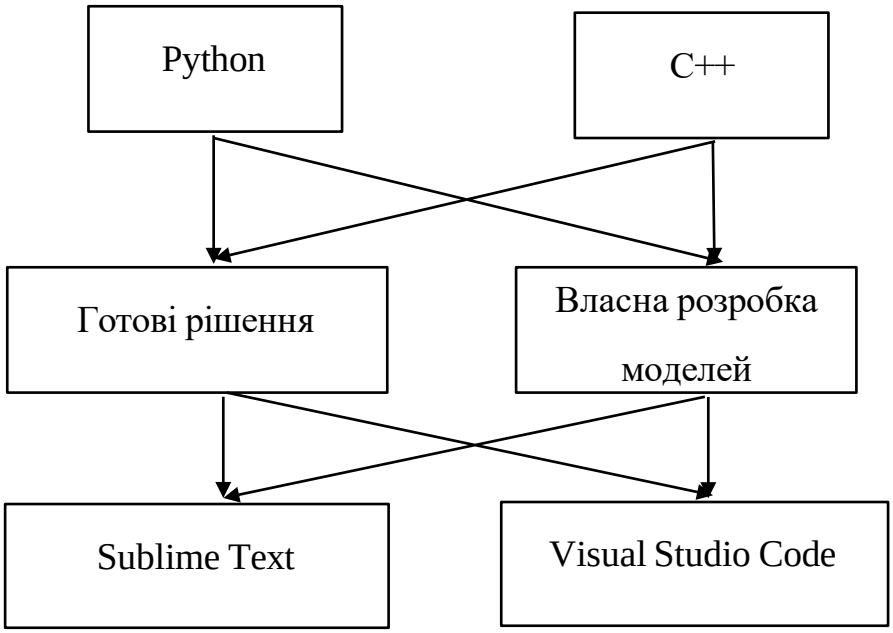

Рисунок 4.1 — Морфологічна карта

Морфологічна карта демонструє множину усіх можливих варіантів основних функцій. Позитивно-негативна матриця наведена в таблиці 4.1.
На основі проведеного аналізу цієї матриці можна зробити висновок, що під час розробки програмного продукту певні варіанти реалізації функцій слід відхилити, оскільки вони не відповідають поставленим завданням програмного виробу. Такі варіанти позначені безпосередньо у морфологічній карті.

Таблиця 4.1 — Позитивно-негативна матриця

Функції	*Варіанти реалізації*	*Переваги*	*Недоліки*
F_1	*А*	Простота у використанні, великий набір бібліотек для різних сфер	Відсутність контролю помилок під час розробки, повільна швидкодія
	Б	Контроль помилок під час розробки, велика швидкодія	Складність у вивченні, використанні, значно менший арсенал бібліотек
F_2	*А*	Швидкість впровадження, передчасно відомі потенційні результати	Неможливість внесення змін, можлива не найкраща оптимальність
	Б	Можливість внести авторські зміни, коригування будови для отримання оптимальності	Довгострокова реалізація, передчасно не відомо чи успішна будова моделі
F_3	*А*	Великий набір функціоналу для налаштування під себе, функціонал для тестування програм	Більша ресурсоємність, складніший інтерфейс
	Б	Простота, легкість програми	Повний доступ за оплатою, менший набір функціоналу

Функція $F1$ — Вибір мови програмування

Обираємо варіант А, тому що простота, швидкість та великий набір реалізованого функціоналу в рамках цієї роботи є більш вигідним.

Функція $F2$ — Вибір підходу до побудови моделі

Обидва варіанти мають суттєві переваги, саме тому використання варіанту А та варіанту Б для розробки можливе.

Функція $F3$ — Вибір середовища розробки

Обираємо варіант А, тому що розробка програмного забезпечення в рамках цієї роботи може потребувати більше функціоналу від середовища розробки програмного забезпечення.

Таким чином, будемо розглядати такі варіанти реалізації ПЗ:

$$F1а – F2а – F3а,$$

$$F1а – F2б – F3а.$$

Для оцінювання якості розглянутих функцій обрана система параметрів, описана нижче.

4.3 Обгрунтування системи параметрів програмного продукту

На підставі даних, розглянутих вище, визначаються головні параметри вибору, які будуть використані для розрахунку коефіцієнта технічного рівня.

Для того, аби охарактеризувати програмний продукт, використовуватимемо наступні параметри:

— X1 – швидкодія мови програмування;

— X2 – об’єм пам’яті для навчання та зберігання даних;

— X3 – час навчання обраних моделей;

— X4 – потенційний об’єм програмного коду.

Найгірші, середні та найкращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію програмного продукту, як показано у таблиці 4.2.

За даними таблиці 4.2 будуються графічні характеристики параметрів (див. рис. Г.11 – Г.14).

Обрані параметри є критично важливими для оцінки ефективності програмного продукту, оскільки вони безпосередньо впливають на продуктивність системи, її адаптивність до змін вхідних даних і можливості масштабування. Отримані результати будуть використані на наступному етапі для порівняння альтернативних реалізацій та вибору оптимального варіанту впровадження.

Таблиця 4.2 — Основні параметри програмного продукту

Назва Параметра	*Умовні позначення*	*Одиниці виміру*	*Значення параметра*		
			гірші	*середні*	*кращі*
Швидкодія мови програмування	X1	оп/мс	10	30	50
Об’єм пам’яті	X2	Мб	1400	800	500
Час навчання обраних моделей	X3	хв	300	180	90
Потенційний об’єм програмного коду	X4	кількість рядків коду	1200	700	400

4.4 Аналіз експертного оцінювання параметрів

Після детального обговорення та проведення аналізу кожен фахівець оцінює ступінь важливості кожного параметра з огляду на конкретну мету — розробку програмного продукту, що спеціалізується на класифікації та розпізнаванні дефектів дорожнього покриття з використанням нейронних мереж.

Процедура визначення коефіцієнтів значущості включає:

— встановлення рівня значимості параметра шляхом присвоєння відповідних рангів;

— перевірку придатності отриманих експертних оцінок для подальшого використання;

— визначення оцінок попарного пріоритету параметрів;

— обробку зібраних результатів та розрахунок коефіцієнтів значущості.

Отримані коефіцієнти значущості дозволяють об'єктивно врахувати пріоритетність параметрів під час прийняття рішень у процесі розробки системи. Це, у свою чергу, сприяє підвищенню обґрунтованості технічних рішень і забезпечує відповідність програмного продукту поставленим цілям. Таким чином, експертне оцінювання стало важливим інструментом для формування ефективної архітектури та функціональності системи.

Результати ранжування параметрів наведені в таблиці 4.3.

Таблиця 4.3 — Результати ранжування параметрів

Позначення параметра	*Назва параметра*	*Одиниці виміру*	*Ранг параметра за оцінкою експерта*							*Сума рангів* R_i	*Відхилення* Δ_i	Δ_i^2
			1	*2*	*3*	*4*	*5*	*6*	*7*			
X1	Швидкодія мови програмування	Оп/мс	2	1	1	2	1	1	2	10	-7,5	56,25
X2	Об’єм пам’яті	Мб	1	2	2	1	2	2	1	11	-6,5	42,25
X3	Час навчання моделі	с	3	3	4	4	3	4	3	24	6,5	42,25
X4	Потенційний об’єм програмного коду	Кількість рядків коду	4	4	3	3	4	3	4	25	7,5	56,25
	Разом		10	10	10	10	10	10	10	70	0	197

Для перевірки ступеню достовірності експертних оцінок, визначимо наступні параметри.

Сума рангів кожного з параметрів і загальна сума рангів:

$$\mathrm{R_i} = \sum_{j=1}^{N} r_{ij} \;\; R_{ij} = \frac{Nm(n+1)}{2} \;=\; 70$$

де N – число експертів,

n – кількість параметрів;

Середня сума рангів:

$$\mathrm{T} = \frac{1}{n} \quad R_{ij} = 17{,}5.$$

Відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_\mathrm{i} = R_i - \mathrm{T}.$$

Сума відхилень по всім параметрам повинна дорівнювати 0. Загальна сума квадратів відхиленн

$$S = \sum_{i=1}^{N} \Delta_i^2 = 153.$$

Порахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2 (n^3 - n)} = \frac{12 * 197}{7^2 (4^3 - 4)} = 0{,}804 > W_k = 0{,}67.$$

Ранжування можна вважати достовірним, тому що знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0,67.

Скориставшись результатами ранжирування, проведемо попарне порівняння всіх параметрів і результати занесемо у таблицю 4.4.

Таблиця 4.4 — Попарне порівняння параметрів

Параметри	*Експерти*							*Кінцева оцінка*	*Числове значення*
	1	*2*	*3*	*4*	*5*	*6*	*7*		
X1 і X2	<	>	>	<	>	>	<	>	1.5
X1 і X3	>	>	>	>	>	>	>	>	1.5
X1 і X4	>	>	>	>	>	>	>	>	1.5
X2 і X3	>	>	>	>	>	>	>	>	1.5
X2 і X4	>	>	>	>	>	>	>	>	1.5
X3 і X4	>	>	<	<	>	<	>	>	1.5

Числове значення, що визначає ступінь переваги *i*–го параметра над *j*– тим, a_{ij} визначається по формулі:

$$a_{ij} = \begin{cases} 1.5 \text{ при } X_i > X_j \\ 1.0 \text{ при } X_i = X_j. \\ 0.5 \text{ при } X_i < X_j \end{cases}$$

З отриманих числових оцінок переваги складемо матрицю $A = \| a_{ij} \|$.

Для кожного параметра зробимо розрахунок вагомості $K_{вi}$ за наступними формулами:

$$K_{вi} = \frac{b_i}{\sum_{i=1}^{n} b_i}$$

$$b_i = \sum_{i=1}^{N} a_{ij}$$

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 2%). Обрахунки буде занесено у таблицю 4.5. На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$$K_{\text{ві}} = \frac{b'_i}{\sum_{i=1}^{N} b'_i},$$

$$b'_i = \sum_{i=1}^{N} a_{ij} b_j .$$

Таблиця 4.5 — Розрахунок вагомості параметрів

Параметри x_i	*Параметри x_j*				*Перша ітер.*		*Друга ітер.*		*Третя ітер*		*Четвета ітер.*	
	X1	X2	X3	X4	b_i	$K_{\text{ві}}$	b_i^1	$K_{\text{ві}}^1$	b_i^2	$K_{\text{ві}}^2$	b_i^3	$K_{\text{ві}}^3$
X1	1	1.5	1.5	1.5	5.5	0.3437	30.25	0.432	166.37	0.526	915	0.62
X2	0.5	1	1.5	1.5	4.5	0.2812	20.25	0.289	91.12	0.289	410	0.273
X3	0.5	0.5	1	1.5	3.5	0.2188	12.25	0.175	42.88	0.136	150	0.01
X4	0.5	0.5	0.5	1	2.5	0.1563	6.25	0.089	15.63	0.049	39	0.025
Всього:					16	1	70	1	316	1	1514	1

4.5 Модуль аналізу видимості

Після того, як дані, зібрані відеореєстратором, будуть позначені, можна створити та навчити модуль виявлення дорожньої розмітки на основі YOLOv3. Цільову дорожню розмітку можна витягти та експортувати як невеликі ділянки зображення. Таким чином, наступним кроком є розробка модуля аналізу видимості, щоб допомогти визначити стан дорожньої розмітки. Дорожня розмітка наноситься головним чином для того, щоб заздалегідь повідомляти водіїв. Таким чином,

важливою властивістю дорожньої розмітки є її яскравість. Однак яскравість – це абсолютна, величина, на яку впливає багато факторів, таких як погода та освітленість. Оскільки зорова система людини більш чутлива до контрасту, ніж до абсолютної яскравості, контраст інтенсивності обрано як метрика видимості дорожньої розмітки. У цьому дослідженні контраст було визначено як різниця між середньою інтенсивністю дорожньої розмітки та середньою інтенсивністю навколишнього покриття. Основний конвеєр цього модуля аналізу видимості контурів. Оскільки розмітка дорожнього покриття вже експортована як ділянки зображення, першим кроком є відокремлення розмітки дорожнього покриття від дорожнього покриття. Оскільки в цьому дослідженні розглядалися лише стрілкоподібні розмітки, частину з розміткою можна легко відокремити від зображення, якщо буде знайдено зовнішній контур розмітки. Контур можна описати як криву, яка з'єднує всі безперервні точки вздовж межі з однаковим кольором або інтенсивністю. Алгоритм трасування контурів, який використовується в цій частині, був запропонований Сузукі та ін. Це був один з перших алгоритмів, який визначив ієрархічні зв'язки меж та розрізнив зовнішні межі від меж отворів. Цей метод був інтегрований у бібліотеку OpenCV. Дистанційне зондування 2020, 12, 3837 8 з 17 Вхідне зображення має бути бінарним, що означає, що зображення має лише два значення: 0 та 1, де 0 позначає чорний фон, а 1 – яскравий передній план або об'єкт. Таким чином, межа повинна головним чином служити краєм. Припустимо, що рij позначає значення пікселя в позиції (i, j) на зображенні. Дві змінні, Найновіший номер межі (NBD), Останній найновіший номер межі (LNBD), створюються для запису зв'язку між пікселями під час процесу сканування. Алгоритм використовує схеми сканування рядок за рядком та зліва направо для обробки кожного NBD та LNBD, де $p_{ij} > 0$.

Крок 1. Якщо $p_{ij} = 1$ та $p_{i,j}-1 = 0$, що вказує на те, що ця точка є початковою точкою зовнішньої межі, збільште NBD на 1 та встановіть (i2, j2) ← (i, j − 1). Якщо $p_{ij} \geq 1$ та $p_{i,j}+1 = 0$, що означає, що воно веде до межі отвору, збільшуємо NBD на

1та встановлюємо $(i_2, j_2) \leftarrow (i, j + 1)$ та $LNBD \leftarrow p_{ij}$ у випадку $p_{ij} > 1$. В іншому випадку переходимо до кроку 3.

Крок 2. З цієї початкової точки (i, j) виконаємо наступні операції для трасування межі.

2.1. Починаючи з пікселя (i_2, j_2), пройдемо по околах пікселя (i, j) за годинниковою стрілкою. У цьому дослідженні для визначення околів вибрано 4-зв'язний випадок, що означає, що лише точки, з'єднані горизонтально та вертикально, розглядаються як околиці. Якщо існує ненульове значення, позначаємо такий піксель як (i_1, j_1). В іншому випадку нехай $p_{ij} = -NBD$ та переходимо до кроку 3.

2.2. Призначаємо $(i_2, j_2) \leftarrow (i_1, j_1)$ та $(i_3, j_3) \leftarrow (i, j)$.

2.3. Взявши піксель (i_3, j_3) за центр, пройдіть околиці проти годинникової стрілки від наступного елемента (i_2, j_2), щоб знайти перший ненульовий піксель, і призначте його як (i_4, j_4).

2.4. Оновіть значення $p_{i3,j3}$ відповідно до кроку 2.4 на рисунку (4.2).

2.5. Якщо $pi3,j3+1 = 0$, оновіть $pi3,j3 \leftarrow -NBD$.

2.6. Якщо $pi3,j3+1$, 0 та $pi3,j3 = 1$, оновіть $p_{i3,j3} \leftarrow NBD$.

2.7. Якщо поточна умова задовольняє $(i_4, j_4) = (i, j)$ та $(i_3, j_3) = (i_1, j_1)$, що означає повернення до початкової точки, перейдіть до кроку 3. В іншому випадку, призначте $(i_2, j_2) \leftarrow (i_3, j_3)$ та $(i_3, j_3) \leftarrow (i_4, j_4)$ і поверніться до підкроку 2.3.

Крок 3. Якщо p_{ij} , 1, оновіть $LNBD \leftarrow p_{ij}$. Нехай $(i, j) \leftarrow (i, j + 1)$ і поверніться до кроку 1 для обробки наступного пікселя. Цей алгоритм зупиняється після обробки найнижчого правого пікселя вхідного зображення.

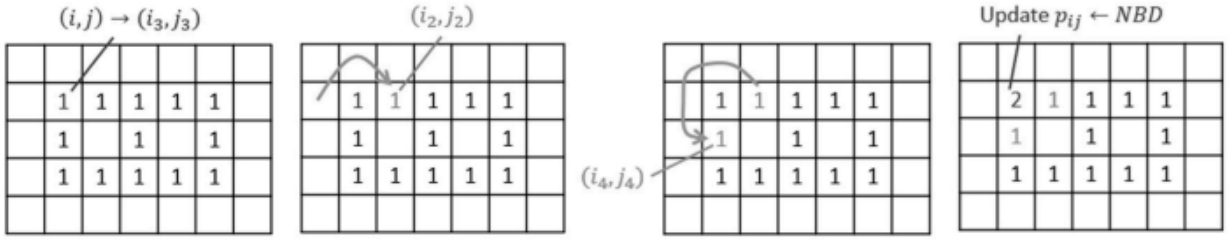

Рисунок 4.2 — Процес трасування контурів за методом Сузукі

Дилетація – це один із методів обробки морфологічних зображень, протилежний ерозії. Основний ефект оператора дилатації на бінарному зображенні полягає у поступовому збільшенні меж пікселів переднього плану, щоб отвори в областях переднього плану стали меншими. Оператор дилатації приймає два фрагменти даних як вхідні дані. Перший вхід – це зображення, яке потрібно дилатати, а другий вхід – це набір координатних точок, відомий як ядро. Ядро визначає точний вплив дилатації на вхідне зображення. Воно припускається , що ядро є квадратом 3×3 з початком координат у центрі. Щоб обчислити вихідне значення дилатації бінарного зображення, кожен піксель фону (тобто значення 0) повинен оброблятися по черзі. Для кожного пікселя фону, якщо хоча б одна координатна точка всередині ядра збігається з пікселем переднього плану (тобто значенням 1), піксель фону повинен бути перевернутий на значення переднього плану. В іншому випадку наступний піксель фону повинен постійно оброблятися. На рисунку 8 показано ефект розширення за допомогою ядра 3×3. Використовуючи метод розширення перед виявленням контурів для ділянок дорожньої розмітки, отвори в розмітці є ознакою.

Після отримання повної зовнішньої межі дорожньої розмітки наступним кроком є відокремлення дорожньої розмітки від навколишнього покриття. У практичних випадках дорожню розмітку не можна фізично відокремити від ділянки зображення через її довільну форму. Найпоширеніший спосіб досягнення мети – використовувати маски для позначення сегментації області. Оскільки існує лише дві категорії об'єктів, тобто дорожня розмітка та покриття, у цьому дослідженні довелося створити дві маски для кожної ділянки зображення. Маскування зображення – це неруйнівний процес редагування зображень, який повсюдно

використовується в графічному програмному забезпеченні, такому як Photoshop, для приховування або відображення деяких частин зображення. Маскування передбачає встановлення деяких значень пікселів на зображенні на 0 або інше фонове значення. Звичайні маски мають лише значення 1 та 0, а області зі значенням 0 повинні бути прихованими (тобто замаскованими). Приклади масок, створених для дорожньої розмітки, Обчислення контрасту інтенсивності Згідно з конвеєром модуля аналізу видимості, останнім кроком є обчислення контрасту між дорожньою розміткою та навколишнім покриттям. Найпростіший спосіб визначити значення контрасту – це просто обчислити різницю між середніми інтенсивностями розмітки та покриття. Однак ця процедура не адаптується до змін у загальній яскравості. Наприклад, різниця яскравості 60 відтінків сірого в темному сценарії (наприклад, на рисунку 4.3).

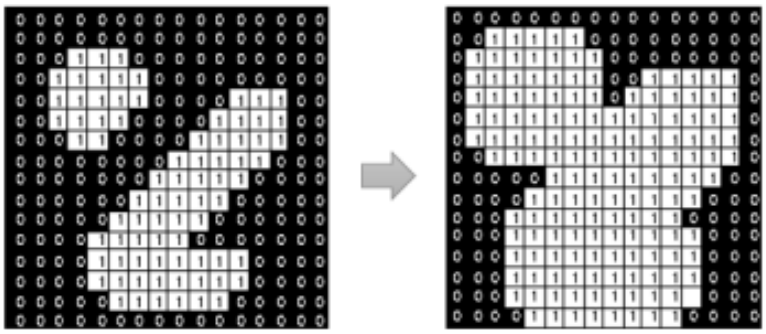

Рисунок 4.3 — Приклад ефекту операції розширення

4.6 Аналіз рівня якості варіантів реалізації функцій

Визначаємо рівень якості кожного варіанту виконання основних функцій окремо.

Абсолютні значення параметрів *X2* (Об’єм пам’яті), *X3* (час навчання моделей) та *X4* (потенційний об’єм програмного коду) відповідають технічним вимогам умов функціонування даного ПП.

Абсолютне значення параметра *X1* (швидкість роботи мови програмування) обрано не найгіршим.

Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховується так (таблиця 4.6):

K_K (j) = $\sum_{i=1}^{n} K_{bi,j} B_{i,j}$,

де n – кількість параметрів;

$K_{вi}$ – коефіцієнт вагомості i–го параметра;

B_i – оцінка i–го параметра в балах.

Таблиця 4.6 — Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	Варіант реалізації функції	Параметри	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1	А	X1	20	8	0.21	1.68
F2	А	X2	700	13	0.342	4.446
F2	Б	X3	100	8	0.21	1.68
F3	А	X4	550	9	0.237	2.13

За даними з таблиці 4.6 за формулою:

$K_K = K_{TY}[F_{1k}] + K_{TY}[F_{2k}] + \ldots + K_{TY}[F_{zk}]$

визначаємо рівень якості кожного з варіантів:

$$K_{K1} = 1.68 + 4.446 + 2.13 = 8.256,$$

$$K_{K2} = 1.68 + 1.68 + 2.13 = 5.49.$$

Як видно з розрахунків, кращим є 1 варіант, для якого коефіцієнт технічного рівня має найбільше значення.

У четвертому розділі було проведено функціональний аналіз програмного продукту, призначеного для виявлення дефектів дорожнього покриття та пошкоджень дорожньої розмітки. Основна увага приділялася визначенню ключових функціональних можливостей системи, її структурних компонентів, а також оцінюванню ефективності їх взаємодії для досягнення поставлених завдань.

У процесі аналізу було детально вивчено функціональні вимоги до програмного забезпечення, які охоплюють:

— обробку та аналіз зображень дорожньої інфраструктури;

— класифікацію типів пошкоджень на основі попередньо навчених моделей;

— інтеграцію з інструментами для машинного навчання та комп’ютерного зору;

— зручну взаємодію з користувачем та представлення результатів аналізу у візуально зрозумілому форматі.

На основі проведеного функціонального аналізу було обґрунтовано вибір ключових компонентів реалізації, які найкраще відповідають зазначеним функціональним вимогам:

— мова програмування Python, що забезпечує широкі можливості для роботи з нейронними мережами, обробки зображень та інтеграції з популярними бібліотеками машинного навчання;

— застосування попередньо навчених моделей, що дозволяє реалізувати функції класифікації та розпізнавання дефектів з високою точністю;

— вибір середовища розробки Visual Studio Code, яке підтримує гнучку конфігурацію, роботу з віртуальними середовищами та має зручну інтеграцію з системами контролю версій.

Таким чином, функціональний аналіз дав змогу визначити оптимальну структуру майбутньої системи, забезпечити її відповідність технічним вимогам і підвищити ефективність виконання основних задач обробки та аналізу зображень дорожньої інфраструктури.

ВИСНОВКИ

У даній бакалаврській кваліфікаційній роботі було проведено комплексне дослідження застосування нейронних мереж для автоматизованого розпізнавання дефектів дорожнього покриття та пошкоджень дорожньої розмітки.

У першому розділі виконано аналіз сучасних підходів до ідентифікації дорожніх дефектів і порушень цілісності розмітки, зокрема з використанням технологій комп’ютерного зору та глибинного навчання. Розглянуто основні методи виявлення пошкоджень, серед яких особливу увагу приділено згортковим нейронним мережам завдяки їх здатності до автоматичного вилучення релевантних ознак із зображень без потреби в ручному формуванні ознак.

У другому розділі було детально проаналізовано архітектуру згорткових нейронних мереж, зокрема шари згортки, активації, агрегації та повнозв’язні шари. Розглянуто різні функції активації, методи агрегації, а також популярні оптимізатори, що використовуються для навчання нейронних мереж. Проведено порівняльний аналіз найпоширеніших моделей, що застосовуються для задач розпізнавання об'єктів.

У третьому розділі обґрунтовано вибір моделі YOLO (You Only Look Once) для реалізації програмного забезпечення системи розпізнавання. Надано детальний огляд архітектури YOLO, описано процес підготовки навчальних даних, включаючи збір та анотацію зображень з дефектами дорожнього покриття та пошкодженнями розмітки. Проведено навчання нейронної мережі, її тестування на контрольних зразках, а також представлено алгоритм функціонування програмного забезпечення, розробленого для реального розпізнавання дефектів і порушень дорожньої розмітки.

У четвертому розділі проведено функціональний аналіз програмного забезпечення, в результаті якого визначено оптимальний варіант реалізації системи. Підтверджено ефективність використання мови Python та популярних бібліотек і платформ, зокрема TensorFlow, Roboflow і Ultralytics, у побудові системи розпізнавання.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Терейковський І.А., Бушуєв. Д.А, та Терейковська Л.О., “ШТУЧНІ НЕЙРОННІ МЕРЕЖІ: БАЗОВІ ПОЛОЖЕННЯ,” 2022. Available:https://ela.kpi.ua/server/api/core/bitstreams/9fee52b6-83fc-4e99-8541-c2767f634c7c/content.

2. K. O'shea and R. Nash, “An Introduction to Convolutional Neural Networks,” Dec. 2015. Available: https://arxiv.org/pdf/1511.08458.

3. IBM, “What are Convolutional Neural Networks? | IBM,” www.ibm.com, 2023. https://www.ibm.com/topics/convolutional-neural-networks.

4. Deep Neural Networks. https://www.tutorialspoint.com/python_deep_learning/python_deep_learning_deep_neural_networks.htm

5. P. Baheti, “12 Types of Neural Networks Activation Functions: How to Choose?” www.v7labs.com, Mar. 08, 2022.

6. N. Malviya, “Evaluation of Object Detection Models — Flops, FPS, Latency, Params, Size, Memory, Storage, mAP…” Medium, May 23, 2024. https://medium.com/@nikitamalviya/evaluation-of-object-detection-models- flops-fps-latency-params-size-memory-storage-map-8dc9c7763cfe.

7. Intersection over Union (IoU) in Object Detection & Segmentation. https://learnopencv.com/intersection-over-union-iou-in-object-detection-and-segmentation/

8. Example of PR curve and point interpolation method. https://qph.cf2.quoracdn.net/main-qimg-bd1357c543dba9e3f5d171855cdb9e2c

9. Z. Kelta, “YOLO Object Detection Explained: A Beginner’s Guide,” www.datacamp.com, Sep. 2022. https://www.datacamp.com/blog/yolo- object-detection-explained

10. J. Kim and J. Cho, “RGDiNet: Efficient Onboard Object Detection with Faster R-CNN for Air-to-Ground Surveillance,” Sensors, vol. 21, no. 5, p. 1677, Mar. 2021, doi: https://doi.org/10.3390/s21051677.

11. J. Jiang, H. Xu, S.-C. Zhang, and Y. Fang, “Object Detection Algorithm Based on Multiheaded Attention,” vol. 9, no. 9, pp. 1829–1829, May 2019, doi: https://doi.org/10.3390/app9091829.

12. Niko Klingler, RetinaNet: Single-Stage Object Detector with Accuracy Focus, dop: https://viso.ai/deep-learning/retinanet/

13. Gautam and S. Singh, “Neural style transfer combined with EfficientDet for thermal surveillance” The Visual Computer, Aug. 2021, doi: https://doi.org/10.1007/s00371-021-02284-2.

14. S. Zhang and Akram Alhelyani, “Maintenance technologies for roads and its prioritization,” Zenodo (CERN European Organization for Nuclear Research), Oct. 2022, doi: https://doi.org/10.5281/zenodo.7264113.

15. Hamzeh Zakeri, Fereidoon Moghadas Nejad, and A. H. Gandomi, Automation and Computational Intelligence for Road Maintenance and Management. John Wiley & Sons, 2022.

16. Example of using laser scanners. https://hexagon.com/solutions/detection-underground-unseen structures?tabId=null

17. F. Alvi, “Computer Vision and Image Processing: Understanding the Distinction and Interconnection,” OpenCV, Dec. 13, 2023.

18. Computer Vision – image classification and object detection. https://ambolt.io/en/image-classification-and-object-detection/

19. A. Olafenwa (she/her), “Image Segmentation With 5 Lines 0f Code,” Medium, Nov. 25, 2020. https://towardsdatascience.com/image- segmentation-with-six-lines-0f-code-acb870a462e8

20. D. Shah, “Object Tracking,” Medium, Apr. 29, 2020. https://medium.com/visionwizard/object-tracking-675d7a33e687

21. Y. Poirier-Ginter and J.-F. Lalonde, “Robust Unsupervised StyleGAN Image Restoration,” Jun. 2023. [Online]. Available: https://arxiv.org/pdf/2302.06733.

22. S. Bigdeli and M. Zwicker, “Image Restoration using Autoencoding Priors,” Apr. 2017. Available: https://arxiv.org/pdf/1703.09964.

23. CVAT, “Documentation” CVAT. https://docs.cvat.ai/docs/.

24. A Brief History of YOLO Object Detection Models From YOLOv1 to YOLOv5. https://machinelearningknowledge.ai/a-brief-history-of-yolo-object-detection-models/

25. G. Jocher, A. Chaurasia, and J. Qiu, “YOLOv8 by Ultralytics,” GitHub, Jan. 01, 2023. https://github.com/ultralytics/ultralytics.

26. Система розпізнавання дорожньої розмітки та виявлення пошкоджень дорожнього покриття/І.С. Шелестун, А. В. Кожем’яко // Матеріали LIV науково-технічній конференції факультету інформаційних технологій та комп’ютерної інженерії. Вінниця 2025 р 2 с. [Електронний ресурс]. Режим доступу: https://conferences.vntu.edu.ua/index.php/all-fitki/all-fitki-2025/paper/view/24198

ДОДАТОК А

Технічне завдання

Міністерство освіти і науки України

Вінницький національний технічний університет

Факультет інформаційних технологій та комп’ютерної інженерії

Кафедра обчислювальної техніки

ЗАТВЕРДЖУЮ

Завідувач кафедри ОТ

проф., д.т.н.. Азаров О.Д.

«21березня 2025 року» 2025 р.

ТЕХНІЧНЕ ЗАВДАННЯ

на виконання бакалаврської кваліфікаційної роботи

«Комп’ютерна система розпізнавання дорожньої розмітки та виявлення пошкоджень дорожнього покриття»

08-54.БДР.020.00.000 ТЗ

Науковий керівник: к.т.н., доц. каф. ОТ

Кожем’яко А. В.

Студент групи 1СП-21Б

Шелестун І.С.

ВНТУ 2025

1. Підстави для виконання бакалаврської кваліфікаційної роботи (БКР)

1.1 Актуальність теми обумовлена зростанням попиту на системи комп’ютерного зору, які базуються на досягненнях штучного інтелекту, глибокого навчання та алгоритмів машинного розпізнавання. Застосування таких підходів, зокрема моделей типу YOLO, дає змогу з високою точністю і швидкістю виявляти пошкодження дорожнього покриття та елементи розмітки.

1.2 Тема бакалаврської кваліфікаційної роботи затверджена наказом ректора університету №97 від 20.03.2025 року.

2.1 Мета роботи – розробка комп’ютерної системи, яка здійснює автоматизоване розпізнавання дорожньої розмітки та виявлення пошкоджень дорожнього покриття з використанням методів комп’ютерного зору та глибокого навчання

2.2 Призначення розробки – створення програмного забезпечення, що дозволяє виявляти та класифікувати елементи дорожньої інфраструктури (розмітку, тріщини, ями тощо) для подальшого аналізу, інформування служб обслуговування доріг або використання в системах допомоги водієві (ADAS).

3 Вихідні дані для виконання БКР

3.1 Аналіз існуючих досліджень і рішень у сфері комп’ютерного зору, глибокого навчання, моделей розпізнавання об’єктів та пошкоджень дорожнього покриття.

3.2 Формування вимог до програмного забезпечення та підбір відповідного інструментарію (Python, OpenCV, TensorFlow/YOLO, Roboflow)

3.3 Створення, підготовка або використання наявних датасетів для тренування моделі.

3.4 Розробка, навчання та тестування моделі розпізнавання.

3.5 Розробка інтерфейсу програми, яка забезпечує завантаження зображень/відео, обробку та виведення результатів.

4 Вимоги до виконання БКР

Головна вимога до виконання бакалаврської кваліфікаційної роботи полягає в розробці ефективної та точної комп’ютерної системи для розпізнавання дорожньої розмітки та виявлення пошкоджень дорожнього покриття на основі методів глибокого навчання. Особливу увагу слід приділити точності розпізнавання, надійності роботи системи та ефективності обробки зображень у реальному часі.

5. Етапи БКР та очікувані результати

5.1 Аналіз літературних джерел, наукових статей та технічної документації з тематики комп’ютерного зору, розпізнавання образів, а також сучасних методів виявлення дорожньої розмітки та пошкоджень дорожнього покриття.

Очікувані результати: формування теоретичної бази, визначення існуючих підходів, аналіз їх переваг і недоліків, постановка технічних вимог до розроблюваної системи.

5.2 Визначення архітектури комп’ютерної системи розпізнавання дорожньої розмітки та пошкоджень дорожнього покриття, розробка концепції її основних модулів, алгоритмів обробки зображень та машинного навчання.

Очікувані результати: концептуальна модель системи, структурна схема взаємодії компонентів, опис функціоналу модулів.

5.3 Розробка програмного забезпечення для автоматичного виявлення та класифікації дорожньої розмітки і пошкоджень дорожнього покриття з використанням методів комп’ютерного зору і нейронних мереж.

Очікувані результати: реалізація алгоритмів, вихідний код системи, підготовлені моделі для тренування.

5.4 Проведення тестування розробленої системи на різних наборах зображень дорожнього покриття, оцінка точності розпізнавання, виявлення та класифікації дефектів.

Очікувані результати: звіти про тестування, виявлені помилки та недоліки, пропозиції щодо покращення.

5.5 Оформлення пояснювальної записки з детальним описом розробки, методів, результатів тестування та аналізом досягнутих цілей, а також створення додатків із вихідними кодами, схемами та ілюстраціями.

Очікувані результати: завершений документ з повним описом роботи, графічними матеріалами, висновками.

5.6 Оформлення пояснювальної записки та додатків.

Підготовка пояснювальної записки згідно з вимогами, включаючи опис архітектури системи, реалізованих функцій, тестування та висновків.

Очікувані результати: повністю оформлена документація, додатки з кодом, графічними матеріалами та схемами.

5.7 Перевірка якості виконання БКР та усунення недоліків.

Проведення внутрішнього контролю якості, внесення коригувань за зауваженнями керівника, підготовка до захисту.

Очікувані результати: удосконалена фінальна версія роботи, готовність до презентації результатів на захисті.

6 Матеріали, що подаються до захисту БКР

До захисту подаються: пояснювальна записка БКР, графічні і ілюстративні матеріали, протокол попереднього захисту БКР на кафедрі, відгук наукового керівника, рецензія, протоколи складання державних екзаменів, анотації до БКР українською та іноземною мовами, довідка про відповідність оформлення БКР діючим вимогам.

7 Порядок контролю виконання та захисту БКР

Виконання етапів графічної та розрахункової документації БКР контролюється керівником згідно зі встановленими термінами. Захист БКР відбувається на засіданні Екзамеційної комісії, затвердженої наказом ректора.

8 Вимоги до оформлювання та порядок виконання БКР

8.1 Вимоги до оформлювання

При оформлюванні БКР використовуються:

– ДСТУ 3008: 2015 «Звіти в сфері науки і техніки. Структура та правила оформлювання»;

– ДСТУ 8302: 2015 «Бібліографічні посилання. Загальні положення та правила складання»;

– міждержавний ГОСТ 2.104-2006 «Єдина система конструкторської документації. Основні написи»;

– методичні вказівки до виконання бакалаврських кваліфікаційних робіт зі спеціальності 123 – «Комп’ютерна інженерія» (освітня програма «Системне програмування»). Кафедра обчислювальної техніки ВНТУ 2023.

8.2 Порядок виконання БКР викладено в «Положення про кваліфікаційні роботи на першому (бакалаврському) рівні вищої освіти СУЯ ВНТУ-03.02.02-П.001.01:21».

ДОДАТОК Б

Протокол перевірки кваліфікаційної роботи

Назва роботи: _____Комп'ютерна система розпізнавання дорожньої розмітки та виявлення пошкоджень дорожнього покриття

Тип роботи: _____ бакалаврська кваліфікаційна робота _____
(бакалаврська кваліфікаційна робота / магістерська кваліфікаційна робота)

Підрозділ _____кафедра обчислювальної техніки_____
(кафедра, факультет, навчальна група)

Коефіцієнт подібності текстових запозичень, виявлених у роботі системою StrikePlagiarism ___36____%

Висновок щодо перевірки кваліфікаційної роботи (відмітити потрібне)

- ✓ Запозичення, виявлені у роботі, є законними і не містять ознак плагіату, фабрикації, фальсифікації. Роботу прийняти до захисту

- ☐ У роботі не виявлено ознак плагіату, фабрикації, фальсифікації, але надмірна кількість текстових запозичень та/або наявність типових розрахунків не дозволяють прийняти рішення про оригінальність та самостійність її виконання. Роботу направити на доопрацювання.

- ☐ У роботі виявлено ознаки плагіату та/або текстових маніпуляцій як спроб укриття плагіату, фабрикації, фальсифікації, що суперечить вимогам законодавства та нормам академічної доброчесності. Робота до захисту не приймається.

Експертна комісія:

_____Азаров О.Д., зав. каф. ОТ _____ _____
(прізвище, ініціали, посада) (підпис)

_____ Тарновський М.Г., доц. каф. _____ _____
(прізвище, ініціали, посада) (підпис)

Особа, відповідальна за перевірку _____ ___Захарченко С.М.____
(підпис) (прізвище, ініціали)

З висновком експертної комісії ознайомлений(-на)

Керівник _____ _____Кожем'яко А.В._____
(підпис) (прізвище, ініціали, посада)

Здобувач _____ _____Шелестун І.С._____
(підпис) (прізвище, ініціали)

ДОДАТОК В

Графічна частина

КОМП’ЮТЕРНА СИСТЕМА РОЗПІЗНАВАННЯ ДОРОЖНЬОЇ РОЗМІТКИ ТА ВИЯВЛЕННЯ ПОШКОДЖЕНЬ ДОРОЖНЬОГО ПОКРИТТЯ

Рисунок В.1 — Блок – схема головного алгоритму обробки інструкцій

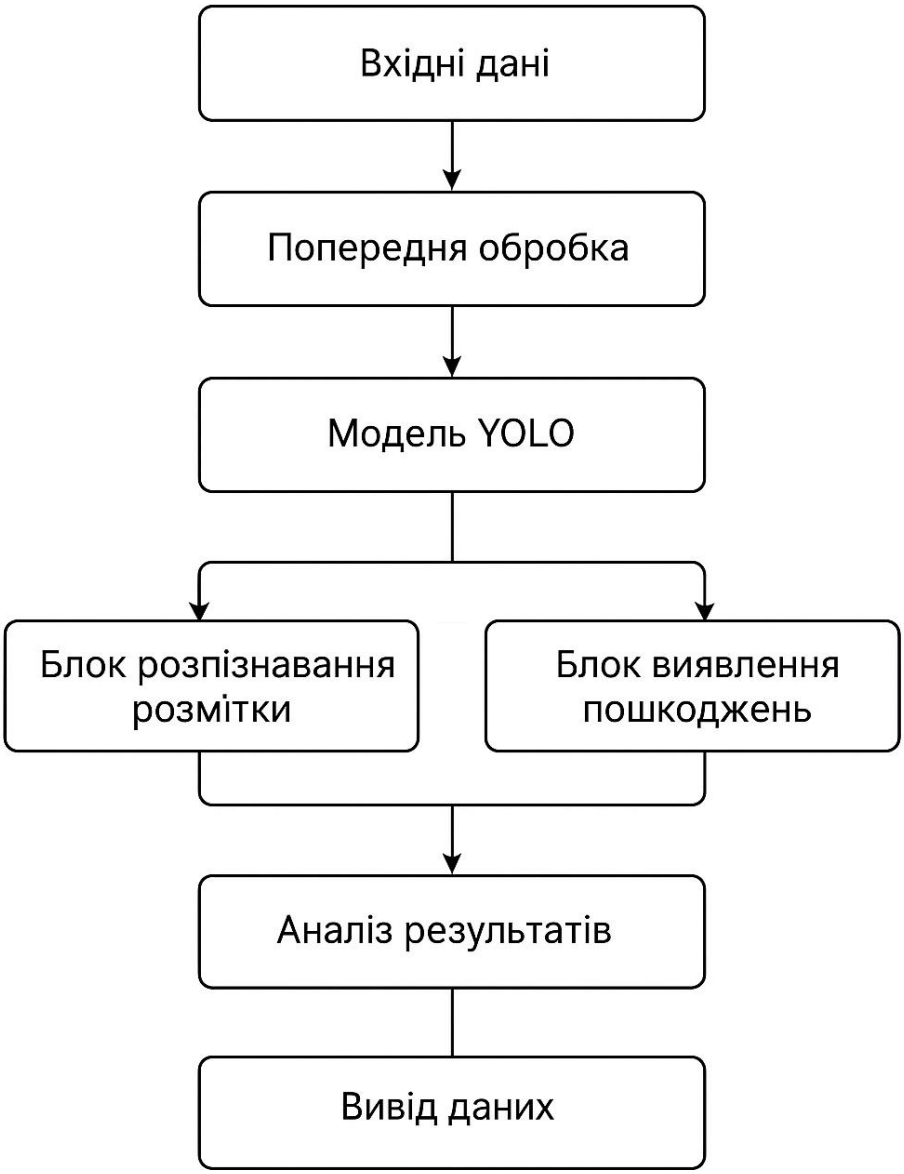

Рисунок В.2 — Структурна схема програми

ДОДАТОК Г
Ілюстративна частина

КОМП’ЮТЕРНА СИСТЕМА РОЗПІЗНАВАННЯ ДОРОЖНЬОЇ РОЗМІТКИ ТА ВИЯВЛЕННЯ ПОШКОДЖЕНЬ ДОРОЖНЬОГО ПОКРИТТЯ

Рисунок Г.1 — Результат роботи програми для виявлення пошкоджень асфальтного покриття за умови низької роздільної здатності

Рисунок Г.2 — Результат роботи програми для виявлення пошкоджень дорожнього покриття на бетонній поверхні

Рисунок Г.3 — Результат роботи програми для виявлення пошкоджень на комбінованому дорожньому покритті

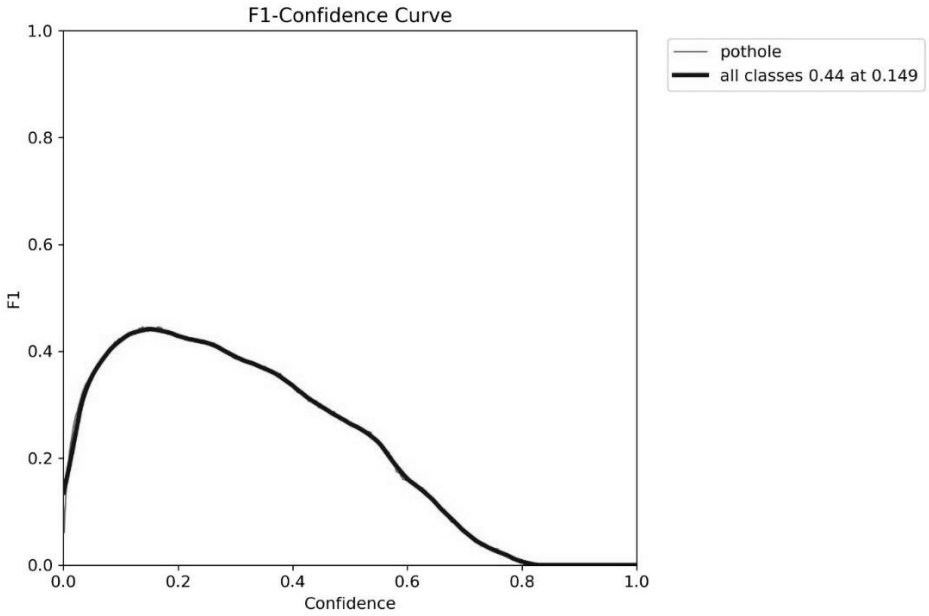

Рисунок Г.4 — Графік кривої F1-впевненості з результатів

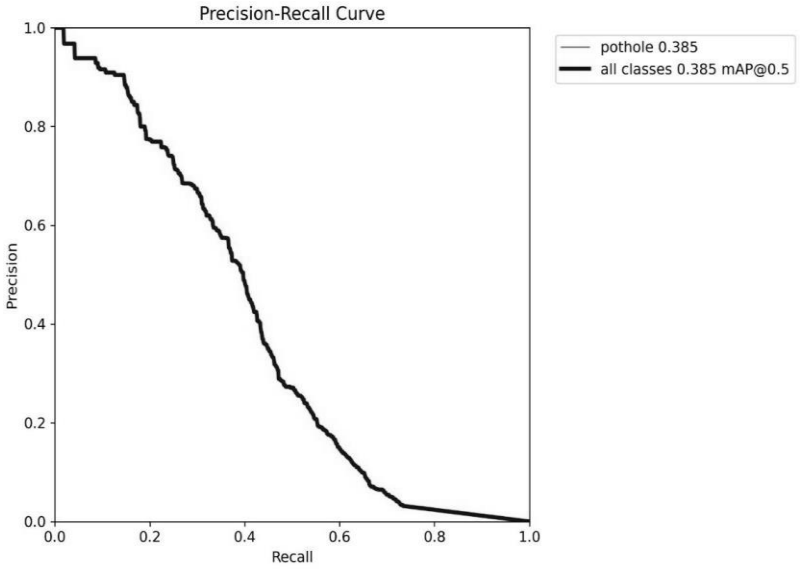

Рисунок Г.5 — Графік PR-кривої з результатів

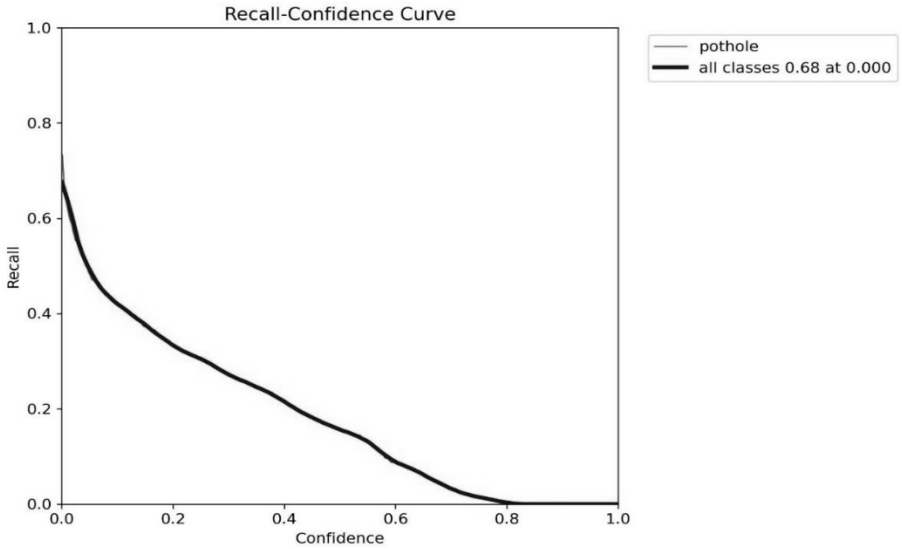

Рисунок Г.6 — Графік кривої відношення точності та впевненості

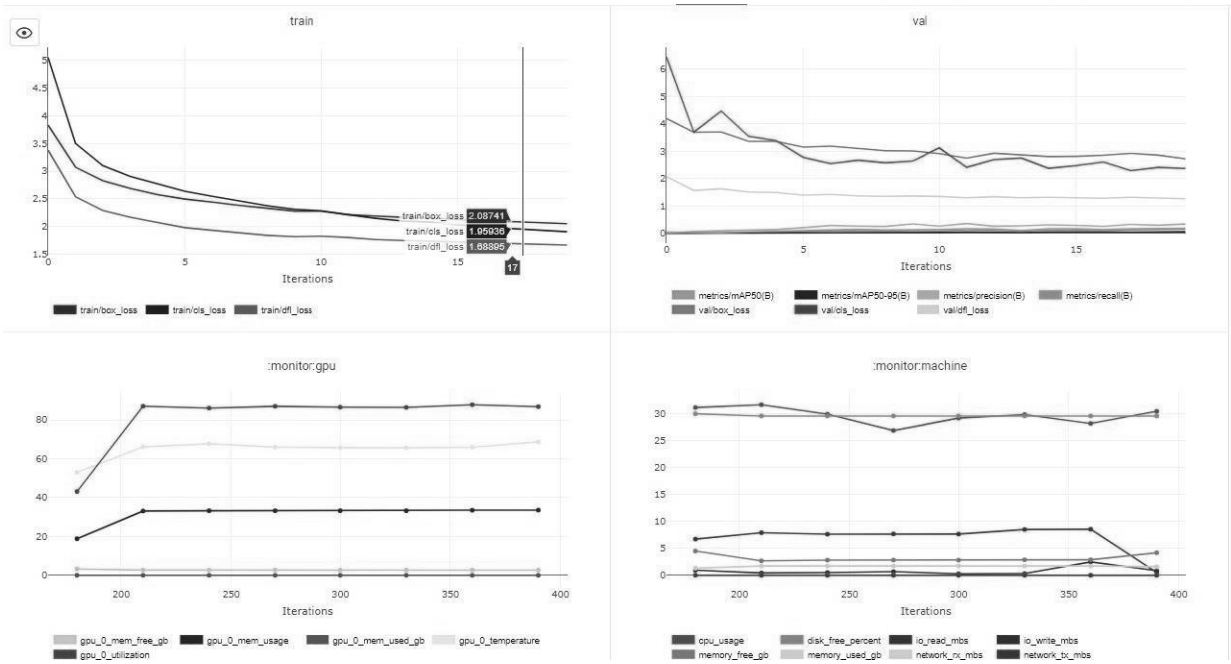

Рисунок Г.7 — Основні метрики системи під час тренування вибору

Рисунок Г.8 — Результат розпізнавання розмітки на перших чотирьох зображеннях за змінних умов освітлення

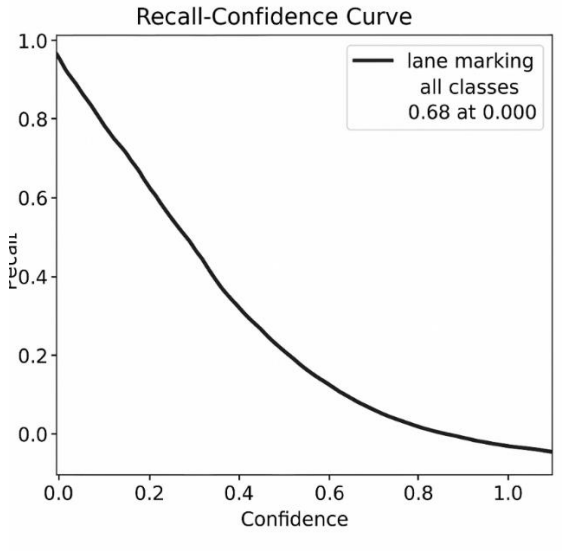

Рисунок Г.9 — Графік кривої відношення точності та впевненості

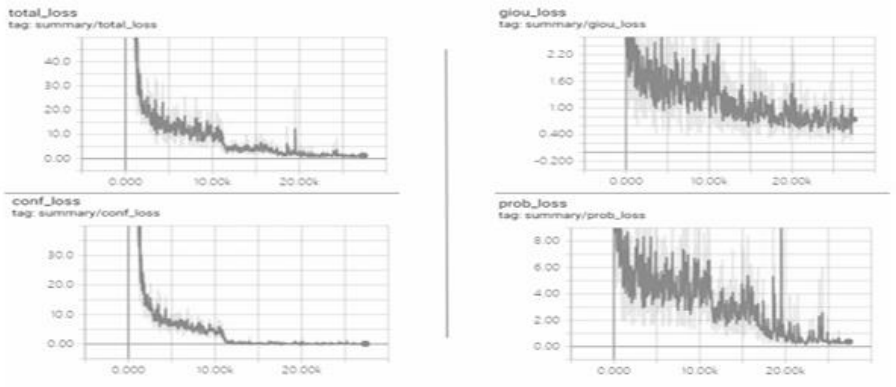

Рисунок Г.10 — Графіки тенденцій різних функцій втрат під час процесу навчання моделі

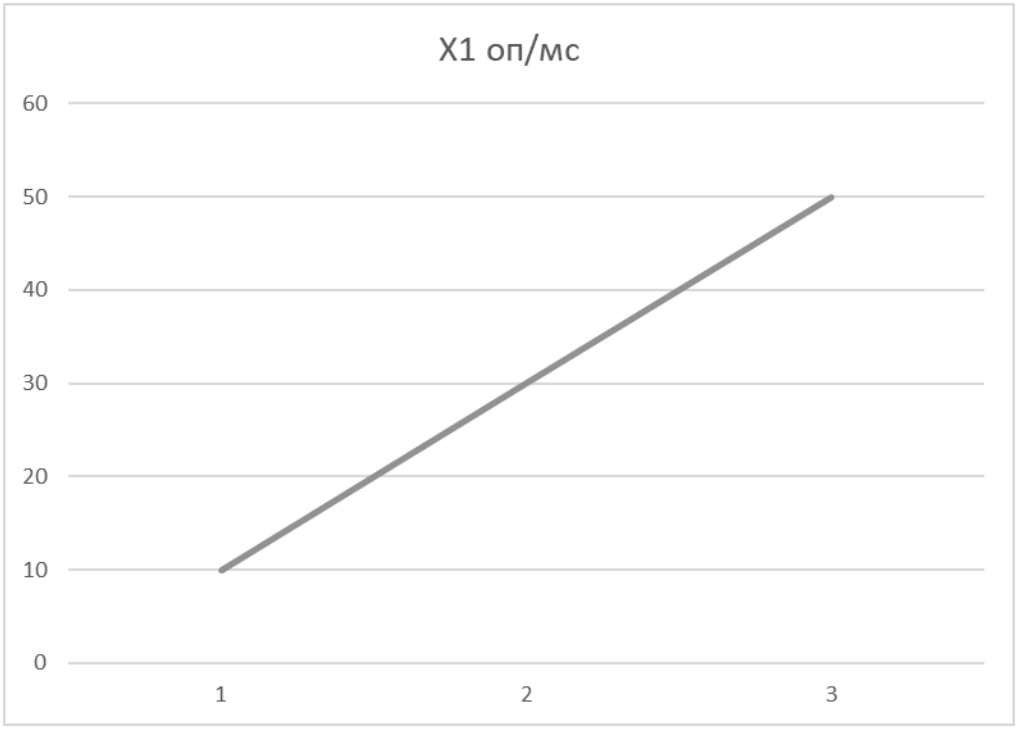

Рисунок Г.11 — X1, швидкодія мови програмування

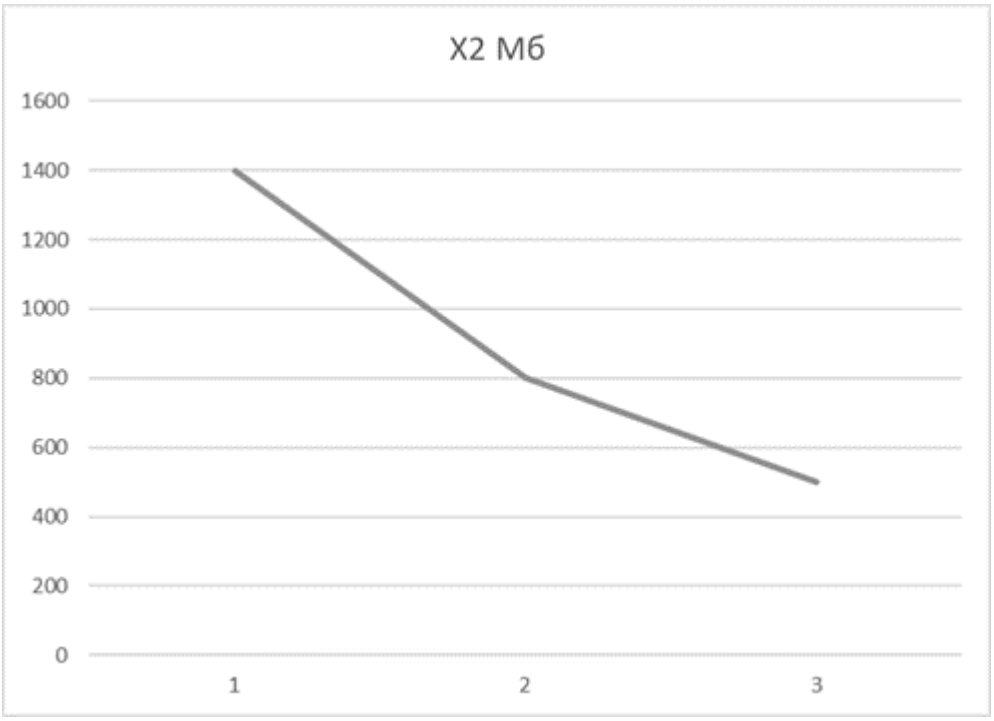

Рисунок Г.12 — X2, об’єм використаної пам’яті

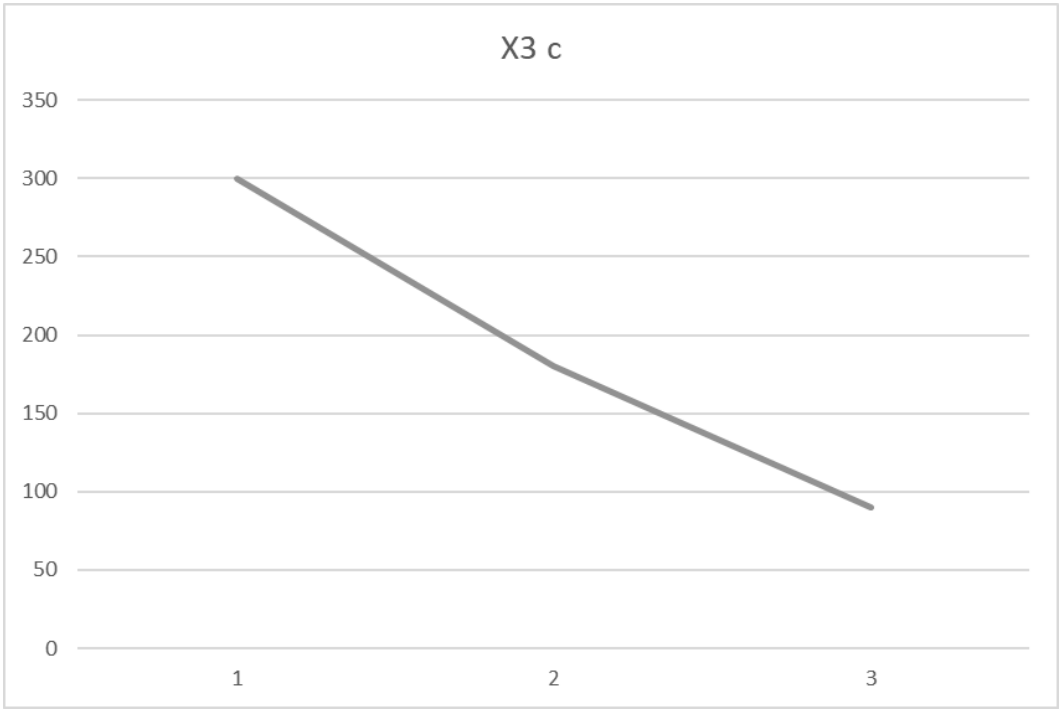

Рисунок Г.13 — X3, час навчання обраних алгоритмів

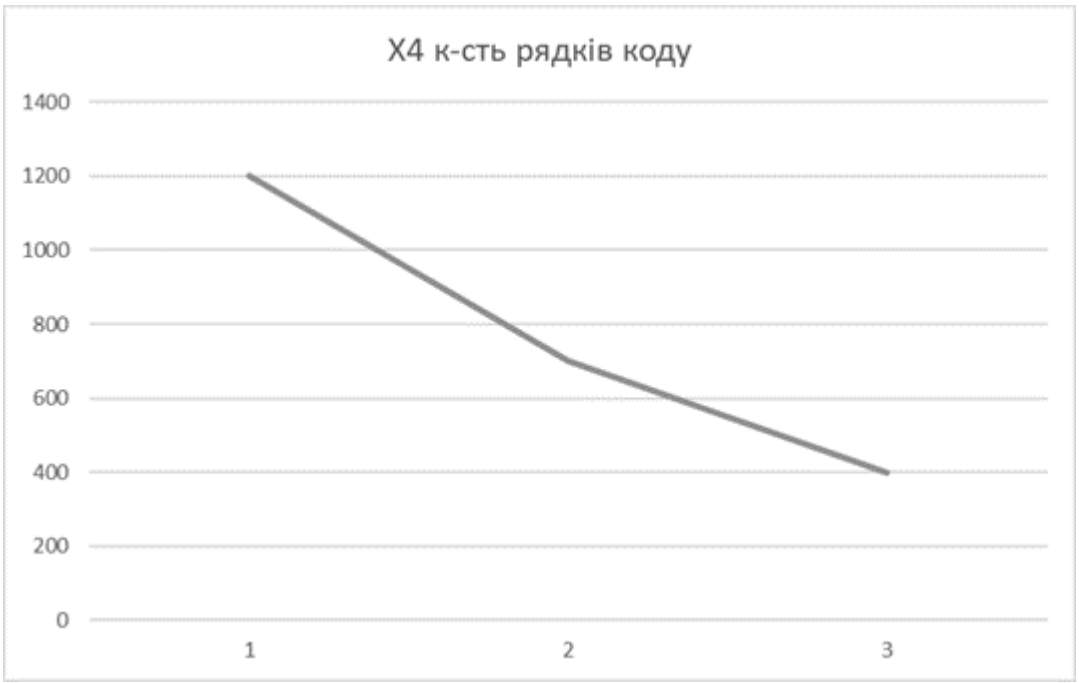

Рисунок Г.14 — X4, потенційний об’єм програмного коду

ДОДАТОК Д

Лістинг програми

```
import os
import cv2
from ultralytics import YOLO
from tkinter import Tk, Button, Label, filedialog, messagebox, Menu
import subprocess
def process_images(image_paths, output_dir, model_path):
    model = YOLO(model_path)
    threshold = 0.5
    area_threshold_yellow = 5000
    area_threshold_red = 20000
    def classify_rectangle(area):
        if area < area_threshold_yellow:
            return (0, 255, 0), "ЗЕЛЕНИЙ"
        elif area < area_threshold_red:
            return (0, 255, 255), "ЖОВТИЙ"
        else:
            return (0, 0, 255), "ЧЕРВОНИЙ"
    output_paths = []
    for path in image_paths:
        image = cv2.imread(path)
        if image is None:
            print(f"Не вдалося відкрити {path}")
            continue
        results = model(image)[0]
        for box in results.boxes.data.tolist():
            x1, y1, x2, y2, score, class_id = box
            if score > threshold:
                area = (x2 - x1) * (y2 - y1)
                class_name = results.names[int(class_id)]
                color, label = classify_rectangle(area)
                if class_name.lower() in ['damaged_road_marking', 'road_line_damaged',
'пошкоджена_розмітка']:
                    color = (255, 0, 0)
                    label = "РОЗМІТКА-ПОШКОДЖЕНА"
                cv2.rectangle(image, (int(x1), int(y1)), (int(x2), int(y2)), color, 4)
                text = f"{class_name.upper()} {score:.2f} {label}"
                cv2.putText(image, text, (int(x1), int(y1 - 10)),
                            cv2.FONT_HERSHEY_SIMPLEX, 1.0, color, 2, cv2.LINE_AA)
                coords = f"({int(x1)}, {int(y1)}), ({int(x2)}, {int(y2)})"
                cv2.putText(image, coords, (int(x1), int(y2) + 30),
```

```
                    cv2.FONT_HERSHEY_SIMPLEX, 0.6, (255, 255, 255), 2,
cv2.LINE_AA)
        base = os.path.basename(path)
        name, ext = os.path.splitext(base)
        output_path = os.path.join(output_dir, f"{name}_out{ext}")
        cv2.imwrite(output_path, image)
        output_paths.append(output_path)
    return output_paths
class ImageApp:
    def __init__(self, root):
        self.root = root
        self.root.title("Виявлення пошкоджень дороги (пакет зображень)")
        self.image_paths = []
        self.output_paths = []
        self.model_path = os.path.join('.', 'runs', 'detect', 'yolov8n_v8_50e14', 'weights',
'last.pt')
        # Меню
        menu_bar = Menu(root)
        file_menu = Menu(menu_bar, tearoff=0)
        file_menu.add_command(label="Завантажити зображення",
command=self.load_images)
        file_menu.add_command(label="Обробити зображення",
command=self.process_images)
        file_menu.add_command(label="Переглянути результати",
command=self.show_images)
        file_menu.add_command(label="Зберегти зображення",
command=self.save_images)
        file_menu.add_separator()
        file_menu.add_command(label="Вийти", command=root.quit)
        menu_bar.add_cascade(label="Меню", menu=file_menu)
        model_menu = Menu(menu_bar, tearoff=0)
        model_menu.add_command(label="Вибрати модель",
command=self.choose_model)
        menu_bar.add_cascade(label="Модель", menu=model_menu)
        root.config(menu=menu_bar)
        # Статус
        self.status = Label(root, text="Готово", bd=1, relief="sunken", anchor="w")
        self.status.pack(side="bottom", fill="x")
    def load_images(self):
        self.image_paths = filedialog.askopenfilenames(filetypes=[("Зображення",
"*.jpg;*.jpeg;*.png")])
        if self.image_paths:
            self.status.config(text=f"Завантажено: {len(self.image_paths)} зображень")
```

```
    def choose_model(self):
        self.model_path = filedialog.askopenfilename(filetypes=[("YOLOv8 .pt файл",
"*.pt")])
        if self.model_path:
            self.status.config(text=f"Модель вибрана:
{os.path.basename(self.model_path)}")
    def process_images(self):
        if not self.image_paths:
            messagebox.showwarning("Увага", "Спочатку завантажте зображення")
            return
        output_dir = os.path.dirname(self.image_paths[0])
        self.output_paths = process_images(self.image_paths, output_dir, self.model_path)
        self.status.config(text=f"Оброблено: {len(self.output_paths)} зображень")
        messagebox.showinfo("Готово", f"Оброблено: {len(self.output_paths)}
зображень")
    def save_images(self):
        if not self.output_paths:
            messagebox.showwarning("Увага", "Немає оброблених зображень для
збереження")
            return
        save_dir = filedialog.askdirectory()
        if save_dir:
            for path in self.output_paths:
                filename = os.path.basename(path)
                os.rename(path, os.path.join(save_dir, filename))
            self.status.config(text=f"Збережено до: {save_dir}")
            messagebox.showinfo("Готово", f"Зображення збережено до: {save_dir}")
    def show_images(self):
        if not self.output_paths:
            messagebox.showwarning("Увага", "Немає зображень для перегляду")
            return
        for path in self.output_paths:
            try:
                subprocess.run(['start', path], shell=True)
            except Exception as e:
                messagebox.showerror("Помилка", f"Не вдалося відкрити {path}: {e}")
if __name__ == "__main__":
    root = Tk()
    app = ImageApp(root)
    root.mainloop()
```