Міністерство освіти у науки України
Харіквський національний університет імені В. Н. Каразіна

Навчально-науковий інститут комп'ютерних наук та штучного інтелекту

Кафедра математичного моделювання та аналізу даних

До захисту допущено
Кафедрою математичного моделювання та аналізу даних
протокол №___ від _____

Завідувач кафедри _____ Володимир СТРУКОВ
(підпис) (імя, прізвище)

«___» _____ 2026 р.

Кваліфікаційна робота
здобувача першого (бакалаврського) рівня вищої освіти

РОЗРОБКА НЕЙРОМЕРЕЖЕВОЇ СИСТЕМИ ДЕТЕКЦІЇ ДОРОЖНІХ ЗНАКІВ НА БАЗІ АРХІТЕКТУР YOLOV8 ТА FASTVIT

Спеціальність (спеціалізація) 122 Комп'ютерні науки

Освітня програма Комп'ютерні системи та мережі

Виконавець _____ Максим МОСКАЛЕЦЬ
(підпис) *(ім'я, прізвище)*

Науковий керівник _____ Константін ЛИСИЦЬКИЙ
(підпис) *(ім'я, прізвище)*

Харків – 2026

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Харківський національний університет імені В.Н.Каразіна

Навчально-науковий інститут комп'ютерних наук та штучного інтелекту

Кафедра математичного моделювання та аналізу даних

ЗАТВЕРДЖУЮ
Завідувач кафедри МДтАД

_____Володимир СТРУКОВ
підпис ім'я, прізвище

"____" _____2026 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувача першого (бакалаврського) рівня вищої освіти

Москалець Максима Анатолійовича

Спеціальність (спеціалізація) 122 «Комп'ютерні науки», _____

Освітня програма Комп'ютерні системи та мережі _____

1. Тема роботи: Розробка нейромережевої системи детекції дорожніх знаків на базі архітектур YOLOv8 та FastViT

керівник роботи ЛИСИЦЬКИЙ К.Є., доцент кафедри математичного моделювання та аналізу даних

затверджені наказом по Університету від 29 січня 2026 року № 4101-5/225

2. Строк здобувачем роботи "____" _____2026 року

3. Вихідні дані до роботи:

1. Набір даних для детекції: Tsinghua-Tencent 100K (TT100K), що містить анотовані зображення дорожніх знаків у реальних умовах;
2. Базові архітектури моделей: YOLOv8 (одностадійний детектор) та FastViT (гібридна архітектура виділення ознак);
3. Мова розробки та інструментарій: Python, фреймворки глибокого навчання PyTorch, Ultralytics, TorchVision, TorchMetrics;

4. Перелік питань, які потрібно розробити:

1. ознайомитись з особливостями задачі детекції та розпізнавання дорожніх знаків у складних умовах реальної дорожньої сцени;
2. провести огляд існуючих методів комп'ютерного зору та сучасних нейромережевих архітектур (одностадійних, двостадійних, гібридних) для виявлення об'єктів;
3. познайомитися з алгоритмом роботи і методиками виділення ознак та регресії координат у сучасних детекторах, створити модель (нейромережеву систему) детекції дорожніх знаків на базі архітектур YOLOv8 та FastViT;
4. провести програмну реалізацію та експериментальне дослідження розроблених моделей, виконати порівняльний аналіз їхньої ефективності за метриками якості локалізації та швидкодії.

5. План роботи

№ з/п	Назви етапів роботи	Строк виконання етапів роботи	Примітка
1	Формулювання мети та повного переліку завдань для виконання кваліфікаційної роботи.	13.01.2026 – 28.01.2026	Виконано
2	Пошук та аналіз інформаційних джерел за темою КР.	28.01.2026 – 18.02.2026	Виконано
3	Проєктування нейромережевої системи на базі архітектур YOLOv8 та FastViT.	18.02.2026 – 16.03.2026	Виконано
4	Програмна реалізація моделей та підготовка датасету TT100K.	16.03.2026 – 01.04.2026	Виконано
5	Проведення експериментів та порівняльний аналіз результатів.	01.04.2026 – 17.04.2026	Виконано
6	Підсумок отриманих результатів та оформлення пакету документів для процедури захисту КР в екзаменаційній комісії ННІ КН та ШІ.	17.04.2026 – 01.05.2026	Виконано

6. Дата видачі завдання 13 січня 2026 р.

Здобувач вищої освіти _____ Москалець М.А.
(підпис) (ініціали, прізвище)

Керівник роботи _____ Лисицький К.Є.
(підпис) (ініціали, прізвище)

АНОТАЦІЯ

Пояснювальна записка до кваліфікаційної роботи складається зі вступу, трьох розділів, висновків, списку використаних джерел та 2 додатків. Загальний обсяг роботи складає 82 сторінок, із яких 74 сторінок основної частини з 14 рисунками, 2 таблицями, 20 найменуваннями списку використаних джерел та 2 додатками.

Метою дослідження є підвищення ефективності детекції та розпізнавання дорожніх знаків і забезпечення високої швидкодії системи в реальному часі шляхом створення та дослідження нейромережевих моделей на основі архітектур глибокого навчання.

Об’єкт дослідження – процеси автоматизованого виявлення та розпізнавання дорожніх знаків на цифрових зображеннях у реальних умовах дорожньої сцени.

Предмет дослідження – моделі побудови нейромережевих систем детекції дорожніх знаків на базі архітектур YOLOv8 та FastViT.

Проблема, яка вирішується в роботі, полягає у забезпеченні високої точності локалізації дрібних об’єктів та стабільної швидкості інференсу в складних реальних умовах (змінне освітлення, погодні фактори, часткові перекриття) шляхом використання моделей глибокого навчання.

Область застосування – інтелектуальні транспортні системи, автономний транспорт, розробка систем допомоги водієві (ADAS) та автоматизований моніторинг дорожньої інфраструктури.

Ключові слова: ДЕТЕКЦІЯ ОБ’ЄКТА, ДОРОЖНІЙ ЗНАК, КОМП’ЮТЕРНИЙ ЗІР, ГЛИБОКЕ НАВЧАННЯ, НЕЙРОННІ МЕРЕЖІ, YOLOv8, FASTVIT, ІНФЕРЕНС, ІНТЕЛЕКТУАЛЬНА ТРАНСПОРТНА СИСТЕМА.

ABSTRACT

The explanatory note to the qualification work consists of an introduction, three chapters, conclusions, a list of references, and 2 appendices. The total volume of the work is 82 pages, of which 74 pages of the main part contain 14 figures, 2 tables, 20 reference items, and 2 appendices.

The purpose of the study is to increase the efficiency of traffic sign detection and recognition and to ensure high system performance in real-time by creating and investigating neural network models based on deep learning architectures.

The object of the study is the processes of automated detection and recognition of traffic signs on digital images in real road scene conditions.

The subject of the study is the models for building neural network traffic sign detection systems based on YOLOv8 and FastViT architectures.

The problem solved in the work is to ensure high accuracy of small object localization and stable inference speed in complex real conditions (variable lighting, weather factors, partial occlusions) using deep learning models.

The area of application is intelligent transport systems, autonomous vehicles, development of advanced driver-assistance systems (ADAS), and automated monitoring of road infrastructure.

Keywords: OBJECT DETECTION, TRAFFIC SIGN, COMPUTER VISION, DEEP LEARNING, NEURAL NETWORKS, YOLO, FASTVIT, INFERENCE, INTELLIGENT TRANSPORT SYSTEM.

ЗМІСТ

ВСТУП

У сучасних умовах стрімкого розвитку інтелектуальних транспортних систем, систем підтримки водія та засобів комп’ютерного зору все більшої актуальності набуває автоматизований аналіз дорожньої сцени. Одним із ключових завдань у цій сфері є своєчасне та точне виявлення дорожніх знаків, оскільки саме вони містять важливу регламентувальну, заборонну, попереджувальну та інформаційну інформацію, необхідну для безпечного руху транспортного засобу. В умовах реального дорожнього середовища водій або автоматизована система повинні швидко інтерпретувати візуальні об’єкти на дорозі, тому задача побудови ефективної системи детекції дорожніх знаків є важливою науково-прикладною проблемою [1; 3].

Актуальність роботи: Серед прикладних областей, у яких зазначена проблема є особливо важливою, слід виокремити системи допомоги водієві, автономний транспорт, навігаційні системи, інтелектуальні комплекси відеоспостереження та засоби моніторингу дорожньої інфраструктури. Специфіка цієї задачі полягає в тому, що дорожні знаки часто мають невеликий розмір у кадрі, можуть бути частково перекриті іншими об’єктами, спотворені перспективою, змінені умовами освітлення або погодними факторами. Додаткову складність становить велика кількість класів знаків і висока візуальна подібність між окремими категоріями. У таких умовах традиційні методи комп’ютерного зору, що базуються на кольоровій сегментації, виділенні контурів або ручному описі ознак, демонструють обмежену стійкість та недостатню якість узагальнення [1; 2].

Сучасний розвиток методів глибокого навчання відкрив нові можливості для розв’язання задачі детекції дорожніх знаків. Особливий інтерес становлять нейромережеві архітектури, здатні автоматично виділяти інформативні ознаки із зображень і виконувати локалізацію та класифікацію об’єктів у межах єдиного обчислювального процесу. Серед таких архітектур важливе місце

займають моделі сімейства YOLO, що орієнтовані на високу швидкодію, а також сучасні гібридні архітектури, зокрема FastViT, які поєднують переваги згорткових мереж і трансформерних підходів [3; 5; 11]. У зв'язку з цим задача розробки нейромережевої системи детекції дорожніх знаків на базі архітектур YOLOv8 та FastViT і проведення їх експериментального порівняння є актуальною як з наукової, так і з практичної точки зору.

Метою дослідження є розробка та дослідження нейромережевої системи детекції дорожніх знаків на базі архітектур YOLOv8 та FastViT, а також експериментальне порівняння їх ефективності за показниками якості детекції та продуктивності.

Об'єкт дослідження — процеси автоматизованого виявлення та розпізнавання дорожніх знаків на цифрових зображеннях у реальних умовах дорожньої сцени.

Методи дослідження: методи системного аналізу та комп'ютерного зору; методи глибокого навчання та детекції об'єктів; архітектури згорткових нейронних мереж і гібридних моделей на основі механізмів уваги; методи попередньої обробки та організації наборів даних для задач детекції; методи оцінювання якості детекційних моделей за метриками Precision, Recall, F1-score, mAP@0.5, mAP@0.5:0.95; методи оцінювання продуктивності моделей за показниками затримки інференсу, FPS, часу навчання та кількості параметрів [15; 18].

Предмет дослідження — моделі, алгоритми та програмні засоби побудови нейромережевих систем детекції дорожніх знаків на базі архітектур YOLOv8 та FastViT.

Завдання дослідження:

1. Виконати аналіз існуючих підходів до детекції та розпізнавання дорожніх знаків, дослідити їх переваги, недоліки та обґрунтувати доцільність використання сучасних нейромережевих архітектур.

2. Проаналізувати архітектурні особливості моделей YOLOv8 і FastViT та спроєктувати структуру нейромережевої системи детекції дорожніх знаків.
3. Обґрунтувати вибір датасету, метрик оцінювання та схеми експериментального дослідження для коректного порівняння моделей.
4. Виконати програмну реалізацію системи, провести навчання, валідацію та тестування моделей YOLOv8 і FastViT, а також здійснити порівняльний аналіз їх ефективності за обраними метриками.

Практичне значення одержаних результатів полягає в тому, що розроблена система може бути використана як основа для створення прикладних рішень у сфері інтелектуальних транспортних систем, систем підтримки водія та автоматизованого аналізу дорожньої сцени. Отримані результати також можуть бути використані для подальшого вдосконалення детекційних моделей, орієнтованих на виявлення дрібних дорожніх об’єктів у складних реальних умовах.

Структура роботи: Кваліфікаційна робота складається зі вступу, трьох розділів, висновків, списку використаних джерел і додатків. У першому розділі проаналізовано задачу детекції дорожніх знаків і сучасні підходи до її розв’язання. У другому розділі розглянуто архітектурні особливості моделей YOLOv8 і FastViT та спроєктовано структуру нейромережевої системи. У третьому розділі описано програмну реалізацію системи, підготовку даних, процес навчання моделей і результати їх порівняльного аналізу.

1. АНАЛІЗ ЗАДАЧІ ДЕТЕКЦІЇ ДОРОЖНІХ ЗНАКІВ ТА ПІДХОДІВ ДО ЇЇ РОЗВ'ЯЗАННЯ

1.1. Постановка задачі детекції та розпізнавання дорожніх знаків

Задача детекції та розпізнавання дорожніх знаків є однією з ключових у сфері комп'ютерного зору, інтелектуальних транспортних систем і систем підтримки водія. Її актуальність зумовлена потребою в автоматизованому аналізі дорожньої обстановки, підвищенні безпеки руху, зменшенні ймовірності помилок водія та забезпеченні коректної роботи навігаційних і допоміжних модулів у транспортних засобах. У сучасних умовах дорожні знаки виступають важливими носіями регламентувальної, попереджувальної та інформаційної функції, тому їх своєчасне виявлення та правильне розпізнавання мають безпосередній вплив на прийняття рішень під час руху [1; 3].

У загальному вигляді задача розпізнавання дорожніх знаків може бути поділена на дві взаємопов'язані підзадачі: детекцію та класифікацію. Детекція передбачає пошук на зображенні або у відеопотоці областей, у яких розташовані дорожні знаки, тобто визначення їх просторового положення. Результатом цього етапу є координати обмежувальних прямокутників, що локалізують відповідні об'єкти. Класифікація, своєю чергою, полягає у віднесенні виявленого об'єкта до певного класу, наприклад заборонного, попереджувального, інформаційного знака або до конкретного типу знака з установленого набору категорій. У сучасних нейромережевих системах ці два етапи можуть виконуватися або послідовно, або в межах єдиної архітектури, що одночасно визначає і місце розташування об'єкта, і його клас [3; 4].

Формально задачу детекції дорожніх знаків можна описати так: для вхідного зображення (I) необхідно визначити множину об'єктів ($O = \{o_1, o_2, \ldots, o_n\}$), де кожен об'єкт характеризується координатами обмежувального прямокутника ($B_i = (x_i, y_i, w_i, h_i)$), міткою класу ($c_i$) та, за

потреби, оцінкою впевненості моделі (p_i). Таким чином, система має не просто знайти факт наявності дорожнього знака, а видати структурований результат, придатний для подальшої обробки іншими модулями інформаційної системи. У випадку аналізу відеопотоку задача ускладнюється тим, що обробка має здійснюватися в режимі, близькому до реального часу, без суттєвого зниження точності [3; 15].

На практиці постановка задачі детекції дорожніх знаків суттєво ускладнюється через особливості реального дорожнього середовища. По-перше, дорожні знаки часто займають невелику частину кадру, особливо на великій відстані від камери, тому належать до категорії дрібних об’єктів. По-друге, їх зовнішній вигляд змінюється залежно від освітлення, кута зйомки, погодних умов, часткового перекриття іншими об’єктами, зносу поверхні, забруднення або пошкодження. По-третє, у межах одного класу можуть спостерігатися незначні варіації вигляду, тоді як між різними класами окремі знаки можуть бути дуже схожими за формою чи кольором. Це підвищує вимоги як до просторової точності локалізації, так і до дискримінаційної здатності моделі [1; 2].

Важливим аспектом постановки задачі є також вибір середовища, в якому система повинна функціонувати. Якщо йдеться про стаціонарний аналіз фотоархівів або автономну обробку наборів даних, основна увага приділяється точності виявлення та стійкості до різних типів спотворень. Якщо ж система орієнтована на використання в транспортних засобах або мобільних пристроях, на перший план виходить компроміс між точністю та швидкодією. У такому випадку недостатньо лише правильно виявити об’єкт, необхідно зробити це за обмежений час, використовуючи доступні обчислювальні ресурси. Саме тому при постановці задачі доцільно враховувати не тільки показники точності, а й швидкість інференсу, кількість параметрів моделі та її придатність до практичного застосування [3; 4; 15].

Для дослідження задачі детекції дорожніх знаків широко використовуються спеціалізовані набори даних, які містять анотовані зображення дорожньої сцени. Одним із відомих прикладів є датасет Tsinghua-Tencent 100K, орієнтований на виявлення дорожніх знаків у реальних умовах. Його особливістю є велика кількість класів, значна варіативність розмірів об’єктів, складність сцен та наявність дрібних знаків, що робить його придатним для оцінювання сучасних моделей детекції [1; 2]. Використання такого набору даних дозволяє наблизити експеримент до реальних умов функціонування системи та сформулювати задачу не в абстрактному, а в прикладному контексті.

У межах цієї кваліфікаційної роботи задача формулюється як розробка нейромережевої системи, що здатна автоматично виявляти дорожні знаки на цифрових зображеннях, локалізувати їх і визначати належність до одного з наперед заданих класів. Система повинна забезпечувати прийнятну точність детекції для знаків різного масштабу та виду, а також достатню швидкодію для практичного використання. Для досягнення цієї мети доцільно розглянути дві різні архітектурні концепції: швидкі одностадійні детектори та гібридні підходи, що поєднують переваги згорткових мереж і трансформерів. Саме це і зумовлює вибір моделей YOLOv8 та FastViT як основи подальшого дослідження [5; 11].

Окремо слід зазначити, що в задачах детекції дорожніх знаків оцінювання результатів не може обмежуватися лише фактом правильної класифікації. Не менш важливим є те, наскільки точно модель визначає координати знайденого об’єкта. Тому для оцінювання якості таких систем традиційно використовуються метрики Precision, Recall, F1-score та mean Average Precision, зокрема mAP@0.5 і mAP@0.5:0.95, які враховують як правильність віднесення об’єкта до певного класу, так і точність його локалізації [10; 18]. З огляду на це постановка задачі у даній роботі передбачає

не лише створення моделі, а й проведення порівняльного аналізу її ефективності за сукупністю кількісних показників.

Отже, задача детекції та розпізнавання дорожніх знаків є складною багатофакторною задачею комп'ютерного зору, що поєднує проблеми локалізації дрібних об'єктів, їх класифікації в складних реальних умовах та забезпечення високої швидкодії системи. Її розв'язання потребує використання сучасних методів глибокого навчання, ретельного вибору набору даних, коректних метрик оцінювання та обґрунтованого вибору архітектур, придатних до практичного застосування в інтелектуальних транспортних системах [1; 5; 15].

1.2. Класичні та нейромережеві підходи до виявлення дорожніх знаків

Розв'язання задачі виявлення дорожніх знаків історично розвивалося від класичних алгоритмів комп'ютерного зору до сучасних нейромережевих моделей глибокого навчання. Такий перехід зумовлений тим, що дорожні сцени є складними, динамічними та містять значну кількість факторів, які ускладнюють локалізацію об'єктів: змінне освітлення, погодні умови, часткове перекриття, різні масштаби знаків у кадрі, фонові об'єкти схожих кольорів і форм. У реальних наборах даних, зокрема в TT100K, ці фактори особливо виражені, що робить задачу детекції дорожніх знаків набагато складнішою, ніж розпізнавання об'єктів у штучно підготовлених умовах [1; 2].

До класичних підходів належать методи, що базуються на попередньо заданих правилах і вручну сформованих ознаках. У задачах виявлення дорожніх знаків найчастіше використовувалися кольорова сегментація, аналіз геометричної форми, порогова обробка, пошук контурів, шаблонне зіставлення, а також дескриптори ознак на кшталт HOG, SIFT або Haar-подібних ознак у поєднанні з традиційними класифікаторами, наприклад SVM

чи каскадами. Логіка таких методів полягає в тому, що дорожні знаки зазвичай мають характерні кольори, відносно стандартні форми та чіткі межі, що дозволяє спочатку виділити потенційні області інтересу, а потім перевірити їх на належність до певного класу. Перевагою класичних методів є відносна простота реалізації, менші вимоги до обчислювальних ресурсів та зрозуміла інтерпретованість окремих етапів обробки. Однак їх ефективність суттєво знижується в реальних умовах, коли форма спотворюється перспективою, колір змінюється через освітлення, а фон містить велику кількість завад [1].

Головним недоліком класичних підходів є слабка узагальнювальна здатність. Такі методи, як правило, добре працюють лише за умови, що сцена близька до тієї, під яку вони налаштовувалися. Якщо ж змінюється освітлення, контрастність, роздільна здатність кадру або масштаб знака, якість виявлення помітно падає. Крім того, каскадні системи з окремими етапами сегментації, відбору кандидатних областей і класифікації накопичують похибки: помилка на початковому етапі часто призводить до того, що знак узагалі не потрапляє на вхід класифікатора. Саме тому класичні методи поступово втратили домінуючу роль у складних прикладних задачах детекції [1; 2].

Новий етап розвитку пов’язаний із застосуванням нейромережевих підходів, насамперед згорткових нейронних мереж. На відміну від класичних алгоритмів, глибокі моделі не потребують ручного проєктування ознак, а навчаються самостійно виділяти інформативні просторові представлення безпосередньо з даних. Це особливо важливо для дорожніх знаків, оскільки ознаки, корисні для їх виявлення, можуть змінюватися залежно від масштабу, фону, кута огляду та стану самого знака. Нейромережеві моделі дозволили перейти від набору окремих евристичних процедур до єдиних trainable-систем, що оптимізуються за детекційними метриками та краще пристосовуються до складних природних сцен [3; 4].

Сучасні нейромережеві підходи до виявлення об’єктів умовно поділяють на дві великі групи: двостадійні та одностадійні. Двостадійні

методи, типовим представником яких є Faster R-CNN, спочатку формують набір кандидатних областей, що потенційно містять об’єкти, а потім уточнюють їх межі та виконують класифікацію. Такий підхід зазвичай забезпечує високу точність локалізації, особливо в задачах зі складним фоном або об’єктами малого розміру, але часто є повільнішим через багатокрокову структуру обробки [4]. Натомість одностадійні детектори, до яких концептуально належить сімейство YOLO, розглядають задачу виявлення як єдиний процес прямої регресії координат рамок та ймовірностей класів. Завдяки цьому вони значно краще підходять для роботи в реальному часі, що є критично важливим для транспортних систем і вбудованих застосувань [3].

Переваги одностадійних моделей полягають у високій швидкодії, простішій інтеграції в прикладні системи та ефективній роботі на GPU. Саме тому вони широко застосовуються в задачах, де важливий баланс між точністю та часом обробки кадру. Водночас двостадійні архітектури залишаються конкурентними в тих випадках, коли пріоритетом є якість локалізації та здатність точніше працювати з дрібними об’єктами. У роботах з порівняння детекторів на наборах дорожніх знаків і світлофорів показано, що жодна архітектура не є універсально найкращою за всіма критеріями: вибір моделі залежить від цільового сценарію використання, доступних ресурсів та вимог до затримки інференсу [2; 4].

Подальший розвиток нейромережевих підходів пов’язаний із використанням трансформерних і гібридних архітектур. Якщо згорткові мережі добре вловлюють локальні патерни, то трансформерні моделі надають кращі можливості для врахування глобального контексту зображення. Для задач детекції дорожніх знаків це може бути корисним у складних сценах, де об’єкт маленький, частково перекритий або присутній на фоні великої кількості схожих структур. Одним із сучасних прикладів такого підходу є FastViT, який поєднує переваги ефективних візуальних трансформерів і

структурної репараметризації, орієнтуючись на поєднання точності та швидкодії [5; 6; 7].

Таким чином, класичні підходи до виявлення дорожніх знаків мають історичне та методичне значення, але в умовах реальних дорожніх сцен поступаються сучасним нейромережевим системам за стійкістю та якістю узагальнення. Нейромережеві моделі, зокрема одностадійні детектори на зразок YOLO та сучасні гібридні архітектури на зразок FastViT, забезпечують значно вищий потенціал для побудови прикладних систем детекції дорожніх знаків. Саме тому в подальшому дослідженні доцільно зосередитися на порівнянні сучасних архітектур, що репрезентують різні підходи до розв'язання задачі: швидкий одностадійний детектор і гібридну модель, побудовану на сучасних механізмах візуального представлення [3; 5].

1.3. Сучасні архітектури детекції об'єктів, їх переваги та недоліки

Сучасні архітектури детекції об'єктів формують окремий напрям розвитку комп'ютерного зору, у межах якого поєднуються задачі локалізації об'єкта на зображенні, визначення його класу та оцінювання впевненості прогнозу. На відміну від задачі звичайної класифікації, де модель повинна лише віднести зображення до певної категорії, детекція вимагає значно складнішого результату: встановити, які саме об'єкти присутні на сцені, де вони розташовані та до яких класів належать. Саме тому детекційні архітектури розвивалися як спеціалізовані моделі, що враховують просторову структуру сцени, багатомасштабність об'єктів та необхідність точної локалізації [4; 9].

У загальному вигляді сучасні архітектури детекції об'єктів доцільно поділяти на кілька основних груп: двостадійні моделі, одностадійні моделі та трансформерно-орієнтовані або гібридні архітектури. Кожна з цих груп має власну логіку побудови, переваги та обмеження. Вибір конкретного підходу

визначається не лише бажаною точністю, а й вимогами до швидкодії, складності реалізації, обсягу доступних обчислювальних ресурсів і характеру самих об’єктів, які потрібно виявляти [2; 4].

До двостадійних архітектур належать моделі, в яких процес детекції розбивається на два логічні етапи. На першому етапі формується набір кандидатних областей, які потенційно можуть містити об’єкти. На другому етапі ці області уточнюються, класифікуються та отримують фінальні координати рамок. Найвідомішим представником такого підходу є Faster R-CNN, у якому для генерації кандидатних областей використовується Region Proposal Network [4]. Перевагою двостадійних моделей є висока точність локалізації, краща робота зі складними сценами та можливість більш детального аналізу кожної знайденої області. Це особливо важливо у випадках, коли об’єкти невеликі, частково перекриті або мають складний фон.

Разом із тим двостадійні архітектури мають і суттєві недоліки. Насамперед це вища обчислювальна складність і більший час обробки одного зображення порівняно з одностадійними моделями. Через наявність кількох етапів обчислення та обробки регіонів інтересу такі системи зазвичай гірше підходять до сценаріїв реального часу. Для транспортних систем, де рішення повинні прийматися швидко, ця особливість може бути критичною. Крім того, двостадійні моделі, як правило, вимагають складнішого налаштування та більших ресурсів під час навчання [2; 4].

Іншу велику групу становлять одностадійні архітектури, у яких локалізація та класифікація виконуються в межах одного проходу мережею. Саме цей підхід став основою для сімейства YOLO, яке розглядає задачу детекції як задачу прямої регресії координат рамок і класів об’єктів. Перевага одностадійних моделей полягає в їх високій швидкодії та кращій придатності до використання в режимі реального часу [3]. У сучасних версіях таких детекторів, зокрема в YOLOv8, ця перевага поєднується з достатньо високою

точністю, що робить їх особливо привабливими для прикладних систем відеоаналізу та транспортних застосувань [11; 12].

Архітектура YOLOv8 є представником сучасного покоління одностадійних детекторів, які поєднують компактність, ефективність та практичну орієнтованість. Серед її сильних сторін можна виділити відносно невелику кількість параметрів, високу швидкість інференсу, зручний інструментарій навчання та валідації, а також підтримку гнучкого налаштування тренувального процесу [11; 14]. Крім того, сучасні реалізації YOLO містять засоби для автоматизованого аналізу метрик, бенчмаркінгу та оптимізації продуктивності, що полегшує їх застосування як у дослідницьких, так і в інженерних задачах [15; 16]. Недоліком одностадійних моделей можна вважати те, що у складних умовах вони іноді поступаються двостадійним за точністю локалізації дрібних або візуально подібних об’єктів, особливо якщо модель недостатньо добре адаптована до конкретного набору даних.

Окрему групу сучасних підходів утворюють трансформерні та гібридні архітектури. Їх поява пов’язана з розвитком механізмів уваги, які спочатку були запропоновані для обробки послідовностей, а згодом успішно адаптовані до задач комп’ютерного зору [6]. На відміну від класичних згорткових мереж, трансформери краще моделюють глобальні взаємозв’язки між різними частинами зображення. Для задачі детекції це означає можливість ефективніше враховувати контекст сцени, що може бути корисним при виявленні об’єктів малого розміру, частково перекритих елементів або структурно складних композицій [7]. Утім, чисто трансформерні моделі часто є важчими, потребують більше даних і обчислювальних ресурсів, що обмежує їх застосування в системах реального часу.

Саме тому останніми роками значного поширення набули гібридні архітектури, які поєднують сильні сторони згорткових мереж і трансформерних механізмів. Одним із прикладів такого підходу є FastViT, у якому ідеї візуальних трансформерів поєднуються з ефективними

архітектурними рішеннями та структурною репараметризацією [5]. Перевагою подібних моделей є прагнення до компромісу між швидкістю та якістю представлення ознак. Вони можуть забезпечити кращу роботу з глобальним контекстом порівняно з суто згортковими мережами, але при цьому залишатися практичнішими за повністю трансформерні системи. До недоліків слід віднести складніший процес реалізації, вищу чутливість до вибору гіперпараметрів і, в окремих випадках, більші вимоги до часу навчання.

При аналізі сучасних архітектур важливо враховувати і базові мережі виділення ознак, або backbone-мережі. Розвиток таких архітектур, зокрема ResNet, істотно вплинув на подальшу еволюцію детекторів, оскільки саме якість глибинного представлення ознак багато в чому визначає успішність локалізації та класифікації об’єктів [8]. У подальшому ця ідея була розвинена в багатомасштабних підходах, де інформація з різних рівнів мережі використовується для виявлення як великих, так і дрібних об’єктів. Для дорожніх знаків це особливо важливо, оскільки в межах одного кадру можуть одночасно бути присутні як великі добре видимі знаки, так і дуже дрібні об’єкти на дальньому плані.

Ще одним важливим аспектом аналізу сучасних архітектур є оцінювання їх ефективності. Для цього використовуються стандартизовані підходи та набори метрик, запозичені з практики оцінювання детекційних систем на складних наборах даних, зокрема COCO [9]. Найбільш поширеними метриками є Precision, Recall, F1-score, mAP@0.5 та mAP@0.5:0.95, які дозволяють оцінити не лише факт виявлення об’єкта, а й точність його локалізації [10; 18]. Таким чином, сучасна архітектура вважається ефективною не лише тоді, коли вона демонструє високу точність на окремих класах, а й коли забезпечує належний баланс між точністю, швидкодією, стабільністю та обчислювальною вартістю.

Отже, сучасні архітектури детекції об’єктів репрезентують кілька різних підходів до розв’язання однієї й тієї самої задачі. Двостадійні моделі зазвичай

орієнтуються на високу точність і детальнішу локалізацію, одностадійні моделі забезпечують вищу швидкість і кращу придатність до реального часу, а гібридні трансформерно-орієнтовані архітектури намагаються знайти баланс між локальною ефективністю згорткових мереж і глобальною контекстною чутливістю механізмів уваги. Саме тому в рамках дослідження детекції дорожніх знаків доцільно порівнювати архітектури, що репрезентують різні концепції побудови детекторів, і оцінювати їх не ізольовано, а в контексті конкретної прикладної задачі [2; 5; 15].

1.4. Напрями розвитку систем детекції дорожніх знаків і обґрунтування вибору підходу

Системи детекції дорожніх знаків постійно еволюціонують під впливом двох взаємопов’язаних факторів: ускладнення практичних умов використання та швидкого розвитку архітектур глибокого навчання. Якщо на ранніх етапах основною метою було саме автоматизувати процес виявлення знаків на зображенні, то сьогодні на перший план виходять питання точності в реальних умовах, стабільності до шумів і спотворень, обробки дрібних об’єктів, зменшення затримки інференсу та адаптації моделей до ресурсних обмежень. Саме тому сучасний розвиток систем детекції дорожніх знаків відбувається не лише в напрямі підвищення якості виявлення, а й у напрямі пошуку збалансованих архітектур, придатних до практичного впровадження [1; 2].

Одним із головних напрямів розвитку є підвищення якості детекції дрібних об’єктів. Дорожні знаки часто займають дуже малу площу кадру, особливо коли камера встановлена в транспортному засобі, а об’єкти знаходяться на значній відстані. Для таких випадків важливими стають архітектури, здатні працювати з багатомасштабними ознаками, зберігати деталі на ранніх рівнях мережі та ефективно інтегрувати інформацію з різних глибинних представлень. Саме ця вимога пояснює розвиток мереж із багатомасштабною обробкою та покращеними механізмами інтеграції ознак,

оскільки звичайна класифікаційна мережа без адаптації до задачі детекції часто виявляється недостатньо чутливою до малих об’єктів [1; 8].

Іншим важливим напрямом є забезпечення роботи в реальному часі. Для транспортних систем затримка між отриманням зображення та видачею результату може мати критичне значення. Саме тому значного поширення набули одностадійні детектори, які зменшують кількість обчислювальних етапів і забезпечують вищу швидкість обробки без надмірної втрати точності. У цьому контексті архітектури сімейства YOLO стали одним із найважливіших практичних рішень, оскільки поєднують зручність навчання, підтримку сучасних засобів оптимізації та високу швидкодію на графічних прискорювачах [3; 11; 12]. Для реальних застосувань саме такий баланс між точністю та продуктивністю часто є визначальним.

Водночас підвищення швидкості не означає відмову від розвитку точнісних характеристик. Сучасні системи дедалі частіше орієнтуються на комбінування локальних і глобальних механізмів виділення ознак. Якщо згорткові мережі добре працюють із локальними текстурними й просторовими патернами, то трансформерні підходи покращують облік глобального контексту сцени, що може бути корисним для складних випадків виявлення. У цьому напрямі особливий інтерес становлять гібридні моделі, які намагаються поєднати сильні сторони обох підходів, не перетворюючись при цьому на надто важкі та повільні системи [5; 6; 7].

Ще одним помітним напрямом є оптимізація архітектур під реальні апаратні обмеження. Якщо раніше багато моделей оцінювалися переважно за абсолютною точністю, то тепер дедалі більшу роль відіграють показники кількості параметрів, швидкості інференсу, використання пам’яті та енергоефективності. Це особливо важливо для вбудованих платформ, мобільних пристроїв і транспортних систем, де обчислювальні ресурси не є необмеженими. Саме тому в дослідженнях дедалі частіше аналізують не лише mAP, а й FPS, latency та співвідношення між якістю та складністю моделі [15;

16]. Такий підхід є більш прикладним і краще відповідає реальним умовам впровадження.

Паралельно розвивається і сам підхід до організації експериментального дослідження. Сучасна практика вимагає не просто показати окремий результат на вибраному прикладі, а забезпечити відтворюваність експерименту, коректний вибір train/validation split, обґрунтування використаних гіперпараметрів і застосування стандартизованих метрик оцінювання [9; 18]. У цьому контексті важливими стають не лише наукові статті, а й офіційні інструменти навчання, валідації та бенчмаркінгу моделей, які дозволяють побудувати більш прозорий та технічно коректний експериментальний процес [13; 14; 17].

Для задачі детекції дорожніх знаків окреме значення має використання спеціалізованих наборів даних, що відображають реальні особливості дорожньої сцени. Саме набори на кшталт TT100K дають змогу оцінювати поведінку моделей не в спрощених лабораторних умовах, а на складних зображеннях із дрібними об’єктами, фоновими перешкодами та значною міжкласовою варіативністю [1; 2]. Отже, напрям розвитку систем детекції дорожніх знаків пов’язаний не тільки з удосконаленням архітектур, а й із підвищенням реалістичності оцінювання та якості експериментальної постановки задачі.

У межах цієї кваліфікаційної роботи доцільно обрати підхід, який дає змогу не лише отримати якісний результат, а й провести змістовне порівняння двох різних архітектурних концепцій. З одного боку, потрібна модель, що добре зарекомендувала себе в задачах детекції об’єктів, має високу швидкодію, доступну реалізацію та розвинену інфраструктуру навчання і валідації. Такою моделлю є YOLOv8, яку доцільно розглядати як практичний базовий детектор для дорожніх знаків [11; 12; 15]. З іншого боку, доцільно включити до дослідження альтернативний підхід, який репрезентує сучасний напрям розвитку архітектур комп’ютерного зору, орієнтований на поєднання

ефективності й покращеного представлення ознак. Такою архітектурною основою є FastViT [5].

Обґрунтування вибору саме YOLOv8 та FastViT полягає в тому, що вони репрезентують два різні, але актуальні напрями розвитку систем детекції. YOLOv8 уособлює еволюцію одностадійних високошвидкісних детекторів, орієнтованих на прикладне використання та роботу в умовах обмеженого часу обробки. FastViT, своєю чергою, відображає сучасний інтерес до гібридних архітектур, які прагнуть поєднати ефективність згорткових підходів із перевагами глобального контекстного аналізу [5; 11]. Порівняння цих підходів у межах однієї предметної задачі дозволяє не лише оцінити їх кількісні характеристики, а й зробити практичні висновки щодо доцільності використання кожної з моделей у системах детекції дорожніх знаків.

Важливо також підкреслити, що вибір підходу в даній роботі здійснюється не за принципом пошуку “універсально найкращої” архітектури, а за принципом зіставлення різних концепцій побудови моделі в умовах спільного датасету, спільної задачі та єдиної системи оцінювання. Такий підхід є більш науково обґрунтованим, оскільки дозволяє оцінювати не абстрактні переваги архітектур, а їх реальну поведінку в конкретному прикладному контексті. Саме це і визначає доцільність побудови дослідження на основі порівняння YOLOv8 та FastViT за сукупністю показників точності, швидкодії та практичної придатності [15; 18].

Отже, сучасний розвиток систем детекції дорожніх знаків відбувається в напрямі підвищення якості виявлення малих об’єктів, забезпечення роботи в реальному часі, інтеграції нових архітектурних рішень та оптимізації моделей під реальні обчислювальні умови. З урахуванням цих тенденцій вибір YOLOv8 і FastViT як основи подальшого дослідження є обґрунтованим, оскільки дозволяє проаналізувати дві різні архітектурні парадигми та оцінити їх потенціал для побудови ефективної системи детекції дорожніх знаків [5; 11; 17].

Висновки до розділу 1

У першому розділі було проаналізовано сутність задачі детекції та розпізнавання дорожніх знаків як однієї з важливих прикладних задач комп'ютерного зору в інтелектуальних транспортних системах, специфіка якої полягає у необхідності одночасного розв'язання двох взаємопов'язаних підзадач: локалізації об'єкта на зображенні та визначення його класу. Складність задачі зумовлюється малим розміром дорожніх знаків у кадрі, впливом освітлення, погодних умов, перспективних спотворень, часткових перекриттів і значною міжкласовою подібністю окремих категорій знаків [1].

У ході аналізу підходів до виявлення дорожніх знаків встановлено, що класичні методи комп'ютерного зору, засновані на кольоровій сегментації, аналізі форми, контурів і ручних дескрипторах ознак, мають обмежену стійкість у реальних умовах дорожнього середовища, натомість нейромережеві підходи, зокрема моделі глибокого навчання, забезпечують вищу якість узагальнення та кращу пристосованість до складних сцен [1; 3; 4].

Також встановлено, що сучасні архітектури детекції об'єктів можна умовно поділити на двостадійні (забезпечують високу точність локалізації, але поступаються за швидкодією), одностадійні (орієнтовані на ефективну роботу в реальному часі) та гібридні трансформерно-орієнтовані, які поєднують переваги згорткових мереж і трансформерів для досягнення балансу між точністю та обчислювальною ефективністю [3; 4; 5].

На основі проведеного аналізу визначено, що для задачі детекції дорожніх знаків доцільно обирати підходи, які враховують не лише точність виявлення, а й швидкість інференсу, кількість параметрів моделі, стійкість до варіативності реальних сцен і придатність до практичного використання. У зв'язку з цим обґрунтовано вибір двох архітектурних підходів для подальшого дослідження: моделі YOLOv8 як представника сучасних швидких одностадійних детекторів та FastViT як представника актуальних гібридних архітектур комп'ютерного зору [5; 11; 15].

2. ОПИС МОДЕЛЕЙ ТА ПРОЄКТУВАННЯ НЕЙРОМЕРЕЖЕВОЇ СИСТЕМИ ДЕТЕКЦІЇ ДОРОЖНІХ ЗНАКІВ

2.1 Обґрунтування вибору моделей для дослідження

У межах даної кваліфікаційної роботи для розв’язання задачі детекції дорожніх знаків було обрано дві різні за архітектурним підходом моделі: YOLOv8 та альтернативну модель детекції на основі FastViT. Такий вибір зумовлений необхідністю порівняння ефективності сучасних підходів до детекції об’єктів, які відрізняються принципами побудови, швидкодією та вимогами до обчислювальних ресурсів.

Модель YOLOv8 належить до класу одностадійних детекторів, які поєднують етапи локалізації та класифікації об’єктів в єдиному нейромережевому проході. Основною перевагою таких моделей є висока швидкість інференсу, що робить їх особливо придатними для задач реального часу, зокрема в інтелектуальних транспортних системах і системах допомоги водієві. Крім того, YOLOv8 є однією з найсучасніших реалізацій сімейства YOLO, яка демонструє високі показники точності при збереженні ефективності обчислень.

У свою чергу, модель на основі FastViT представляє інший підхід до побудови систем детекції, що базується на використанні сучасних гібридних архітектур виділення ознак. FastViT поєднує властивості згорткових нейронних мереж і трансформерів, забезпечуючи ефективне представлення зображення та потенційно кращу узагальнювальну здатність. Використання FastViT як backbone у складі детектора типу Faster R-CNN дозволяє дослідити альтернативний підхід до задачі детекції дорожніх знаків, орієнтований на якість представлення ознак, а не лише на швидкість роботи.

Таким чином, вибір саме цих двох моделей обумовлений прагненням провести порівняльний аналіз між швидким і ефективним одностадійним детектором (YOLOv8) та більш складною, але потенційно гнучкішою

двостадійною системою з сучасним backbone (FastViT). Це дозволяє отримати більш повне уявлення про можливості різних архітектурних підходів у задачі детекції дорожніх знаків.

2.2. Архітектурні особливості моделі YOLOv8

Модель YOLOv8 належить до сучасного покоління одностадійних детекторів об’єктів, призначених для швидкого та точного виявлення об’єктів на зображеннях. Її архітектура орієнтована на досягнення компромісу між точністю локалізації, швидкістю інференсу та обчислювальною ефективністю, що робить її придатною для використання у прикладних системах комп’ютерного зору, зокрема в задачах детекції дорожніх знаків [11; 12].

YOLOv8 реалізує підхід, за якого детекція об’єктів виконується в межах одного проходу мережею. На відміну від двостадійних детекторів, де спочатку генеруються кандидатні області, а потім виконується їх класифікація та уточнення координат, YOLOv8 безпосередньо прогнозує розташування об’єктів і їхні класи з вхідного зображення. Саме це забезпечує високу швидкодію моделі та робить її придатною для сценаріїв, де важлива робота в режимі, близькому до реального часу [3; 11].

Архітектурно модель YOLOv8 складається з трьох основних функціональних блоків: backbone, neck і head. Backbone відповідає за виділення ознак із вхідного зображення. На цьому етапі відбувається поступове зменшення просторової роздільної здатності та водночас збільшення семантичної насиченості ознак. У результаті мережа формує багаторівневі представлення зображення, які містять інформацію про контури, текстури, форми та більш складні візуальні структури. Саме від якості цього етапу значною мірою залежить здатність моделі розрізняти дрібні об’єкти, до яких належать дорожні знаки [11; 14].

У ролі neck у YOLOv8 використовується механізм багатомасштабного агрегування ознак. Його призначення полягає в об’єднанні інформації з різних рівнів глибинних представлень, отриманих у backbone. Це особливо важливо для детекції об’єктів різного масштабу, оскільки великі об’єкти краще описуються глибокими семантичними ознаками, тоді як для дрібних об’єктів важливо зберігати детальніші просторові характеристики. У задачі детекції дорожніх знаків така багатомасштабність має принципове значення, оскільки в межах одного кадру знаки можуть суттєво відрізнятися за розміром залежно від відстані до камери [1; 11].

Фінальним блоком є detection head, який формує прогноз моделі. На цьому етапі для кожного об’єкта визначаються координати обмежувального прямокутника, класова належність та оцінка впевненості в прогнозі. Однією з важливих особливостей YOLOv8 є використання сучасного підходу до детекції, орієнтованого на більш гнучке прогнозування параметрів об’єкта та покращення якості локалізації. У практичному сенсі це дозволяє підвищити точність виявлення невеликих і візуально близьких об’єктів, що є актуальним саме для дорожніх знаків [11; 15].

На рисунку 2.1 подано спрощену схему архітектури моделі YOLOv8. Як видно зі схеми, вхідне зображення спочатку проходить через backbone, де формуються багаторівневі ознаки. Далі ці ознаки надходять до neck, де виконується їх багатомасштабне поєднання, після чого в head здійснюється безпосередня детекція об’єктів. Також на схемі показано, що під час навчання модель використовує окремі компоненти функції втрат для регресії координат рамок і класифікації об’єктів. Зокрема, інтеграція сучасних функцій втрат, таких як CIoU (Complete Intersection over Union) та DFL (Distribution Focal Loss), дозволяє мережі точніше адаптуватися до меж об'єктів. Це суттєво зменшує кількість хибних спрацьовувань та підвищує загальну просторову чутливість системи в умовах мінливого дорожнього середовища.

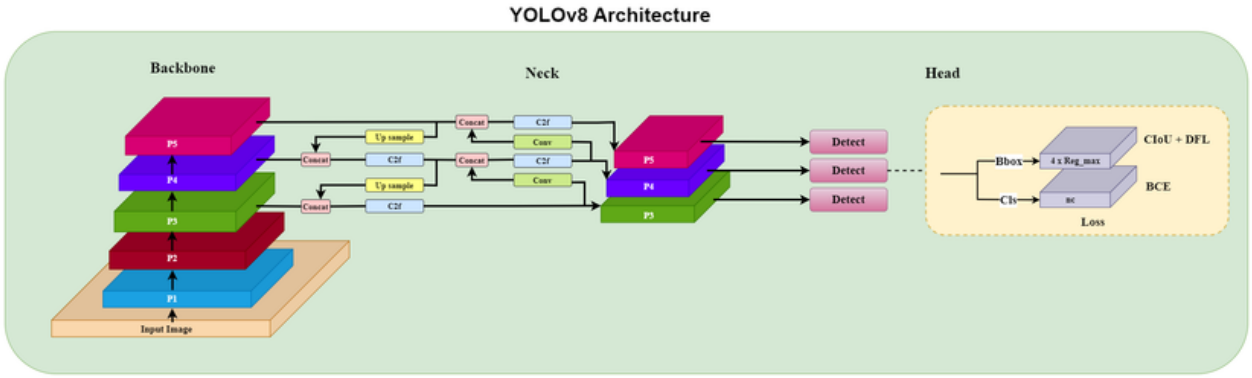

Рисунок 2.1 – Архітектура моделі YOLOv8

Важливою рисою YOLOv8 є орієнтація не лише на високу точність, а й на зручність практичного використання. Модель підтримує навчання, валідацію, бенчмаркінг і гнучке налаштування параметрів через офіційний інструментарій Ultralytics. Це значно спрощує проведення експериментів, порівняння різних конфігурацій та інтеграцію моделі у прикладні програмні системи [12; 13]. Для даної кваліфікаційної роботи ця особливість є важливою, оскільки дозволяє не лише дослідити точність моделі, а й оцінити її продуктивність за показниками швидкодії, що має значення в системах аналізу дорожньої сцени [15; 16].

Ще однією перевагою YOLOv8 є наявність кількох варіантів масштабування моделі, що відрізняються кількістю параметрів та обчислювальною складністю. Це дає змогу обирати конфігурацію залежно від доступних ресурсів і поставлених вимог до точності та швидкості. У практиці детекції об’єктів такий підхід є зручним, оскільки дозволяє адаптувати модель як для дослідницьких запусків, так і для більш продуктивних фінальних експериментів [11; 14].

Разом із тим модель YOLOv8, як і інші одностадійні детектори, має певні обмеження. Попри високу швидкодію, її точність у складних випадках може залежати від вибору розміру вхідного зображення, якості анотацій,

гіперпараметрів навчання та специфіки датасету. Особливо це стосується дрібних об’єктів і класів, які мають незначні візуальні відмінності між собою. Тому ефективність моделі в задачі детекції дорожніх знаків значною мірою визначається не лише її архітектурою, а й правильністю організації експериментального процесу [14; 15].

Отже, YOLOv8 є сучасною одностадійною архітектурою детекції об’єктів, яка поєднує високу швидкість роботи, багатомасштабне представлення ознак та зручний інструментарій для навчання й оцінювання. Саме ці характеристики обумовлюють доцільність її використання як базової моделі для побудови системи детекції дорожніх знаків і подальшого порівняння з альтернативним підходом на основі FastViT [11; 17].

2.3. Архітектурні особливості моделі FastViT

Модель FastViT належить до сучасних гібридних архітектур комп’ютерного зору, у яких поєднуються переваги згорткових нейронних мереж і трансформерних підходів. Її розроблення було орієнтоване на досягнення двох взаємопов’язаних цілей: збереження високої якості виділення ознак та забезпечення швидкого інференсу, придатного для практичного використання. На відміну від класичних візуальних трансформерів, які часто потребують значних обчислювальних ресурсів, FastViT проектувалася як ефективна архітектура, що враховує вимоги до швидкодії та оптимізації під реальне виконання [5; 6; 7].

У контексті цієї роботи FastViT розглядається не як готовий детектор об’єктів, а як архітектурна основа для виділення ознак, на базі якої будується друга модель системи детекції дорожніх знаків. Такий підхід є цілком обґрунтованим, оскільки якість роботи детектора значною мірою залежить від того, наскільки інформативні, стійкі та багаторівневі ознаки формує backbone-

мережа. Саме тому аналіз архітектурних особливостей FastViT є важливим етапом перед проєктуванням повної системи детекції [5].

Загальна структура FastViT включає початковий блок Stem, послідовність кількох обчислювальних етапів Stage 1–Stage 4, проміжні операції зміни масштабу представлень, а також фінальний модуль формування вихідних ознак. Початковий блок Stem виконує первинне перетворення вхідного зображення та формує базові просторові представлення. Уже на цьому етапі модель зменшує розмірність простору ознак і водночас виділяє локальні структури, важливі для подальшої обробки. Саме така побудова дозволяє FastViT ефективно працювати з візуально складними даними та передавати далі більш інформативні проміжні представлення [5].

Після початкового етапу зображення послідовно проходить через кілька обчислювальних стадій, кожна з яких виконує подальше ускладнення ознак. На ранніх рівнях модель більше орієнтується на локальні візуальні патерни, тоді як на глибших рівнях зростає роль складніших структурних залежностей. Така багаторівнева організація є важливою для задачі детекції дорожніх знаків, оскільки дрібні об’єкти потребують збереження просторової деталізації, а складні сцени водночас вимагають формування більш абстрактних ознак для впевненого розмежування класів [5; 8].

Однією з ключових особливостей FastViT є використання ефективних блоків, у яких поєднуються згорткові операції, нормалізація, нелінійні функції активації та модулі змішування ознак. У структурі моделі важливу роль відіграють блоки ConvFFN, що реалізують обробку ознак із використанням згорткових механізмів і виконують функцію ефективного перетворення представлень. У глибших частинах архітектури також використовуються елементи, пов’язані з механізмами уваги, завдяки чому модель набуває здатності краще враховувати глобальні залежності між різними частинами зображення. Саме це дозволяє віднести FastViT до гібридних архітектур, а не до суто згорткових мереж [5; 6].

На рисунку 2.2 подано загальну архітектурну схему FastViT. Із рисунка видно, що модель складається з послідовно з’єднаних етапів обробки, де після початкового Stem виконуються кілька стадій виділення ознак, розділених операціями зміни масштабу. Окремо показано внутрішню структуру окремих блоків, а також відмінності між режимом навчання та інференсу. Це важливо, оскільки FastViT проектувалася з урахуванням не лише якості ознак, а й оптимізації практичної швидкодії моделі під час застосування [5].

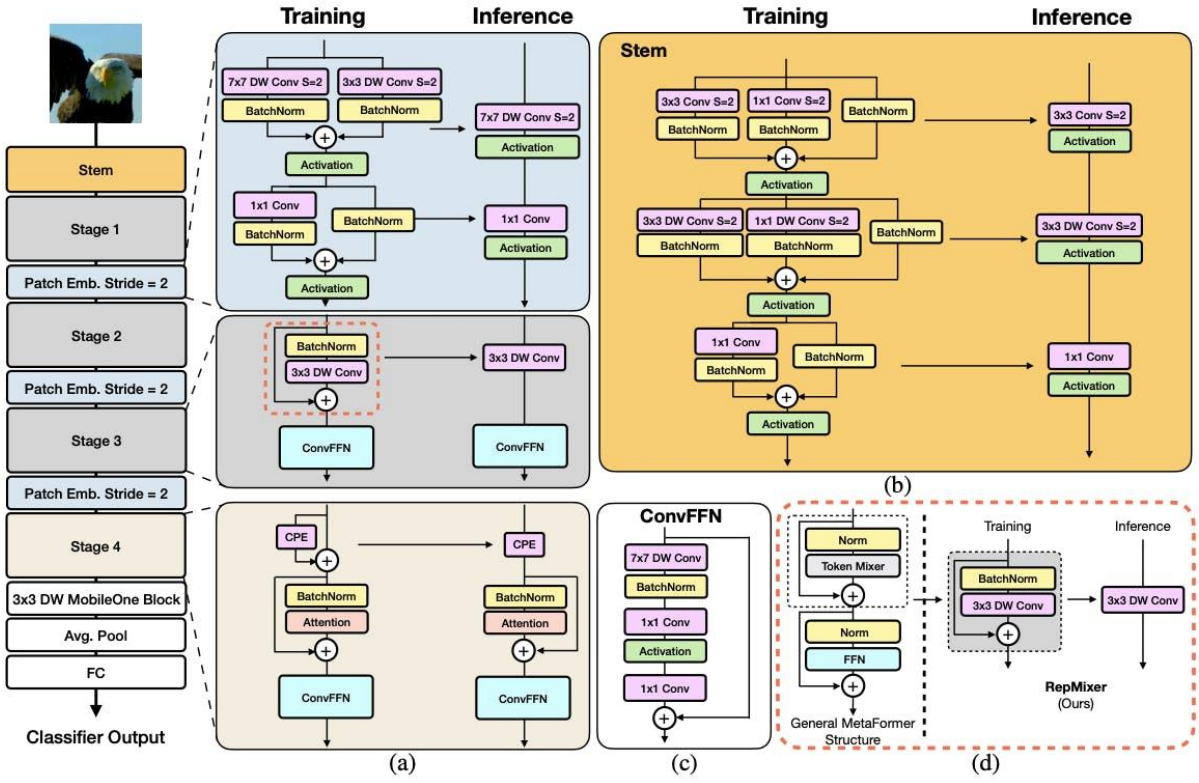

Рисунок 2.2 – Архітектурна схема FastViT як моделі виділення ознак

Суттєвою особливістю FastViT є використання структурної репараметризації, завдяки якій під час навчання можуть застосовуватися більш складні багатогілкові обчислювальні блоки, тоді як під час інференсу вони перетворюються на простішу й швидшу форму. Такий підхід дозволяє отримати виграш як у процесі оптимізації моделі, так і на етапі її практичного

використання. З інженерної точки зору це одна з найважливіших причин, чому FastViT розглядається як архітектура, придатна не лише для дослідницьких, а й для прикладних задач комп’ютерного зору [5].

Перевагою FastViT є те, що вона намагається зберегти сильні сторони згорткових підходів, зокрема ефективну роботу з локальними просторовими структурами, але водночас використовує сучасні механізми побудови представлень, характерні для трансформерних моделей. У результаті архітектура має потенціал до кращого врахування контексту сцени, ніж традиційні компактні згорткові мережі. Для задачі детекції дорожніх знаків це є важливим, оскільки знак може бути невеликим, частково перекритим або розташованим у візуально складному середовищі, де локальних ознак недостатньо без урахування ширшого контексту [5; 7].

Разом із тим архітектура FastViT має і певні обмеження. Насамперед, у порівнянні з дуже легкими одностадійними детекторами, побудова повної системи детекції на основі такого backbone може бути складнішою в реалізації та вимагати більш ретельного налаштування. Крім того, ефективність моделі значною мірою залежить від способу інтеграції FastViT у детекційну систему, вибору голови детекції, параметрів навчання та організації багатомасштабної обробки ознак. Таким чином, сам по собі FastViT не є завершеним рішенням для задачі детекції, а виступає високоякісною архітектурною основою, на базі якої доцільно будувати повну модель [5; 19].

Для даної кваліфікаційної роботи модель FastViT є доцільною з кількох причин. По-перше, вона репрезентує сучасний напрям розвитку архітектур комп’ютерного зору, пов’язаний із поєднанням згорткових та трансформерних ідей. По-друге, її використання дозволяє побудувати альтернативну до YOLOv8 систему детекції та провести змістовне порівняння різних архітектурних підходів. По-третє, FastViT має практичну цінність як ефективний backbone, що може бути адаптований до задачі виявлення

дорожніх знаків і протестований за тими самими метриками, що й базова модель [5; 15].

Отже, FastViT є сучасною гібридною архітектурою виділення ознак, яка поєднує ефективні згорткові блоки, механізми покращеного представлення ознак та структурну репараметризацію для прискорення інференсу. Саме ці властивості обумовлюють доцільність її використання в межах даної роботи як основи другої моделі для подальшого проєктування системи детекції дорожніх знаків і порівняння з архітектурою YOLOv8 [5].

2.4. Проєктування структури нейромережевої системи детекції дорожніх знаків

Проєктування нейромережевої системи детекції дорожніх знаків у межах даної кваліфікаційної роботи виконується з урахуванням двох основних вимог: забезпечення достатньої точності виявлення дорожніх знаків та можливості проведення коректного порівняльного аналізу різних архітектурних підходів. Саме тому запропонована система будується не як окрема вузькоспеціалізована модель, а як єдина експериментальна структура, у межах якої дві різні архітектури працюють із тим самим набором даних, спільними принципами оцінювання та єдиною логікою формування результатів.

Загальна структура системи передбачає наявність кількох взаємопов’язаних модулів: модуля вхідних даних, модуля підготовки даних, модуля навчання моделей, модуля валідації та тестування, а також модуля порівняльного аналізу результатів. Вхідними даними для системи є набір анотованих зображень дорожніх сцен, у яких для кожного об’єкта задано координати обмежувального прямокутника та класову належність. На виході система формує структурований результат у вигляді знайдених об’єктів, їх

координат, класів і значень довіри прогнозу, а також узагальнених показників якості роботи моделей.

У функціональному вигляді систему можна подати як відображення згідно з формулою (2.1):

$$S : I \to O, \tag{2.1}$$

де I — вхідне зображення, а O — множина знайдених об’єктів. Кожен елемент множини O може бути описаний виразом (2.2):

$$o_i = (B_i, c_i, p_i), \tag{2.2}$$

де B_i — координати обмежувального прямокутника, c_i — клас дорожнього знака, p_i — оцінка впевненості моделі. Такий запис є достатнім для опису загальної логіки функціонування системи та не перевантажує підпункт надмірною формалізацією.

Проєктована система має дві основні модельні гілки. Перша гілка базується на використанні YOLOv8 як одностадійного детектора об’єктів. У цьому випадку обробка зображення, виділення ознак, багатомасштабне поєднання представлень і безпосередня детекція виконуються в межах єдиної архітектури. Друга гілка базується на використанні FastViT як моделі виділення ознак, на основі якої побудовано альтернативну систему детекції дорожніх знаків. У межах цієї гілки FastViT виконує функцію backbone-мережі, а подальша локалізація та класифікація об’єктів реалізується в детекційному модулі, що працює з багатомасштабними ознаками. Таким чином, у системі реалізується порівняння двох підходів: готового сучасного одностадійного детектора та альтернативної детекційної системи, побудованої на базі гібридної архітектури виділення ознак.

Важливою частиною проєктування є уніфікація вхідних даних для обох моделей. Для цього використовується єдиний датасет у форматі, придатному для детекції об’єктів, із фіксованим набором класів дорожніх знаків. Такий підхід забезпечує однакову предметну область дослідження і дає змогу порівнювати моделі не на різних наборах прикладів, а в межах єдиного експериментального середовища. Крім того, під час проєктування системи передбачено використання одних і тих самих навчальної, валідаційної та тестової вибірок, що зменшує ризик некоректного зіставлення результатів.

Структура системи також включає етап підготовки даних. На цьому етапі виконується зчитування конфігурації набору даних, перевірка доступності зображень і анотацій, перетворення розмітки до формату, який підтримується обраною моделлю, а також підготовка train та validation наборів. Для YOLOv8 і FastViT реалізаційно використовуються різні механізми навчання, але логіка вхідних даних залишається узгодженою: обидві моделі отримують ті самі зображення та ту саму інформацію про розміщення й класи дорожніх знаків.

На етапі навчання система повинна підтримувати налаштування основних гіперпараметрів: кількості епох, розміру вхідного зображення, розміру пакета, швидкості навчання, вагової регуляризації та інших параметрів оптимізації. Це необхідно не лише для підвищення якості моделей, а й для проведення порівняльного дослідження. У проєктованій системі передбачено, що для кожної моделі гіперпараметри можуть бути адаптовані до її архітектурних особливостей, однак оцінювання виконується в межах спільної схеми експерименту. Такий підхід є більш реалістичним, ніж штучне нав’язування абсолютно однакових параметрів різним архітектурам, оскільки дозволяє оцінювати їх у режимі практичного застосування.

Окремий модуль системи відповідає за валідацію та тестування. Його призначення полягає у формуванні об’єктивної оцінки якості роботи моделей після навчання. У межах цієї роботи система проектується так, щоб фінальні

результати для обох моделей подавалися в єдиному форматі. Це означає, що після завершення навчання для кожної моделі зберігаються основні кількісні показники: точність детекції, повнота, F1-міра, значення mAP, час навчання, затримка інференсу, швидкість обробки кадрів і кількість параметрів. Завдяки цьому результати можна безпосередньо порівнювати у вигляді таблиць, графіків і текстових висновків.

Суттєвим елементом проєктованої системи є модуль порівняльного аналізу. Його функція полягає не просто у виведенні окремих чисел, а у формуванні узагальненого аналітичного результату. У межах такого модуля повинні поєднуватися характеристики якості детекції та характеристики продуктивності, оскільки для задачі дорожніх знаків важливо враховувати як точність знаходження об’єктів, так і швидкість роботи системи. Саме тому структура системи детекції проєктується як експериментальна платформа, що дозволяє не лише отримати результати для кожної моделі окремо, а й оцінити їхню практичну придатність у порівняльному аспекті.

З позиції програмної організації система має модульний характер. Це означає, що окремі частини системи, пов’язані з підготовкою даних, навчанням YOLOv8, навчанням FastViT-орієнтованої моделі, оцінюванням та побудовою підсумкових звітів, розділяються на логічно самостійні компоненти. Така модульність є важливою не лише для зручності реалізації, а й для подальшої масштабованості системи. У перспективі це дає можливість додавати нові архітектури, змінювати набір метрик, використовувати інші датасети або модифікувати окремі блоки без повної перебудови всієї системи.

На рисунку 2.3 доцільно подати загальну структурну схему розроблюваної системи детекції дорожніх знаків. На такому рисунку мають бути відображені: вхідні зображення, блок підготовки даних, дві модельні гілки — YOLOv8 та FastViT-орієнтований детектор, блок оцінювання якості та блок порівняльного аналізу результатів. Така схема дозволить наочно показати, що в межах роботи досліджується не ізольована модель, а

повноцінна нейромережева система з єдиною логікою експериментального аналізу.

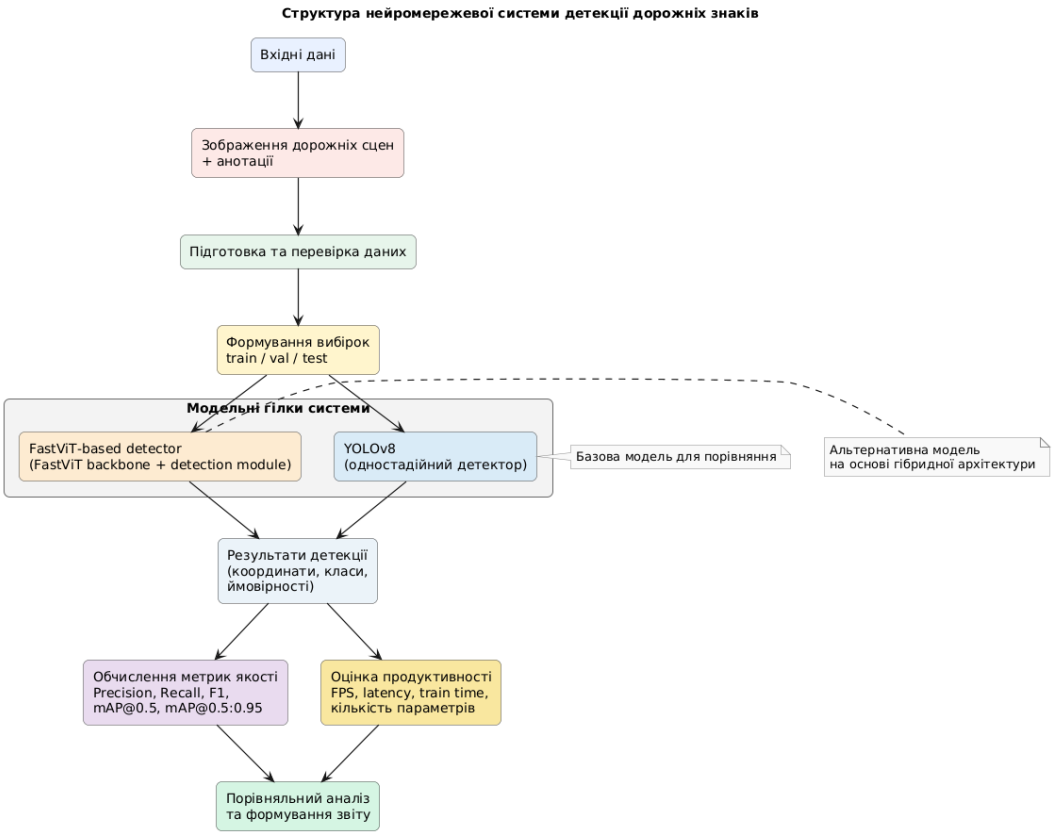

Рисунок 2.3 – Структура нейромережевої системи детекції дорожніх знаків

Отже, у межах даної роботи проєктується модульна нейромережева система детекції дорожніх знаків, яка забезпечує спільну обробку даних, навчання двох альтернативних архітектурних підходів, оцінювання результатів за єдиною системою метрик та формування підсумкового порівняльного аналізу. Така структура є доцільною як з наукової, так і з практичної точки зору, оскільки дозволяє не лише успішно реалізувати моделі, а й отримати обґрунтовані висновки щодо їхньої придатності для інтеграції в сучасні автомобільні системи (ADAS), спираючись на баланс точності та швидкодії.

2.5. Вибір датасету, метрик оцінювання та схеми експериментального дослідження

Одним із ключових етапів проєктування нейромережевої системи детекції дорожніх знаків є вибір набору даних, на якому здійснюватиметься навчання, валідація та подальше порівняння моделей. Для цієї роботи доцільно використовувати датасет Tsinghua-Tencent 100K (TT100K), оскільки він спеціально орієнтований на задачу виявлення та класифікації дорожніх знаків у реальних умовах. В офіційному описі зазначено, що набір даних створено на основі 100000 панорам Tencent Street View; він містить 10000 зображень із приблизно 30000 екземплярами дорожніх знаків, а також охоплює значну варіативність освітлення, погодних умов і масштабів об’єктів. Крім класових міток, для кожного знака наведено bounding box та піксельну маску, що робить датасет придатним саме для задач детекції [1; 2].

Вибір TT100K для даної кваліфікаційної роботи є обґрунтованим з кількох причин. По-перше, цей датасет безпосередньо відповідає предметній області дослідження, оскільки містить дорожні знаки, зафіксовані в природних дорожніх сценах, а не в лабораторно спрощених умовах. По-друге, він добре підходить для оцінювання моделей у випадку дрібних об’єктів, що є принципово важливим для систем детекції дорожніх знаків. По-третє, використання спеціалізованого набору даних дає змогу проводити дослідження не в загальному контексті детекції об’єктів, а саме в умовах, максимально наближених до реального транспортного застосування [1; 2].

У межах реалізації системи набір даних подається через конфігураційний YAML-файл, у якому задаються шляхи до зображень та анотацій, а також перелік класів. Такий спосіб організації експерименту є зручним, оскільки забезпечує єдину конфігурацію для обох модельних гілок та спрощує відтворюваність дослідження. Для коректного порівняння моделей необхідно, щоб YOLOv8 і FastViT-орієнтована модель працювали з одними й тими самими навчальною та валідаційною вибірками. Саме тому в даній

роботі експериментальна схема передбачає використання спільного YAML-конфігу, що задає єдину структуру train/val/test даних для обох архітектур [12; 13].

Не менш важливим етапом є вибір метрик оцінювання. Оскільки задача детекції передбачає одночасно і локалізацію об’єкта, і визначення його класу, для її оцінювання недостатньо лише показника правильної класифікації. Саме тому в роботі доцільно використовувати сукупність метрик: Intersection over Union (IoU), Precision, Recall, F1-score, Average Precision (AP) та mean Average Precision (mAP). У документації Ultralytics і TorchMetrics наголошується, що IoU є базовою мірою перекриття між передбаченою та еталонною рамками, AP узагальнює співвідношення між precision і recall, а mAP відображає середню якість детекції за всіма класами [10; 15; 18].

Формально метрика IoU визначається за формулою (2.3) як відношення площі перетину прогнозованої та еталонної рамок до площі їх об’єднання:

$$IoU = \frac{B_{pred} \cap B_{gt}}{B_{pred} \cup B_{gt}}, \tag{2.3}$$

де B_{pred} — прогнозована обмежувальна рамка, B_{gt} — еталонна рамка. Чим ближче значення IoU до 1, тим точніше локалізовано об’єкт [10; 15].

Для оцінювання точності й повноти виявлення використовуються метрики Precision та Recall, які обчислюються за формулами (2.4) та (2.5):

$$Precision = \frac{TP}{TP+FP,} \tag{2.4}$$

$$Recall = \frac{TP}{TP+FN,} \tag{2.5}$$

де TP — кількість істинно позитивних виявлень, FP — хибнопозитивні спрацювання, FN — пропущені об’єкти. На основі цих двох величин обчислюється F1-score за формулою (2.6):

$$F1 = \frac{2*Precision*Recall}{Precision+Recall}, \quad (2.6)$$

Precision дозволяє оцінити, наскільки “чистими” є передбачення моделі, Recall показує, яку частину реальних об’єктів модель змогла знайти, а F1-score узагальнює ці дві характеристики [15; 18].

Основною інтегральною метрикою для порівняння детекторів у роботі доцільно обрати mAP@0.5 та mAP@0.5:0.95. Перша відображає середню точність за фіксованого порога IoU = 0.5, а друга є суворішою, оскільки усереднює значення AP на наборі порогів IoU. Документація Ultralytics прямо вказує на mAP50 і mAP50-95 як на ключові показники валідації моделі, а TorchMetrics у класі MeanAveragePrecision надає результати, зокрема, у вигляді map та map_50, що відповідає mAP@0.5:0.95 та mAP@0.5 відповідно [13; 15; 18].

Окрім показників якості детекції, у межах даної роботи доцільно оцінювати й продуктивність моделей. З огляду на прикладний характер задачі, важливими є такі характеристики, як час навчання, затримка інференсу одного зображення, кількість кадрів за секунду (FPS) і кількість параметрів моделі. Саме поєднання точнісних і швидкісних показників дозволяє зробити змістовний висновок не лише про те, яка модель точніша, а й про те, наскільки вона придатна до практичного застосування в системах аналізу дорожньої сцени [15; 16; 20].

Схема експериментального дослідження в даній роботі повинна бути побудована так, щоб забезпечити максимальну коректність порівняння. На

першому етапі виконується підготовка набору даних та перевірка анотацій. На другому етапі обидві моделі навчаються на спільній навчальній вибірці з використанням однакової загальної постановки задачі. На третьому етапі виконується валідація моделей на спільній validation-вибірці. Для YOLOv8 валідація може виконуватися штатними засобами Ultralytics, які надають mAP50, mAP75, mAP50-95 та інші показники. Для другої моделі на базі FastViT застосовується єдина логіка оцінювання на тій самій валідаційній вибірці з використанням TorchMetrics та додаткових розрахунків Precision, Recall і F1-score [12; 13; 18].

Важливо підкреслити, що в роботі не ставиться мета штучно зробити архітектурно різні моделі абсолютно однаковими за всіма внутрішніми параметрами навчання. Натомість порівняння організовується на спільному наборі даних, за єдиною логікою постановки задачі та за спільним набором фінальних метрик. Такий підхід є більш практичним і науково коректним, оскільки дозволяє оцінювати моделі в умовах, наближених до реального використання, а не в штучно симетричному експерименті [12; 15; 20].

Отже, для даної кваліфікаційної роботи обрано датасет TT100K як спеціалізований набір даних для детекції дорожніх знаків у реальних умовах, а також сукупність метрик, що охоплює як якість локалізації та класифікації об’єктів, так і практичну продуктивність моделей. Запроєктована схема експериментального дослідження забезпечує можливість коректного зіставлення YOLOv8 та FastViT-орієнтованої системи в межах єдиного прикладного завдання [1; 2; 13; 18].

Висновки до розділу 2

У другому розділі було розглянуто архітектурні особливості двох моделей, обраних для дослідження, а також виконано проєктування нейромережевої системи детекції дорожніх знаків. Встановлено, що модель

YOLOv8 є сучасним одностадійним детектором, орієнтованим на високу швидкодію та практичне застосування, тоді як FastViT доцільно розглядати як сучасну гібридну архітектуру виділення ознак, на основі якої може бути побудована альтернативна система детекції [5; 11; 12].

У ході проєктування було визначено загальну структуру нейромережевої системи, яка включає модуль підготовки даних, дві модельні гілки, блок оцінювання якості та модуль порівняльного аналізу результатів. Така структура є доцільною, оскільки дозволяє досліджувати дві різні архітектурні концепції в межах спільної задачі, спільного набору даних і єдиної логіки отримання підсумкових результатів [12; 13].

Також у розділі було обґрунтовано вибір датасету TT100K як спеціалізованого набору даних для детекції дорожніх знаків та визначено систему метрик оцінювання, що включає IoU, Precision, Recall, F1-score, mAP@0.5, mAP@0.5:0.95, а також показники продуктивності. Це створює необхідне методичне підґрунтя для переходу до програмної реалізації моделей, організації навчання та подальшого експериментального порівняння їх роботи у третьому розділі [1; 2; 15; 18].

3. ПРОГРАМНА РЕАЛІЗАЦІЯ, НАВЧАННЯ ТА ДОСЛІДЖЕННЯ РЕЗУЛЬТАТІВ

3.1. Програмна реалізація системи та середовище розробки

Програмна реалізація нейромережевої системи детекції дорожніх знаків у межах даної кваліфікаційної роботи була виконана у вигляді єдиного програмного комплексу, призначеного для навчання, валідації, тестування та порівняльного аналізу двох моделей: YOLOv8 та альтернативної системи детекції на основі FastViT. Основна ідея реалізації полягала в тому, щоб створити таку структуру програмних модулів, яка забезпечує спільну роботу з датасетом, уніфіковану логіку підготовки вхідних даних, окремий запуск моделей, подальше обчислення метрик і автоматичне формування підсумкових звітів. Такий підхід відповідає сучасній практиці реалізації задач детекції об’єктів засобами Python, Ultralytics, PyTorch і TorchVision [11; 12; 19].

Як основну мову розробки було використано Python, оскільки вона має розвинену екосистему бібліотек для машинного навчання, комп’ютерного зору та візуалізації результатів. Для побудови й навчання моделі YOLOv8 було використано бібліотеку Ultralytics, яка надає готові засоби тренування, валідації, бенчмаркінгу та збереження результатів. Реалізація другої моделі виконувалася на базі PyTorch, TorchVision, timm, TorchMetrics, а також допоміжних бібліотек NumPy, Pandas, Matplotlib та Pillow. Такий набір інструментів дозволив поєднати готовий інженерний пайплайн для YOLOv8 із більш гнучкою кастомною реалізацією моделі на базі FastViT [5; 11; 18; 19].

Середовищем виконання обчислювальних експериментів виступала платформа Windows. Навчання моделей проводилося з використанням графічного прискорювача NVIDIA GeForce RTX 5060 Ti з доступною відеопам’яттю близько 16 ГБ. У ході запусків використовувалася конфігурація Python 3.9.13, PyTorch 2.8.0+cu128 та підтримка CUDA, що забезпечувало

виконання тренування й інференсу на GPU. Така апаратно-програмна конфігурація є достатньою для реалізації експериментальної системи детекції дорожніх знаків і водночас дає змогу оцінювати не лише точність моделей, а й їхню швидкодію в умовах, наближених до практичного використання [12; 20].

Центральним файлом програмної реалізації є full_diploma_system.py, у якому зосереджено основний функціонал системи. Саме в цьому файлі реалізовано засоби зчитування конфігурації датасету, роботу з файлами розмітки, підготовку вхідних даних, побудову моделі на основі FastViT, запуск навчання YOLOv8, обчислення метрик якості, бенчмаркінг продуктивності та генерацію підсумкового звіту. Окрім головного файла, у роботі використовувалися допоміжні скрипти для окремих експериментів, зокрема окремого запуску YOLOv8, окремого навчання FastViT та відновлення навчання другої моделі після переривання. Така організація зробила систему більш гнучкою та дозволила виконувати як повні, так і часткові експериментальні запуски без потреби повторно виконувати весь конвеєр.

З функціональної точки зору програмний комплекс доцільно поділити на кілька основних модулів. Перший модуль відповідає за роботу з даними. У його межах реалізовано зчитування YAML-конфігурації датасету, визначення шляхів до зображень та анотацій, перевірку доступності файлів, рекурсивний пошук зображень, а також перетворення координат рамок із формату YOLO до формату, придатного для моделей на базі PyTorch. Для другої моделі було створено спеціальний клас YoloDetectionDataset, який забезпечує завантаження зображень і анотацій, побудову списків bounding boxes і класових міток, а також формування структури target, необхідної для детекційних моделей TorchVision. Завдяки цьому обидві модельні гілки працювали з одним і тим самим набором даних, що є важливою умовою коректності подальшого порівняння.

Другий модуль системи пов’язаний із реалізацією моделі YOLOv8. У цій частині використовуються готові засоби бібліотеки Ultralytics, що дозволяють запускати тренування, валідацію та тестування із заданим набором гіперпараметрів. У межах реалізації передбачалася можливість змінювати розмір вхідного зображення, кількість епох, batch size, параметри оптимізації, patience, а також інші параметри тренувального процесу. Окремою перевагою такого підходу є автоматичне збереження найкращих ваг, журналів навчання, графічних матеріалів і таблиць із метриками, що спрощує подальший аналіз [12; 13; 15].

Третій модуль реалізує альтернативну модель детекції на основі FastViT. У ньому використано архітектуру FastViT як backbone-мережу для виділення ознак, після чого на її основі формується детекційна система з використанням компонентів TorchVision. Для цього в коді реалізовано клас TimmBackboneWithFPN, який поєднує FastViT із пірамідою ознак FPN, а також функцію побудови детектора, сумісного зі структурою Faster R-CNN. Така реалізація дозволила сформувати другу модельну гілку, що репрезентує інший архітектурний підхід до задачі детекції дорожніх знаків. Окремо були реалізовані процедури навчання, валідації, збереження найкращих ваг і відновлення навчання після переривання.

Четвертий модуль призначений для оцінювання якості роботи моделей. Для YOLOv8 використовуються штатні засоби валідації Ultralytics, які формують словник із основними показниками якості, зокрема mAP50, mAP50-95, precision і recall. Для другої моделі оцінювання реалізоване окремо через функцію evaluate_fastvit, що використовує MeanAveragePrecision із бібліотеки TorchMetrics. Додатково у системі було реалізовано власну функцію detection_precision_recall_f1, яка дає змогу обчислювати TP, FP, FN, а також Precision, Recall і F1-score на основі фактичних результатів детекції. Така реалізація є важливою, оскільки забезпечує подання підсумкових результатів обох моделей у єдиному форматі [15; 18].

П’ятий модуль відповідає за оцінювання обчислювальної продуктивності моделей. У програмному коді реалізовано спеціалізовані функції `benchmark_yolo` та `benchmark_fastvit`, за допомогою яких в автоматичному режимі визначаються середня затримка інференсу (Latency) на одне зображення, загальна кількість кадрів за секунду (FPS), а також обсяг спожитої відеопам'яті. Додатково обчислюється кількість параметрів нейромереж і фіксується точний час тренувального процесу. Аналіз цих характеристик став важливою частиною експериментального дослідження, оскільки для інтелектуальних систем детекції дорожніх знаків (наприклад, у комплексах ADAS) потрібно враховувати не лише якість виявлення об’єктів, а й суворі вимоги до швидкості роботи в реальному часі [15; 16; 20].

Окреме значення у програмній реалізації має модуль автоматичного логування та формування результатів експерименту. У системі передбачено автоматизовану генерацію зведених JSON-файлів, CSV-таблиць і графічних матеріалів прямого порівняння. Зокрема, у спеціальному каталозі результатів формуються такі файли, як `yolo_summary.json`, `fastvit_summary.json`, `comparison_results.csv`, `final_summary.json`, а також аналітичні графіки `fps_comparison.png` і `map5095_comparison.png`. Такий підхід дозволяє не лише надійно зберігати результати експериментів, а й безпосередньо використовувати їх під час написання пояснювальної записки та формування ілюстративного матеріалу без ризику ручного спотворення даних.

На рисунку 3.1 доцільно подати загальну архітектурну структуру програмної реалізації системи, де відображено головний керівний скрипт `full_diploma_system.py` та його взаємодію з підпорядкованими модулями: підготовки даних, YOLOv8, FastViT, оцінювання та генерації звітів. Такий рисунок дає можливість наочно показати логіку обробки інформації та довести, що в межах роботи реалізовано не набір розрізнених скриптів, а цілісний, добре структурований модульний програмний комплекс.

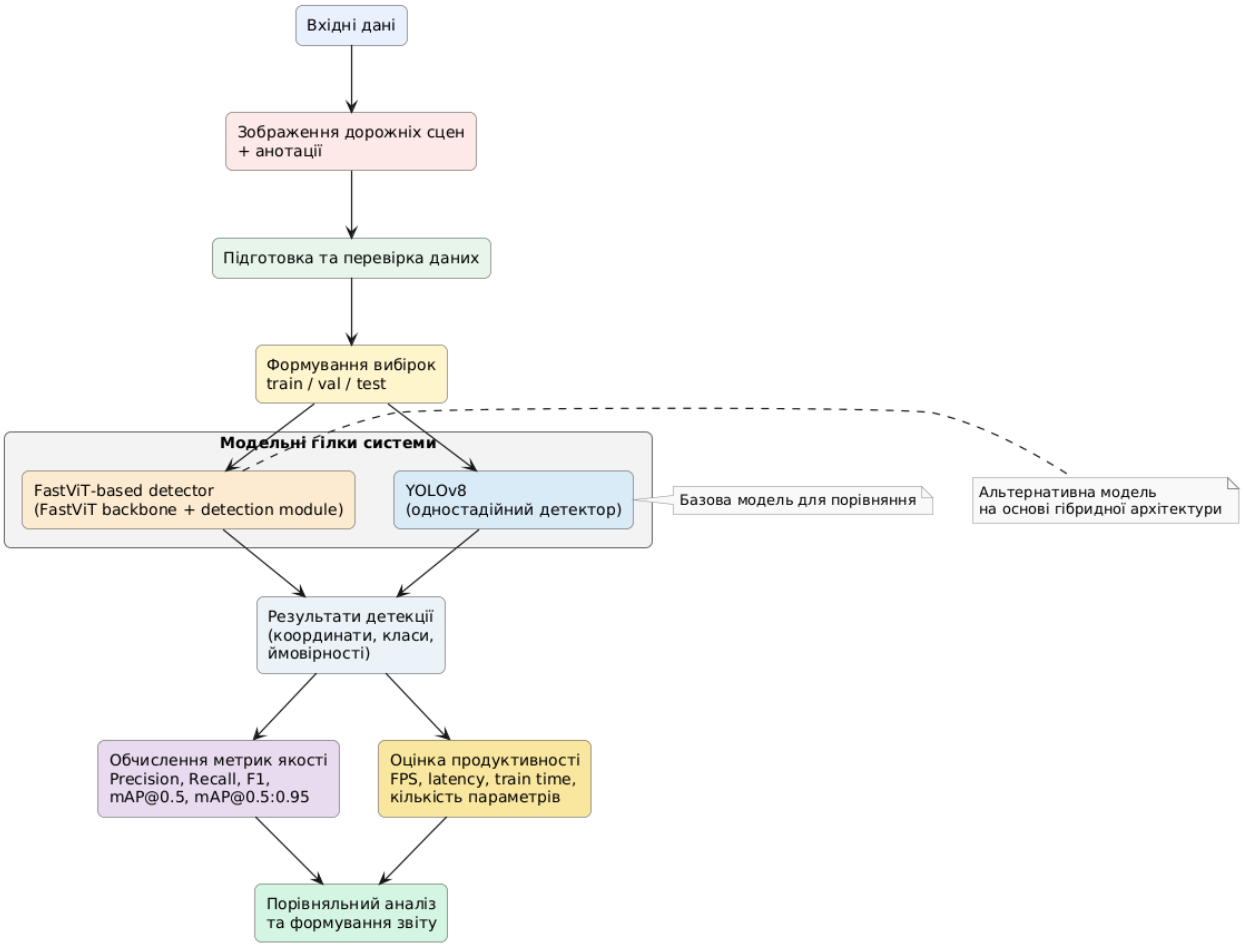

Рисунок 3.1 – Структура програмної реалізації системи детекції дорожніх знаків

У ході експериментів система формувала окремі каталоги запусків, у яких автоматично зберігалися результати навчання та порівняльного аналізу. Одним із таких каталогів є, наприклад, diploma_experiments/run_20260307_175003, у якому були збережені підсумкові файли comparison_results.csv, final_summary.json, окремі підсумки моделей, а також графіки порівняння. Така організація результатів є зручною з практичної точки зору, оскільки кожний запуск утворює окремий каталог із повним набором підсумкових матеріалів, що забезпечує відтворюваність експерименту та спрощує подальший аналіз.

На рисунку 3.2 наведено файлову структуру розробленого програмного проєкту. Ця ілюстрація наочно демонструє логічну організацію вихідного коду, конфігураційних файлів, директорій з наборами даних та каталогів із результатами експериментальних запусків у середовищі розробки.

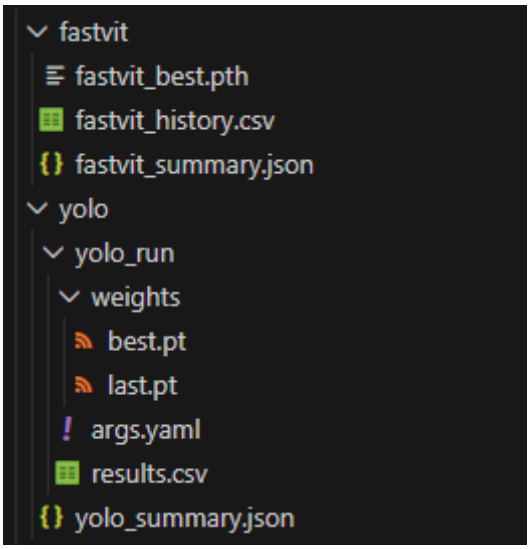

Рисунок 3.2 – Файлова структура програмного проєкту

Отже, програмна реалізація системи детекції дорожніх знаків була побудована як модульний комплекс, що поєднує спільну підготовку даних, дві альтернативні модельні гілки, засоби оцінювання якості та продуктивності, а також автоматичне формування підсумкових звітів. Така реалізація є достатньо гнучкою для проведення експериментального дослідження, зручною для повторного запуску окремих моделей і придатною для подальшого порівняльного аналізу YOLOv8 та FastViT у наступних підпунктах розділу.

3.2. Підготовка даних і налаштування процесу навчання моделей

Підготовка даних є одним із визначальних етапів побудови системи детекції дорожніх знаків, оскільки саме від коректності організації вхідного

набору даних, структури анотацій і параметрів навчання залежить як точність локалізації об’єктів, так і стабільність роботи моделей у процесі тренування. У межах даної роботи для обох модельних гілок використовувався спільний датасет TT100K, підключений через єдиний YAML-конфігураційний файл, що містив шляхи до навчальної та валідаційної вибірок, а також перелік класів дорожніх знаків. Такий підхід забезпечив узгодженість вхідних даних для моделей YOLOv8 і FastViT та створив основу для їх подальшого коректного порівняння [1; 2; 12].

У програмній реалізації підготовка даних починалася зі зчитування YAML-файлу конфігурації. На цьому етапі система визначає кореневий каталог датасету, шляхи до підкаталогів train, val і test, а також список класів, із якими працюватиме модель. У коді це реалізовано через функцію load_yolo_data_cfg, яка не лише зчитує конфігурацію, а й перетворює відносні шляхи на абсолютні, формуючи єдиний словник параметрів датасету. Така організація дає змогу уникнути помилок, пов’язаних із файловою структурою, а також спрощує повторне використання системи з іншими наборами даних подібного формату.

Після завантаження конфігурації виконується пошук зображень і відповідних файлів анотацій. Для цього в системі реалізовано функції рекурсивного пошуку зображень у вкладених каталогах і побудови шляху до файлу розмітки для кожного зображення. Оскільки вихідна розмітка зберігалася у форматі YOLO, що задає клас об’єкта та нормалізовані координати рамки у вигляді xcenter,ycenter,width,heightx_{center}, y_{center}, width, heightxcenter,ycenter,width,height, для другої модельної гілки було необхідно виконати перетворення цих координат до формату x1,y1,x2,y2x_1, y_1, x_2, y_2x1,y1,x2,y2, який використовується детекційними моделями TorchVision. У коді це реалізовано через функції xywhn_to_xyxy і clamp_box, які перетворюють координати в абсолютні значення та обмежують їх межами

зображення. Така процедура є важливою для забезпечення коректного навчання FastViT-орієнтованої моделі [19].

Для роботи з другою моделлю було створено власний клас датасету YoloDetectionDataset, який забезпечує завантаження зображення, перетворення його у тензор і формування структури target, що містить координати рамок, класові мітки, площі об’єктів і службову інформацію. При цьому для FastViT-орієнтованого детектора класи були зміщені на одиницю, оскільки у Faster R-CNN нульовий клас резервується під фон. Така реалізація забезпечила сумісність одного й того самого набору анотацій із двома різними архітектурними підходами. Для YOLOv8 окремих перетворень розмітки не вимагалося, оскільки модель безпосередньо використовує формат YOLO, заданий у вихідному наборі даних [11; 12; 19].

Важливою частиною підготовки даних була також організація механізму подачі прикладів у модель. Для цього в системі застосовувалися DataLoader з функцією collate_fn, яка забезпечує коректне групування зображень і анотацій у батчі змінного розміру. Для моделі на базі FastViT використовувалися параметри pin_memory, persistent_workers і num_workers, що дозволяло оптимізувати роботу з GPU та адаптувати подачу даних до операційної системи Windows. У практичних експериментах окрему увагу було приділено стабільності роботи з multiprocessing, оскільки при навчанні на Windows некоректні налаштування workers могли призводити до помилок ініціалізації процесів. Саме тому в частині запусків для FastViT використовувалося значення workers=0, що забезпечувало стабільну роботу навіть ціною певного зниження швидкості завантаження даних [19; 20].

Підготовка даних для моделі YOLOv8 додатково включала використання штатних механізмів бібліотеки Ultralytics. У процесі навчання застосовувалися автоматичне кешування списків анотацій, перевірка коректності файлів розмітки, формування train та validation cache, а також аугментації, визначені параметрами тренувального запуску. Серед них

використовувалися зміни кольорових характеристик (hsv_h, hsv_s, hsv_v), горизонтальне віддзеркалення (fliplr), масштабування (scale), трансляція (translate) та обмежене поворотне перетворення (degrees). Застосування таких аугментацій є виправданим для задачі детекції дорожніх знаків, оскільки дозволяє підвищити стійкість моделі до змін умов спостереження та покращити узагальнювальну здатність детектора [12; 14].

Окрему увагу в роботі було приділено налаштуванню процесу навчання моделей. Для YOLOv8 використовувалися два режими запуску: швидкий експериментальний режим для первинного відпрацювання пайплайну та фінальний режим для отримання підсумкових результатів. У фінальному запуску для моделі YOLOv8s були встановлені такі параметри: imgsz=640, epochs=80, batch=16, fraction=1.0, optimizer=AdamW, lr0=0.0005, weight_decay=0.0005, patience=30, cos_lr=True, cache=ram, а також стандартні параметри аугментації. Така конфігурація дозволила використати весь доступний навчальний набір і забезпечити достатньо тривалий тренувальний процес для стабілізації якості моделі. У результаті саме цей запуск став базою для фінального оцінювання YOLOv8 [12; 14].

Для FastViT-орієнтованої системи налаштування процесу навчання було складнішим, оскільки модель будувалася як власна реалізація детектора на основі FastViT-backbone та компонентів TorchVision. У фінальних експериментах використовувалися різні варіанти конфігурацій, однак загальна логіка полягала у використанні повного train-набору, спільної validation-вибірки, фіксованого розміру вхідного зображення та явного задання ключових гіперпараметрів. Для швидкого, але коректного порівняння з YOLOv8 окремо реалізовувався запуск з параметрами, наближеними до умов базової моделі: imgsz=416, epochs=50, patience=10, фіксований batch_size, а також можливість resume після переривання навчання. Водночас для повнішого дослідження застосовувалися й конфігурації з img_size=640, epochs=18 та іншими значеннями batch size і кількості workers. Такий підхід

дозволив не лише перевірити працездатність архітектури, а й оцінити її поведінку в різних режимах тренування.

У процесі навчання FastViT використовувалися оптимізатор AdamW, scheduler типу CosineAnnealingLR, змішана точність обчислень через torch.amp.autocast і GradScaler, а також механізм часткового заморожування backbone на початкових епохах. Це означає, що на старті навчання параметри FastViT-backbone могли тимчасово не оновлюватися, а основна оптимізація стосувалася детекційної частини моделі. Такий підхід є доцільним для стабілізації навчання та поступового перенесення попередньо навчених ознак на нову предметну область. Крім того, у коді була реалізована можливість збереження best і last checkpoint, що дозволяло як обирати найкращу модель за метриками валідації, так і відновлювати навчання після аварійного завершення [5; 19; 20].

Важливим етапом налаштування процесу навчання було забезпечення відтворюваності експериментів. Для цього у програмній реалізації використовувалася функція seed_everything, яка встановлювала початкові значення генераторів випадкових чисел для Python, NumPy та PyTorch. Додатково фіксувалися параметри середовища, пов’язані з deterministic-поведінкою окремих компонентів. Такий підхід є важливим у дослідницькій роботі, оскільки дозволяє мінімізувати вплив випадкових факторів на результати навчання та спростити повторення експериментів у майбутньому [20].

На рисунку 3.3 доцільно подати схему підготовки даних і передачі їх у модельні гілки, де буде показано YAML-конфігурацію, зображення, анотації, формування train і validation наборів, а також подачу даних до YOLOv8 і FastViT-based detector. Така схема дає змогу наочно продемонструвати, що обидві моделі навчалися в межах єдиної системи організації даних.

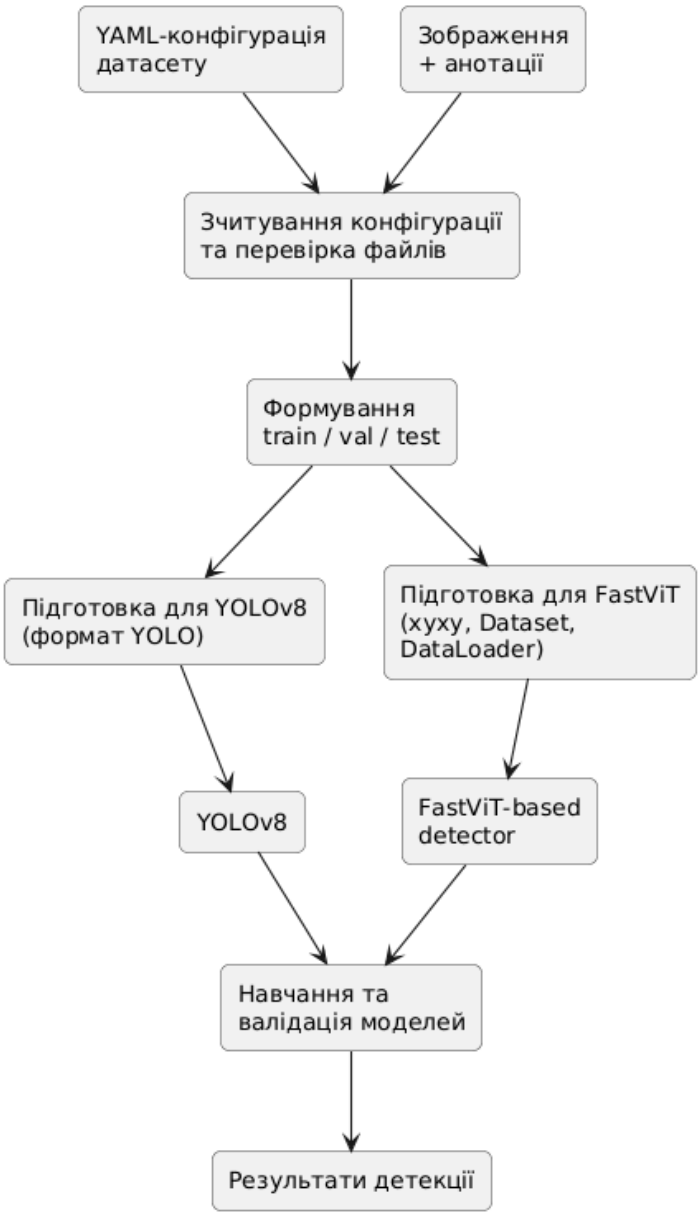

Рисунок 3.3 – Схема підготовки даних і передачі їх у модельні гілки системи

Для ілюстрації реального процесу підготовки даних та навчання моделі наведено приклади пакетів зображень (train batch), автоматично сформованих засобами бібліотеки Ultralytics (рис. 3.4). Ця візуалізація є важливою складовою експериментального розділу, оскільки наочно демонструє вигляд розмітки об’єктів (обмежувальних рамок), різноманітність класів дорожніх знаків у вибірці, а також реальну складність дорожніх сцен (зміну масштабу,

перекриття, умови освітлення), з якими безпосередньо працює модель під час тренування.

Рисунок 3.4 – Приклад зображень з анотаціями дорожніх знаків із навчальної вибірки

Отже, у межах даної роботи було реалізовано єдину систему підготовки даних та налаштування процесу навчання, яка забезпечує коректну роботу

двох модельних гілок на спільному датасеті. Підготовка даних включала зчитування конфігурації, перевірку й перетворення анотацій, формування наборів даних і налаштування механізму подачі прикладів у модель. Налаштування процесу навчання охоплювало вибір гіперпараметрів, оптимізаторів, scheduler-ів, механізмів змішаної точності, відновлення після переривання та забезпечення відтворюваності експериментів. Саме така організація створила технічну основу для подальшого навчання, валідації та порівняння моделей YOLOv8 і FastViT у наступному підпункті.

3.3. Навчання, валідація та тестування моделей YOLOv8 і FastViT

Після завершення етапу підготовки даних і налаштування програмного середовища було виконано навчання, валідацію та тестування двох модельних гілок системи: YOLOv8 та альтернативної моделі детекції на основі FastViT. Організація цього етапу передбачала окремий запуск кожної моделі на спільному наборі даних, збереження ваг найкращих станів, контроль проміжних метрик та подальше формування підсумкових результатів. Для моделі YOLOv8 використовувалися штатні засоби бібліотеки Ultralytics, тоді як для FastViT-орієнтованої системи було реалізовано окремий цикл навчання, валідації, збереження checkpoint і відновлення обчислень після переривання [12; 13; 19].

Навчання моделі YOLOv8 виконувалося у фінальному режимі на повній навчальній вибірці датасету. Для цього використовувалася конфігурація з параметрами imgsz=640, epochs=80, batch=16, fraction=1.0, optimizer=AdamW, lr0=0.0005, weight_decay=0.0005, patience=30, cos_lr=True, а також набором стандартних аугментацій. У процесі навчання модель демонструвала стабільне зростання якості детекції вже на перших епохах. Так, після першої епохи валідаційні показники становили mAP50 = 0.328 та mAP50-95 = 0.227, після другої епохи вони зросли до 0.467 та 0.345 відповідно, а після третьої досягли 0.551 та 0.421. Надалі спостерігалося поступове, але стале поліпшення

результатів, що свідчило про коректну збіжність моделі та адекватність обраних гіперпараметрів.

У середині тренувального процесу YOLOv8 продовжувала покращувати валідаційні результати. Зокрема, на 10-й епосі модель досягла mAP50 = 0.726 та mAP50-95 = 0.583, на 20-й епосі — 0.790 та 0.644, а на 30-й епосі — 0.821 та 0.676. Після цього приріст метрик став менш різким, однак залишався позитивним. На 40-й епосі було зафіксовано mAP50 = 0.837 та mAP50-95 = 0.692, а після 50-ї епохи модель вийшла на рівень mAP50 = 0.847 та mAP50-95 = 0.701. Подальші епохи давали вже незначні покращення, що є типовою поведінкою моделі, яка наближається до режиму насичення за якістю.

З технічної точки зору важливим є те, що навчання YOLOv8 було завершено з використанням механізму збереження best.pt та last.pt, а також із можливістю відновлення тренування після переривання. У ході експерименту дійсно виникала необхідність продовжити навчання після зупинки на пізніх епохах, і відновлення було успішно виконане зі збереженого файлу last.pt. Це підтверджує працездатність обраної організації тренувального процесу та дає змогу розглядати систему як придатну до тривалих експериментів, що можуть вимагати кількох запусків.

Після завершення 80 епох навчання та фінальної перевірки найкращих ваг модель YOLOv8s показала такі підсумкові результати на валідаційній вибірці: Precision = 0.864, Recall = 0.737, mAP50 = 0.853, mAP50-95 = 0.707. Окремо система Ultralytics зафіксувала характеристики швидкодії для процесу валідації: близько 0.6 мс на попередню обробку, 2.7 мс на інференс та 21.3 мс на постобробку одного зображення. Отримані результати свідчать про високу якість локалізації та класифікації дорожніх знаків і підтверджують придатність YOLOv8 як базової моделі для даної задачі [11; 13; 15].

На рисунку 3.5 доцільно подати криві навчання або фрагмент динаміки основних метрик YOLOv8, зокрема зміну mAP50-95 упродовж епох. Це

дозволить наочно продемонструвати процес збіжності моделі та підтвердити, що підвищення якості відбувалося поступово і стабільно.

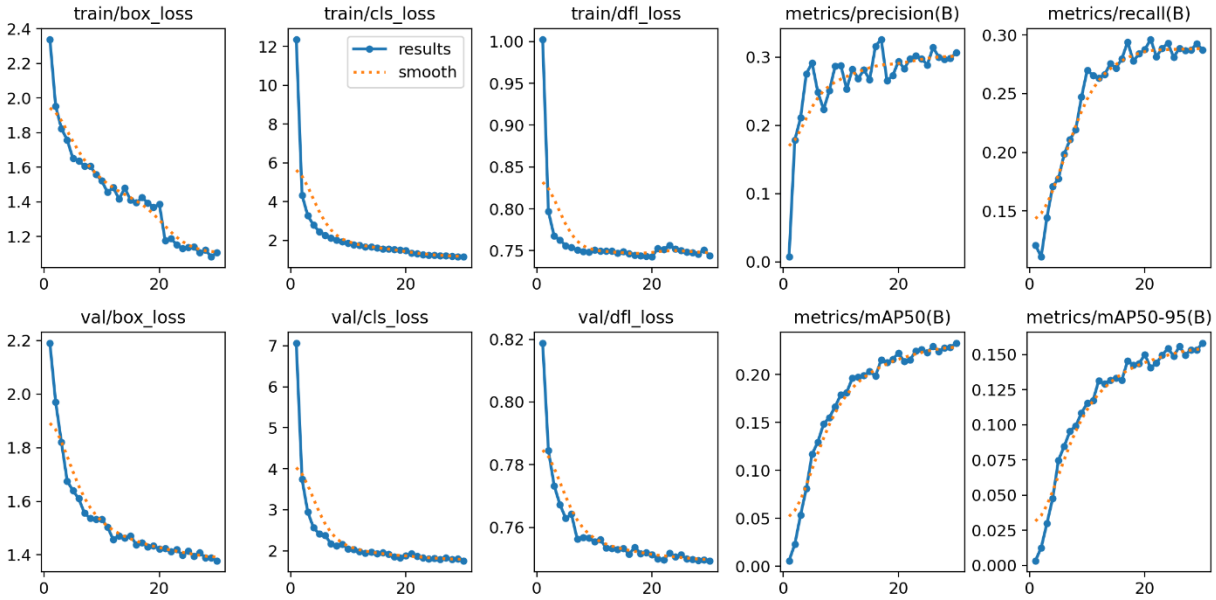

Рисунок 3.5 – Динаміка зміни показників моделі YOLOv8 у процесі навчання

Друга гілка дослідження, побудована на основі трансформерної архітектури FastViT, реалізовувалася як незалежна двостадійна система детекції. У цій конфігурації FastViT було інтегровано як базову мережу виділення просторових ознак (backbone) у комбінації з архітектурою Faster R-CNN, яка відповідає за генерацію пропозицій регіонів (Region Proposal Network, RPN). Оскільки така гібридна структура не підтримується стандартними рішеннями «з коробки», для неї було спроєктовано та програмно реалізовано власний об'єктно-орієнтований цикл навчання (Training Loop) на базі фреймворків PyTorch та TorchVision. Цей конвеєр охоплював усі ключові етапи глибинного машинного навчання: динамічне формування батчів із застосуванням спеціалізованої функції агрегації, виконання прямого проходу (forward pass) через трансформерні блоки,

складне обчислення багатокомпонентної функції втрат (помилки класифікації та регресії рамок), зворотне поширення градієнта, оптимізацію вагових коефіцієнтів, а також регулярну крокову валідацію. У практичному аспекті навчання цієї моделі виявилося значно ресурсомісткішим і тривалішим, ніж навчання YOLOv8, що пояснюється складнішою організацією двостадійного детекційного конвеєра, наявністю трансформерних механізмів уваги та загалом вищою обчислювальною вартістю архітектури.

Для об'єктивного оцінювання FastViT було організовано ізольований чесний експеримент у режимі прямого порівняння з YOLOv8. Навчання проводилося за ідентичних умов: використовувалися повний тренувальний набір даних, спільний YAML-конфігуратор, єдина валідаційна вибірка, просторовий розмір зображення 416x416, ліміт у 50 епох, механізм ранньої зупинки (patience=10), розмір пакету batch_size=8, workers=0 та базова вага `fastvit_t8.apple_in1k`. Важливою методологічною особливістю цього запуску було цілеспрямоване заморожування (freezing) backbone-мережі на початкових епохах. Це дозволило стабілізувати перенесення попередньо навчених на ImageNet ознак на специфічну задачу детекції дорожніх знаків та уникнути руйнування градієнтів. Завдяки цьому під час перших епох модель продемонструвала стрімке та послідовне покращення якості. Так, після першої епохи значення становили: mAP50 = 0.2232, mAP50-95 = 0.1316, Precision = 0.4575, Recall = 0.2596, F1 = 0.3313. Після другої епохи метрики впевнено зросли до mAP50 = 0.3098, mAP50-95 = 0.1957, Precision = 0.4957, Recall = 0.3516, F1 = 0.4114. Завершення третьої епохи зафіксувало подальшу позитивну динаміку: mAP50 = 0.3644, mAP50-95 = 0.2407, Precision = 0.4776, Recall = 0.4183, F1 = 0.4460.

Наведена динаміка переконливо доводить, що адаптована FastViT-орієнтована модель є цілком працездатною та здатною до ефективного навчання складної задачі локалізації дорожніх знаків. Водночас емпірично встановлено, що процес її тренування вимагає непропорційно великих витрат

машинного часу на одну епоху порівняно з одностадійною YOLOv8. За фактичними спостереженнями, опрацювання лише перших трьох епох конфігурації FastViT займало близько двох годин інтенсивних обчислень на графічному процесорі, що свідчить про істотно вище навантаження на підсистему пам'яті (VRAM). З огляду на це критично важливого інженерного значення набули реалізовані в коді алгоритми періодичного збереження станів моделі (checkpointing) — як найкращої ваги, так і останнього стану. Цей функціонал забезпечив відмовостійкість системи, дозволяючи безпечно призупиняти та відновлювати багатоденні експерименти без ризику втрати прогресу оптимізації [5; 19; 20].

Незважаючи на швидке зростання метрик на ранніх етапах, що свідчило про високий репрезентативний потенціал моделі до подальшого узагальнення, подальше навчання виявило специфічні обмеження трансформерних архітектур на датасетах з домінуванням дрібних об'єктів. Результати повного експериментального циклу підтверджують працездатність моделі та стабільність її архітектури, проте демонструють, що для розкриття максимального потенціалу FastViT та виходу метрик на рівень сучасних YOLO-подібних систем, гібридна модель вимагає застосування потужніших кластерних обчислювальних ресурсів та суттєвого збільшення розміру навчальної вибірки (batch size).

У межах комплексного тестування також було реалізовано етап початкового бенчмаркінгу, під час якого модель YOLOv8n показала такі базові результати: mAP50 = 0.2175, mAP50-95 = 0.1472, Precision = 0.5012, Recall = 0.2270, F1 = 0.3125 при продуктивності інференсу FPS = 93.42. Своєю чергою, конфігурація FasterRCNN + FastViT-T8 показала mAP50 = 0.2152, mAP50-95 = 0.1292, Precision = 0.4553, Recall = 0.2671, F1 = 0.3367, досягнувши швидкодії FPS = 71.28. Хоча цей тестовий запуск мав пілотний характер і відображав поведінку моделей лише на початкових фазах збіжності, він відіграв фундаментальну роль у верифікації працездатності обох модельних гілок,

підтвердив відсутність апаратних помилок і довів правильність роботи модуля автоматизованого формування звітів.

Для якісного аналізу результатів роботи розроблених систем у цьому підпункті наведено приклади валідаційних зображень із накладеними прогнозованими обмежувальними рамками, які автоматично згенеровані засобами програмного комплексу. Ця візуалізація є необхідним інструментом оцінювання, оскільки вона наочно розкриває механіку локалізації знаків на зображеннях високої складності. Аналіз таких графічних матеріалів дозволяє виявити специфіку типових помилок, а також переконливо підтверджує кореляцію розрахованих числових метрик із фактичною ефективністю роботи детектора.

Рисунок 3.6 – Приклад результатів детекції дорожніх знаків на валідаційних зображеннях

Таким чином, етап навчання, валідації та тестування моделей показав, що YOLOv8 продемонструвала високу якість детекції та стабільне завершене навчання, досягнувши підсумкових значень mAP50 = 0.853 та mAP50-95 = 0.707. Альтернативна модель на основі FastViT підтвердила свою працездатність і продемонструвала позитивну динаміку зростання метрик уже на ранніх етапах тренування, однак її навчання виявилося більш тривалим і ресурсомістким. Отримані результати створюють достатню основу для подальшого порівняльного аналізу двох моделей за точнісними та продуктивнісними характеристиками у наступному підпункті.

3.4. Порівняльний аналіз результатів роботи моделей за обраними метриками

Після завершення етапів навчання, валідації та тестування було виконано порівняльний аналіз результатів роботи моделей YOLOv8 та FastViT-based detector за обраними метриками якості та продуктивності. Для задачі детекції дорожніх знаків такий аналіз має включати не лише точність виявлення об’єктів, а й швидкодію, затримку інференсу, кількість параметрів моделі та витрати часу на навчання. Саме такий підхід дозволяє оцінити моделі не ізольовано, а в прикладному контексті, де важливим є баланс між якістю детекції та придатністю до практичного використання [15; 18].

Важливо підкреслити, що для прямого коректного порівняння двох моделей у даній роботі доцільно використовувати саме той запуск, у якому для обох архітектур були отримані завершені підсумкові файли summary та спільний comparison_results.csv. У наявних результатах таким запуском є каталог
diploma_experiments/run_20260307_175003, де збережено:

- comparison_results.csv
- yolo_summary.json

- fastvit_summary.json
- fps_comparison.png
- map5095_comparison.png

Саме цей запуск далі використовується як основа прямого попарного порівняння. Разом із тим у роботі також було отримано окремий тривалий фінальний запуск YOLOv8s на 80 епох, який показав значно кращі результати, однак для FastViT у такому самому завершеному форматі парного результату в архіві немає. Тому цей довгий запуск YOLO буде розглянуто окремо як додатковий результат поглибленого навчання, але не як абсолютно симетричну основу для прямого зіставлення з FastViT.

У таблиці 3.1 наведено основні результати, отримані у спільному порівняльному запуску run_20260307_175003.

Таблиця 3.1 - Порівняння моделей за основними метриками

Модель	mAP50	mAP50-95	Precision	Recall	F1	Latency	FPS	Params	Train time
YOLOv8n	0.2175	0.1472	0.5012	0.2270	0.3125	10.70	93.42	3.02	33.16
FasterRCNN+FastViT-T8	0.2152	0.1292	0.4553	0.2671	0.3367	14.03	71.28	20.47	41.22

Як видно з таблиці 3.1, у спільному завершеному запуску модель YOLOv8n мала незначну перевагу за метрикою mAP50: 0.2175 проти 0.2152. Абсолютна різниця становить лише близько 0.0023, тобто за порогом IoU = 0.5 обидві моделі показали близькі результати. Це означає, що на базовому рівні знаходження об’єктів обидва підходи продемонстрували працездатність і змогли навчитися виявляти дорожні знаки.

Проте суворіша метрика mAP50-95, яка краще враховує точність локалізації на різних порогах IoU, уже чіткіше відокремлює моделі. Для YOLOv8n значення становило 0.1472, тоді як для FasterRCNN+FastViT-T8 —

0.1292. Отже, у цьому аспекті YOLOv8n мала перевагу приблизно 0.0180 в абсолютних значеннях, що відповідає майже 14 % відносної переваги. Це свідчить про те, що в умовах спільного експериментального запуску YOLO не лише знаходила об’єкти, а й точніше локалізувала їхні межі.

За показником Precision модель YOLOv8n також виявилася кращою: 0.5012 проти 0.4553.

Це означає, що серед усіх передбачень YOLO частка правильних виявлень була більшою. Іншими словами, YOLOv8n формувала більш “чисті” детекції, допускаючи менше хибнопозитивних спрацьовувань, ніж FastViT-based detector.

Водночас за метрикою Recall перевагу показала модель FastViT-based detector: 0.2671 проти 0.2270 у YOLOv8n.

Це означає, що FastViT знаходила більшу частку реальних об’єктів у валідаційній вибірці, але досягала цього ціною меншої вибірковості. Саме тому в неї нижчий Precision, але вищий Recall. Така поведінка є типовою для моделі, яка менш консервативно видає передбачення.

Аналогічно і за F1-score перевага в спільному запуску залишилася за FastViT-based detector: 0.3367 проти 0.3125 у YOLOv8n.

Оскільки F1-score є гармонійним середнім між Precision і Recall, його значення показує, що в цьому конкретному запуску FastViT забезпечила більш збалансоване співвідношення між повнотою та точністю. Отже, за сукупністю лише трьох базових показників Precision, Recall і F1 не можна однозначно сказати, що одна модель абсолютно домінує: YOLOv8n точніша за Precision, FastViT сильніша за Recall і F1. Саме тому для коректного висновку необхідно враховувати також mAP50-95 і показники продуктивності.

Не менш важливою частиною порівняння є продуктивність моделей. У задачах детекції дорожніх знаків, особливо для застосування в транспортних

системах, модель повинна не лише правильно виявляти об’єкти, а й робити це швидко.

За показником Latency модель YOLOv8n мала явну перевагу: 10.70 мс проти 14.03 мс у FasterRCNN+FastViT-T8. Різниця становить приблизно 3.32 мс, тобто YOLOv8n працює швидше приблизно на 23.7 % у сенсі затримки одного проходу.

За показником швидкодії (FPS) перевага одностадійної архітектури YOLOv8n стає ще помітнішою: 93.42 FPS проти 71.28 FPS у гібридної моделі FasterRCNN+FastViT-T8.

Таким чином, YOLOv8n виявилася приблизно на 31 % швидшою за кількістю оброблених кадрів за секунду. Саме цей результат є одним із найвагоміших аргументів на користь використання YOLO як базової моделі для практичних систем аналізу дорожньої сцени, де висока кадрова частота є критично важливою для забезпечення миттєвої реакції автомобіля під час руху в реальному часі.

Ще одна суттєва відмінність між досліджуваними моделями спостерігається за загальною кількістю параметрів, що підлягають оптимізації. YOLOv8n має лише 3.02 млн параметрів, тоді як гібридна FasterRCNN+FastViT-T8 — 20.47 млн. Тобто друга модель є приблизно у 6.8 раза більшою за кількістю параметрів. Це безпосередньо впливає на підвищені вимоги до відеопам’яті (VRAM), загальну тривалість процесу навчання та суттєво ускладнює задачу практичного розгортання алгоритму на малопотужних вбудованих пристроях (Edge Devices). З погляду структурної компактності та ресурсоефективності модель YOLOv8n є значно привабливішою.

Загальний час навчання на тестовому запуску також виявився меншим у моделі YOLOv8n: 33.16 хв проти 41.22 хв у FasterRCNN+FastViT-T8.

Хоча абсолютна різниця тут не така драматична, як у випадку з кількістю параметрів або показником FPS, вона додатково свідчить на користь оптимізованої архітектури YOLOv8n, оскільки економія машинного часу дозволяє швидше проводити ітеративні експерименти з підбору гіперпараметрів. У сукупності отримані результати переконливо показують, що в межах спільного експериментального запуску YOLOv8n демонструє найкращий баланс між якістю просторової детекції та загальною ефективністю виконання.

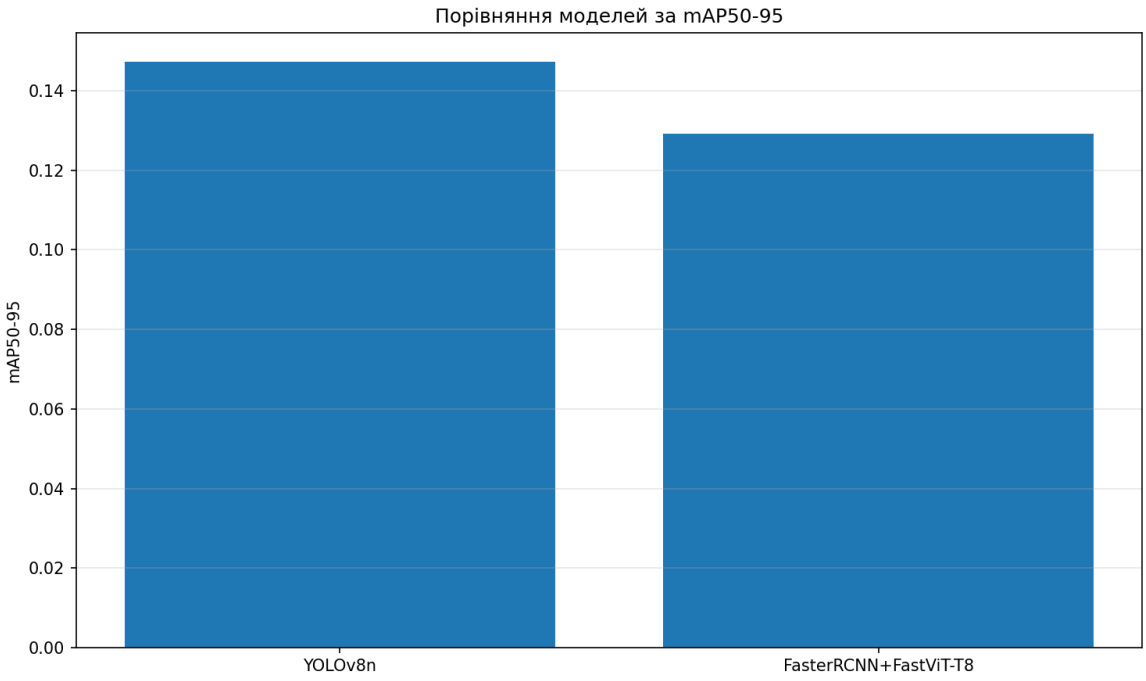

Рисунок 3.7 – Порівняння моделей за метрикою mAP50-95

На рисунку 3.8 наведено графік порівняння розроблених моделей за показником FPS, який наочно демонструє суттєву перевагу архітектури YOLOv8n за швидкістю інференсу. Ця візуалізація додатково підтверджує високу обчислювальну ефективність моделі, що є критично важливим критерієм для її впровадження у системи аналізу дорожньої обстановки в реальному часі.

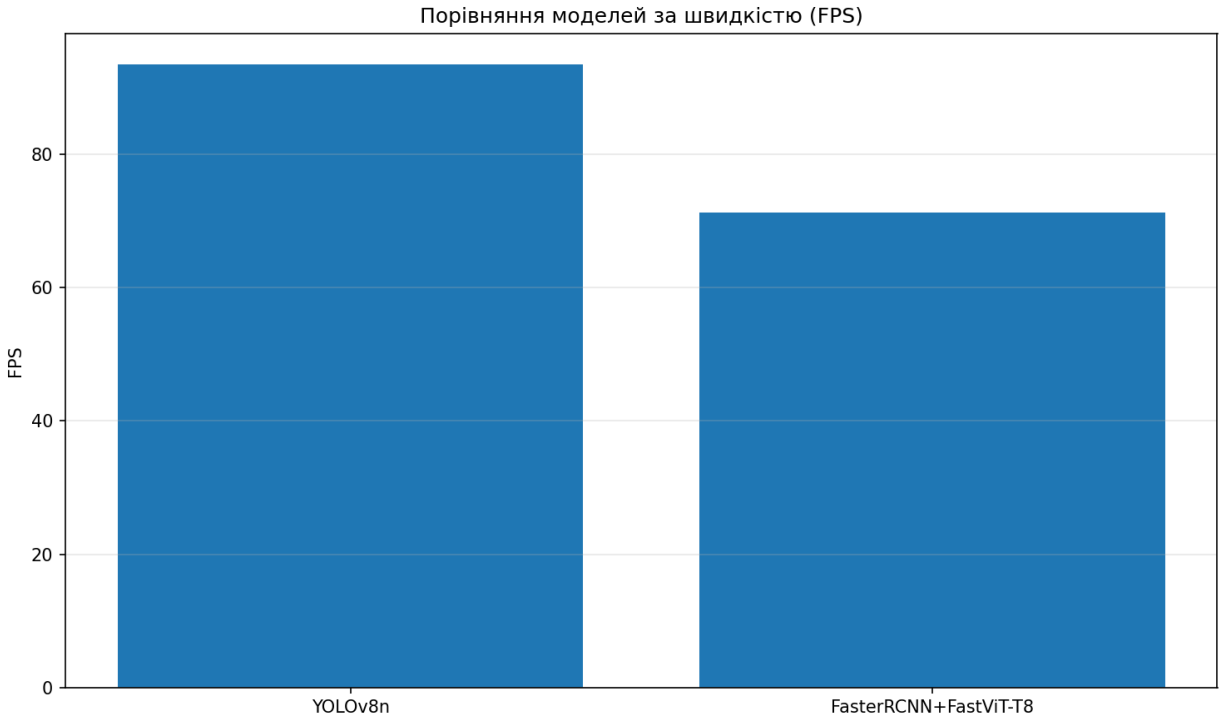

Рисунок 3.8 – Порівняння моделей за швидкістю роботи (FPS)

Окрім інтегральних підсумкових таблиць, у файлах запуску YOLOv8 доступні додаткові графічні матеріали, які дозволяють глибше проаналізувати поведінку моделі. Зокрема, в каталозі `runs/detect/yolov8_fast_experiment` були автоматично згенеровані такі файли, як BoxPR_curve.png, BoxF1_curve.png, BoxP_curve.png, BoxR_curve.png, confusion_matrix.png, confusion_matrix_normalized.png та results.png. Аналіз цих візуальних матеріалів, зокрема нормалізованих матриць помилок, дає змогу детально дослідити специфіку хибних спрацьовувань системи, наприклад, рівень плутанини між візуально спорідненими класами знаків або елементами складного фону.

Графік PR-curve (Precision-Recall curve), наведений на рисунку 3.9, дозволяє оцінити залежність між Precision та Recall при зміні порога спрацьовування моделі. Для детекції дорожніх знаків цей графік є важливим, оскільки демонструє, як змінюється поведінка моделі при переході від

консервативного до більш “сміливого” режиму виявлення. Чим ближче крива до верхнього правого кута, тим кращим є співвідношення між точністю та повнотою. Відповідно, використання цієї кривої дозволяє обґрунтовано обрати оптимальне значення порогу впевненості (Confidence Threshold) під час розгортання моделі у реальних умовах, мінімізуючи ризик пропуску важливих знаків без надмірного зростання кількості хибних детекцій.

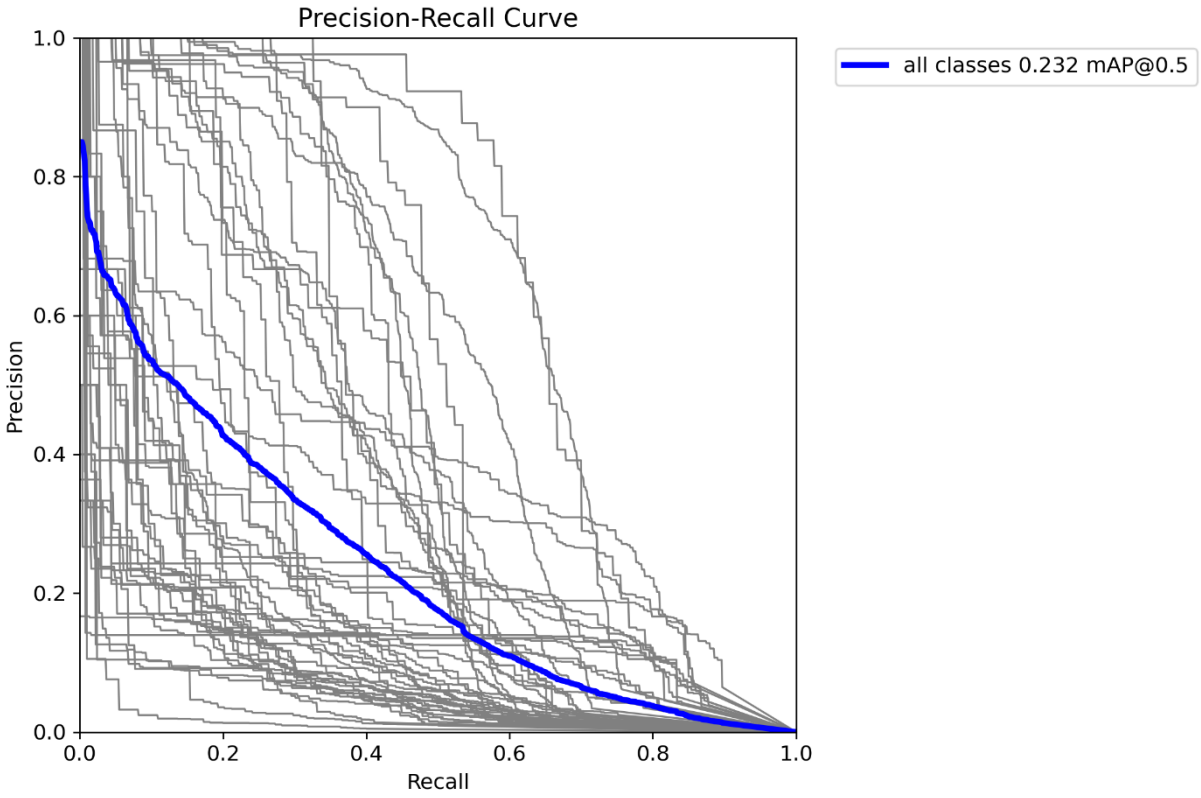

Рисунок 3.9 – PR-крива моделі YOLOv8

Графік F1-curve (рис. 3.10) є корисним для визначення такого діапазону порогів, у якому модель демонструє найбільш збалансоване співвідношення між Precision і Recall. Для даної задачі це особливо важливо, оскільки надто високий поріг може зменшити кількість хибних спрацьовувань, але водночас призвести до пропуску реальних дорожніх знаків.

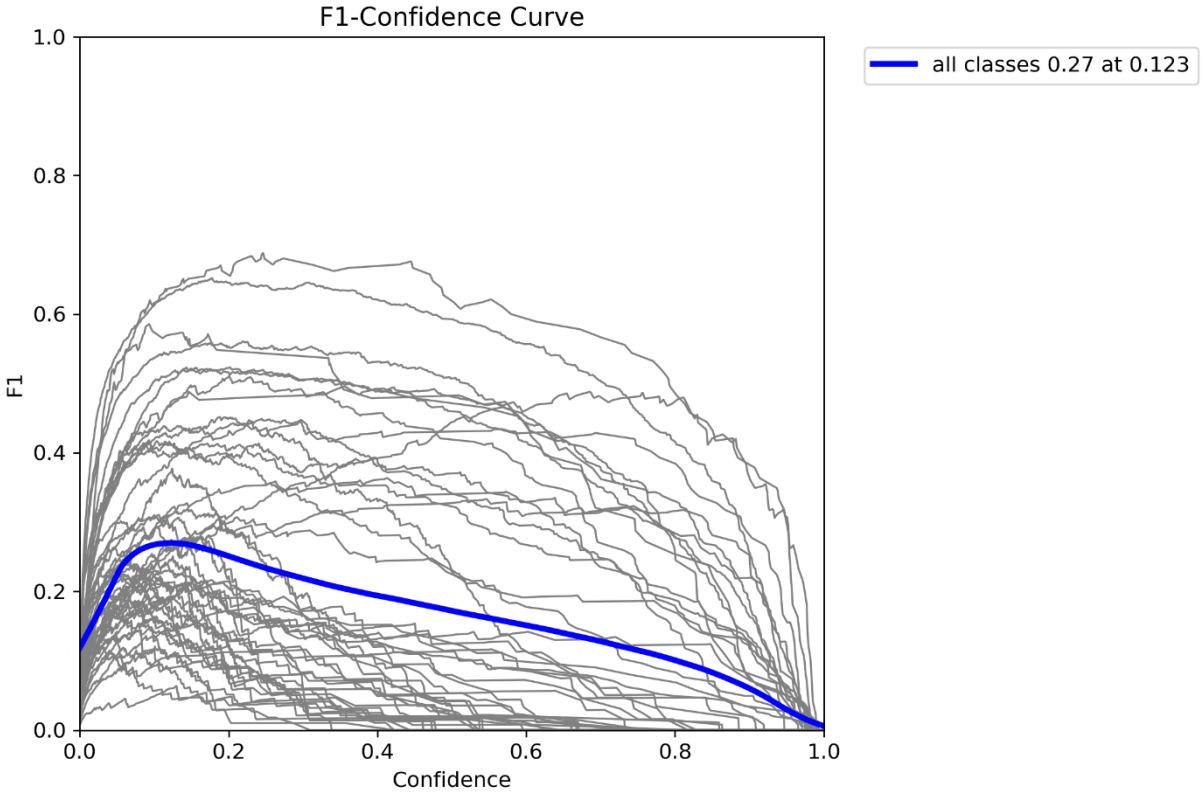

Рисунок 3.10 – Крива зміни F1-score моделі YOLOv8

Нормалізована матриця помилок (confusion matrix), представлена на рисунку 3.11, дозволяє детально оцінити, для яких класів дорожніх знаків модель працює з максимальною точністю, а для яких демонструє зниження якості розпізнавання. Крім того, вона дає змогу виявити конкретні пари класів, між якими найчастіше виникає алгоритмічна плутанина, а також відстежити випадки хибного детектування фонових об'єктів як знаків (False Positives). Для задачі аналізу дорожньої сцени цей інструмент є особливо показовим. Це зумовлено тим, що багато категорій знаків мають майже ідентичну візуальну структуру, форму та кольорову гаму (наприклад, знаки обмеження швидкості з різними числовими значеннями), що суттєво ускладнює їх класифікацію за умов перспективних спотворень або розмиття в русі. Використання саме нормалізованої версії матриці є критично важливим, оскільки вона дозволяє об'єктивно оцінювати точність розпізнавання рідкісних знаків навіть за умов

сильного дисбалансу класів у навчальній вибірці. Загалом, аналіз виявлених хибних спрацьовувань на основі цієї матриці надає цінну інформацію для подальшого вдосконалення системи, наприклад, шляхом цілеспрямованої аугментації проблемних зображень або налаштування вагових коефіцієнтів функції втрат.

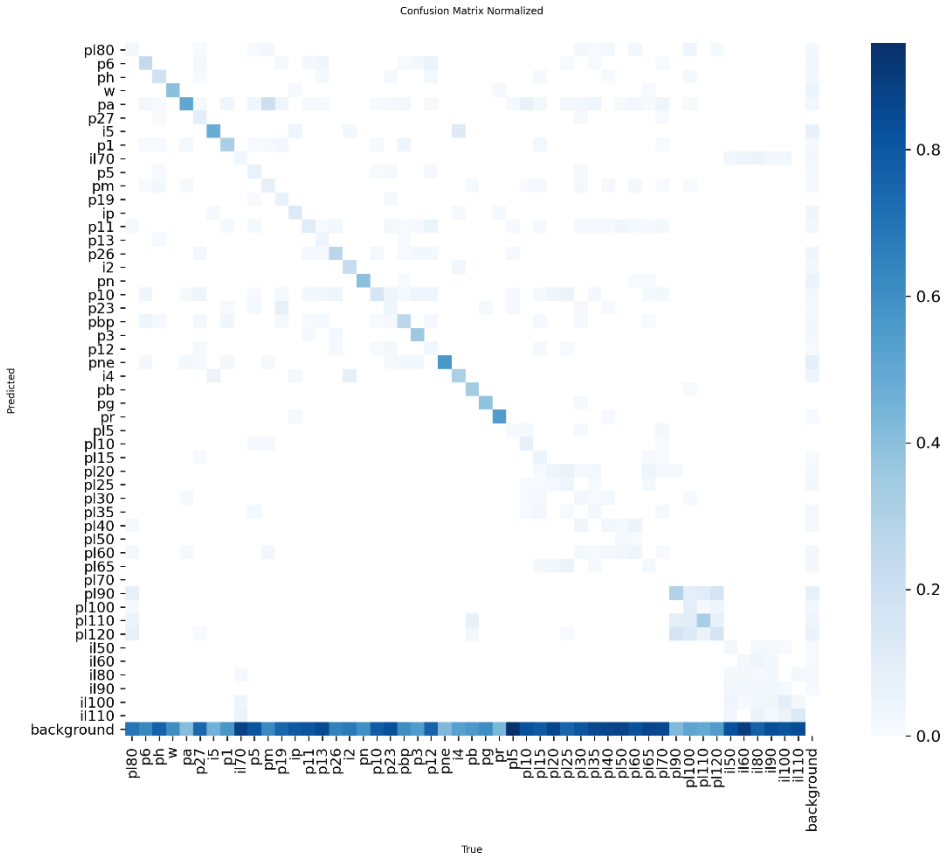

Рисунок 3.11 – Нормалізована матриця помилок моделі YOLOv8

Окремо варто звернути увагу на файл results.png, який відображає зміну основних тренувальних і валідаційних показників YOLOv8 у процесі навчання. Саме цей матеріал наочно показує, що в міру зростання номера епохи втрати зменшуються, а метрики якості зростають і поступово виходять

на плато. Це узгоджується з числовими результатами, наведеними в попередньому підпункті, та підтверджує коректну збіжність моделі.

Окремо в межах дослідження було отримано тривалий фінальний запуск моделі YOLOv8s на 80 епох, збережений у каталозі `diploma_experiments/run_20260307_214942/yolo/yolo_run`. За його результатами модель досягла високих показників на валідаційній вибірці, які зведено у таблиці 3.2. Максимальне значення mAP50-95 у логах було досягнуте приблизно на 67-й епосі, після чого показники фактично вийшли на плато.

Таблиця 3.2 - Додатковий результат тривалого навчання моделі YOLO

Модель	Precision	Recall	mAP50	mAP50-95	Епох
YOLOv8s	0.8625	0.7404	0.8522	0.7073	80

Ці значення є суттєво вищими за результати короткого порівняльного запуску YOLOv8n. Це свідчить про те, що архітектура YOLO має значний потенціал до підвищення якості за рахунок довшого навчання та використання більшої модифікації моделі. Однак з методичної точки зору ці результати не слід безпосередньо змішувати з парним порівнянням із FastViT-T8 із запуску run_20260307_175003, оскільки умови експерименту в цих випадках уже не є однаковими.

По-перше, у спільному завершеному запуску модель YOLOv8n продемонструвала кращі результати за mAP50-95, Precision, Latency, FPS, кількістю параметрів і часом навчання. Це дає підстави вважати її більш ефективною з позиції практичного використання, особливо якщо пріоритетом є швидкодія та компактність системи.

По-друге, модель FasterRCNN+FastViT-T8 показала вищі значення Recall і F1-score. Це означає, що вона була менш вибірковою та знаходила більшу частку об’єктів, однак робила це ціною більшої кількості хибнопозитивних передбачень і нижчої інтегральної якості локалізації на суворішій метриці mAP50-95.

По-третє, тривалий запуск YOLOv8s підтвердив, що архітектура YOLO добре масштабується за якістю та здатна досягати значно вищих результатів у разі повнішого і довшого навчання. Таким чином, навіть якщо в короткому спільному запуску обидві моделі були відносно близькими за mAP50, поглиблений експеримент із YOLO показав значно вищий рівень якості детекції.

Отже, результати порівняльного аналізу дають підстави зробити висновок, що YOLOv8 є більш доцільною моделлю для побудови практичної системи детекції дорожніх знаків, оскільки забезпечує кращий баланс між точністю, швидкістю роботи та компактністю архітектури. Водночас FastViT-based detector слід розглядати як перспективний альтернативний підхід, який підтвердив працездатність і здатність до навчання, однак у наявних завершених результатах поступився YOLOv8 за сукупністю найбільш важливих для прикладної системи характеристик.

3.5. Рекомендації щодо покращення моделей детекції дорожніх знаків

На основі проведеного експериментального дослідження, аналізу процесу навчання моделей та отриманих результатів доцільно сформулювати низку рекомендацій щодо підвищення ефективності системи детекції дорожніх знаків. Запропоновані напрями вдосконалення стосуються як моделі YOLOv8, що продемонструвала високий рівень якості та швидкодії, так і альтернативної моделі на основі FastViT, яка підтвердила свою

працездатність, але виявилася більш ресурсомісткою та менш ефективною за сукупністю показників у завершеному порівняльному експерименті.

Для моделі YOLOv8 основним напрямом покращення є подальше підвищення точності детекції за рахунок масштабування архітектури та оптимізації процесу навчання. У проведеному дослідженні фінальні результати були отримані для модифікацій YOLOv8n та YOLOv8s, однак використання більших варіантів моделі, таких як YOLOv8m або YOLOv8l, потенційно дозволить підвищити значення mAP50-95 за рахунок більшої кількості параметрів і складніших представлень ознак. Водночас таке покращення потребує додаткових обчислювальних ресурсів, тому доцільність його застосування залежить від вимог до швидкодії системи.

Важливим напрямом є також збільшення тривалості навчання та більш точне налаштування гіперпараметрів. Як показали результати експерименту, навіть після 80 епох модель продовжувала демонструвати незначне зростання якості, що свідчить про можливість подальшого покращення за рахунок більш тривалого тренування або використання більш складних стратегій зміни learning rate. Додатково доцільно дослідити вплив параметрів оптимізатора, зокрема значень learning rate, weight decay та параметрів scheduler, на фінальні метрики якості.

Ще одним ефективним способом підвищення якості YOLOv8 є вдосконалення аугментацій даних. У роботі вже використовувалися стандартні перетворення, однак доцільним є застосування більш складних методів, таких як Mosaic, MixUp, Copy-Paste або адаптивні аугментації, що враховують специфіку дорожніх знаків як дрібних об’єктів. Це дозволить покращити узагальнювальну здатність моделі та підвищити її стійкість до змін освітлення, масштабу та складності сцени.

Окрему увагу варто приділити задачі детекції малих об’єктів, яка є ключовою для даної предметної області. Для цього можуть бути використані

такі підходи, як збільшення роздільної здатності вхідних зображень, модифікація масштабів якорів або застосування спеціалізованих варіантів архітектури, оптимізованих для малих об’єктів. Також доцільним є використання test-time augmentation, що дозволяє покращити результати детекції без зміни процесу навчання.

Щодо моделі на основі FastViT, то основні напрями покращення пов’язані з оптимізацією архітектури детекційного конвеєра та зменшенням обчислювальної складності. У поточній реалізації використовується комбінація FastViT-backbone з детектором Faster R-CNN, що є двостадійним підходом і природно має вищу затримку інференсу. Тому одним із перспективних напрямів є інтеграція FastViT у більш швидкі одностадійні детектори або використання lightweight-варіантів архітектури, що дозволить зменшити latency та підвищити FPS.

Додатково доцільно вдосконалити механізм побудови ознак, зокрема оптимізувати конфігурацію FPN (Feature Pyramid Network). Це може включати зміну кількості рівнів піраміди, адаптацію каналів ознак або використання більш сучасних варіантів, таких як BiFPN. Такі зміни можуть позитивно вплинути на якість детекції малих об’єктів і підвищити значення метрик mAP50-95.

Ще одним важливим напрямом є оптимізація процесу навчання FastViT-моделі. У роботі було застосовано заморожування backbone на початкових епохах, однак подальші дослідження можуть включати більш гнучкі стратегії fine-tuning, поступове “розморожування” шарів, а також підбір оптимальної тривалості цього етапу. Крім того, доцільним є збільшення кількості епох навчання, оскільки результати початкових етапів показали позитивну динаміку, що свідчить про потенціал моделі до подальшого покращення.

З метою зменшення ресурсомісткості FastViT-орієнтованої системи також доцільно застосовувати методи оптимізації, такі як квантизація, pruning

або knowledge distillation. Це дозволить зменшити кількість параметрів і підвищити швидкодію моделі без істотної втрати якості. Особливо актуальними ці підходи є у випадку розгортання системи на обмежених апаратних платформах.

Окрім специфічних рекомендацій для кожної моделі, існують і загальні напрями покращення системи детекції дорожніх знаків. Передусім це розширення та покращення якості датасету. Збільшення кількості зображень, балансування класів і додавання прикладів у складних умовах (ніч, дощ, часткові перекриття) можуть суттєво підвищити узагальнювальну здатність моделей. Також доцільним є використання більш складних методів аугментації, орієнтованих саме на дорожню сцену.

Ще одним перспективним напрямом є застосування ансамблевих методів, які поєднують результати кількох моделей. Наприклад, комбінування YOLOv8 і FastViT-based detector може дозволити використати переваги обох підходів: високу точність і швидкодію YOLO та вищу повноту виявлення FastViT. Такий підхід може забезпечити покращення інтегральних метрик якості, хоча й потребує додаткових обчислювальних ресурсів.

Таким чином, проведений аналіз показує, що обидві моделі мають потенціал до подальшого вдосконалення. Для YOLOv8 основними напрямами є підвищення точності за рахунок масштабування та оптимізації навчання, тоді як для FastViT ключовими є зменшення обчислювальної складності та вдосконалення архітектури детекційного конвеєра. Реалізація запропонованих рекомендацій може суттєво підвищити ефективність системи детекції дорожніх знаків і є перспективним напрямом подальших досліджень.

Висновки до розділу 3

У третьому розділі було розглянуто програмну реалізацію нейромережевої системи детекції дорожніх знаків, підготовку даних,

налаштування процесу навчання моделей, а також результати їх валідації та порівняльного аналізу. Реалізована система об’єднала спільний модуль роботи з датасетом, окремі модельні гілки для YOLOv8 і FastViT-based detector, засоби оцінювання якості та продуктивності, а також автоматичне формування підсумкових звітів у вигляді таблиць, графіків і JSON-файлів.

У ході експериментів було встановлено, що модель YOLOv8 продемонструвала стабільне навчання, високу швидкодію та найкращий баланс між точністю детекції й обчислювальною ефективністю. У завершеному тривалому запуску YOLOv8s було отримано значення Precision = 0.8625, Recall = 0.7404, mAP50 = 0.8522, mAP50-95 = 0.7073, що підтвердило високу придатність цієї архітектури до задачі детекції дорожніх знаків.

Альтернативна модель на основі FastViT підтвердила працездатність і здатність до навчання на тому самому наборі даних. У спільному завершеному порівняльному запуску FasterRCNN+FastViT-T8 показала близьке до YOLOv8n значення mAP50 = 0.2152, а також вищі Recall = 0.2671 і F1 = 0.3367, однак поступилася за mAP50-95, швидкістю інференсу, кількістю параметрів і часом навчання. Це свідчить про перспективність такого підходу, але також показує його вищу ресурсомісткість та складність практичного застосування.

Порівняльний аналіз результатів дозволив встановити, що для побудови прикладної системи детекції дорожніх знаків найбільш доцільним є використання архітектури YOLOv8, оскільки саме вона забезпечує найкраще поєднання точності, швидкодії, компактності та зручності практичної реалізації. Отримані результати підтвердили досягнення мети експериментального дослідження та створили основу для формування загальних висновків кваліфікаційної роботи.

ВИСНОВКИ

У кваліфікаційній роботі було розв’язано актуальне науково-практичне завдання, що полягало у розробці та дослідженні нейромережевої системи детекції дорожніх знаків на базі архітектур YOLOv8 та FastViT. Актуальність теми зумовлена зростанням ролі інтелектуальних транспортних систем, систем підтримки водія та засобів комп’ютерного зору, для яких автоматичне виявлення дорожніх знаків є важливою складовою безпечного аналізу дорожньої сцени.

У теоретичній частині роботи було проаналізовано задачу детекції дорожніх знаків, її складність та особливості в реальних умовах. Показано, що основними ускладнювальними факторами є малий розмір об’єктів у кадрі, вплив освітлення, погодних умов, перспективних спотворень, часткових перекриттів та міжкласової подібності окремих знаків. Проведений огляд класичних і сучасних нейромережевих підходів дав змогу встановити, що саме моделі глибокого навчання є найбільш доцільними для розв’язання даної задачі, оскільки вони забезпечують вищу стійкість до складних сцен та кращу узагальнювальну здатність.

У роботі було розглянуто дві сучасні архітектурні концепції: YOLOv8 як одностадійний швидкий детектор об’єктів та FastViT як сучасну гібридну архітектуру виділення ознак, на основі якої реалізовано альтернативну модель детекції. Для дослідження було спроєктовано модульну нейромережеву систему, що включає спільну підготовку даних, окремі модельні гілки, блок оцінювання та модуль автоматичного формування підсумкових звітів. Така структура дозволила забезпечити єдину логіку експерименту та провести коректне порівняння моделей у межах спільної задачі.

У програмній частині було реалізовано повний експериментальний конвеєр: зчитування конфігурації датасету, підготовку зображень і анотацій, навчання моделей, валідацію, тестування, обчислення метрик якості та

оцінювання продуктивності. Для експериментів використано датасет TT100K, який містить анотовані зображення дорожніх знаків у реальних умовах і є придатним для аналізу задачі детекції дрібних об’єктів. Для оцінювання моделей використовувалися метрики Precision, Recall, F1-score, mAP50, mAP50-95, а також показники latency, FPS, кількість параметрів і час навчання.

У результаті експериментального дослідження було встановлено, що в завершеному спільному порівняльному запуску модель YOLOv8n показала mAP50 = 0.2175, mAP50-95 = 0.1472, Precision = 0.5012, Recall = 0.2270, F1 = 0.3125, FPS = 93.42, тоді як модель FasterRCNN+FastViT-T8 продемонструвала mAP50 = 0.2152, mAP50-95 = 0.1292, Precision = 0.4553, Recall = 0.2671, F1 = 0.3367, FPS = 71.28. Отже, FastViT-based detector показала вищі Recall і F1, але поступилася YOLOv8n за точністю локалізації, швидкодією, компактністю та часом навчання. Це дозволяє зробити висновок, що в рамках спільного завершеного запуску модель YOLOv8n виявилася більш ефективною саме для прикладного використання.

Додатково було виконано тривале навчання моделі YOLOv8s, яке дало значно кращі результати: Precision = 0.8625, Recall = 0.7404, mAP50 = 0.8522, mAP50-95 = 0.7073. Це підтвердило високий потенціал архітектури YOLO до покращення якості при довшому навчанні та використанні більш продуктивної конфігурації моделі. Для FastViT також було підтверджено позитивну динаміку навчання та працездатність архітектурного підходу, однак повний експеримент для цієї моделі виявився істотно більш ресурсомістким і тривалим.

Отже, мету кваліфікаційної роботи досягнуто: розроблено нейромережеву систему детекції дорожніх знаків, реалізовано дві альтернативні модельні гілки, проведено їх експериментальне дослідження та виконано порівняльний аналіз результатів. На основі отриманих даних встановлено, що YOLOv8 є найбільш доцільною архітектурою для побудови практичної системи детекції дорожніх знаків, оскільки забезпечує найкраще

поєднання точності, швидкодії, компактності та зручності реалізації. Водночас підхід на основі FastViT є перспективним напрямом подальших досліджень, особливо в аспекті вдосконалення гібридних backbone-мереж та оптимізації їх для задач детекції дрібних об’єктів.

Перспективи подальшого розвитку роботи полягають у завершенні повного довготривалого навчання FastViT-орієнтованої моделі в симетричних до YOLOv8 умовах, розширенні набору експериментів із різними конфігураціями backbone, використанні додаткових способів аугментації даних, а також у дослідженні можливості розгортання системи на вбудованих або мобільних обчислювальних платформах.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Zhu Z., Liang D., Zhang S.-H., Huang X., Li B., Hu S.-M. Traffic-Sign Detection and Classification in the Wild // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016. P. 2110–2118. URL: https://openaccess.thecvf.com/content_cvpr_2016/papers/Zhu_Traffic-Sign_Detection_and_CVPR_2016_paper.pdf (дата звернення: 09.03.2026).

2. Zhu Z., Liang D., Zhang S.-H., Huang X., Li B., Hu S.-M. Tsinghua-Tencent 100K: Traffic-Sign Detection Benchmark. URL: https://cg.cs.tsinghua.edu.cn/traffic-sign/ (дата звернення: 09.03.2026).

3. Redmon J., Divvala S., Girshick R., Farhadi A. You Only Look Once: Unified, Real-Time Object Detection // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016. P. 779–788. URL: https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/Redmon_You_Only_Look_CVPR_2016_paper.pdf (дата звернення: 09.03.2026).

4. Ren S., He K., Girshick R., Sun J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks // Advances in Neural Information Processing Systems. 2015. Vol. 28. URL: https://papers.nips.cc/paper_files/paper/2015/file/14bfa6bb14875e45bba028a21ed38046-Paper.pdf (дата звернення: 09.03.2026).

5. Vasu P. K. A., Gabriel J., Zhu J., Tuzel O., Ranjan A. FastViT: A Fast Hybrid Vision Transformer Using Structural Reparameterization // Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). 2023. P. 5785–5795. URL: https://openaccess.thecvf.com/content/ICCV2023/papers/Vasu_FastViT_A_Fast_Hybrid_Vision_Transformer_Using_Structural_Reparameterization_ICCV_2023_paper.pdf (дата звернення: 09.03.2026).

6. Vaswani A., Shazeer N., Parmar N., et al. Attention Is All You Need // Advances in Neural Information Processing Systems. 2017. Vol. 30. URL: https://papers.nips.cc/paper/7181-attention-is-all-you-need (дата звернення: 16.03.2026).

7. Dosovitskiy A., Beyer L., Kolesnikov A., et al. An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale // International Conference on Learning Representations (ICLR). 2021. URL: https://arxiv.org/abs/2010.11929 (дата звернення: 16.03.2026).

8. He K., Zhang X., Ren S., Sun J. Deep Residual Learning for Image Recognition // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016. P. 770–778. URL: https://arxiv.org/abs/1512.03385 (дата звернення: 16.03.2026).

9. Lin T.-Y., Maire M., Belongie S., et al. Microsoft COCO: Common Objects in Context // European Conference on Computer Vision (ECCV). 2014. P. 740–755. URL: https://arxiv.org/abs/1405.0312 (дата звернення: 02.04.2026).

10. Rezatofighi H., Tsoi N., Gwak J., Sadeghian A., Reid I., Savarese S. Generalized Intersection over Union: A Metric and a Loss for Bounding Box Regression // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2019. P. 658–666. URL: https://openaccess.thecvf.com/content_CVPR_2019/papers/Rezatofighi_Generalized_Intersection_Over_Union_A_Metric_and_a_Loss_for_CVPR_2019_paper.pdf (дата звернення: 02.04.2026).

11. Ultralytics. Explore Ultralytics YOLOv8. URL: https://docs.ultralytics.com/models/yolov8/ (дата звернення: 02.04.2026).

12. Ultralytics. Model Training with Ultralytics YOLO. URL: https://docs.ultralytics.com/modes/train/ (дата звернення: 21.04.2026).

13. Ultralytics. Model Validation with Ultralytics YOLO. URL: https://docs.ultralytics.com/modes/val/ (дата звернення: 21.04.2026).

14. Ultralytics. Configuration - Ultralytics YOLO Docs. URL: https://docs.ultralytics.com/usage/cfg/ (дата звернення: 21.04.2026).

15. Ultralytics. Performance Metrics Deep Dive. URL: https://docs.ultralytics.com/guides/yolo-performance-metrics/ (дата звернення: 27.04.2026).

16. Ultralytics. Model Benchmarking with Ultralytics YOLO. URL: https://docs.ultralytics.com/modes/benchmark/ (дата звернення: 27.04.2026).

17. Ultralytics. Object Detection. URL: https://docs.ultralytics.com/tasks/detect/ (дата звернення: 27.04.2026).

18. Lightning AI. Mean-Average-Precision (mAP) — TorchMetrics Documentation. URL: https://lightning.ai/docs/torchmetrics/stable/detection/mean_average_precision.html (дата звернення: 03.05.2026).

19. PyTorch. TorchVision Object Detection Finetuning Tutorial. URL: https://docs.pytorch.org/tutorials/intermediate/torchvision_tutorial.html (дата звернення: 03.05.2026).

20. PyTorch. Performance Tuning Guide. URL: https://docs.pytorch.org/tutorials/recipes/recipes/tuning_guide.html (дата звернення: 03.05.2026).

ДОДАТОК А. Лістинг програмного коду модуля ініціалізації та навчання моделі YOLOv8

Лістинг А.1

```python
from ultralytics import YOLO

def train_and_evaluate_yolo():
    """Ініціалізація, запуск навчання та валідація моделі YOLOv8"""

    # 1. Завантаження базової конфігурації моделі
    model = YOLO('yolov8s.pt')

    # 2. Запуск процесу навчання з визначеними гіперпараметрами
    print("[YOLOv8] Початок тренувального процесу...")
    train_results = model.train(
        data='mydata/YOLOv8_TT100K.yaml',
        epochs=50,
        imgsz=416,
        batch=-1,          # Автоматичний підбір розміру батчу під VRAM
        device=0,
        name='yolov8_fast',
        workers=8,         # Паралельне завантаження даних
        patience=10        # Механізм ранньої зупинки (Early Stopping)
    )

    # 3. Автоматична валідація на тестовій вибірці після навчання
    print("[YOLOv8] Оцінювання якості моделі...")
    metrics = model.val()

    # 4. Збереження результатів
    print(f"Підсумковий mAP50: {metrics.box.map50:.4f}")
    print(f"Підсумковий mAP50-95: {metrics.box.map:.4f}")

    return model, metrics

if __name__ == '__main__':
    train_and_evaluate_yolo()
```

ДОДАТОК Б. Лістинг програмного коду модуля навчання, валідації та порівняльного аналізу гібридної архітектури FastViT

Лістинг Б.1

```
import torch
from torch.utils.data import DataLoader
from full_diploma_system import (
    build_fastvit_detector, evaluate_fastvit, YoloDetectionDataset,
collate_fn
)

def train_fastvit_core(data_cfg, device, epochs=50, batch_size=8):
    """Основний цикл навчання гібридної моделі на базі FastViT-backbone"""

    # 1. Підготовка даних
    train_ds = YoloDetectionDataset(data_cfg["train"],
class_names=data_cfg["names"])
    val_ds = YoloDetectionDataset(data_cfg["val"],
class_names=data_cfg["names"])

    train_loader = DataLoader(train_ds, batch_size=batch_size,
shuffle=True, collate_fn=collate_fn)
    val_loader = DataLoader(val_ds, batch_size=1, shuffle=False,
collate_fn=collate_fn)

    # 2. Ініціалізація моделі, оптимізатора та масштабатора градієнтів
(AMP)
    model = build_fastvit_detector(
        num_classes=data_cfg["nc"],
        img_size=416,
        model_name="fastvit_t8.apple_in1k",
        pretrained_backbone=True
    ).to(device)

    optimizer = torch.optim.AdamW(model.parameters(), lr=1e-4,
weight_decay=5e-4)
    scaler = torch.amp.GradScaler("cuda")

    best_map5095 = 0.0

    # 3. Головний тренувальний цикл
    for epoch in range(1, epochs + 1):
        model.train()
        epoch_loss = 0.0

        for images, targets in train_loader:
            images = [img.to(device, non_blocking=True) for img in images]
```

Закінчення лістингу Б.1

```
            targets = [{k: v.to(device) for k, v in t.items()} for t in
targets]

            optimizer.zero_grad(set_to_none=True)

            # Змішана точність для прискорення на GPU
            with torch.amp.autocast("cuda"):
                loss_dict = model(images, targets)
                losses = sum(loss for loss in loss_dict.values())

            scaler.scale(losses).backward()
            scaler.step(optimizer)
            scaler.update()

            epoch_loss += losses.item()

        # 4. Валідація та збереження найкращих ваг (Checkpointing)
        val_metrics, _ = evaluate_fastvit(model, val_loader, device)
        current_map5095 = float(val_metrics.get("map", 0.0))

        print(f"[Epoch {epoch}/{epochs}] Loss:
{epoch_loss/len(train_loader):.4f} | mAP50-95: {current_map5095:.4f}")

        if current_map5095 > best_map5095:
            best_map5095 = current_map5095
            torch.save(model.state_dict(), "fastvit_best.pth")

    return model
```