Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
(повна назва)

Кафедра Електронних обчислювальних машин
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Методи класифікації зображень з використанням машинного навчання

(тема)

Виконав:
здобувач 2 року навчання,
групи СПм-23-5
Михайло ПРОКОПЧИК
(власне ім'я, прізвище)

Спеціальність
123 «Комп'ютерна інженерія»
(код і повна назва спеціальності)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма
Системне програмування
(повна назва освітньої програми)

Керівник: доц. Віталій ТКАЧОВ
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ЕОМ _____ Андрій КОВАЛЕНКО
(підпис) (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління

Кафедра Електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 «Комп'ютерна інженерія»
(код і повна назва)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системне програмування
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
"____" _____ 20__ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Прокопчику Михайлу Володимировичу
(прізвище, ім'я, по батькові)

1. Тема роботи Методи класифікації зображень з використанням машинного навчання

затверджена наказом по університету від " 21 " квітня 2025 р. № 296Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 16 червня 2025 р.

3. Вхідні дані до роботи
датасет для навчання нейромережевих моделей (складається з навчальної та тестової частин),
інтегроване середовище розробки для мови Python – PyCharm,
бібліотеки машинного навчання - TensorFlow, NumPy, Pandas, OpenCV

4. Перелік питань, що потрібно опрацювати у роботі
- аналіз існуючих методів класифікації зображення
- пошук та створення набору даних для навчання
- емпіричне дослідження методів класифікації зображень за допомогою методів машинного навчання
- аналіз отриманих результатів

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій презентаційний матеріал у вигляді 31 слайд

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

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

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Огляд класичних методів класифікації зобра-	22.04.25-29.04.25	
2	жень		
3	Вибір та обґрунтування методики дослідження	30.04.2025-05.05.25	
4	Вибір інструментальних засобів	06.05.25-09.05.25	
5	Розробка методу класифікації	10.05.25-21.05.25	
6	Проведення експериментів	22.05.25-02.06.25	
7	Оформлення матеріалів кваліфікаційної роботи	03.06.25-05.06.25	
8	Подання кваліфікаційної роботи керівникові та	06.06.25-09.06.25	
	її попередній захист		
9	Подання кваліфікаційної роботи на рецензу-	10.06.25-12.06.25	
	вання		

Дата видачі завдання " 21 " квітня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. Ткачов В.М.
(підпис) (посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 111 с., 7 табл., 26 рис., 20 джерел.

МАШИНЕ НАВЧАННЯ, ML, CNN, ЗОБРАЖЕННЯ, МЕТОД, OPENCV.

Метою кваліфікаційної роботи є емпіричний аналіз та порівняння існуючих методів класифікації зображень з низькою роздільною здатністю, виявлення найкращої архітектури немережевого класифікатора та впровадження покращень для роботи із зображеннями низькою роздільною здатністю на прикладі класифікації БПЛА. Покращення роботи за часом та зменшення використовуваних ресурсів під час виконання класифікації методом за допомогою пре обробки вхідних зображень.

У ході виконання кваліфікаційної роботи було проаналізовано існуючи архітектури непромережених класифікаторів, таких як, класичні методи вододіл та затоплення, та з використанням машинного навчання CNN, Transformer, RNN. Було доказано переваги CNN для класифікації зображень в порівнянні з іншими типами. Було проаналізовано та досліджено різні реалізації CNN архітектури емпіричним шляхом для отримання результатів якості класифікації. Було проаналізовано усі популярні набори даних для навчання і виявлено мінімальну кількість зображень з БПЛА. Створено нових вид класифікатора на базі CNN архітектури зі зміною шарів stride-конвульсії та пулінгу. Проаналізовані існуючи методи пре обробки зображень та вибрані відповідні методи для збільшення точності та якості класифікації.

ABSTRACT

Master’s thesis: 111 pages, 7 table, 26 figures, 20 sources.

МАШИНЕ НАВЧАННЯ, ML, CNN, ЗОБРАЖЕННЯ, МЕТОД, OPENCV.

The major goal of this thesis is to empirically analyze and compare existing methods for classifying low-resolution images, identify the best architecture for a non-network classifier, and implement improvements for working with low-resolution images using the example of UAV classification. Improving performance in terms of time and reducing the resources used during classification using the method of pre-processing input images.

In order to research existing architectures of non-network classifiers were analyzed, such as classical watershed and flooding methods, and using machine learning CNN, Transformer, RNN. The advantages of CNN for image classification compared to other types were proven. Various implementations of CNN architecture were analyzed and researched empirically to obtain classification quality results. All popular training datasets were analyzed and the minimum number of images from UAVs was identified. A new type of classifier based on CNN architecture was created with changes in stride-convolution and pooling layers. Existing image preprocessing methods were analyzed and appropriate methods were selected to increase classification accuracy and quality.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ 8
ВСТУП 9
1.1 Огляд проблемної області 10
1.2 Обґрунтування актуальності теми роботи 12
1.3 Огляд класичних методів обробки цифрових зображень 13
1.3.1 Алгоритм заповнення 13
1.3.2 Алгоритм вододілу 15
1.4 Огляд методів обробки цифрових зображень з використанням методів машинного навчання 17
1.4.1 Детальний опис принципу та особливості роботи згорткових нейронних мереж 20
1.5 Порівняльний аналіз класичних та нейромережевних методів класифікації зображень 23
1.6 Обмеження згорткових нейронних мереж для рішення задачі класифікації зображень 26
2 МЕТОДИКА ВИКОРИСТАННЯ CNN ДЛЯ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ 29
2.1 Набори даних для використання 29
2.2 Основним компонентом CNN 31
2.3 Архітектури CNN 38
3 АНАЛІЗ ТА ПРОЕКТУВАНЯ МЕТОДУ КЛАСИФІКАЦІЇ НА ОСНОВІ CNN АРХІТЕКТУРИ 43
3.2 Блок предобробки зображення 46
3.3 Блок аналізу базової інформації про зображення для виклику оптимального класифікатору 50
3.4 Блок класифікації на основі методу на базі CNN 51
3.4.1 Новий блок методу: PUNSH-Conv 51

3.4.2 Non-strided згортка 53
3.5 Блок постобробки результатів класифікації 53
4 АНАЛІЗ ОТРИМАНИХ РЕЗУЛЬТАТІВ 55
4.1 Оцінка впливу роздільної здатності, наявності шумів та розмиття на точність класифікації зображень (Accuracy) на основі запропонованого та традиційних методів 56
4.2 Оцінка впливу використання методів передобробки на точність класифікації зображень (Accuracy) на основі запропонованого методу 60
4.3 Оцінка впливу використання модуля визначення базової інформації про зображення на навантаження обчислювача 61
5 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ 63
5.1 Вибір мови програмування 63
5.2 Структура Програмного Забезпечення 65
5.2.1 Метод на його навчання 66
ВИСНОВКИ 69
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ 70
ДОДАТОК А 72
ДОДАТОК Б 89
ДОДАТОК В 91
ДОДАТОК Г 95
ДОДАТОК Ґ 97
ДОДАТОК Д 103
ДОДАТОК Е 105
ДОДАТОК Є 107
ДОДАТОК Ж 110

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

БПЛА – безпілотний літальний апарат

ЕОМ – електронна обчислювальна машина

КТ – комп'ютерна томографія

НМ – нейронна мережа

ШНМ – штучна нейронна мережа

CMYK – блакитний, пурпуровий, жовтий, ключовий (англ., Cyan, Magenta, Yellow, Key)

CNN – згорткова нейронна мережа (англ., Convolutional Neural Network)

FPN – пірамідальна мережа мережа (англ., Feature Pyramid Network)

ML – машинне навчання(англ., Machine Learning)

PUNSH – воротне переставляння пікселів (англ., Pixel Unshuffling)

RBG – червоний, синій, зелений (англ., Red Blue Green)

RNN – рекурентна нейронна мережа (англ., Recurrent Neural Network)

VGG – група візуальної геометрії (англ., Visual Geometry Group)

ВСТУП

У сучасному світі стрімко зростає обсяг візуальних даних, що вимагають автоматизованої обробки. Одним із ключових завдань у цій сфері є класифікація зображень – процес, що дозволяє виявляти та розпізнавати об’єкти без участі людини. Особливої актуальності набуває розпізнавання БПЛА, які широко використовуються в різних сферах і можуть становити потенційну загрозу в контексті безпеки.

Традиційні методи, такі як Floodfill чи Watershed, залишаються ефективними у простих випадках, проте часто не справляються зі складними або змінними умовами зйомки. На зміну їм приходять моделі машинного навчання, зокрема CNN, RNN та Transformers, що демонструють високу точність і здатність до адаптації, але можуть погану працювати з аугментованими зображеннями.

У цій роботі розглядається підхід до класифікації зображень із на основі CNN з підходом поліпшення роботи з аугментованими зображеннями. Основна мета – створити програмне рішення для розпізнавання БПЛА, здатне працювати в режимі реального часу. Крім того, створюється власний метод класифікації, та проводиться порівняльний аналіз його та існуючиз сучасних методів, щоб визначити найбільш ефективний підхід до розв’язання поставленої задачі.

Результати дослідження мають прикладну цінність і можуть бути використані в системах відеоспостереження, контролю повітряного простору та інших сферах, де потрібна надійна і швидка обробка візуальної інформації.

1 ОБРОБКА ЗОБРАЖЕНЬ СУЧАСНИМИ МЕТОДАМИ ТА З ВИКОРИСТАННЯМ МЕТОДІВ МАШИННОГО НАВЧАННЯ

1.1 Огляд проблемної області

Протягом останнього десятиліття зростає поширеність БПЛА, які використовуються для таких цілей, як доставка товарів, дистанційне зондування, геодезія, інспекція та розваги. Як і більшість технологій, БПЛА можуть бути використані зловмисно. Мотивацією для такого зловмисного використання може бути бажання викликати роздратування, порушити приватність, заподіяти фізичну шкоду або навіть закрити повітряний простір з відповідними економічними наслідками.

Після виявлення зловмисного БПЛА можна розглянути різні заходи протидії, включаючи захоплення або знищення БПЛА, блокування його бездротового зв’язку, щоб він не міг керуватися або передавати дані, а також ідентифікацію та накладення штрафу на власника. Залежно від конкретного сценарію, виявлення може залежати від того, чи повідомить особа про зловмисний БПЛА владі, можливо, додавши до повідомлення фотографію БПЛА. Було б дуже корисно мати можливість автоматично передбачити виробника та конкретну ідентифікацію продукту зловмисного БПЛА на основі фотографії за допомогою навченої моделі класифікації зображень, навіть якщо БПЛА знаходиться відносно далеко від особи, яка робить фотографію.

Класифікація зображень використовується в великій кількісті сфер роботи починаючи з медичної сфери, де класифікація зображень використовується для діагностики легенів, та моніторингом хвороб, ідентифікація об’єктів на зображеннях зі спутників та завершуючи Mil-технологіями, де класифікація зображень допомагає класифікувати ціль та автоматизація процесів моніторингу стану різних об’єктів на зображенні. Ще

однією з ключових зон класифікації зображень в Mil-Tech є класифікація БПЛА. На рисунку 1.1 можна побачити опис галузей використання класифікації зображень.

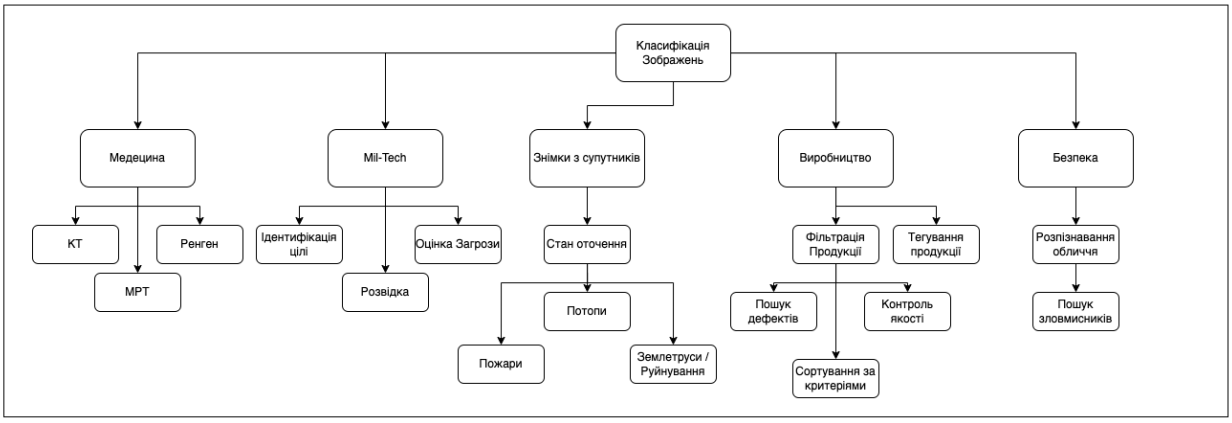

Рисунок 1.1 – Сфери використання класифікації зображення

Зображення – це двомірна матриця пікселів. Значення кожного пікселя пропорційне яскравості відповідної точки в матриці.

Найпоширенішими видами зображень є векторні та растові. Кожен з цих видів має різні реалізації, ми сфокусуємося тільки на їх основних характеристиках.

Векторні зображення це візуальні зображення, які створюються безпосередньо з геометричних фігур, визначених на декартовій площині, таких як точки, лінії, криві та багатокутники.

Растрове зображення це двовимірне зображення, представлена у вигляді прямокутної матриці або сітки пікселів. Растрове зображення технічно характеризується шириною і висотою зображення в пікселях та кількістю бітів на піксель.

Ще кожен з видів зображення використовує певну модель кольору. Найпоширенішими моделями є RGB та CMYK. RGB це адитивна колірна модель, тобто вона створює кольори шляхом додавання різних інтенсивностей червоного, зеленого та синього світла. CMYK це субтрактивна колірна модель, яка створює кольори шляхом поєднання різних

відсотків блакитного, пурпурного, жовтого та чорного чорнил. На рисунку 1.2 показано приклад зображення у виді матриці.

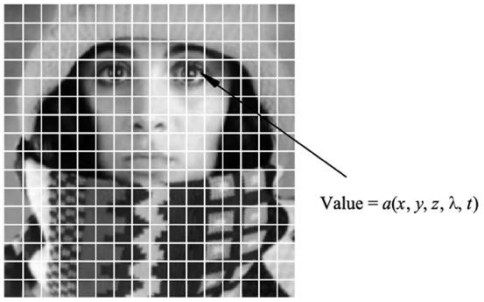

Рисунок 1.2 – Відображення зображення як матриці

Тема збереження зображень є досить широкою. Було зроблено аналіз видів зображення, моделей кольору та способом збережень у виді матриці.

1.2 Обґрунтування актуальності теми роботи

БПЛА є досить малими об’єктами та несуть за собою великий рівень небезпеки, їх класифікація на зображені може допомогти знизити або позбавити рівень небезпеки над критично важливою інфраструктурою.

Якість класифікацію БПЛА, тобто чітке розпізнавання типу та моделі БПЛА є критично важливою частиною протидії зловмисникам, де за допомогою правильно класифікованого БПЛА можна виконати оптимальні дії знешкодження небезпеки, такими як його знищення, використання методів РЄБ. Класифікацію БПЛА можливо роботи різними методами класифікації, такими як радіопошук, звукий або пошук на зображенні. На поточний момент найвища якість класифікації є класифікація на зображені. Усі поточні рішення обробляють зображення великої роздільної здатності та/або без різних природних обставин, таких як туман або дощ. Це зменшує область та час використання класифікаторів БПЛА на основі методів класифікації зображень.

1.3 Огляд класичних методів обробки цифрових зображень

На поточний момент класичні методи досить часто використовуються в сфері медицині, а саме у комп’ютерній томографії. Там кожен воксель має значення в шкалі Хаунсфілда. Значення в одиницях Хаунсфілда базуються на коефіцієнті лінійного ослаблення тканин μ, який трансформується з використанням щільності дистильованої води та повітря (–1000Hu), тому значення кожного пікселя відображає фактичну щільність тканини, наприклад: тканина легень знаходиться в діапазоні (–700Hu, –600Hu). Завдяки цій властивості навіть найпростіші методи, такі як порогове значення, все ще застосовуються радіологами у повсякденній роботі.

Є два найпопулярніші класичні методи для аналізу та обробки зображень:

- алгоритм заповнення (Floodfill);
- алгоритм вододілу (Watershed).

1.3.1 Алгоритм заповнення

Алгоритм заливки, часто іменований як "заливка відром", є класичним методом сегментації зображень, який ґрунтується на принципах зв’язності у дискретному просторі пікселів. З формальної точки зору, цей метод розглядає зображення як неорієнтований граф, де кожен піксель відповідає вершині, а зв’язки між сусідніми пікселями – ребрам. Процес заливки починається з вибору початкового (насіннєвого) пікселя, значення інтенсивності якого слугує еталоном. Далі алгоритм рекурсивно або ітеративно досліджує його сусідів у межах певного радіусу (звичайно за схемою 4– або 8–зв’язності), перевіряючи, чи різниця інтенсивності між ними та базовим пікселем не перевищує встановлений поріг толерантності.

Якщо ця умова виконується, відповідний піксель включається до

області заливки, і процедура повторюється для його сусідів. Таким чином, формується область із пікселів зі схожими радіометричними властивостями, що дозволяє ефективно виділяти однорідні ділянки в умовах слабкого контрасту або нерівномірного освітлення. Цей підхід широко застосовується як на етапі первинної сегментації, так і для корекції масок після застосування порогових методів, зокрема при обробці медичних зображень, де важливо точно виділити анатомічні структури зі схожими градаціями сірих рівнів Алгоритми заливки демонструють високу стійкість до незначних шумів у зображенні та можуть адаптуватися до змін локального контексту, зберігаючи при цьому обчислювальну ефективність, що робить їх корисними в реальному часі та вбудованих системах [1].

Хоча алгоритм сегментації і є корисним в медичній області, він може використовувати і в інших сферах, на рисунку 1.3 можна побачити приклад використання заповнення для розпізнання кількості монет.

Рисунок 1.3 – Приклад використання алгоритму floodfill

У випадку КТ це дуже корисно для сегментації цілих груп тканин, таких як легені та трахея, кістки та черевна порожнина. Однак слід звернути увагу на початкову точку і значення допуску, оскільки обидва ці фактори мають величезний вплив на результат алгоритму.

1.3.2 Алгоритм вододілу

Алгоритм вододілу є одним із найпоширеніших методів сегментації зображень, заснованих на аналізі градієнтів інтенсивності, що дозволяє інтерпретувати зображення як топографічну поверхню. У цій інтерпретації інтенсивність пікселів розглядається як висота: пікселі з низькою інтенсивністю утворюють "долини", а з високою – "хребти" або "гірські вершини". Суть алгоритму полягає у симуляції заповнення цих топографічних басейнів "водою", яка починає підніматися з локальних мінімумів (рисунок 1.4). У ході цього підйому, як тільки два потоки з різних джерел зустрічаються, алгоритм формує роздільну межу, яка слугує як сегментуюча лінія між регіонами [2].

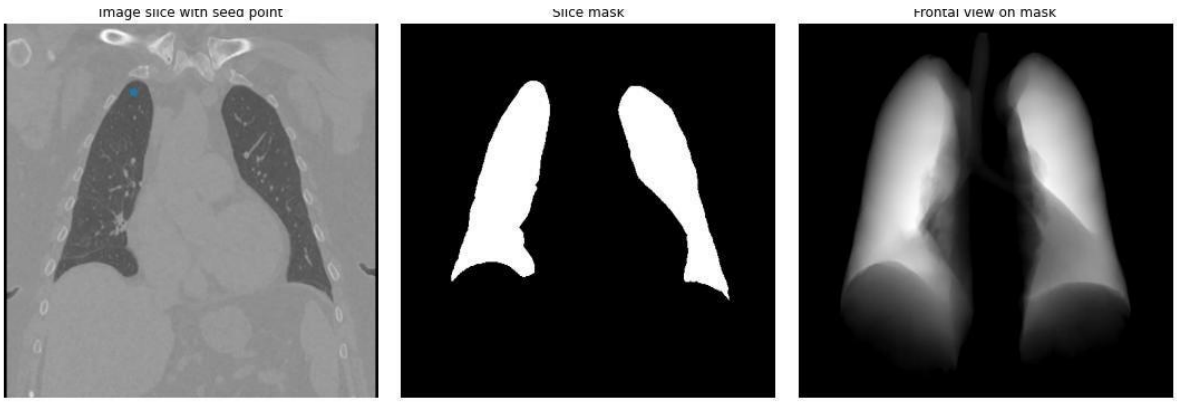

Рисунок 1.4 – КТ легенів з використанням watershed алгоритму

Цей процес дозволяє отримати детальну сегментацію на основі регіонів, особливо в задачах, де структура об’єкта добре корелює з локальними екстремумами інтенсивності, як-от у випадках морфометричного аналізу органів, виділення долей легень або структур шлунково–кишкового тракту. Однак для коректної роботи алгоритму вододілу необхідна попередня обробка, що включає генерацію так званих маркерів (markers), які виконують роль джерел для розливу. Зазвичай маркери поділяються на три типи:

маркери фону, що охоплюють області, поза межами об'єкта інтересу наприклад, повітря або зовнішні тканини, внутрішні маркери, які строго локалізуються всередині об'єкта, і нейтральна або пошукова зона, що включає невизначені пікселі (рисунок 1.5).

Формування цих маркерів часто здійснюється за допомогою трансформації відстаней після попередньої бінаризації, що дозволяє точно задати топологічні мінімальні точки. У присутності шумів або маловиражених країв застосовуються морфологічні фільтри, що дозволяють згладити градієнтну карту, тим самим зменшуючи кількість хибних регіонів. Незважаючи на концептуальну простоту, реалізація алгоритму вимагає ретельного налаштування, оскільки надлишкова кількість локальних мінімумів може призвести до перенасичення сцени неінформативними межами, що потребує додаткового етапу постобробки або включення алгоритмів об'єднання регіонів.

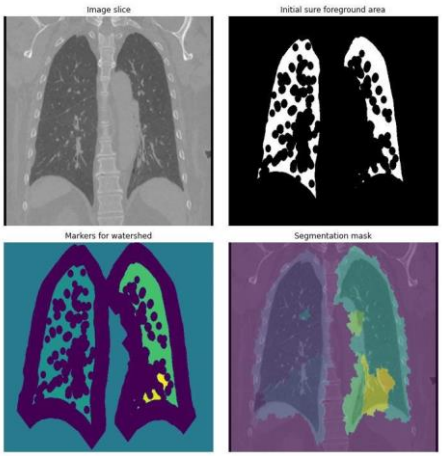

Рисунок 1.5 – Використання watershed та floodfill алгоритмів для аналізу зображення легенів

Основна відмінність між вододілом і заливкою полягає в тому, що вона зберігає розбиття областей на основі найнижчих точок маски. Крім того, вододіл дуже залежить від початкових областей і має тенденцію включати область навколо тканини в цьому конкретному випадку.

Однак ці методи класифікації працюють тільки з певним, перед

обробленим типом зображень, що робить їх область використання дуже вузькою. Як було зазначено, їх практична сфера використання залишається тільки в області медицини.

1.4 Огляд методів обробки цифрових зображень з використанням методів машинного навчання

Класифікація зображень з використанням методів машинного навчання є поширеним підходом для отримання необхідної інформації з зображення. Вони використовується в майже усіх областях, таких як Mil-Tech, безпека та охорона природи, виробництві та медицині, хоча в осатаній області, хоча вони і є досить розповсюдженими, вони можуть уступати місце класичним методам, бо велика кількість методів, заснованих на машинному навчанні працюють як чорні скриньки і в певних областях медицини, це робить їх використання неможливим.

Методи засновані на використані методів машинного навчання використовують різні типи архітектури та підходи для класифікації.

Але необхідно зауважити, що процес класифікації зазвичай має наступні кроки:

1 передобробка зображення, яка передбачає зміну розміру зображення, нормалізацію, видалення шумів, корекція кольору, сегментацію;

2 використання нейромережевого класифікатора (алгоритм унікальний для кожного з методів);

3 пост обробка зображень, яка може передбачати наступні опціональні кроки вибір класу з найвищою ймовірністю, калібрування ймовірностей, фільтрація, згладжування та ансамблювання, семантичне групування, форматування вихідних даних.

Типовим для усіх методів класифікаціє є етап навчання, хоча і тут є унікальність певних методів(використання вчителя, або без нього), поширеним методом є наступні кроки:

1 пошук існуючих наборів даних для необхідної області класифікації;

2 використання методів аргументації зображення, якщо кількість зображень є малою, над набором для тренування;

3 використання різних підходів для навчання на рівні EOM (GPU, CPU, оптимізатори);

4 використання варіаційного набору для виявлення перенавчання.

Найпоширенішими є методи класифікації засновані на наступних архітекторах та підходах:

- CNN;
- RNN;
- Transformer.

На рисунку 1.6 зображено деякі реалізації архітектур, які було показані вище.

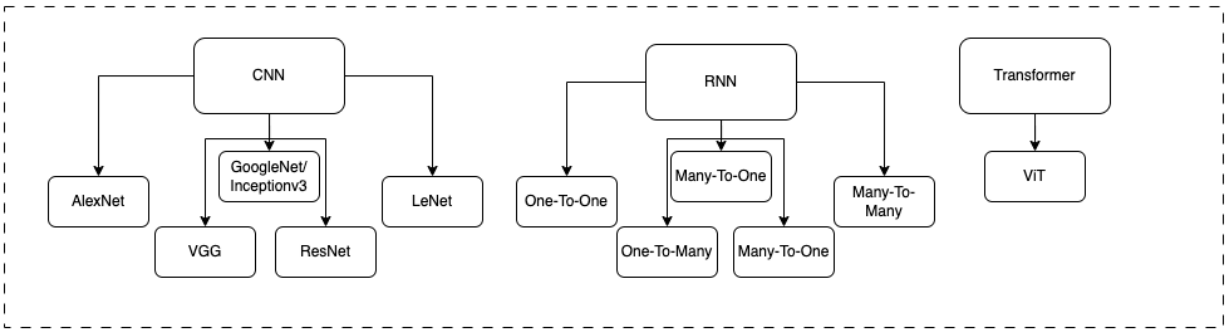

Рисунок 1.6 – Приклад реалізації архітектур методів класифікації зображень на основі методів машинного навчання

У таблиці 1.1 можна побачити порівняння описаних вище архітектур. CNN демонструють низку архітектурних переваг над RNN та ViT у задачах класифікації зображень. На відміну від RNN, які орієнтовані на послідовну обробку даних та не враховують просторову структуру зображень, CNN ефективно моделюють локальні просторові залежності завдяки використанню згорткових фільтрів. Крім того, CNN забезпечують повну паралелізацію обчислень, тоді як рекурентна природа RNN обмежує швидкодію та масштабованість. У порівнянні з ViT, CNN мають вбудовану

індуктивну біасу, зокрема локальність і трансляційну інваріантність, що дозволяє досягати високої узагальнюючої здатності навіть при обмежених об’ємах навчальних даних. Натомість ViT потребує великих датасетів для ефективного навчання, не реалізує ієрархічного вилучення ознак і характеризується високою обчислювальною складністю через квадратичну залежність механізму самоуваги від кількості патчів.

Таблиця 1.1 – Порівняльний аналіз методів класифікації зображень [3]

Критерій	CNN	RNN	ViT
1	2	3	4
Тип даних	Просторові	Послідовні	Просторові, подані як послідовність патчів
Врахування просторової структури	Природна через згортки, локальність і трансляційну інваріантність	Відсутня	Частково враховується через self-attention
Ієрархія ознак	Є; моделюється згортковими шарами зростаючої глибини	Відсутня; немає механізму формування ієрархічних представлень	Відсутня; неявна ієрархія, потребує додаткових модифікацій
Узагальнення на малих датасетах	Високе завдяки індуктивній біасі	Низьке; модель схильна до переобучення при нестачі даних	Низьке; потребує великих об’ємів даних через відсутність архітектурної індукції

Продовження таблиці 1.1

1	2	3	4
Обчислювальна ефективність	Висока; повна паралелізація обчислень	Низька; обробка послідовна, не масштабована, виникає затухання градієнтів	Низька при великій кількості патчів; квадратична складність self-attention
Архітектурна адаптованість	Спеціалізована для зорових задач	Неадаптована до зображень; архітектура створена для часових/послідовних даних	Потребує масштабування, об’ємів даних і обчислювальних ресурсів; відсутній вбудований inductive bias

Аналіз таблиці показує, що CNN забезпечують більш ефективне, стійке та обчислювально оптимізоване рішення для задач класифікації.

1.4.1 Детальний опис принципу та особливості роботи згорткових нейронних мереж

Основним елементом, що відрізняє CNN від інших типів нейронних мереж, є операція згортки – потужний інструмент, який дозволяє мережі ефективно виявляти корисні ознаки на зображеннях незалежно від їхнього положення.

Згортка в CNN – це процес застосування фільтра до зображення. Фільтр – це невелика матриця, яку ми перемножуємо з пікселями зображення. Він рухається по зображенню, розглядаючи невеликі ділянки зображення по черзі, що дозволяє аналізувати локальні ознаки. На рисунку 1.7 можна побачити приклад етапу згортки в CNN.

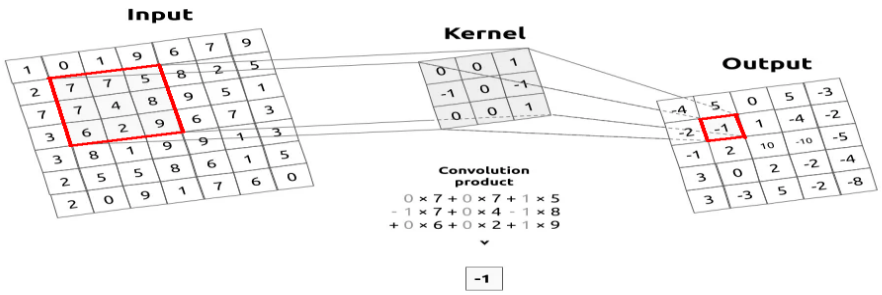

Рисунок 1.7 – Приклад операції згортки, де на вхідне зображення (Input) накладається фільтр (Kernel), що означає що їхнє значення перемножуються та складаються разом, утворюючи нові значення для наступного зображення (Output)

Формально процес згортки ми можемо визначити наступним чином:

$$I_{y,x} * h = \sum_{i=-n_2}^{n_2} \sum_{j=-m_2}^{m_2} I(y-i, x-j) * h(i,j), \qquad (1.1)$$

де n_2 – це половина висоти фільтра, m_2 – це половина довжини фільтра, x – позиція стовпчика певного пікселя на зображенні, y – позиція рядка певного пікселя на зображенні.

Так само з визначенням $n_2 = \left[\frac{n}{2}\right], m_2 = [\frac{m}{2}]$, де n – висота фільтра, а m – довжина фільтра h.

За приклад візьмемо зображення I, та фільтр h, які визначені нижче – ми можемо застосувати формулу згортки для них.

$$h = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}. \qquad (1.2)$$

Щоб застосовувати згортку, потрібно визначити ще кілька деталей щодо того, як фільтр буде пересуватись по вхідному зображенню. Перше значення, яке потрібно визначити, це крок того, на скільки пікселів будуть переміщатися значення x та y на зображенні. Цей параметр називають stride.

На рисунку 1.10 ліворуч можна побачити переміщення фільтра на один крок, тобто stride=1. На зображенні 1.8 stride=2, тому зелена ділянка для обробки знаходиться на два пікселі праворуч від червоної. Також треба зауважити, що чим більше значення stride, тим менший розмір зображення ми отримаємо.

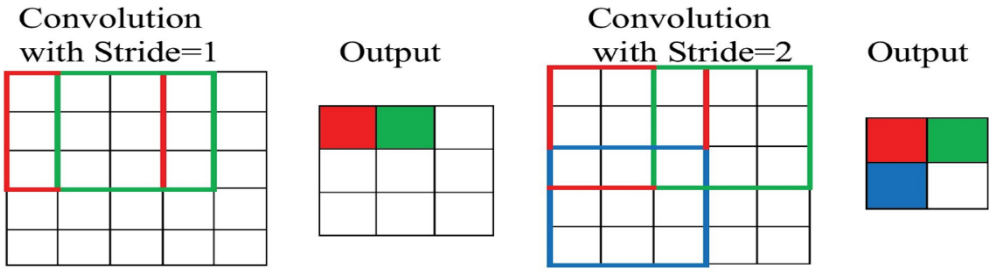

Рисунок 1.8 – Приклад використання параметру stride

Вхідне зображення має додані рядки та стовпчики заповнені нулями, тобто padding=1, та stride=1, тому зображенні 1.9 на виході фактично не відрізняється від зображення на вході.

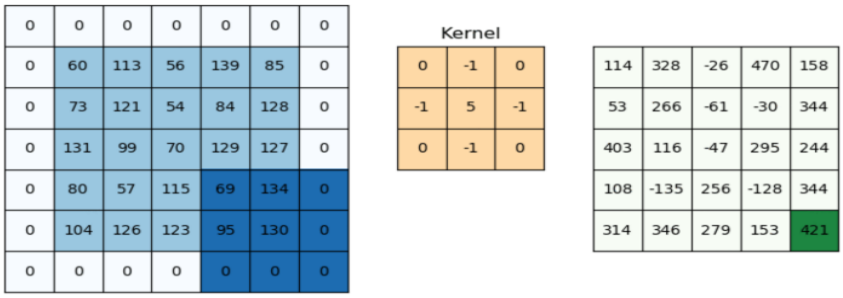

Рисунок 1.9 – Використання параметру stride у синергії з параметром padding

Як можна помітити з прикладів вище, розмір та значення в результаті можуть відрізнятись залежно від цих двох параметрів. Тому задля простоти ми будемо враховувати, що значення для параметру stride=1, а для padding=0. Згідно з цими значеннями, ми можемо порахувати розмір вихідного зображення для матриці *I*, яку ми визначили раніше. Оскільки її розмір 4×4, а фільтр, що будемо до неї застосовувати, розміром 3×3, то вихідна матриця буде мати розміри 2×2. Це значення можна вирахувати за формулою нижче:

$$n(\text{out}) = \frac{n(in)+2padding-n}{stride} + 1,$$
$$m(\text{out}) = \frac{m(in)+2padding-m}{stride} + 1, \quad (1.3)$$

де n_{in}, m_{in} – висота та довжина вхідного зображення, n_{out}, m_{out} – висота та довжина вихідного зображення.

Отже, маючи всі необхідні дані, можемо почати згортку, яка буде виконана для елементів I_{11}, I_{12}, I_{21}, I_{22} так, як це зображено на малюнку нижче (рисунок 1.10).

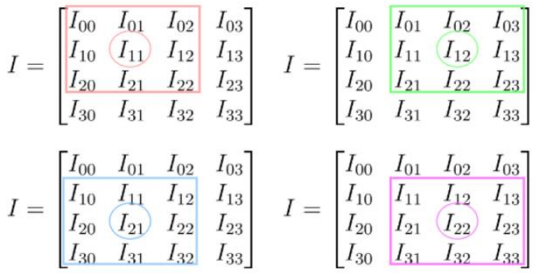

Рисунок 1.10 – Операція згортки

Таким чином, CNN відрізняються від інших типів нейронних мереж своєю архітектурною адаптованістю до просторової структури вхідних даних, що забезпечує їхню виняткову ефективність у задачах комп'ютерного зору. Їх здатність до автоматичного виділення багаторівневих ознак зображення суттєво підвищує точність аналізу та знижує залежність від ручного втручання.

1.5 Порівняльний аналіз класичних та нейромережевних методів класифікації зображень

Далі буде наведено таблицю порівняння різних методів класифікації зображень за їх точністю та ресурсами для навчання та використання (таблиця 1.2).

Таблиця 1.2 – Порівняння різних типів класифікації зображень [4]

Метод	Принцип дії	Переваги	Недоліки	Обчислювані ресурси
1	2	3	4	5
Watershed	Ґрунтується на метафорі "затоплення" рельєфу зображення, де кожен мінімум є джерелом затоплення, а границі сегментації виникають на стиках басейнів.	Добре працює для зображень з чітко вираженими контурами; не потребує навчання; точне в розділенні дотичних об’єктів.	Дуже чутливий до шуму та локальних мінімумів; необхідна попередня обробка (наприклад, маркери або фільтрація); низька гнучкість.	CPU–орієнтований, обчислювальна складність O(n), пам’ять <1 ГБ; дуже швидкий для зображень <1 Мп.
FloodFill	Рекурсивний або ітеративний алгоритм заливки пікселів, що мають однакові або подібні значення, починаючи з точки запуску.	Простий в реалізації; ефективний при чітких граничних значеннях пікселів.	Погано працює зі складними структурами або слабким контрастом; не здатен до адаптації або узагальнення.	Виконується на CPU з O(n) складністю; низьке споживання пам’яті (<500 МБ); реальний час для малих зображень.

Продовження таблиці 1.2

1	2	3	4	5
CNN	Використовує згортки (kernels) для автоматичного виділення просторових ознак. Складається з шарів згортки, підвибірки (pooling) та повнозв’язних шарів.	Висока точність; автоматичне навчання ознак; стійкість до трансляцій і масштабувань; добре масштабується на великі датасети.	Вимагає великої кількості навчальних даних; потребує високих обчислювальних ресурсів; "чорний ящик" – важко інтерпретувати.	Параметри: 5–100 млн (ResNet–50: ~25M); GPU (NVIDIA V100): ~1–3 год навчання на ImageNet; RAM >12 ГБ.
RNN	Архітектура з рекурентними зв’язками, що дозволяє враховувати послідовність вхідних даних (часовий контекст).	Добре обробляє послідовні дані; може використовуватись для відеоаналізу (кадри у часі).	Обмежене зберігання довгострокового контексту; складність у стабільному навчанні; неефективні для просторової обробки.	Параметри: 10–60 млн; потрібен GPU з великою пам’яттю (LSTM: до 8–16 ГБ), час навчання2–4× CNN).

Продовження таблиці 1.2

1	2	3	4	5
ViT	Використовують механізм уваги (self–attention) для обробки всіх частин вхідного зображення паралельно.	Добре моделює глобальні залежності; високий потенціал у візуальних задачах; state–of–the–art результати.	Дуже ресурсоємні; складні для навчання; потребують великих наборів даних.	ViT–B/16: ~86 млн параметрів; потребує TPU/GPU A100; RAM >32 ГБ; навчання на ImageNet ~4–8 год.

Як можна зазначити з таблиці 1.2 доказано що методи класифікації на основі CNN архітектури підходять більше, ніж класичні чи інші методи на основі машинного навчання, через можливість масштабуватися, має ієрархію ознак, має високу обчислювану ефективність.

1.6 Обмеження згорткових нейронних мереж для рішення задачі класифікації зображень

Традиційно виявлення як малих, так і великих об'єктів розглядається як широка проблема виявлення та класифікації об'єктів. Класичним способом є піраміда зображень [3], яка змінює розмір вхідних зображень до декількох масштабів і навчає спеціальний детектор для кожного масштабу. Для покращення точності.

Класичним прикладом для легкої класифікації зображення є зображення, котре можна описати наступники критеріями:

- велика роздільна здатність;

- мала кількість шумів;
- оптимальна яскравість, збільшена або зменшена контрастність відсутність розмиття;
- об’єкт для класифікації займає велику відсоток зображення.

Таке зображення назвемо якісними вхідними даними.

З іншого боку, можуть бути зображення, із наступними характеристиками:

- велика кількість шумів;
- низька роздільна здатність зображення;
- наявність розмиття, збільшеної/зменшеної контрастність;
- малий об’єкт на зображені для класифікації.

Такі зображення назвемо неякісними. Низька якість зображення може бути причиною низького відсотку точності класифікації обраними методами.

Для підвищення точності було запропоновано SNIP [4], який виконує вибіркове зворотне поширення на основі різних розмірів об'єктів у кожному детекторі. SNIPER [5] покращує ефективність SNIP, обробляючи лише контекстні області навколо кожного об'єкта, а не кожен піксель у піраміді зображень, таким чином зменшуючи час навчання. Використовуючи інший підхід до ефективності FPN [6] використовує різномасштабні ознаки, притаманні шарам згортки, використовуючи бічні зв'язки, і об'єднує ці ознаки за допомогою низхідної структури. Згодом було створено PANet [7] та BiFPN [8] для покращення FPN у потоці інформації про ознаки за рахунок використання коротших шляхів. Крім того, SAN [9] було впроваджено для відображення різномасштабних ознак на підпростір, інваріантний до масштабу, щоб зробити детектор більш стійким до зміни масштабу. Всі ці моделі одностайно використовують покрокову згортку та максимальне об’єднання.

Однією з перших спроб вирішити цю проблему з низькою роздільною власністю є LR-CNN для тонкої класифікації з мінливою роздільною здатністю [10], де пропонується модель CNN «від кінця до кінця» шляхом

додавання етапу надвисокої роздільної здатності перед класифікацією. Після цього «Fine-to-coarse knowledge transfer for low-res image classification» [11] пропонує перенести детальні знання, отримані з тренувальних зображень з високою роздільною здатністю, на тестові зображення з низькою роздільною здатністю. Однак цей підхід вимагає тренувальних зображень з високою роздільною здатністю, що відповідають конкретному застосуванню, наприклад, класам, які не завжди доступні. Така сама вимога щодо тренувальних зображень з високою роздільною здатністю також необхідна для декількох інших дослідженнях, таких як «Вивчення розпізнавання з дуже низькою роздільною здатністю за допомогою глибинних мереж» [11]. Нещодавно у «Покращення дрібнозернистої класифікації для зображень з низькою роздільною здатністю» було запропонована функцію втрат, яка включає роздільність на рівні атрибутів (де атрибут означає детальні, ієрархічні мітки класів), щоб модель могла вивчати дискримінаційні ознаки, специфічні для класу [11]. Однак детальні ієрархічні мітки класів важко отримати, що обмежує застосування цього методу.

2 МЕТОДИКА ВИКОРИСТАННЯ CNN ДЛЯ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ

2.1 Набори даних для використання

Прискорення CNN є актуальним завданням у всіх сферах їх застосування, включаючи обробку та класифікацію зображень, виявлення об’єктів, сегментацію та інші суміжні задачі комп’ютерного зору. Серед цих напрямів класифікація зображень вважається найбільш поширеним і репрезентативним завданням для CNN, оскільки саме вона найчастіше використовується в наукових дослідженнях для оцінювання ефективності запропонованих підходів. У більшості наукових публікацій, присвячених оптимізації або прискоренню CNN, демонстрація результатів здійснюється саме на класифікаційних задачах, що пояснюється їхньою універсальністю, наявністю широкого спектра відкритих еталонних наборів даних та порівняльною простотою інтерпретації результатів. До типових наборів даних, що застосовуються для таких демонстрацій, належать, зокрема:

- ImageNet це великий анотований набір даних, який широко використовується в дослідженнях комп’ютерного зору, зокрема для задач класифікації зображень, виявлення об'єктів та локалізації. Його було започатковано у 2009 році дослідницькою групою під керівництвом Фей-Фей Лі в Стенфордському університеті. ImageNet побудований на основі ієрархії WordNet, у якій кожна сутність представлена набором зображень. Кожен synset містить середню кількість від кількох сотень до кількох тисяч зображень, а загальна кількість зображень у повному наборі перевищує 14 мільйонів. Кожне зображення вручну анотоване для того, щоб відповідати певному класу в ієрархії, що забезпечує високу якість даних;

- CIFAR-10 та CIFAR-100 (набори даних CIFAR-10 та CIFAR-100 є анотованими підмножинами великого немаркованого набору, який містить близько 80 мільйонів зображень. CIFAR-10 включає десять класів: літак,

автомобіль, птах, кіт, олень, собака, жаба, кінь, корабель і вантажівка. Обидва набори складаються із кольорових зображень розміром 32×32 пікселі, що відрізняє їх від сіромасштабного MNIST і забезпечує більшу різноманітність та складність. Наприклад, клас «птах» може містити як птаха в польоті на фоні неба, так і крупним планом страуса. Об’єкти, створені людиною, такі як вантажівки чи кораблі, можуть мати різні форми, кольори й контексти, що підвищує варіативність даних. Ця візуальна різноманітність значно ускладнює задачу класифікації, особливо у випадку CIFAR-100, де налічується сто класів, що робить цей набір суттєво складнішим порівняно з MNIST і CIFAR-10);

- ILSVRC2012 (ILSVRC, що є підмножиною ImageNet, започаткований у 2010 році, а особливо відомий за версією 2012 року, став еталоном для оцінки ефективності алгоритмів розпізнавання об’єктів у великих масштабах. Цей набір даних є найскладнішим серед розглянутих, адже містить близько 1,2 мільйона зображень у тренувальному наборі та 50 000 у валідаційному, охоплюючи 1000 різних класів. Розміри зображень варіюються, однак зазвичай їх уніфікують до 256×256 пікселів для стандартизації обробки. ILSVRC є підмножиною ширшої бази ImageNet, яка налічує понад 14 мільйонів зображень, структурованих за ієрархією у 21 841 категорію і анотованих із застосуванням краудсорсингових технологій);

- MNIST (база даних рукописних цифр MNIST є, ймовірно, найвідомішим набором даних у галузі машинного навчання. Вона містить 70 000 зображень розміром 28×28 пікселів у відтінках сірого, розділених на 60 000 тренувальних і 10 000 тестових прикладів, що належать до одного з десяти класів. MNIST була використана у дослідженні зі згорткових нейронних мереж [12] і досі користується популярністю через свій компактний розмір і відносну простоту, що забезпечує швидке проведення експериментів та отримання результатів);

- та інші.

Наведені датасети забезпечують стандартизоване середовище для

експериментальної перевірки ефективності нових архітектур або методів оптимізації згорткових мереж.

Точність класифікації на ILSVRC2012 вважається ключовим індикатором ефективності моделей у реальних умовах, тому цей набір даних широко використовується як остаточний тест для оцінки загальної продуктивності згорткових нейронних мереж. Якщо MNIST і CIFAR найчастіше застосовуються на початкових етапах розробки та перевірки алгоритмів, то між ними та ILSVRC2012 існує суттєва різниця за масштабом та рівнем складності, що формує відповідні виклики при створенні високопродуктивних моделей глибокого навчання.

Недоліками описаних датасетів є неможливість використання їх в проблемній області Mil-Tech та класифікації дронів. Тому, вказані датасети не можуть бути використанні для донавчання нейромережевих моделей для класифікації зображень у вихідному вигляді, а мають бути розширені зображеннями з наборів даних DICD та MMAUD, котрі заточені на моделей дронів бренду DJI.

2.2 Основним компонентом CNN

Нейронні мережі представлені у вигляді послідовного набору перетворень, або шарів, кожен з яких виконує певні обчислювальні операції над вхідними даними. Основним компонентом CNN, що визначає їхню архітектуру, є згортковий шар. У CNN дані зазвичай подаються у вигляді багатовимірних масивів – двовимірних, тривимірних або чотиривимірних, які відображають різні рівні просторової та каналової інформації. Ці масиви також іменуються картами ознак або каналами. В загальному випадку N-вимірний масив називають тензором порядку N, хоча в практиці глибокого навчання терміни «масив» та «тензор» часто використовуються як синоніми.

Концепція згортки в CNN базується на принципах лінійної фільтрації зображень, відомих ще до появи сучасних нейронних мереж. Під час

виконання згортки фіксований лінійний фільтр, або ядро згортки, послідовно переміщується по вхідному двовимірному масиву, що відповідає карті ознак або каналу, і обчислює локальні зважені суми. Результатом цієї операції є новий двовимірний масив, який містить відфільтровані ознаки. Таким чином, згортка виступає специфічною формою лінійної трансформації вхідних даних, яка зберігає локальні просторові взаємозв'язки, що є фундаментальним для створення ефективних моделей розпізнавання образів:

$$V(x,y) = \sum_{i,j} W(x-i, y-j)U(i,j) = \sum_{i,j} W(i,j)U(x-i, y-j), \qquad (2.1)$$

або як взаємне кореляція:

$$V(x,y) = \sum_{i,j} W(i,j)U(x+i, y+j). \qquad (2.2)$$

Формулу (2.1) можна перетворити на (2.2) і навпаки, перевернувши фільтр W. У багатьох CNN у рамках навчання, а також у решті роботи використовується формула крос–кореляції, але за традицією відповідний шар ще називають згортковим. Межі підсумовування визначаються розміром фільтра W, який тут передбачається квадратом d × d. Індекси поза межами в U обробляються шляхом доповнення вхідних даних тензору, зазвичай з нулями. Ця підкладка зазвичай невелика і не заважає алгоритмів, розглянутих у цій кваліфікаційній роботі.

Тоді як (2.1) і (2.2) обробляють одноканальні (відтінки сірого) зображення, кольорові зображення представлені у вигляді стека з 3 каналів (або карт) або одного 3D–масиву. Так само дані, що проходять між згортковими шарами, також представлені тривимірним масивом із кількістю каналів, які традиційно більші за 3. Із запровадженням третього виміру на вхід U фільтр W також стає тривимірним масивом:

$$V(x,y) = \sum_{i=1}^{d}\sum_{j=1}^{d}\sum_{c=1}^{C} W(i,j,c)U(x+i,y+j,c)\,. \qquad (2.3)$$

Нарешті, застосування кількох 3D–фільтрів призводить до кількох вихідних карт, які є зібрані в єдиний вихідний масив 3D. Потім фільтри організовуються як єдиний 4D–масив:

$$V(x,y,k) = \sum_{i=1}^{d}\sum_{j=1}^{d}\sum_{c=1}^{C} W(i,j,c,k)U(x+i,y+j,c). \qquad (2.4)$$

Це перетворення тривимірних масивів (або тензорів третього порядку) називається узагальненою згорткою. Він входить до складу нейронних мереж як згортковий шар, так і масив W розглядається як параметр моделі. Альтернативні назви для W можуть бути: згортка ваги, згорткове ядро, ядерний тензор. У згорткових шарах кожен нейрон з'єднаний з невеликою підгрупою нейронів у попередній шар, і ця підмножина називається рецептивним полем. Число з плаваючою комою операції в згортці можна оцінити як $HWCNd^2$, де H, W і C є розміри вхідного масиву, N – кількість фільтрів, d – розмір фільтра. Інші блоки CNN включають:

- у повнозв'язному шарі кожен нейрон з'єднаний з усіма нейронами попереднього шару. Цей шар приймає вхідний вектор x з елементами C і множить його на матрицю вагових коефіцієнтів $W \epsilon R^{C\times N}$, що створює вихідний вектор y з N елементів у CN операції з плаваючою комою. Якщо вхід не є вектором, він змінюється та обробляється як вектор;
- нелінійність. Лінійні шари, такі як згортковий і повнозв'язаний шар, перемежовуються нелінійностями. Найпопулярнішою нелінійністю є функція випрямленої лінійної одиниці (ReLU) f(x) = max(0,x). Це обчислювально дешева операція застосовується поелементно, тому її вартість незначна порівняно з іншими компонентами CNN. Оскільки нелінійність майже завжди присутня після повнозв'язного або згорткового рівня, її часто опускають в описі архітектури;
- пакетна нормалізація. Шар пакетної нормалізації нормалізує вхідні дані для нульового середнього та одиниці стандартного відхилення.

Запровадження пакетної нормалізації може значно прискорити конвергенцію навчання та покращити фінальний результат. Нормалізація така ж повсюдна, як і нелінійність у сучасних архітектурах, її також можна опустити в схемах архітектури. Згорткові шари зазвичай мають найбільшу кількість операцій і споживають найбільше пам’яті та часу в CNN, як показано на рисунку 2.2. Кілька конкретних випадків і модифікацій загальної згортки, які використовуються для швидшого збереження збірки моделі, показані на рисунку 1.1 і детально описані нижче.

1×1 згортка. Складність згортки зменшується для менших просторових розмірів ядра, а найменший можливий розмір дорівнює 1. У цьому випадку згортка (2.1) зменшується до лінійної комбінації вхідних каналів:

$$V(x, y, k) = \sum_{c=1}^{C} W(c, k)U(x, y, c). \quad (2.5)$$

Окрім невеликої кількості операцій, цю операцію можна ефективно реалізувати множення матриці.

Групова згортка. Ще один спосіб зменшити вартість згортки – скоротити її з’єднання між вхідним і вихідним каналами. Ідея реалізована шляхом поділу вхідних і вихідних каналів на кілька груп G_k і розрізання всіх з’єднянь між різними групами:

$$V(x, y, k) = \sum_{i=1}^{d} \sum_{j=1}^{d} \sum_{c \in Gk} W(i, j, c, k)U(x + i, y + j, c). \quad (2.6)$$

Групова згортка спочатку була запропонована як спосіб побудови моделі, яка може бути розпаралелена між двома графічними процесорами, а потім ця концепція була відроджена як блок швидких CNN.

Поглиблена згортка. Якщо кількість груп збігається з кількістю вхідного та вихідного каналів, згортка називається згорткою по глибині. У цьому випадку кожен канал фільтрується незалежно одним фільтром:

$$V(x, y, k) = \sum_{i=1}^{d} \sum_{j=1}^{d} W(i, j, k) U(x + i, y + j, k). \quad (2.7)$$

Кількість каналів у вихідному масиві у цьому випадку збігається з числом каналів у вхідному масиві. Згортка по глибині потребує у кілька разів менше операцій з плаваючою точкою порівняно зі звичайною згорткою того ж розміру, однак реальна продуктивність значною мірою залежить від ефективності її реалізації. В практичних застосуваннях існуючі згорткові нейронні мережі не завжди відповідають вимогам щодо швидкодії. Одним із шляхів подолання цієї проблеми є використання більш потужного апаратного забезпечення або оптимізація реалізації операції згортки. Також можливе налаштування моделей із застосуванням швидких апроксимацій.

Перехід на сучасне апаратне забезпечення (що включає перехід від CPU до GPU) є найпростіший спосіб підвищення продуктивності, оскільки сучасні фреймворки глибокого навчання підтримують цей перехід без необхідності додаткового програмування. Збільшення швидкості обробки та обсягу пам’яті на сучасних GPU є ключовим фактором, що розширює можливості нейронних мереж. Хоча питання апаратної архітектури та реалізації операцій здебільшого виходять за межі цієї роботи, певні аспекти впровадження можуть суттєво впливати на ефективність обчислень.

Реалізація операції згортки (див. рис. 2.1) передбачає наявність кількох вкладених циклів, зазвичай шести. Малий розмір ядра спричиняє неефективність внутрішніх циклів, зумовлену частими переходами керування інструкціями JMP. Крім того, етапи прямого та зворотного поширення помилки вимагають доступу як по рядках, так і по стовпцях до вхідних даних і ядра, що є складною для ефективної реалізації операцією з урахуванням особливостей структури пам’яті.

Варіанти згортки, що використовуються в сучасних конструкціях CNN представлені на рисунку 2.1. Стандартна згортка з фільтрами 3×3. Згортка 1×1, яка переставляє вхідні карти та не фіксує співвідношення сусідніх пікселів. Вхідні та вихідні карти є поділено на дві групи (позначено синім і зеленим), між якими немає зв’язків їх. У згортці по глибині всі карти

обробляються незалежно.

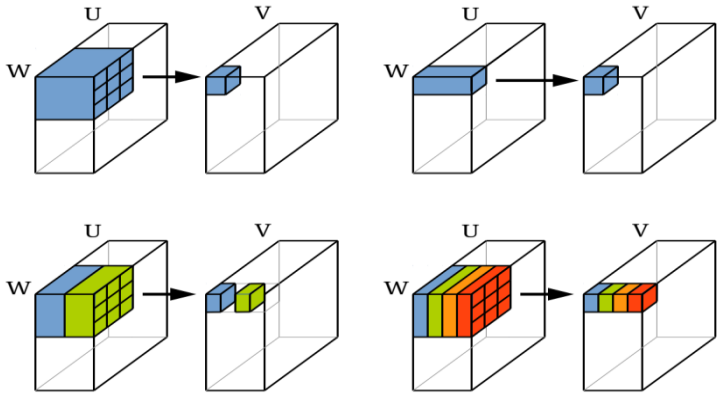

Рисунок 2.1 – Варіанти згортки у сучасних розробках CNN

Це так звана розгорнута згортка. Основною ідеєю є зведення згортки до множення двох матриць шляхом дублювання вхідних даних. Зниження дозволяє використовувати високооптимізовані реалізації множення матриці, варіанти BLAS бібліотеки, які розроблялися протягом багатьох років для різних обчислень архітектури, включаючи процесор і графічний процесор. Зменшення показано на рисунку 2.3.

Сучасна реалізація згортки на GPU настільки ефективна, що для відносно неглибокої архітектури, такої як AlexNet, найбільша частина робочого часу витрачається на повнозв'язані шари. У випадку з процесором також для більш глибоких архітектур основна частина часу споживається згортковими шарами. Наприклад, в ResNet–50, один повністю підключений рівень цієї архітектури займає менше ніж 0,5% від загального часу роботи як на CPU, так і на GPU.

Ще один спосіб створення швидшої реалізації для згортки базується на згортковій теоремі, яка стверджує, що кругові згортки в просторовій області еквівалентні поточковим добуткам в області Фур'є. Позначаючи перетворення Фур'є як F і зворотне перетворення як F−1, ми можемо виразити згортку двох 2D відображає f і g таким чином:

$$f * g = \mathcal{F}^{-1}\big(\mathcal{F}(f)\mathcal{F}(g)\big). \qquad (2.8)$$

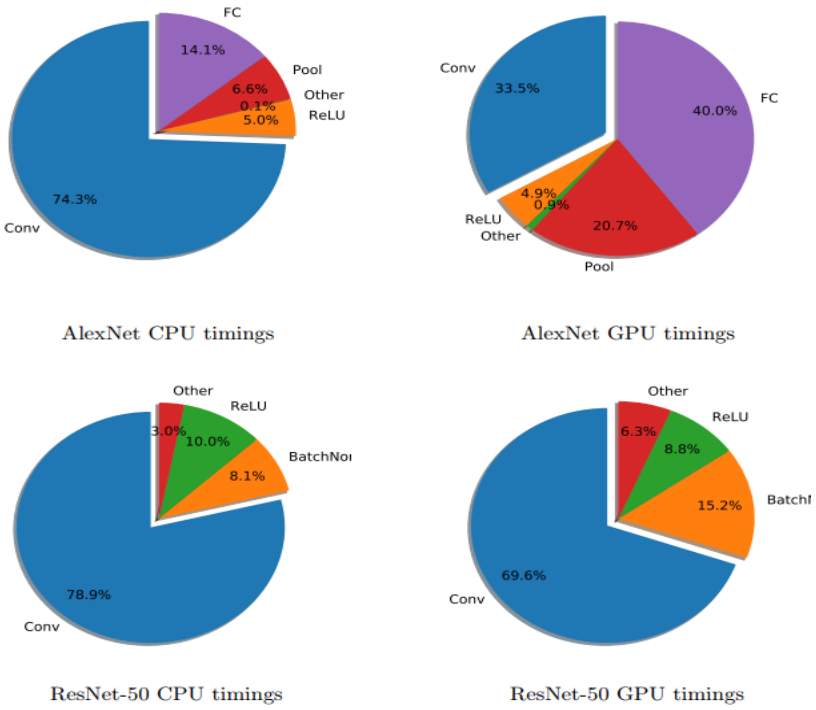

Рисунок 2.2 – Таймінг різних рівнів для архітектур AlexNet і ResNet–50 включені ЦП (Intel Core i9–9880H) і графічний процесор (GeForce GTX 1080 Ti), виміряні у рамках Pytorch за допомогою вбудованого профілювача

Введений масив U перетворюється на матрицю P за допомогою операції im2col. Стовпці P створено з розгаданих патчів з U, розмір патча визначається розміром фільтрів у ваговому масиві W. Матриця патчів P потім множиться на вагову матрицю W’ (отримано з W шляхом зміни форми), в результаті чого вихідна матриця V’. Остаточний вихідний масив V отримано з V’ іншим переформуванням.

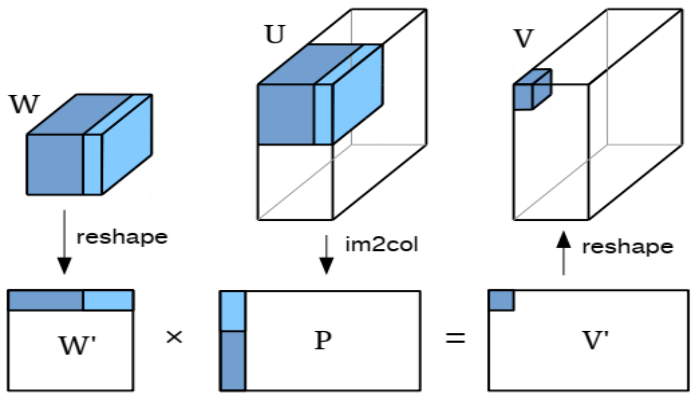

Рисунок 2.3 – Зведення згортки до множення матриці через розгортання

Синім виділено один патч у U з відповідним стовпцем P, відфільтроване в W з відповідним рядком W’ і один вихідний піксель, створений цим фільтром у цьому патчі. Представлено не тільки з фільтрів W, оскільки важко візуалізувати чотиривимірний масив.

2.3 Архітектури CNN

Основними структурними елементами CNN є згорткові шари, шари pooling, функції активації та повнозв’язні шари.

Згорткові шари виконують операцію згортки між вхідним зображенням і ядрами фіксованого розміру (наприклад, 3×3 або 5×5), що дозволяє локально виявляти просторові закономірності, такі як границі, кути та текстури. У процесі навчання ці ядра набувають значень, які відповідають найбільш інформативним шаблонам. Субдискретизація, зокрема max-pooling, зменшує просторову роздільність тензора ознак, зберігаючи водночас найсильніші відповіді нейронів і підвищуючи інваріантність до невеликих зсувів чи спотворень у зображенні. Активаційні функції, найчастіше ReLU, вводять нелінійність у модель, дозволяючи мережі апроксимувати складні залежності між вхідними даними та виходом. На завершальних етапах повнозв’язні шари агрегують вилучені ознаки для класифікації або регресії (рисунок 2.4).

LeNet – це проста архітектура CNN, спочатку запропонована і досі часто використовується для набору даних MNIST. Вона складається з двох згорткових, двох об’єднаних і двох повністю з’єднаних шарів. Оригінальна архітектура включала нелінійний тангенс та блоки RBF, які зазвичай замінюються ReLU і звичайним повнозв’язним шаром у сучасній реалізації цієї архітектури.

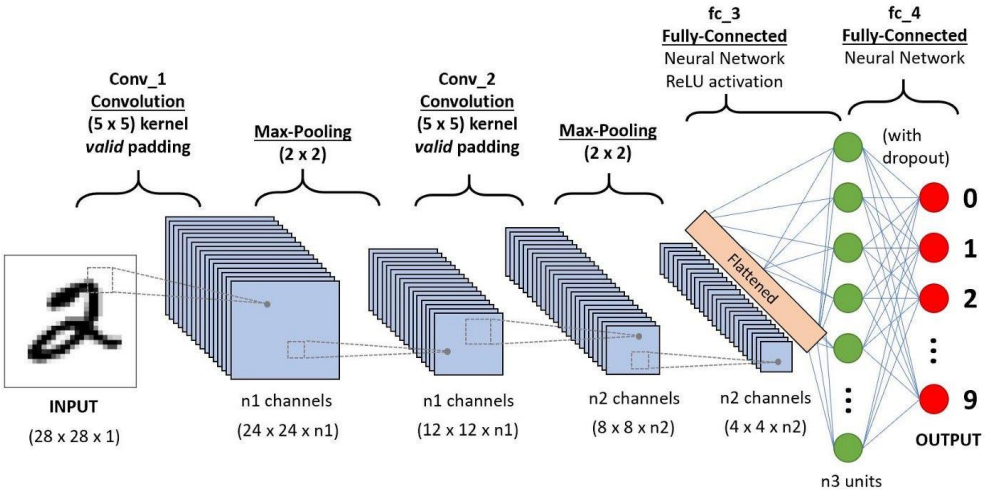

Рисунок 2.4 – Архітектура CNN

AlexNet була першою згортковою нейронною мережею, успішно натренованою на великому за обсягом наборі даних, що стало проривом і забезпечило перемогу на змаганні ILSVRC2012. Модель AlexNet вирізнялася певними особливостями: використанням фільтрів різних розмірів, зокрема великих фільтрів на початкових шарах, а також порівняно невеликою глибиною мережі. Хоча сучасні CNN перевершують AlexNet за всіма параметрами, вона досі широко застосовується як базова еталонна модель. AlexNet часто порівнюють із найглибшими і найточнішими архітектурами сучасних CNN, однак вона не конкурує за швидкістю з найпродуктивнішими моделями при однаковому рівні точності.

Ранні архітектури CNN використовували великі згорткові фільтри, наприклад, 5×5 в AlexNet і LeNet, причому сама AlexNet застосовувала фільтри розміром 11×11 на першому шарі. Великий розмір таких фільтрів забезпечує плавність навчання, що свідчить про значно менші потреби у визначенні параметрів фільтра через інтерполяцію чи інші методи апроксимації. Подібність деяких фільтрів до вертикальних або горизонтальних операторів виявлення країв свідчить про використання специфічних методів роздільних фільтрів. Дослідження з питань роздільності та розширення до тензорних розкладів розглядаються у наступному розділі.

VGG – це сімейство CNN, що визначається двома ключовими характеристиками: значною глибиною та ексклюзивністю використання фільтрів 3×3, що є найменшим розміром для захоплення поняття ліворуч/праворуч, вгору/вниз, центр.

Успіх архітектур VGG поклав початок збільшення глибини сучасних архітектур CNN: чим більше шарів можна скласти, тим краще. Архітектури VGG дуже повільні та навіть важчі за кількістю параметрів, переважно через масивні повністю пов’язані шари. Мережі VGG все ще популярні серед дослідників, оскільки мають просту структуру, і навіть домінують у деяких застосуваннях, таких як дрібнозерниста класифікація та стилізація зображення.

Слід зазначити, що деяка межа глибини CNN все ще існує: коли мережа стає глибшою, її точність насичується і після досягнення деякої межі запуск швидко деградує. Ця проблема спричинена не переобладнанням, а несправністю тренувального процесу. Було запропоновано новий підхід до проекту, щоб полегшити градієнт розповсюдження через мережу і, отже, сприяюнню процесу навчання. Головна ідея полягає в тому, щоб організувати блоки CNN як залишкову функцію де x – вхід, а h(x) – блок згорткових шарів.

$$f(x) = h(x) + x. \tag{2.9}$$

Залишкове навчання долає проблему зниження точності та дозволяє ефективно навчати CNN до тисячі шарів, хоча ці надзвичайні розміри можливі лише на наборах даних із невеликими розміри введення, як CIFAR. Члени сімейства моделей ResNet позначаються кількістю шарів у мережі, серед яких ResNet середнього розміру є найпоширенішою моделлю. Спроби покращити оригінальну архітектуру ResNet включають ResNeXt, яка вводить групові згортки до залишкового блоку, і DenseNet, що створює додаткові зв’язки між залишковими блоками.

Сімейство Inception також відоме як GoogleNet. Основна ідея полягає в початковій архітектурі, яка заснована на з'ясуванні того, як оптимальна локальна розріджена структура у мережі згорткового бачення може бути наближена та охоплена легкодоступними щільними компонентами. Цей принцип призвів до побудови блоку Inception, який складається з кількох гілок, кожна з яких має фільтри певного розміру. Цей блок повторюється кілька разів разів, що призводить до дуже глибокої нейронної мережі.

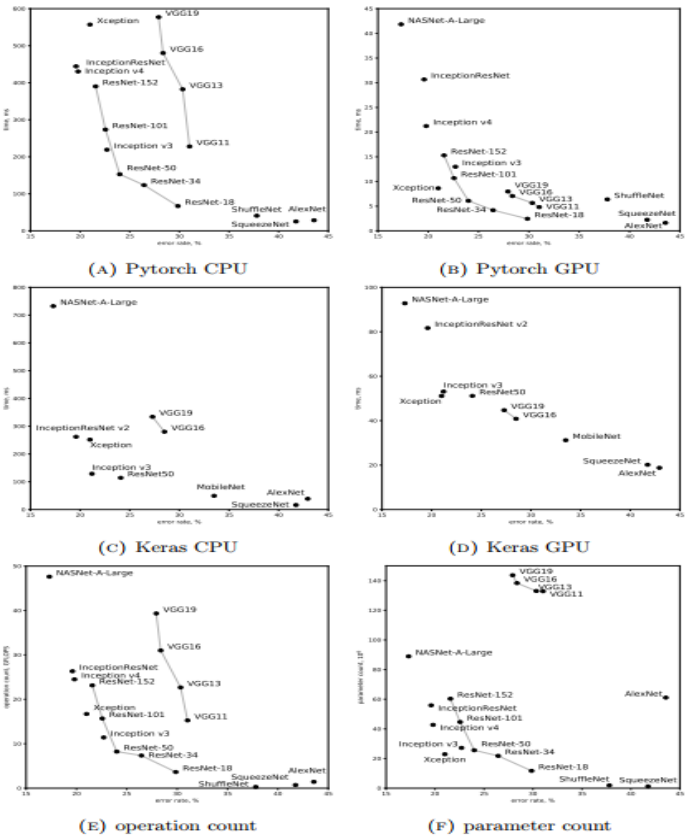

Рисунок 2.5 – Компроміс між часом висновку та помилкою класифікації ILSVRC Top–1 для деяких архітектур CNN

У поточних час початкова сім'я включає декілька версії моделі Inception, модель Xception, яка використовує згортки, що розділяються по глибині, і гібридну модель під назвою Inception-ResNet.

Точність має пріоритет над швидкістю та компактністю в основних дослідженнях CNN, які певний час рухаються в напрямку збільшення глибини CNN. Тим не менш, навчання найглибшої з сучасних CNN

неможливе без уваги до ефективності спроектованої архітектури.

Таблиця 2.1 – Порівняння моделей CNN мереж [13]

Назва Мережі	Відсоток помилки	Кількість натренерованих зображень	Назва набіру даних
LeNet-5	1.15%	60 тисяц	MNIST
AlexNet-8	10.34%	15 мільонів	ImageNet
GoogleNet-22	7.71%	138 мільонів	ImageNet, Place365
VGGNet-19	6.70%	1.2 мільонів	ImageNet
ResNet	5.29%	37 тисяч	ImageNet

Отже, наведена таблиця узагальнює ключові характеристики, переваги та обмеження основних підходів до обробки зображень із використанням методів машинного навчання. Аналіз представлених даних свідчить про поступову еволюцію від ручного проєктування ознак до повністю автоматизованих глибоких архітектур, таких як CNN та їх похідні. Ці результати підкреслюють значущість вибору моделі відповідно до специфіки задачі, доступності мічених даних та обчислювальних ресурсів, а також демонструють потенціал інтеграції класичних та сучасних методів для досягнення оптимальної продуктивності в аналізі зображень.

У цьому контексті окрему увагу слід приділити особливостям архітектур CNN, зокрема їхній типово застосовуваній структурі, що включає згортки з кроком та операції підвибірки. Незважаючи на те, що така побудова зазвичай є ефективною для зображень високої якості, її використання на ранніх шарах мережі може призводити до суттєвих втрат дрібних деталей. Втім, у складніших сценаріях, де присутнє розмиття або об’єкти мають малий розмір, така припущена надмірність даних вже не забезпечується. У результаті архітектура страждає від втрати дрібнозернистої інформації та формує менш інформативні ознаки, що негативно позначається на загальній ефективності моделі.

3 АНАЛІЗ ТА ПРОЕКТУВАНЯ МЕТОДУ КЛАСИФІКАЦІЇ НА ОСНОВІ CNN АРХІТЕКТУРИ

Серед існуючих методів класифікації зображень, що наведені в розділі 2, найкращі аналітичні показники має CNN. В роботі запропоновано метод класифікації зображень на основі моделі ResNet, яка основана на архітектурі CNN. Запропоновано модицікація моделі ResNet для підвищення точності класифікації зображень з низькою роздільною здатрністю.

Як сказано вище, основою запропонованого методу буде нейромережевий класифікатор на основі CNN архітектури на базі ResNet, а саме ResNet-18 та ResNet-50. Ці дві реалізації було обрані, тому що ResNet використовує опорні зв'язки, що дозволяє навчати мережу глибиною до 152 шарів. Це також значно зменшує загальну кількість параметрів, використовуючи лише один повністю з'єднаний шар. Наприкінці використовується шар softmax для нормалізації прогнозів класів.

Додаючи до вище сказано цей класифікатор виграв турнир ILSVRC 2015. ResNet-18 і ResNet-50 використовують загалом чотири конволюції з кроком 2 і один шар максимального об'єднання з кроком 2 для зменшення розміру кожного вхідного зображення в 25 разів (рисунок 3.1).

За результатами дослідження, представленого в статті "Impact of Low Resolution on Image Recognition with Deep Neural Networks" [14], архітектура ResNet демонструє значне погіршення якості класифікації при низькій роздільності зображень. Причиною є втрата локальних та структурних ознак на ранніх етапах обробки, згортки та субдискретизація над малими вхідними тензорами призводять до втрати критичної інформації. У такому випадку мережа не може сформувати достатньо інформативні ознаки для подальших рівнів, що порушує ієрархічну структуру вилучення ознак і знижує ефективність класифікації.

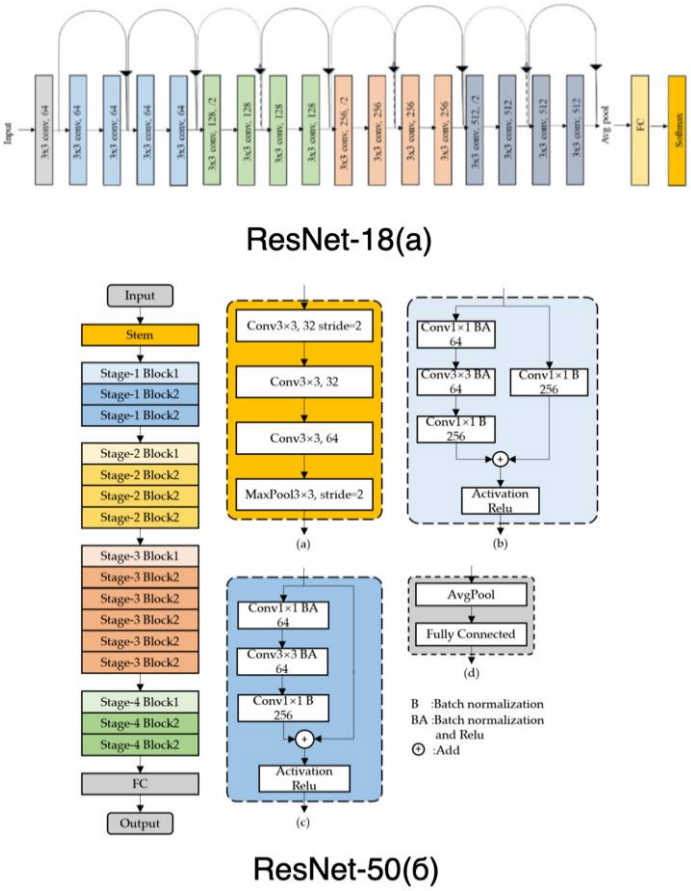

Рисунок 3.1 – Архітектури: а) ResNet-18, б) та ResNet-50

З метою підвищення точності класифікації класифікації зображень з низькою роздільною здатністю в роботі представлено метод підвищеної точності на основі поліпшеною архітектурою ResNet методу, де шари non-strided згортки та пулінгу замінюється новим блоком PUNSH-Conv та додаткових препроцесорів для нормалізації вхідних зображень (рисунок 3.2).

На рисунку 3.2 наведено IDEF0 нотації методу.

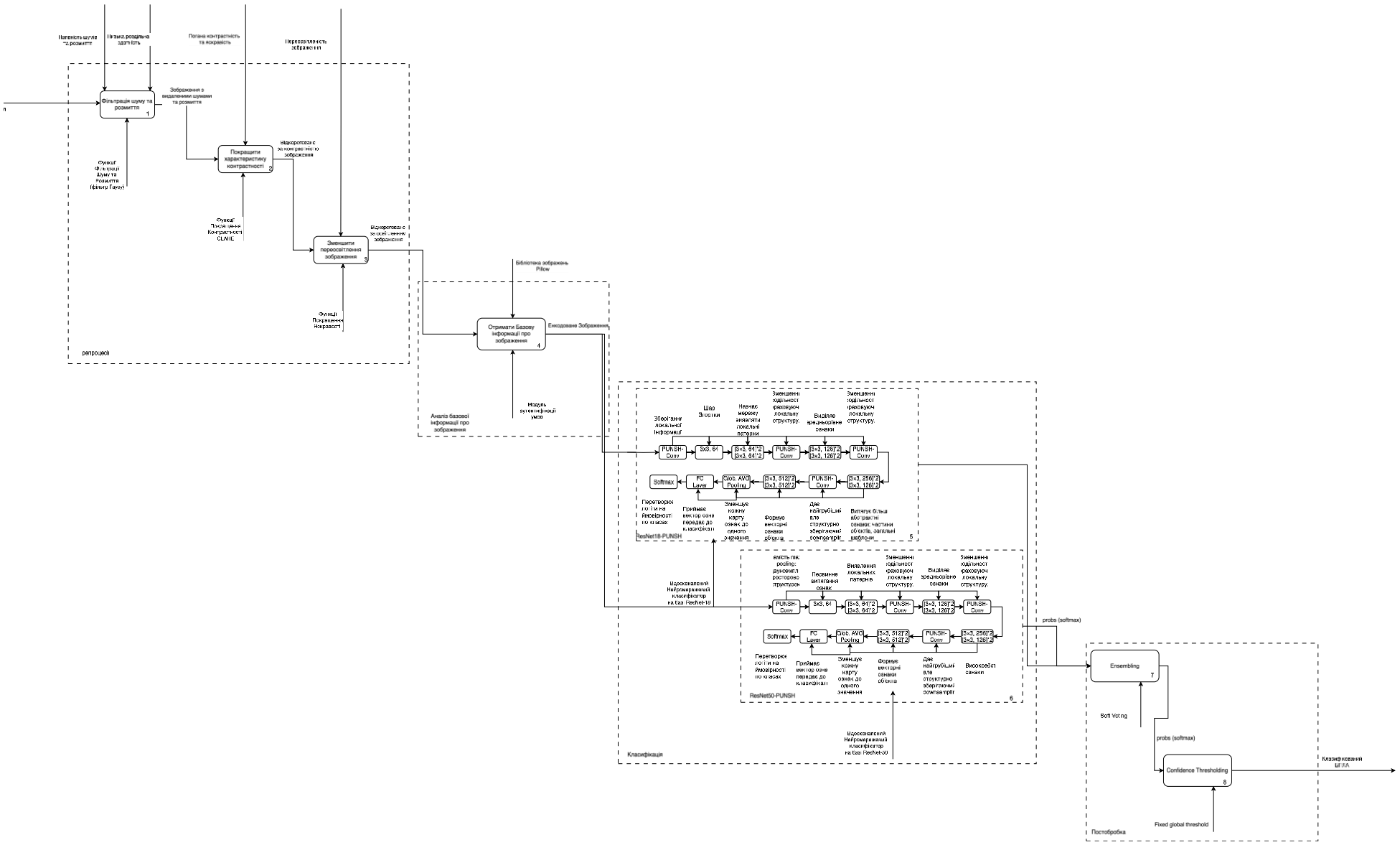

Рисунок 3.2 – IDEF0 нотація методу з використанням нормалізаторів та поліпшеної архітектури ResNet

Як показано на рисунку 3.2 є чотири основних компоненти методу:

- нормалізації (предобробка зображення; його нормалізація);
- аутентифікації вхідних умов(отримання базової інформації про зображення та вибору оптимального класифікатора);
- нейромережевий класифікатори на базі поліпшеної моделі ResNet-18 та ResNet-50 з покращеної архітектурою для роботи з неякісними зображеннями;
- постобробка результатів класифікації(валідація вихідних даних класифікатора або класифікаторів).

3.2 Блок предобробки зображення

Першим блоком методу, на який поступають вхідні дані є блок предпроцесінгу, його IDEF0 нотацію можна побачити на рисунку 3.3, де описано покрокові дії цього модулю.

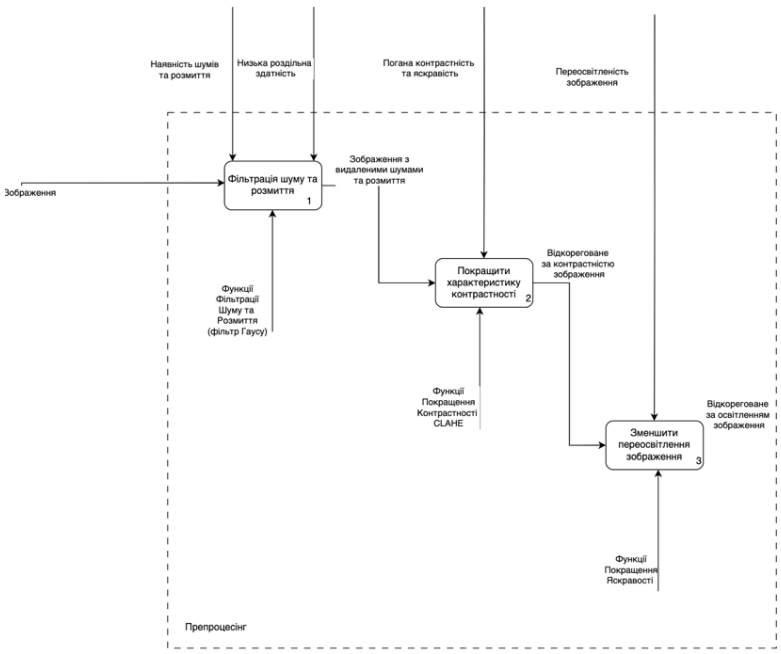

Рисунок 3.3 – Модуль предобробки зображення за допомогою нормалізації зображення

Ціль цього модулю, перед переходом до наступного, є наступні результати:

- зменшити кількість шумів та розмиття;
- покращення контрастності;
- зменшення зон пере освітлення.

Як було сказано вище, процес нормалізації є поступовим, де кожен крок поліпшує якість зображення за допомогою функції та методів.

Першим кроком є фільтрація шумів та розмиття за допомогою Гаусовського фільтру. Цей фільтр був обраний бо він краще всього працює з невідомими вхідними зображеннями, тобто не знаючими конфіденти розмиття та шумів. Також цей фільтр краще працює в умовах як багатоядерних процесорів, так я з малою кількістю порівнюючи з фільтром на основі функції Вінера [15]. Хоча основною перевагою, в нашому випадку, є те, що він нормалізує усі вхідні зображення(мала/велика кількість шумів та/або розмиття). У формулі 3.1 описано функцію Гаусу а реалізацію можна побачити у додатку А.

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}}. \quad (3.1)$$

Код зазначений у додатку А автоматично покращує зображення, застосовуючи Гаусове розмиття та підсилення контурів для зменшення шумів. Додатково може підсилювати синій канал, якщо він недостатньо виражений порівняно з червоним і зеленим. Параметри фільтрації визначаються на основі різкості зображення, оціненої через дисперсію Лапласіана. Весь процес виконується з перевірками на валідність і тип зображення.

На рисунку 3.3 можна побачити різницю між нормалізованим зображенням БПЛА та без неї.

Після використання попередньої функції далі йде крок покращення

контрастності за допомогою алгоритму CLARE що дозволяє покращити локальний контраст та виразність структурних ознак без необхідності додаткового перенавчання або зміни параметрів моделі.

а) б)

Рисунок 3.3 – Робота фільтру Гауса: а) вхідне зображення; б) результат роботи фільтру Гауса

На відміну від HE, який часто призводить до втрати деталей у світлих або темних ділянках.

Таким чином, CLAHE виступає як зовнішній підсилювач якості вхідних даних, що підвищує стійкість моделі до варіативності освітлення та знижує ймовірність помилкової класифікації в складних умовах. Метод CLAHE реалізован за допомогою бібліотеки OpenCV.

Лістинг 3.1 – Реалізація методу CLAHE для покращення контрасності (файл clahe.py)

```
import cv2
import numpy as np

def apply_clahe(image: np.ndarray, clip_limit=3.0,
tile_grid_size=(8, 8)) -> np.ndarray:
    lab = cv2.cvtColor(image, cv2.COLOR_BGR2LAB)
    l, a, b = cv2.split(lab)

    clahe = cv2.createCLAHE(clipLimit=clip_limit,
tileGridSize=tile_grid_size)
    cl = clahe.apply(l)

    limg = cv2.merge((cl, a, b))
    enhanced_image = cv2.cvtColor(limg, cv2.COLOR_LAB2BGR)
    return enhanced_image
```

На рисунку 3.4 показано результат роботи методу CLAHE, де було збільшено контрастність зображення.

Рисунок 3.4 – Результат роботи метода CLAHE

Останній блок фільтрації є поліпшення яскравості. Для поліпшення яскравості використовується метод IAGCWD [16].

Цей метод заснований на зміні гамма-кривої відповідно до статистичних характеристик яскравості зображення. Основною ідеєю є побудова адаптивної функції корекції, яка враховує розподіл яскравості та підсилює деталі в затемнених або засвічених ділянках без втрати загальної контрастності. Для цього використовується модифікована кумулятивна гістограма.

Як можна побачити на рисунку 3.5 не було виконано жодних змін, тому що зображення і так має велику контрастність. Код методу можна побачити у додатку В.

Рисунок 3.5 – Результат роботи метода IAGCWD

3.3 Блок аналізу базової інформації про зображення для виклику оптимального класифікатору

Після блоку фільтрації є модуль аналізу базової інформації про зображення (рисунок 3.6). Цей модуль використовується для виклику моделі ResNet18-PUNSH та/або ResNet50-PUNSH в залежності від параметрів зображення.

Рисунок 3.6 – Модуль отримання та аналізу вхідного зображення на основі базової інформації

Алгоритм аналізу базової інформації описан далі:

- приведення зображення до уніфікованого формату;
- оцінка різкості зображення;
- оцінка рівня шуму;
- оцінка експозиції;
- оцінка контрастності;
- формування вектору ознак якості зображення;
- класифікація рівня якості або типу зображення;
- вибір відповідної моделі класифікації.

У додатку Г є частина код який отримує та аналізує базову інформації про зображення.

3.4 Блок класифікації на основі методу на базі CNN

Наступним блоком методу є блок класифікації, цей модуль і буде класифікувати зображення на основі згорткової нейромережі (рисунок 3.7).

Блоком класифікації є дві реалізації нейромережевого класифікатора на основі ResNet50 та Resnet-18. Як було зазначено вище архітектура CNN і в вчасності ResNet моделі мають досить низьку точність класифікації зображень з малою роздільною здатністю, наявністю шумів та розмиття, збільшеною або зменшеною контрастністю.

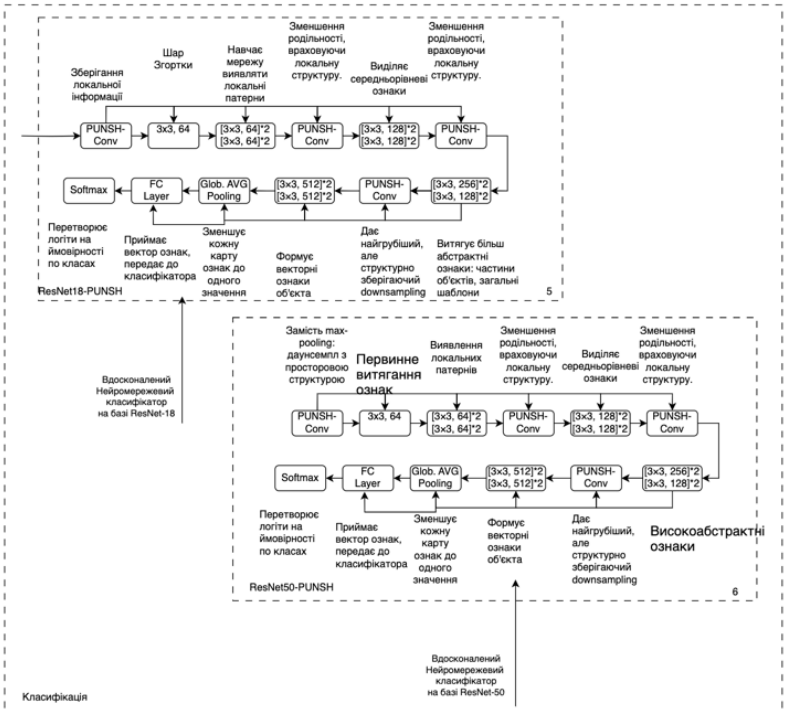

Рисунок 3.7 – Модуль класифікації на базі згорткової архітектури

Для вирішення проблеми класифікації БПЛА на поганій якості зображеннях було створено новий класифікатор на основі моделей ResNet, де було видалено шари stride-пулінгу та stride-сгортки.

3.4.1 Новий блок методу: PUNSH-Conv

Компонент PUNSH узагальнює техніку перетворення сирого зображення для зниження роздільної здатності карт ознак всередині та по всій CNN, як показано нижче. Розглянемо будь-яку проміжну карту ознак X розміром S × S × C1, виріжмо послідовність підкарт ознак як 3.2:

$f_{0,0} = X[O : S : scale, 0{:}S{:}scale], f_{0,0}$= X[1 : S: scale, 0 : S : scale],…,

$$f_{scale-1,0} = X[scale - 1 : S : scale, 0 : S{:}scale];$$

$f_{0,1} = X[O : S : scale, 1 : S{:}scale], f_{1,1}$,….,

$$f_{scale-1,1} = X[scale - 1 : S : scale, 10 : S{:}scale];$$

$f_{0,scale-1} = X[O : S : scale, scale - 1{:}S{:}scale], f_{1,scale-1}$,…,

$$f_{scale-1,scale-1} = X[scale - 1 : S : scale, scale - 1 : S{:}scale]; \quad (3.2)$$

Загалом, для будь-якої оригінальної карти ознак X підкарта fx,y формується всіма вхідними даними X(i, j), для яких i + x та j + y діляться на масштаб. Отже, кожна підкарта зменшує X у масштабі. На рис. 3.1 наведено приклад, коли масштаб = 2, де ми отримуємо чотири підкарти f0,0, f1,0, f0,1, f1,1, кожна з яких має форму 3.3 і зменшує X у 2 рази:

$$\left(\frac{S}{2}, \frac{S}{2} C_1\right). \quad (3.3)$$

Далі ми об’єднуємо ці підкарти ознак уздовж виміру каналу і таким чином отримуємо карту ознак X’, яка має зменшений просторовий вимір у масштабі і збільшений вимір каналу у масштабі в квадраті. Іншими словами, PUNSH перетворює карту ознак X(S, S, C1) у проміжну карту ознак 3.4:

$$X'\left(\frac{S}{scale}, \frac{S}{scale}, scale^2 C_1\right). \quad (3.4)$$

На рис. 3.8 наведено ілюстрацію з використанням масштабу = 2.

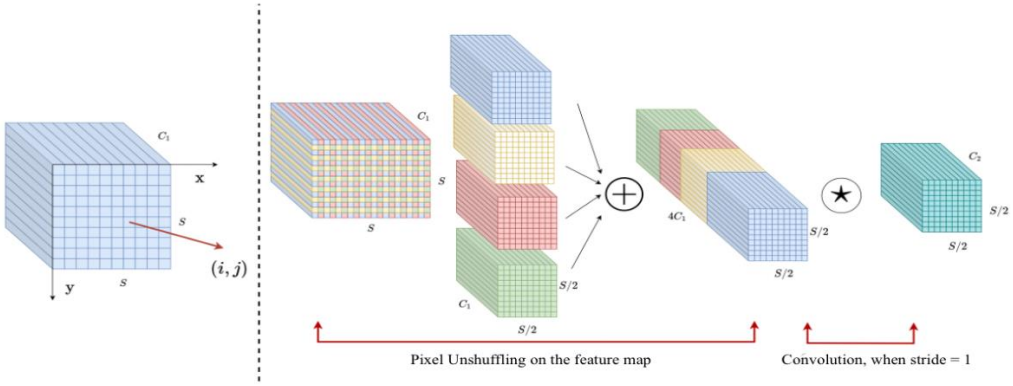

Рисунок 3.8 – Ілюстрація PUNSH-Conv коли параметр scale дорівнює 2

3.4.2 Non-strided згортка

Після шару трансформації ознак PUNSH додаємо некрокуючий (тобто stride=1) шар згортки з фільтрами C2, де C2 < scale^2 C1, і далі трансформуємо, як показано у (3.5).

$$X'\left(\frac{S}{scale},\frac{S}{scale},scale^2C_1\right)\rightarrow X''\left(\frac{S}{scale},\frac{S}{scale},scale^2C_2\right). \qquad (3.5)$$

Причиною використання некрокуючої конволюції є збереження якомога більшої кількості інформації про дискримінаційні ознаки. В іншому випадку, наприклад, при використанні фільтра 3 × 3 з кроком = 3, карти ознак будуть «зменшені», але кожен піксель буде відібраний тільки один раз, якщо крок = 2, відбудеться асиметричний відбір зразків відбуватиметься, де парні та непарні рядки/стовпці будуть відбиратися різну кількість разів. Загалом, крок з розміром кроку, більшим за 1, спричинить недискримінаційну втрату інформації, хоча на поверхні це виглядає як перетворення карти ознак 3.6 також, але без X':

$$X(S,S,C_1)\rightarrow X''\left(\frac{S}{scale},\frac{S}{scale},C_2\right). \qquad (3.6)$$

3.5 Блок постобробки результатів класифікації

Після методу класифікації йде модуль пост обробки на основі алгоритмів Ensembling та Confidence Thresholding (рисунок 3.9). Хоча необхідно зазначити, що модуль аналізу інформації про базове зображення може викликати тількі одну модель і в цього випадку тільки компонент Confidence Thresholding буде використано.

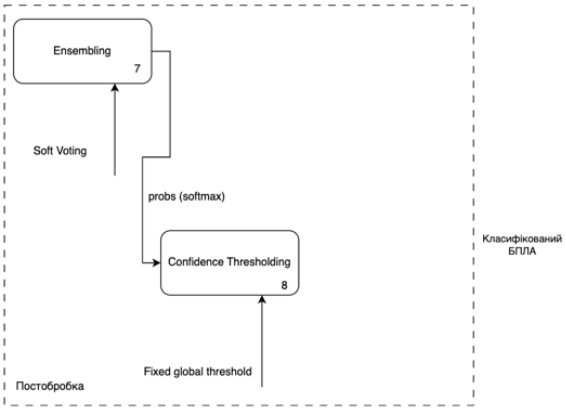

Рисунок 3.9 – Архітектура блоку постобробки

Ensembling реалізується через soft voting, коли кілька моделей прогнозують ймовірнісні розподіли класів, після чого обчислюється середнє арифметичне цих векторів для кожного зразка. Клас із найвищою середньою ймовірністю обирається як остаточний прогноз.

Confidence thresholding полягає у фільтрації результатів за порогом довіри: якщо максимальне значення вектору softmax нижче заданого порогу, прогноз вважається ненадійним і може бути відкинутий або позначений як невизначений. Це дозволяє зменшити кількість хибно-позитивних класифікацій у випадках низької впевненості моделі.

4 АНАЛІЗ ОТРИМАНИХ РЕЗУЛЬТАТІВ

Методика проведення дослідження включає порівняльний аналіз наступних методів класифікації зображень:

- методи на базі RNN архітектури;
- методи на базі CNN архітектури;
- ViT;
- KNN;
- запропонований в роботі метод.

Для навчання методів було використана типи аугментації зображення для підвищити стійкість моделі до типових спотворень, які виникають у реальних умовах – зокрема, шумів, розмиття, нестабільного освітлення та часткової оклюзії (рисунок 4.1). Далі наведено список використаних методів:

- гаусове розмиття є імітацією дефокусування або рухового розмиття камери;
- додавання гаусового шуму для симуляція низької якості сенсора або перешкод у передачі зображення;
- випадкове змінення яскравості та контрасту, яке дозволяє моделі краще працювати при різному освітленні;
- випадкове перекриття частини зображення, яке допомагає моделі фокусуватись на релевантних ділянках об'єкта;
- горизонтальне відображення, яке враховує симетричність деяких моделей дронів при різних орієнтаціях у повітрі;
- Random affine transforms з невеликими зсувами та масштабами дозволяє змоделювати варіативність у позиції та відстані до дрона.

Рисунок 4.1 – Використані методи аугментації

В роботі досліджуються наступні залежності та впливи:

- вплив роздільної здатності, наявності шумів та розмиття на точність класифікації (Accuracy) – експеримент 1;

- використання методів предобробки для покращення точність – експеримент 2;

- використання модулю визначення оптимальної мережі на основі базової інформації про зображення для зменшення використання CPU ресурсів – експеримент 3.

4.1 Оцінка впливу роздільної здатності, наявності шумів та розмиття на точність класифікації зображень (Accuracy) на основі запропонованого та традиційних методів

Для оцінки точності класифікації зображень ми використовуємо набори даних DICD [17] та MMAUD [18].

DICD є відкритим набором. Кожен клас має 4001 навчальних зображень, 4000 зображень для перевірки. Кожне зображення має різну роздільну здатність. MMAUD Dataset складається з 14715 зображень з різну роздільною, включаючи 7512 навчальних зображень і 5253 тестових

зображень. Ми використовуємо точність top-1 як метрику для оцінки ефективності класифікації.

Використовується випадковий пошук по сітці, щоб налаштувати гіперпараметри, включаючи швидкість навчання, розмір партії, імпульс, оптимізатор і згасання ваги. На рис. 3.3 показано зразок графіка прочісування гіперпараметрів, створеного за допомогою wandb MLOPs. Результатом є оптимізатор SGD зі швидкістю навчання 0,01793 і імпульсом 0,9447, розміром міні-пакета 256, регуляризацією спаду ваги 0,002113 і 200 епохами навчання. Далі ми навчаємо нашу модель ResNet50-PUNSH на DICD. Гіперпараметри взяті з статті про ResNet50 [19], де використовується оптимізатор SGD з початковою швидкістю навчання 0,1 і імпульсом 0,9, розміром партії 128, регуляризацією згасання ваги 0,0001 і 200 епохами навчання. Як для ReseNet18-PUNSH, так і для ReseNet50-PUNSH ми використовуємо ту саму функцію затухання, що і в ResNet, щоб зменшувати швидкість навчання із збільшенням кількості епох (рисунок 4.2).

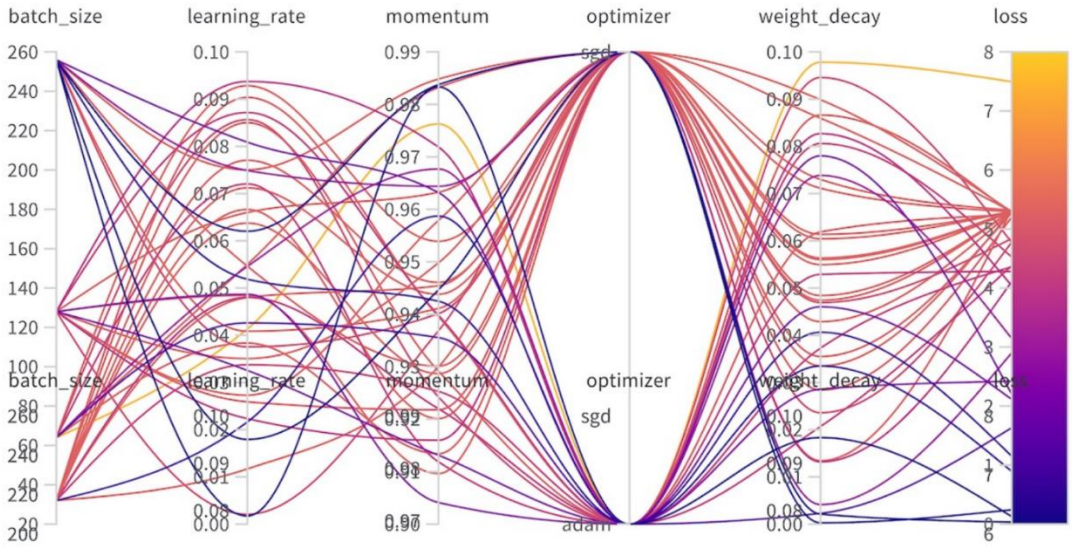

Рисунок 4.2 – Налаштування гіперпараметрів у класифікації зображень для ResNet18-PUNSH та ResNet50-PUNSH

Точність на DICD оцінюється на валідаційному наборі даних, оскільки базові дані в тестовому наборі даних недоступні. Точність на MMAUD

розраховується на тестовому наборі даних. У таблиці 4.1 підсумовано результати впливу якості зображень на точність та якість класифікації БПЛА.

Сніпети реалізації базових методів ViT та KNN можна побачити у додатках Е та Є відповідно.

Таблиця 4.1 – Порівняння ефективності класифікації зображень

Метод	Набір Даних	Точність	Якість Зображень
1	2	3	4
Моделі на основі методу нейронних мереж зі згортковою архітектурою			
ResNet18-PUNSH	DICD	50.13%	Низька або Середня
ResNet18	DICD	43.68%	Низька або Середня
Convolutional Nystromformer for Vision	DICD	48.03%	Низька або Середня
WaveMix-128/7	DICD	49.13%	Низька або Середня
ResNet50-PUNSH	DICD	87.01%	Низька або Середня
ResNet50	DICD	81.54%	Низька або Середня
Stochastic Depth	DICD	79.95%	Низька або Середня
Prodpoly	DICD	82.78%	Низька або Середня
ResNet18-PUNSH	MMAUD	65.50%	Висока
ResNet18	MMAUD	60.68%	Висока

Продовження таблиці 4.1

Метод	Набір Даних	Точність	Якість Зображень
1	2	3	4
Convolutional Nystromformer for Vision	MMAUD	51.03%	Висока
WaveMix-128/7	MMAUD	52.13%	Висока
ResNet50-PUNSH	MMAUD	95.01%	Висока
ResNet50	MMAUD	93.54%	Висока
Stochastic Depth	MMAUD	93.05%	Висока
Prodpoly	MMAUD	93.78%	Висока
Метод RNN			
RNN	DICD	28.32%	Низька або Середня
RNN	MMAUD	67.70%	Висока
Метод ViT			
ViT	DICD	37.57%	Низька або Середня
ViT	MMAUD	83.17%	Висока
Метод KNN			
KNN	DICD	39.12%	Низька або Середня
KNN	MMAUD	79.38%	Висока

Як зазначено у таблиці 3.3 найкращі результати було отримана на основі методів архітектури CNN, де найбільш точними важким(велика кількість шарів НМ) та легким(менша кількість шарів НМ) методи ResNet50-PUNSH та ResNet18-PUNSH, які було розроблені під час кваліфікаційної роботи. Найбільш неточним методом виявився метод на основі RNN, це

обумовлене тим, що ця архітектура більш спрямована на роботі з класифікації тексту. Досить низькою якістю зображення має метод ViT, це обумовлене малим набіром даних для тренувань, для підвищення точності було б необхідно збільшити набір за допомогою синтетичних даних.

4.2 Оцінка впливу використання методів передобробки на точність класифікації зображень (Accuracy) на основі запропонованого методу

Для порівняння точності з використанням методів передобробки буде зроблено порівняння методу на основі PUNSH-Conv блоку, з ввімкнутими та вимкнутими методами передобробки, які були запропоновані вище. У таблиці 3.4 описано порівняння цих методів.

Таблиця 4.2 – Результати порівняння базових моделей та основі PUNSH блоку, та їх використання з методами постобробки

Метод	Набір Даних	Точність
ResNet18-PUNSH	MMAUD	65.50%
ResNet18-PUNSH + предобробка	MMAUD	66.43%
ResNet50-PUNSH	MMAUD	95.01%
ResNet50-PUNSH + предобробка	MMAUD	95.37%

Як зазначено в таблиці 4.2 методи перед та пост обробки збільшують точність класифікації. Більш великим приростом є використання методів з моделю ResNet18-PUNSH, можна припустити, що такий приріст в точності обумовлено кращою якістю вхідних зображень в класифікатор. Основним компонентом для збільшення точності є методи предобробки, томущо вони покращуюсь якість зображення за допомогою приведення їх до стандартної форми.

4.3 Оцінка впливу використання модуля визначення базової інформації про зображення на навантаження обчислювача

Використання ресурсів під час класифікації зображення використовуючи нейромережеві класифікатори є однією з недосліджених тем.

Пропонується проаналізувати чи можливо створити модуль для вибору оптимального класифікатора у залежності від якості вхідних зображень на основі базової інформації про зображення. Чи можливо детермінувати вибір класифікатора тільки на основі базової інформації про зображення та чи залишиться якість класифікації зображення на тому самому рівні.

Для отримання даних щодо використаних ресурсів CPU та пошуку виступи під час виконання дії класифікації буде використано бібліотеку psutil. Дослід буде проведений на методі, запропонованого в роботі.

Вхідними даними будуть вибірка з набору зображень з БПЛА. На рисунку 4.3 можна побачити що найбільш всього ресурсів CPU використовувала модель ResNet50-PUNSH(а), з іншого боку найменш ресурсозатратною була модель ResNet18-PUNSH(б); хоча модель а і використовувала найбільшу кількість ресурсів, вона була найточніша, а модель б отримала меншу точність, як можна помітити наш метод використав середню кількість ресурсів CPU, порівнюючи між а та б, та втратив в точності. Тестування було проведено на ЕОМ з наступними характеристиками процесору 2,3 GHz 8-Core Intel Core i9H.

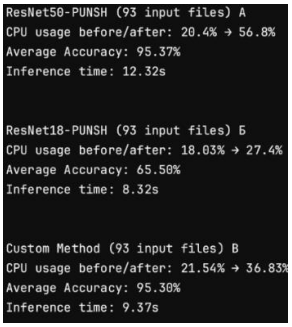

Рисунок 4.3 – Порівняння моделей ResNet50-PUNSH(а), ResNet18-PUNSH(б)

та запропонованого методу(в) по використаню ресурсів CPU

У таблиці 4.3 можна побачити результати порівняння класичних методів та моделей та запропонованого методу.

Таблиця 4.3 – Результати порівняння базових моделей на базі PUNSH блоку та запропонованого методу

Метод/Модель	Збільшення використання CPU, у відстоках, середне значення	Точність
ResNet50-PUNSH	22.4%	95.37%
ResNet18-PUNSH	9.37%	65.50%
Запропонований метод	15.32%	95.30%

Можна зазначити, що перевага в використані ресурсів є те що обертається класифікатор за допомогою базової інформації про зображення, хоча він і має невелику похибку під час роботи(що обумовлене використанням тільки базової інформації про зображення), цей метод досить сильно знижує використання ресурсів під час класифікації зображення.

Запропонований метод покращив якість класифікації зображень БПЛА на неякісних знімках, а також оптимізував використання ресурсів CPU без втрати якості класифікації. Завершуючи вище описане цей метод перевершую усі базові методу під час класифікації зображень з низькою роздільною здатністю, наявністю шумів та проблемами з контрастністю.

5 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

5.1 Вибір мови програмування

Вибір мови програмування для реалізації методу класифікації зображень є дуже важливим, через те, що класифікація зображень є дуже ресурс затратним процесом для ЕОМ, тому необхідно вибрати таку мову, для реалізації, яка буде містити в собі наступні риси:

- можливість масштабуватися;
- наявність бібліотек для навчання нейронних мереж;
- простота роботи всередині мови.

На поточний момент є наступні популярні мови програмування для створення програмних застосунків на основі методів машинного навчання: Python, C/C++, Julia та CUDA [20].

Як було зазначено вище, в роботі реалізовано метод класифікації зображень на основі PUNSH блоку в CNN архітектурі для побудови програмного забезпечення для класифікації БПЛА на зображені.

Мова програмування Python є найпоширенішою для створення та реалізації алгоритмів комп’ютерного зору та машинного навчання завдяки великій кількості бібліотек, таких як, TensorFlow, PyTorch, OpenCV, scikit-learn та інші. Високий рівень абстракції в цій мові, де ми не фокусуємося на видалені ресурсів після їх використання, за допомогою GC, та лаконічним синтаксисом сприяють швидкому створенню прототипів, зокрема моделей для класифікації об’єктів на зображеннях. Хоча, з іншого боку, ця мова не компілюється, а є інтерпретованою, що може стати вузьким місцем у випадку обробки великих масивів відеоданих або при розгортанні на обмежених вбудованих системах. Це більше відноситься до Cpython, але ще PyPy, який зі свого боку є теж інтерпретатором, але є JIT, що може пришвидшити роботу програми.

З іншого боку є мова програмування C/C++, де, навпаки, дуже

маленька кількість абстракції та важкий, не завжди зрозумілий синтаксис. Хоча ці мови забезпечують вищу ефективність виконання та кращий контроль за ресурсами, що особливо важливо при розробці реального часу або при оптимізації обчислень на рівні системного програмування. Використання OpenCV у середовищі C++ дозволяє досягати високої швидкодії при попередній обробці зображень та побудові класичних моделей розпізнавання. Додаючи інформацію про бібліотеки можна зазначити що в цих мовах нема таких сучасних та зручних фреймворків як в мові програмування Python для машинного навчання.

Julia намагається об’єднати найкращі риси мов програмування, які були зазначені вище, де швидкість роботи програми приближається до C/C++, а синтаксис намагається бути схожим своєю лаконічністю та простотою на Python. Але через те що Julia досить нова мова програмування, у котрої офіційна перша версія була випущена у 2012 році, на відміну від Python та C/C++ котрі були випущені у 1991 та 1972/1985 роках відповідно ще не має повноцінної інфраструктури бібліотек для створення ПО на основі машинного навчання, хоча можна зазначити що від має Flux.jl та Metalhead.jl, які і не можуть конкурувати з бібліотеками Python, але надають достатню кількість абстракції та допоміжних функцій.

CUDA, у свою чергу, не є повноцінною мовою програмування загального призначення, а розширенням для C/C++, яке дозволяє програмування на рівні графічного процесора. Її використання є доцільним у випадках, коли необхідно реалізувати високооптимізовані ядра для паралельної обробки зображень або виконання глибинних нейронних мереж без використання готових фреймворків. CUDA забезпечує максимальний контроль над GPU, але вимагає глибоких знань апаратної архітектури й значних зусиль для розробки, що робить її малопридатною для початкової побудови систем класифікації, проте незамінною на етапі оптимізації.

Виходячи з цього результати порівняння можна побачити у таблиці 5.1.

Таблиця 5.1 – Результати порівняння мов програмування

	Python	C/C++	Julia	CUDA
Швидкість	Мала чи середня, через те, що мова є інтерпритованою(Cpython, та PyPy)	Дуже швидка	Дуже швидка	Дуже швидка
Наявність бібліотек	Велики кількість бібліотек від таких компаній як Google, OpenAI, Amazon ітд	Мала кількість бібліотек	Мала кількість бібліотек	Мала кількість бібліотек
Простота	Дуже простий	Велика вхідна точка	Дуже простий	Велика вхідна точка

Тому, виходячи з таблиці 5.1 можна зазначити, що для створення ПЗ на основі моделі підходить мова програмування Python, де є майже все необхідне (2 з 3 пунктів), що робить її підходящим варіантом для створення ПЗ.

5.2 Структура Програмного Забезпечення

Ціль програмного забезпечення є в наступному – це створення моделі на основі методу, який був зазначений у голові 3, та використання цієї моделі, де вхідним параметром буде шлях до файлу, а виходом програми буде зображення з визначним дроном.

Програмна реалізація повинна отримувати файл за локальним шляхом та віддавати такий самий файл, але з лейблом, який буде вказувати, яка модель дрону використана. На рисунку 4.1 показано як відбувається процес обробки запиту користувача. Поточна реалізації буде використовувати гібридний класифікатор на основі нейромережі ResNet-50-PUNSH для

зображень з низькою якістю та ResNet18-PUNSH для зображень середньо-високої якості, тому що вона показала найкращі результати порівнюючи з класичними класифікаторами на основі CNN архітектури. Також були додані 3 рівня фільтрації зображення, а також модуль аутентифікації умов для отримання інформації щодо якості та розширення вхідних зображення з подальшим використанням ResNet50-PUNSH чи ResNet18-PUNSH нейроних класифікаторів.

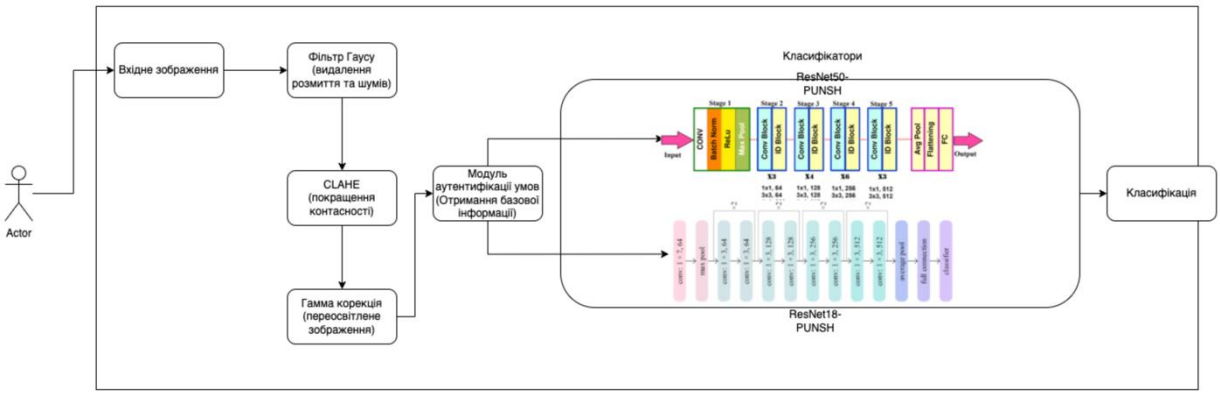

Рисунок 5.1 – Архітектура ПЗ

Поточна реалізація не містить засоби обробки зображення та отримання його через веб-форми, що може бути розглянуто як покращення ПЗ у майбутньому. Ще одною стороною для покращення є типи оптимізаторів на основі рантайму ONNX чи TensorRT для пришвидшення роботи нейронних класифікаторів.

5.2.1 Метод на його навчання

Усі моделі були побудовані на основі методу на базі PUNSH блоку, і вони реалізують модифіковану архітектуру ResNet на PyTorch з елементами оптимізації просторової інформації. На відміну від оригінального варіанту ,початковий блок згорток замінено модулем Focus, який агрегує просторову роздільну здатність у канал, забезпечуючи збереження контексту при

зменшенні розміру. Також, у деяких резідуальних блоках використовується трансформація space_to_depth, що переводить локальні просторові патерни в канал, підвищуючи виразність ознак.

Основним конструктивним елементом є BottleNeck-блок із трьома згортковими шарами (1×1, 3×3, 1×1), де за потреби застосовується stride або згадана трансформація. Резідуальні з'єднання адаптуються до зміни розмірності. Фінальна частина мережі включає глобальний адаптивний пулінг і повнозв'язний шар для класифікації, код моделі можна побачити у Додатку А.

Для навчання моделі було використано скрипт, теж реалізований на мові програмування Python. Він реалізує процес навчання згорткової нейронної мережі в рамках однієї епохи, використовуючи фреймворк PyTorch. На початку мережа переводиться в режим навчання, що дозволяє задіяти специфічні для цього режиму компоненти, зокрема, випадкове вимикання нейронів (dropout) та оновлення статистик пакетної нормалізації (batch normalization). Вхідні дані, у вигляді зображень та відповідних міток, у разі використання графічного процесора, переміщуються в його пам'ять для подальшої обробки.

Перед прямим поширенням сигналу нульовими стають накопичені значення градієнтів у параметрах моделі, що забезпечує коректне обчислення нових градієнтів у поточному кроці оптимізації. Після отримання прогнозу за допомогою прямого поширення, функція втрат (у даному випадку крос-ентропійна) обчислює різницю між прогнозом моделі та істинними значеннями. За допомогою алгоритму зворотного поширення похибки обчислюються градієнти, які використовуються оптимізатором для оновлення параметрів моделі.

Протягом навчання відбувається автоматичне логування значень втрат та норм градієнтів останнього шару в системи моніторингу, такі як TensorBoard і Weights & Biases. За наявності warm-up фази, навчальний граф розігрівається відповідно до заданого плану, що сприяє стабільнішій початковій динаміці оптимізації. Після завершення кожної епохи додатково

виконується запис гістограм ваг усіх параметрів моделі, що дозволяє відстежувати їх динаміку під час тренування. Код можна побачити у Додатку Б. Частина зображень була аргументована для поліпшення якості класифікації.

5.3 Аналіз отриманих результатів

Пісня навчання моделі їх було відправлено декілька фотографій БПЛА, та зображень без них, де вона правильно класифікувала їх. На рисунку 4.2 можна побачити один з прикладів цього, де він правильно класифікував дрон DJI Inspire.

Рисунок 5.2 – Приклад класифікованого зображення

Хоча метод, та модель, які були реалізовані правильно класифікують зображення, наступним кроком може бути додавання методів детекції об’їктів та після цього класифікація їх, ще одним з можливих варіантів покращення моделі є додавання можливості працювати не тільки з зображеннями, але й відео форматом.

ВИСНОВКИ

У ході кваліфікаційної роботи було зроблено аналіз існуючих методів класифікації зображень за критеріями точності класифікації БПЛА в важких умовах. Було виявлено найкращі базові методи кваліфікації – ResNet, та було створено метод на основі них з підвищеною точністю кваліфікації зображень низької роздільної здатністю, наявністю шумів та розмиття та проблемами з контрастністю. Було виявлено проблему в базовій архітектурі та було впроваджено заміну шарів stride-згортки та пулінгу новим блоком PUNSH-Conv. Було додано до методу алгоритми пред та після обробки на основі модулю аналізу базової інформації, а також методів передобробки на основі функції Гаусу для видалення шумів та розмиття, методів CLAHE та IAGCWD для поліпшення контрастності та створено алгоритм вибору більш та менш потужними нейромережовими класифікаторами для оптимізації використання ресурсів CPU. Емпірично було доведено переваги метода над іншими більш класичними методами для класифікації БПЛА на зображені.

Результати роботи можуть бути користі для створення систем детекції та класифікації БПЛА для подальшої протидії їм за алгоритмом для певної моделі, типу БПЛА.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Image coloring with flood fill : стаття. URL: https://medium.com/@chamod.14_80003/image-coloring-with-flood-fill-ca305c30a9ab (дата звернення:25.06.2025).

2. Image Segmentation with Watershed Algorithm: стаття. URL: https://docs.opencv.org/4.x/d3/db4/tutorial_py_watershed.html (дата звернення:25.06.2025).

3. José M, Inês D, Jorge B. Comparing Vision Transformers and Convolutional Neural Networks for Image Classification: A Literature Review. 2023, Vol. 13, No 9. P. 5-16.

4. S. Takahashi та ін. Comparing Vision Transformers and Convolutional Neural Networks for Image Classification: A Literature Review. 2024. Vol. 10, No 1007. P. 47-253.

5. E. H. Adelson та ін. Pyramid methods in image processing : підручник / за заг. Ред. P. J. Burt. Princeton, 1984. С. 33-41.

6. Singh, B., Davis, L.S. An analysis of scale invariance in object detection – snip : IEEE CVPR, 2018, С. 3578-3587.

7. Singh, B., Najibi, M., Davis, L.S. Sniper: Efficient multi-scale training. Advances in neural information processing systems : IEEE CVPR, 2018, С. 31.

8. Lin, T.Y та ін. Feature pyramid networks for object detection: IEEE Conference on Computer Vision and Pattern Recognition, 2017, 2117-2125.

9. Liu, S. та ін. Path aggregation network for instance Segmentation: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, С. 8759-8768.

10. Tan, M., Pang R., Le Q.V. Efficientdet: Scalable and efficient object detection : Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern, 2020, С. 10781-10790.

11. Kim, Y., Kang, B.N., Kim, D. : Learning relationship between

convolutional features for multi-scale object detection : ECCV, 2018, С. 316-331.

12. Харіс М., Шахнарович Г., Укіта Н. Завдання керованого суперрозділення: Виявлення об’єктів у зображеннях низької роздільної здатності : International Conference on Neural Information Processing. 2021. С. 387–395.

13. Порівняння моделей CNN : стаття. URL: https://www.slideshare.net/slideshow/lecture-5-convolutional-neural-network-models/265802507#103 (дата звернення: 26.05.2025).

14. Koziarski M., Cyganek B., Impact of Low Resolution on Image Recognition with Deep Neural Networks: An Experimental Study. AGH University of Krakow. 2018. 10.2478. С. 31-57.

15. Порівняння Фільтрів Гауса та Вінера для фільтрації зображень. TELKOMNIKA, June 2019, С.1455-1460.

16. Gang Cao та ін. Contrast Enhancement of Brightness-Distorted Images by Improved Adaptive Gamma Correction : Communication University of China. 2017. 1709.04427. С. 3-18.

17. Aerial Unmanned Drone Dataset : стаття. URL: https://github.com/ntu-aris/MMAUD (дата звернення:25.06.2025).

18. DICD : стаття. URL: https://www.kaggle.com/datasets/balajikartheek/drone-type-classification (дата звернення:25.06.2025).

19. Кім Й., Канг Б.Н., Кім Д. SAN, вивчення зв’язків між ознаками згортки для багатомасштабного виявлення об’єктів. Матеріали Європейської конференції з комп’ютерного зору (ECCV). 2018. С. 316–331.

20. A. Ipsen. Top 10 Programming Languages for Machine Learning : стаття. URL: https://www.pluralsight.com/resources/blog/upskilling/top-programming-languages-2025 (дата звернення: 15.03.2025).