Вінницький національний технічний університет
(повне найменування вищого навчального закладу)

Факультет інформаційних технологій та комп'ютерної інженерії
(повне найменування інституту)

Кафедра обчислювальної техніки
(повна назва кафедри)

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Система розпізнавання та класифікації об'єктів на аерофотознімках з використанням нейромереж»

Виконав: студент 2-го курсу, групи 1КІ-20м
спеціальності 123 – Комп'ютерна інженерія
(шифр і назва напряму підготовки, спеціальності)
Ярмощук Д. О.
(прізвище та ініціали)

Керівник: к.т.н., доцент каф. ОТ
Ткаченко О. М.
(прізвище та ініціали)
«____» _____ 2021 р.

Опонент: к.т.н., професор каф. ЗІ
Кондратенко Н.Р.
(прізвище та ініціали)
«____» _____ 2021 р.

Допущено до захисту
Завідувач кафедри ОТ
д.т.н., проф. Азаров О.Д.
(прізвище та ініціали)
«____» _____ 2021 р

Вінниця 2021

АНОТАЦІЯ

УДК 004.4

Ярмощук Д. О. Система розпізнавання та класифікації об’єктів на аерофотознімках з використанням нейромереж. Магістерська кваліфікаційна робота зі спеціальності 123 – комп'ютерна інженерія, освітня програма – комп'ютерна інженерія. Вінниця: ВНТУ, 2021. 118 с.

Магістерська кваліфікаційна робота присвячена проектуванню та розробці системи розпізнавання та класифікації об’єктів на аерофотознімках. Було проведено огляд предметної області та розглянуто методи виконання задачі.

Для реалізації системи було обрано нейронну мережу архітектури SegNet що працює на базі HTTP-сервера, який надає інтерфейс відправки зображень на обробку та отримання результатів роботи.

Проведено експериментальну перевірку роботи нейромережі та HTTP-серверу.

Також було проведено аналіз перспектив комерційного використання розробки, за результатами якого розробка може бути комерційно успішною.

Ключові слова: нейромережа, розпізнавання, класифікація, аерофотознімки, семантична сегментація, згорткові нейромережі, SegNet, HTTP, мікросервісна архітектура, горизонтальне масштабування.

ABSTRACT

UDC 004.4

Yarmoshchuk D. O. Aerial photography object recognition and classification system using neural networks. Master's thesis in the specialty 123 — computer engineering, educational program — computer engineering. Vinnytsia: VNTU, 2021. 118 p.

This work is devoted to the design and development of the system of recognition and classification of objects in aerial photographs. A review of the subject area was conducted and methods of performing the task were considered.

To implement the system, a neural network of SegNet architecture was selected, which works on the basis of HTTP-server, which provides an interface for sending images for processing and obtaining results.

An experimental test of the neural network and HTTP server was performed.

An analysis of the prospects for the commercial use of the development was also conducted, as a result of which the development can be commercially successful.

Keywords: neural network, object recoginition, object classification, aerial photography, semantic segmentation, convolutional neural networks, SegNet, HTTP, microservice architecture, horizontal scaling.

Вінницький національний технічний університет

(повне найменування вищого навчального закладу)

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра обчислювальної техніки

Рівень вищої освіти ІІ-й (магістерський)

Спеціальність – 123 – Комп'ютерна інженерія

Освітньо-професійна програма – Комп'ютерна інженерія

ЗАТВЕРДЖУЮ

Завідувач кафедри

обчислювальної техніки

проф., д.т.н. О. Д. Азаров

«___»_____2021 р.

З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ КВАЛІФАКАЦІЙНУ РОБОТУ СТУДЕНТУ

Ярмощуку Дмитру Олександровичу

(прізвище, ім'я, по-батькові)

1 Тема роботи «Система розпізнавання та класифікації об'єктів на аерофотознімках з використанням нейромереж», керівник роботи Ткаченко Олександр Миколайович, к.т.н., доцент затверджені наказом вищого навчального закладу від 06.03.2020 року № 277.

2 Строк подання студентом роботи 21.12.2021

3 Вихідні дані до роботи: проаналізувати основні методи та засоби для розпізнавання і класифікації об'єктів на аерофотознімках, способи вирішення задачі семантичної сегментації з використанням нейромережі, розробити систему для аналізу аерофотознімків.

4 Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити): огляд та аналіз технологій розпізнавання та класифікації об'єктів на зображеннях, дослідження можливості використання нейромереж для реалізації задачі, розробка системи для аналізу зображень.

5 Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень): лістинг сервісу розпізнавання, лістинг коду тренування нейромережі, результати роботи системи у вигляді фото.

6 Консультанти розділів роботи приведені в таблиці 1.

Таблиця 1 — Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1,2,3	**Ткаченко О.М., к.т.н., доцент каф. ОТ**		
5	**Кавецький В.В., к.е.н., доцент каф. ЕПВМ**		

7 Дата видачі завдання 03.09.2021.

8 Календарний план роботи наведено в таблиці 2.

Таблиця 2 — Календарний план роботи

№ з/п	Назва етапів виконання магістерської кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Огляд і аналіз джерел інформації	03.10.2021	Виконано
2	Розробка технічного завдання	10.10.2021	Виконано
3	Аналіз сучасного стану технологій	24.10.2021	Виконано
4	Обгрунтування вибраних методів	7.11.2021	Виконано
5	Реалізація і тестування системи	21.11.2021	Виконано
6	Розрахунок економічної частини	3.12.2021	Виконано
7	Оформлення пояснювальної записки та презентації	15.12.2021	Виконано

Студент _____ Ярмощук Д. О.

(підпис) (прізвище та ініціали)

Керівник роботи _____ Ткаченко О. М.

(підпис) (прізвище та ініціали)

ЗМІСТ

ВСТУП

З давніх часів для людської цивілізації існує необхідність орієнтування на місцевості, прокладення зручних і оптимальних маршрутів на ній, розподілу земельних ділянок та його документування. Ця необхідність виникала в різних галузях людської діяльності: військовій, торгівельній, сільськогосподарській. З часом ці потреби тільки зростали. Тому розвиток таких дисциплін як картографія та топографія є дуже важливою частиною розвитку людської цивілізації в цілому.

Автоматичне розпізнавання об'єктів на аерофотознімках місцевості дозволяє складати максимально точні карти місцевості в короткі терміни та проводити моніторинг різного роду людської діяльності: визначати стан посівів на полях, аналізувати густоту міської забудови, кількість зелених насаджень, створення детальних карт автомобільних шляхів.

Виходячи із розглянутого, завдання створення комп’ютерної системи що може розпізнавати та класифікувати об’єкти на аерофотознімках, є **актуальною задачею**.

Метою дослідження магістерської роботи є підвищення швидкості оброблення знімків мтсцевості за рахунок створення комп’ютерної системи, що може розпізнавати та класифікувати об’єкти на аерофотознімках з використанням нейромереж.

Задачі дослідження магістерської роботи:

— здійснити аналіз сучасних методів розпізнавання об’єктів на аерофотознімках;

— вирішити задачу семантичної сегментації об’єктів на аерофотознімках;

— провести навчання нейромережі для розпізнання об’єктів;

— створити програму для введення вхідних даних(аерофотознімків) та отримання результатів обробки зображень.

Об’єкт дослідження магістерської роботи — процес аналізу зображень, отриманих в результаті аерофотозйомки.

Предмет дослідження магістерської роботи — методи і програмні засоби оброблення цифрового зображення для пошуку та класифікації об’єктів на знімках місцевості.

Методи дослідження магістерської роботи: використовувались методи теорії множин для формування множини ознак для розпізнаванню об’єкта, методи математичної статистики для аналізу отриманих результатів розпізнавання в рамках використання їх в роботі нейромережі, мережеві технології, а саме протокол HTTP, принципи мікросервісної архітетури.

Наукова новизна отриманих результатів магістерської роботи полягає у вдосконаленні методу розпізнавання на основі нейромереж, в якому, на відміну від існуючих застосовано мережу архітектури SegNet, модель якої працює на базі HTTP-серверу, що дозволило зменшити використання системних ресурсів для процесу розпізнавання та отримати можливість горизонтального масштабування системи.

Практичне значення одержаних результатів магістерської роботи полягає у створенні програми для введення вхідних даних(аерофотознімків) та отримання результатів обробки зображень нейромережею.

Апробація. Основні результати роботи повідомлено та затверджено на Всеукраїнській науково-практичній онлайн-конференції «Молодь у науці: дослідження, проблеми, перспективи» (МН-2021) (Вінниця, 05.01.2021 - 14.05.2021)

Публікації. За результатами дослідження опубліковано тези доповіді: Система розпізнавання та класифікації об’єктів на аерофотознімках з використанням нейромереж [Текст] Д. О. Ярмощук // Молодь в науці: дослідження, проблеми, перспективи (МН-2021) Тез. доп. - Вінниця, 2021. - Режим доступу https://conferences.vntu.edu.ua/index.php/mn/mn2021/paper/view/13144/11053 [1].

1 АНАЛІЗ МЕТОДІВ СТВОРЕННЯ ТА ОБРОБКИ ФОТОЗНІМКІВ МІСЦЕВОСТІ

1.1. Сучасні методи створення аерофотознімків.

В 1858 році французом Гаспаром Фелікс Турнашоном було зроблено перший знімок місцевості(рис. 1.1) з повітря, а саме з повітряної кулі. Навіть враховуючи рівень фототехніки тих часів було важко не побачити можливостей які надають повітряні знімки місцевості. Військова розвідка, створення точних карт місцевості, при чому з набагато більшою ефективністю та точністю, ніж раніше.

Рисунок 1.1 — Перший в історії фотознімок з повітря

Створення перших літаків на початку XX століття зробило аерофотографування ще потужнішим інструментом через більшу швидкість літаків та їх незалежність(порівняно з повітряними кулями) від погодних умов.

Ці якості дозволяли покрити фотознімками набагато більшу площу за одиницю часу.

Аерофотографія еволюціонує і до сьогоднішнього дня, створення безпілотних апаратів та дронів, покращення якості та мініатюризація фотообладнання призвели до значного підвищення якості та здешевшання аерофотографії.

В середині XX століття у космос було запущено перший штучний супутник Землі та зроблено першу фотографію нашої планети з космосу. Так з’явилась ще одна можливість отримувати велику кількість знімків нашої планети з ще більшою ефективністю ніж з використанням літаків.

Зараз можливо отримати знімок будь-якої частини земної кулі у високій роздільній здатності. Для прикладу розглянемо сучасний знімок центрального району міста Вінниці з повітря(рис. 1.2). Чітко можна розрізнити будівлі, границі між ними, визначити автомобільні і пішохідні дороги, зелені насадження.

Рисунок 1.2 — Сучасний знімок міста Вінниці з повітря

Наведене зображення є прикладом планової аерозйомки місцевості. Такі знімки найкраще підходять для аналізу та оброки, тому що вони зняті під прямим кутом до земної поверхні, що значно спрощує роботу із зображеннями, дозволяє просто та точно розраховувати розміри та відстань між об’єктами

Іншим видом аерофотозйомки є панорамна зйомка, яка здійснюється під певним кутом до земної поверхні. Панорамні знімки(рис. 1.3) використовуються не настільки широко як планові, але вони дозволяють побачити те, що при плановій зйомці побачити неможливо, наприклад фасади будівель. Ці дані можливо використати для побудови тривимірної моделі місцевості, але така задача потребує великої кількості якісних знімків місцевості з різних ракурсів.

Рисунок 1.3 — Панорамний знімок міста Вінниці з повітря

Але самі по собі необроблені знімки місцевості не можуть знайти застосування в багатьох сферах господарства. Тому задача обробки, дешифрування та аналізу цих знімків є не менш важливою ніж безпосередньо фотографування. Але настільки великі об’єми даних не можуть бути оброблені вручну — дані вже можуть застаріти на момент закінчення ручної оброки. Тому виникає задача автоматизованої обробки знімків місцевості, зокрема семантичної сегментації.

1.2. Задача семантичної сегментації аерофотознімків

Для того, щоб знаходити та визначати типи об’єктів зображених на аерофотознімках, потрібно виришити задачу семантичної сегментації зображення.

Сегментація — це процес обробки зображення з ціллю розділення його на декілька сегментів. Сегментом є множина пікселів, яку також називаються об’єктом. Найчастіше метою сегментації є перетворення представлення зображення для того, щоб полегшити його аналіз, або спростити передачу каналами зв'язку. Сегментація дозволяє визначити зображені об'єкти та межі (лінії, криві, і т. д.) між ними. Також можна сказати що сегментація зображень — це процес присвоєння таких міток кожному пікселю зображення, що пікселі з однаковими мітками мають спільні візуальні характеристики[1]. На рисунку 1.4 наведено примітивну сегментацію аерофотознімку. Загальну класифікацію методів сегментації зображень наведено на рисунку 5.

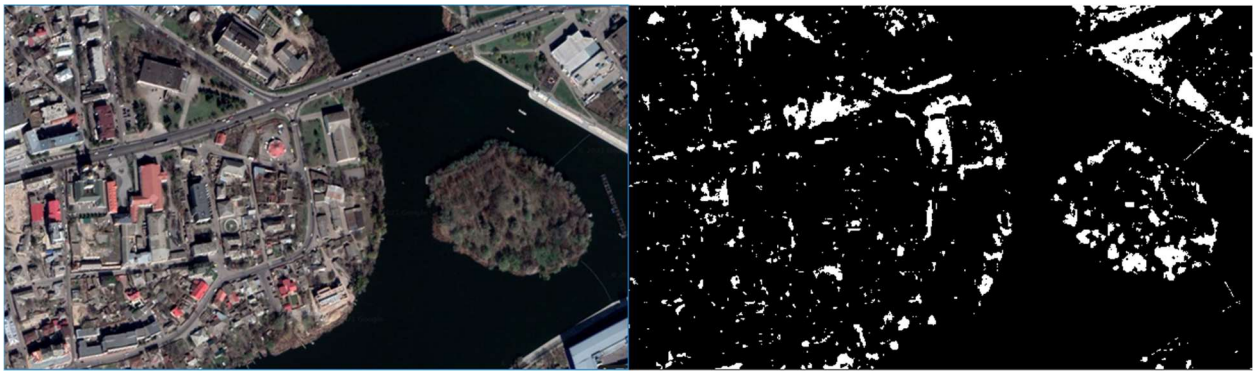


Рисунок 1.4 — Сегментація зображення за кольором

Детальніше розглянемо деякі з них.

Найпростішим методом сегментації зображення є метод порогових значень. Цей метод заснований на фільтруванні порогових значень, щоб перетворити чорно-біле зображення на своєрідну карту зображення де виділені області з кольорами що перевищують поріг. Головною особливістю цього методу є вибір порогового значення (або значень, коли вибрано кілька рівнів).

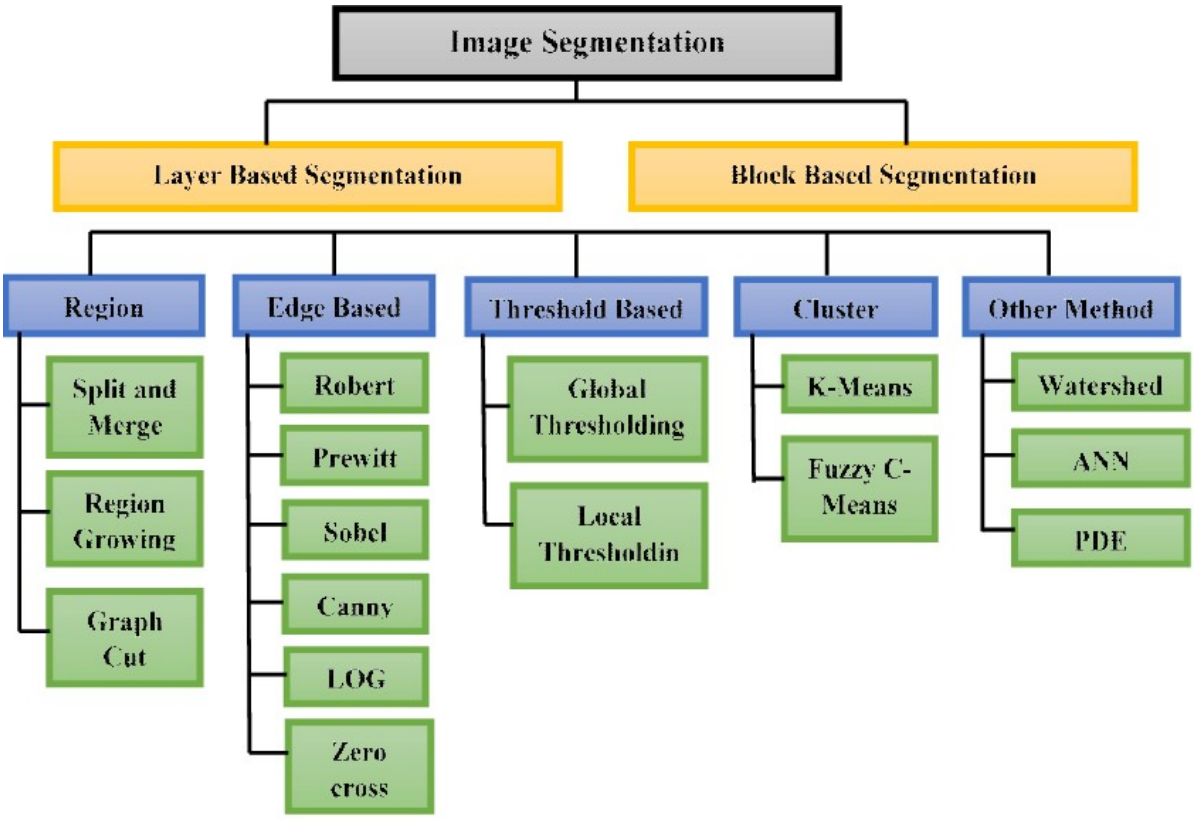


Рисунок 1.5 — Загальна класифікація методів сегментації зображень.

Найчастіше використовується кілька популярних методів, включаючи метод максимальної ентропії, порогове значення збалансованої гістограми, метод Отсу (максимальна дисперсія). Метод знаходить своє застосування в комп'ютернії томографії, де особливість радіограм дозволяє створити на їх основі чіткі зображення. В найновіших методах пропонується використання багатовимірних нечітких нелінійних порогів на основі правил. У цих роботах рішення щодо приналежності кожного пікселя до сегмента ґрунтується на кількох правилах, виведених з використанням нечіткої логіки та еволюційних алгоритмів.

Алгоритм K-середніх є ітераційною технікою, яка використовується для поділу зображення на певну кількість кластерів. Основний алгоритм складається з таких кроків:

а) вибір K центрів кластерів випадковим чином або на основі якогось евристичного методу, наприклад K-means++;

b) Вибір для кожного пікселя кластера таким чином щоб відстань між пікселем і центром кластера була мінімальною;

c) Перерахунок центрів кластерів шляхом усереднення всіх пікселі в кластері;

d) Повторне виконання кроків b і c, доки не буде досягнута конвергенція (пікселі не змінюють кластери).

Відстань тут — це сума квадратів або абсолютних значень різниця між пікселем і центром кластера. Різниця, як правило, залежить від кольору пікселя, інтенсивності, текстури та розташування або зваженої комбінації цих факторів. K можна вибрати вручну, випадковим чином або евристично. Конвергенція є гарантованою, але отриманий результат може бути не найбільш оптимальним. Якість результату залежить від початкового набору кластерів і величини K. Візуально процес кластеризації показано на рисунку 1.6.

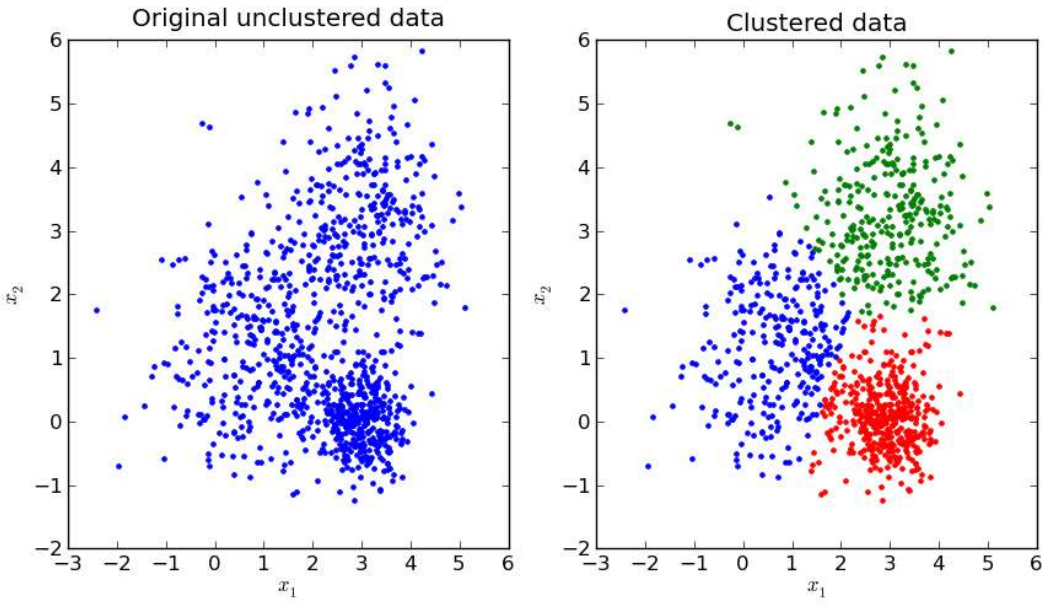


Рисунок 1.6 — Результат застосування методу K-середніх до множини точок

Метод вододілу розглядає градієнтну величину зображення як географічну карту. Пікселі з найвищою інтенсивністю градієнту відповідають так званим «лініям вододілу», які представляють межі сегменту. «Вода», розміщена на будь-якому пікселі, обмеженому загальною лінією вододілу, стікає вниз до загального

локального мінімуму інтенсивності (LIM). Пікселі, що «стікають» до загального мінімуму, утворюють «басейн», що представляє сегмент.

Більшість із вищезгаданих методів сегментації засновані лише на інформації про колір пікселів зображення. Але для того щоб успішно сегментувати зображення цього найчастіше замало, адже потрібно враховувати контекст для кожного пікселя: які пікселі його оточують, які їх характеристики. Методи сегментації, з можливістю навчання, як наприклад з використанням нейронної мережі, довзоляють вирішити цю проблему.

Нейронна мережа може обробляти невеликі ділянки зображення для розрізнення простих елементів, таких як краї та прості фігури. Інша нейронна мережа може потім об’єднати ці функції, щоб позначити області зображення відповідно. Прикладом такої мережі є карта Кохонена[1].

1.3. Сучасний стан технології нейромереж

Штучні нейронні мережі (ШНМ) — це обчислювальні системи, які працюють за принципом, схожим на біологічні нейронні мережі. Такі системи «вчаться» виконувати завдання, але не є запрограмованими на конкретні завдання. Наприклад, під час розпізнавання зображень вони можуть навчитися розпізнавати зображення, що містять автомобілі, аналізуючи приклади зображень які вручну позначені як "автомобіль" або "не автомобіль". Вони роблять це без будь-якого попереднього знання об’єкта. Натомість вони автоматично генерують характеристики, на основі яких відбувається розпізнавання на наборі даних, на яких вони навчаються. Щоб створити модель розпізнавання зображень, потрібно підготувати її на прикладах зображень предметів. Навчену нейромережу потім можна використовувати для аналізу конкретних зображень. Моделі, побудовані таким чином, можуть розпізнавати лише об’єкти, типи яких відомі моделі, приклади зображень яких застосовувались при навчанні мережі.

Нейронні мережі будуються на основі штучних нейронів, які імітують реальні нейрони. Кожна з'єднання між штучними нейронами є аналогом синапсів

в біологічному мозку, через них можливо передавати сигнал іншим нейронам. Штучний нейрон, який отримує сигнал, обробляє його і передає оброблений сигнал підключеним до нього нейронам.

У нейронних мережах сигнал є дійсним числом, а вихід кожного нейрона обчислюється деякою нелінійною функцією за параметрами на його входах.

З’єднання між нейронами називаються ребрами. Нейрони і ребра зазвичай мають вагу, значення якої підбирається в процесі навчання. Фактично "навчання" мережі – це процес підборку таких ваг, які дозволяють мережі вирішувати поставлені перед нею завдання. Вага визначає вплив даного нейрона на кінцевий сигнал мережі, або поточного її шару. Нейрони можуть мати поріг передачі, тобто сигнал буде посилатися на наступні нейрони, тільки якщо рівень сигналу перевищує певний поріг.

Нейрони об’єднуються в шари. Різні шари можуть виконувати різні перетворення на своїх входах. Сигнали переходять від першого шару (вхідного шару) до останнього шару (вихідного шару), а в деяких типах нейронних мереж, сигнали можуть проходити через одні й ті самі шари кілька разів.

На рисунку 1.7 наведено приклад структури штучної нейронної мережі, що складається з 5 вхідних нейронів (позначені зеленим), які отримують вихідну інформацію, 2 шари прихованих нейронів, по 7 нейронів у кожному (позначені оранжевим кольором). Ці нейрони обробляють вхідну інформацію і формують результат, які вони посилають до нейронів вихідного рівня.

Нейронів вихідного рівня — 4, позначені синім кольором. Після обробки сигналів моделі на цих нейронах надходять результати обробки. Специфіка того, що таке сигнал і яке він несе значення, залежить від конкретної мережі та її завдання. Нейронні мережі з імпульсним зв’язком (PCNN) — це нейронні мережі, що були вперше застосовані Рейнхардом Екхорном для моделювання зорової кори кішки і розроблені для високопродуктивної біоміметичної(імітуючою аналогічні процеси у тварин) обробки зображень.

Модель Екхорна надала простий та ефективний інструмент для вивчення зорової кори дрібних ссавців, і незабаром була визнаною такою що має значний

потенціал в галузі обробки зображень. У 1994 році ця модель була розширена Джоном Л. Джонсоном, який і дав їй теперішню назву.

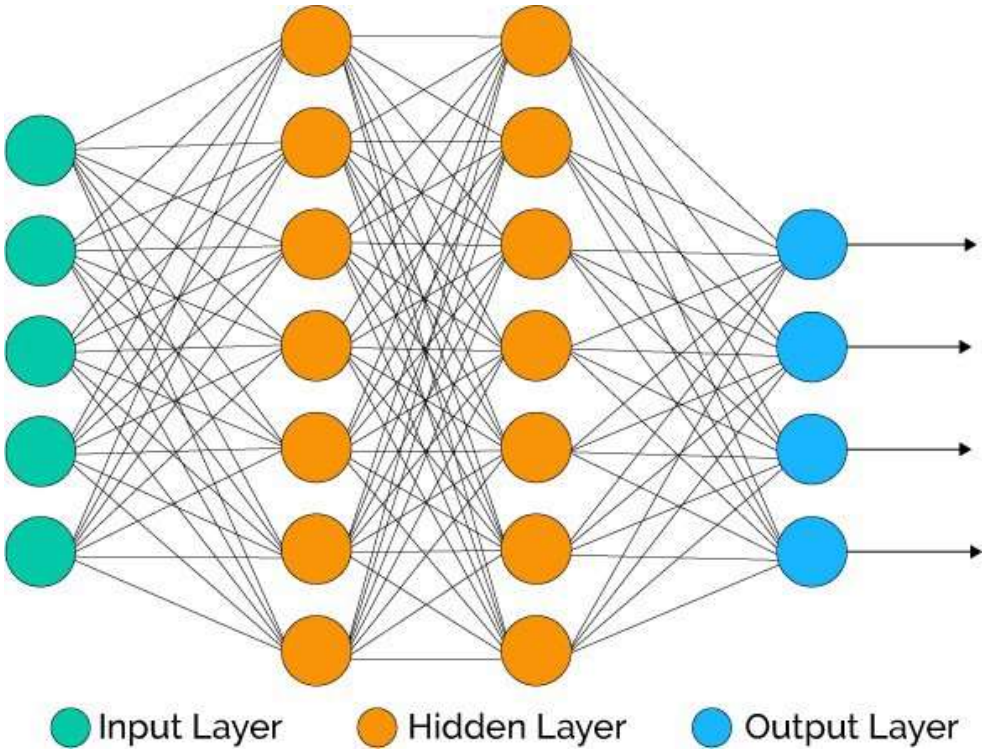


Рисунок 1.7 – Структура простої нейронної мережі

PCNN використовувалися для різноманітних додатків обробки зображень, включаючи: сегментацію зображень, генерацію ознак, розпізнавання обличчя, виявлення руху, зменшення шуму тощо. PCNN — це двовимірна нейронна мережа. Кожен нейрон в мережі відповідає одному пікселю у вхідному зображенні, отримання інформації про колір відповідного пікселя (наприклад, інтенсивність) як вхідний сигнал. Кожен нейрон також з'єднується з сусідніми нейронами, отримуючи від їхні локальні сигнали. Ці сигнали об'єднуються внутрішньою системою активації, що накопичує сигнали, поки ті не перевищить динамічний поріг, в результаті чого виходить імпульсний вихідний сигнал.

У порівнянні зі звичайними засобами обробки зображень, PCNN мають кілька значних переваг: стійкість до шуму, можливість обробляти будь-які геометричні фігури та стійкість до незначних коливань в рамках цих фігур.

Згорткова нейронна мережа (англ. convolutional neural network, CNN) — спеціальна архітектура штучних нейронних мереж, запропонована Яном

Лекуном в 1988 і націлена на ефективне розпізнавання образів, є однією з технологій глибокого навчання (англ. deep learning). Використовує деякі особливості зорової кори, а саме емулювання її простих клітин що реагують на прості лінії під різними кутами, і емулювання складних клітини, реакція яких залежить від активації певного набору простих клітин[2]. Головна ідея згорткових нейронних мереж полягає в чергуванні згорткових шарів (англ. convolution layers) та субдискретизуючих шарів (англ. subsampling layers чи pooling layers, шарів підвиборки). Структура мережі — однонаправлена (без зворотних зв'язків), багатошарова. Для навчання використовуються стандартні методи, найчастіше метод зворотного розповсюдження помилки. Функція активації нейронів (передавальна функція) — можлива будь-яка, вибирається та, яка найкраще підходить для конкретного випадку. Свою назву згорткові мережі отримали через широке використання операції згортки, суть якої в тому, що кожен фрагмент зображення множиться на матрицю (ядро) згортки поелементно, а результат підсумовується та записується в аналогічну позицію вихідного зображення.

Шар згортки (англ. convolutional layer) – це ключовий шар згорткової нейронної мережі. Шар згортки має для кожного з каналів вхідних даних свій фільтр, ядро якого обробляє попередній шар за фрагментами (підсумовуючи результати поелементної суми для кожного фрагмента). Ваги ядра згортки встановлюються у процесі навчання.

Особливістю згорткового шару є порівняно невелика кількість параметрів, що встановлюється під час навчання, оскільки в один момент часу воно працює лише з невеликим фрагментом вхідних даних. Так, наприклад, якщо вихідне зображення має розмірність 256×256 пікселів по трьох каналах(три кольори) то у повнозв’язній мережі буде існувати 256*256*3= 196 608 параметрів, а згортковий шар який використовує фільтри з ядром 3×3 пікселя з виходом на 6 каналів буде мати лише 9*3*6=162 параметрів і при збільшенні розмірності вхідного зображення ця різниця буде значно збільшуватись.

Згорткові шари послідовно застосовують попередньо навчені фільтри до вхідних зображень, щоб створити карти ознак, які дозволяють робити висновок про наявність цих ознак у вхідних даних. Комбінування декількох згорткових рівнів дозволяє шарам, що розташовані ближче до вхідного зображення, розпізнавати об’єкти більш низького рівня (наприклад, лінії) а подальші шари можуть працювати з більш складними та абстрактними поняттями, складними фігурами чи конкретними об’єктами. Така обробка зображення може створити проблему коли невеликі зміни(повороти, зсув, зашумлення) у вхідне зображення може зменшити точність мережі через те що зміняться створювані карти ознак. Основним рішенням даної проблеми є введення субдискретизуючого шару. В ньому створюється образ вхідного сигналу з нижчою роздільною здатністю, що містить важливі ознаки об’єкта і без дрібних деталей.

Результат обчислення кожної операції згортки потрапляє на функцію активації, яка найчастіше представлена деякою нелінійною функцією. Шар активації часто тісно зв’язано з шаром згортки. Традиційно використовуються функції типу гіперболічного тангенсу чи сигмоїди. Однак у 2000-х роках було запропоновано нову функцію активації — ReLU (скорочення від англ. rectified linear unit), яка дозволила суттєво прискорити процес навчання та збільшити швидкодію обчислень, через максимальну простоту обчислення значення функції. Функція має значення нуля для всіх аргументів менших за 0, або набуває значення аргументу для всіх додатніх вхідних значень.

Після кількох ітерації обробки зображення шарами згортки та субдискретизація робочий сигнал представляє собою дуже абстрактну апроксимацію вхідного зображення. Ця апроксимація далі оброблюється невеликою повнозв’язною мережею(що теж в свою чергу може мати декілька шарів), вихідне значення якої є кінцевим результатом обробки зображення.

У порівнянні з повнозв'язною нейронною мережею (типу перцептрону) — згорткова має набагато менше ваг, оскільки одне ядро ваг використовується для всього зображення.

В певний момент часу операція згортки працює тільки з невеликим фрагментом зображення, що створює можливість зробити алгортим паралельним. При цьому конкретна операція згортки є доволі простою в плані обчислювальних ресурсів. Ці особливості приводять до того, що згорткові мережі показують надзвичайно[3] високу швидкодію при навчанні і роботі якщо використовувати графічні процесори.

На рисунку 1.8 зображено структуру простої згорткової нейромережі, на якому видно як саме чергуються шари згортки та субдискретизації.

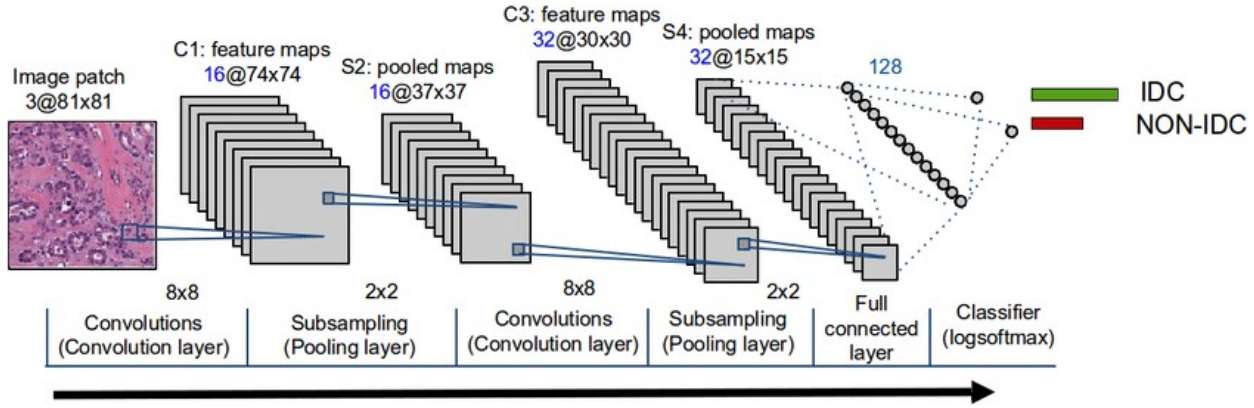


Рисунок 1.8 — Структура простої згорткової нейромережі.

Комбінація згорткових та субдискретизуючих шарів приводить до того, що мережа є доволі стійкою до таких змін у вхідному зображенні як повороти, зсуви, різні комбінації співідношення сторін. Також можна відзначити стійкість до різних шумів у вхідних даних.

Навчання мережі проходить з використанням класичного методу зворотного розповсюдження помилки, який є добре вивченим і оптимізованим за час використання.

Недоліком(як і в більшості інших типів нейромереж) є велика кількість внутрішніх параметрів мережі, які важко тонко конфігурувати і які суттєво впливають на результат. Існує кілька вивірених і добре працюючих конфігурацій мереж, але при створенні нової мережі важко відразу створити мережу що буде працювати так як того вимагає поставлена задача.

U-Net — це згорткова нейронна мережа, яка приймає зображення і призначає мітку(категорію) для кожного пікселя. U-Net була розроблена для виявлення меж клітин на зображеннях з мікроскопах[4]. U-Net дотримується класичної архітектури нейронних мереж. Структура енкодера мережі як і в інших згорткових мережах являє собою послідовність згорткових та субдискретизуючих шарів. Цю послідовність також називають енкодером. Але особливістю що відрізняє U-Net від інших архітектур нейромереж є наявність шарів збільшення розмірності або декодера. На кожен шар субдискретизації припадає один шар збільшення розмірності. Архітектуру U-Net зображено на рисунку 1.9.

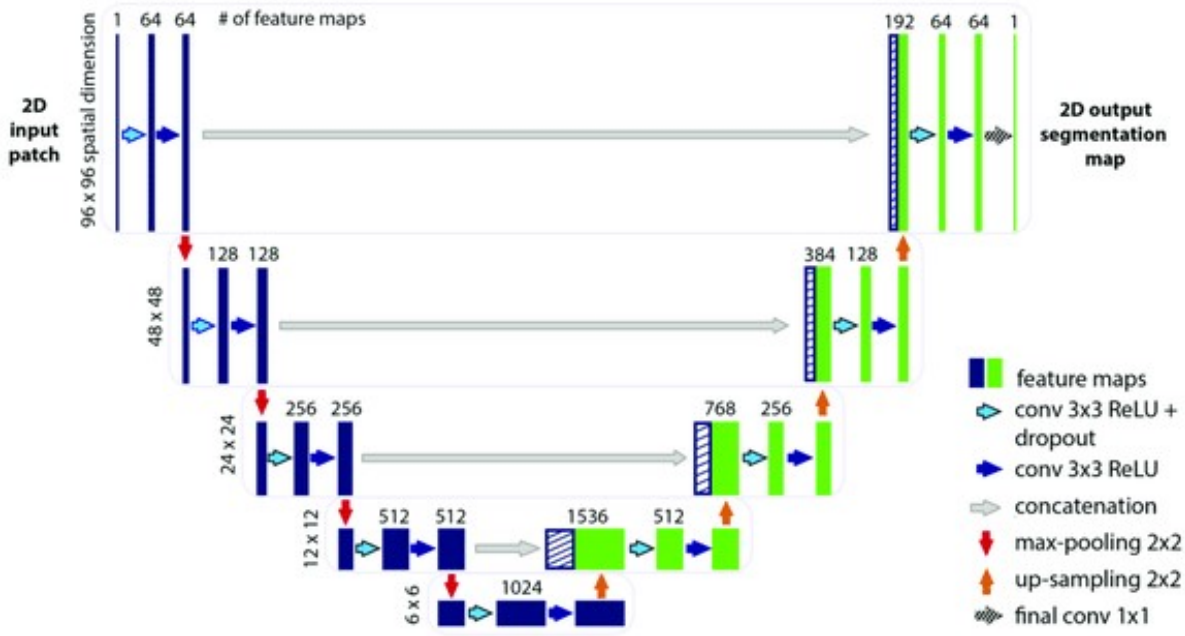


Рисунок 1.9 — Архітектура мережі U-Net

Така архітектура дозволяє визначати контекст зображення. Структура декодера використовує шари збільшення розмірності[5] для підвищення дискретизації, щоб комбінувати ознаки нижчих і вищх рівнів для збільшення точності розпізнавання, збереження малих, але важливих деталей зображення і для того, щоб кінцеві розміри результуючого зображення були близькі до розмірів вхідного зображення.

Наступним кроком в області рішень для семантичної сегментації зображень можна вважати мережу SegNet. Її архітектуру(рис 1.10) було розроблено саме для ефективної піксельної[6] семантичної сегментації. Одним з перших прикладів її застосування були рішення для аналізу дорожньої сцени, що повинно чітко розпізнавати елементи дороги, будівель, об’єкти на дорозі (автомобілі, пішоходи) і розуміти як всі ці об’єкти взаємодіють між собою.

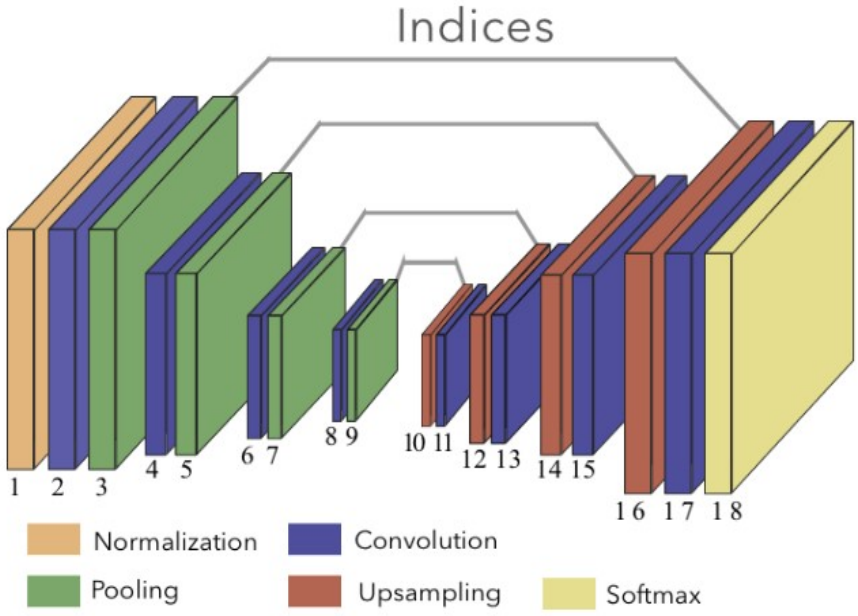


Рисунок 1.10 — Архітектура мережі SegNet

SegNet також може успішно використовуватись і для задач які потребують використанням семантичної сегментації з урахуванням контексту зображення, як от обробка аерофотознімків. Приклад результату такої обробки наведено на рисунку 1.11.

Головною відмінністю від U-Net є те що замість використання індексів об'єднання, усі карти об'єктів передаються від кодера до декодера, потім за допомогою конкатенації для виконання згортки. Це робить модель більшою та потребує більше пам'яті, але при цьому вона є швидшою і більш точною.

На додаток до завдань семантичної сегментації на рівні пікселів, які призначають певну категорію кожному пікселю, найсучасніші рішення в області сегментації включають екземплярну сегментацію[7], в яких на додаток до

визначення категорії кожен об’єкт в даній категорії може бути однозначно ідентифіковано, а також завдання паноптичної сегментації[8], яка поєднує ці два підходи для забезпечення більш повної сегментації зображення. На додаток до завдань семантичної сегментації на рівні пікселів, які призначають певну категорію кожному пікселю, найсучасніші рішення в області сегментації включають екземплярну сегментацію[7], в яких на додаток до визначення категорії кожен об’єкт в даній категорії може бути однозначно ідентифіковано, а також завдання паноптичної сегментації[8], яка поєднує ці два підходи для забезпечення більш повної сегментації зображення.

Рисунок 1.11 — Приклад обробки аерофотознімку з використанням SegNet

1.4. HTTP

HTTP — це протокол прикладного рівня моделі OSI що призначений для отримання різних типів медіа ресурсів в мережі Інтернет. Він є основою будь-якого обміну даними в Інтернет і використовує технологію клієнт-сервер, що означає існування клієнтів, що відправляють запити на отримання ресурсів та серверів що надають ресурси на запит клієнта.

Клієнти та сервери комунікують через обміну окремими повідомленнями. Повідомлення, надіслані клієнтом(найчастіше веб-браузером) називаються

запитами, а повідомлення, надіслані сервером як відповідь, називаються відповідями.

Спроектований на початку 1990-х років, HTTP надає широкі та прості у використанні можливості для розширення функціональності, завдяки чому він змінюється і розвивається і досьогодні. Завдяки своїй розширюваності він використовується не тільки для отримання гіпертекстових документів як було задумано з початку його існування, але і зображення, відео, побудови складних інтерактивних систем.

HTTP був розроблений таким чином, щоб бути простим і зрозумілим для людини, що забезпечує легший процес розробки та тестування. Механізм заголовків HTTP, що були представлені представлені в HTTP/1.0, дозволяють легко розширювати і експериментувати з цим протоколом. Нову функціональність можна ввести навіть шляхом запровадження простої конвенції між клієнтом і сервером про значення нового заголовка.

Популярність HTTP призвела до його підтримки всіма основними мовами програмування, що надають вбудовані інструменти, або користувацькі бібліотеки що дозволяють будувати як і HTTP-клієнти, так і HTTP-сервери.

Сервер HTTP може складатись з одного комп'ютера, але може бути і набором серверів, які включаються в себе окремі сервери балансування навантаження для рівномірного розподілення запитів на окремі сервери, які в свою чергу можуть відправляти запити на інші сервери, наприклад для кешування даних, сервери баз даних та ін.

Ще більше переваги HTTP розкриваються при використанні мікросервісної архітектури що робить окремі сервери ще більш незалежними, і надає ще більше можливостей для горизонтального масштабування сервісів.

HTTPS є сучасним розширенням HTTP для підтримки шифрування даних, що передаються між сервером і клієнтом. Його швидкодія може бути трохи меншою такої в HTTP через необхідність проведення процесу шифрування даних, але цієї різницею можна знехтувати, особливо враховуючи той факт що з'єднання є безпечним і захищеним від прослуховування третьою стороною.

1.5. Мікросервісна архітектура

Мікросервісна архітектура – варіант структурного стилю сервіс-орієнтованої архітектури (SOA) що струтурує додаток як набір слабо пов’язаних сервісів. Є антиподом монолітної архітектури. Служби є невеликими і включають невелику частину загального функціоналу системи, протоколи міжсервісної взаємодії є чітко визначеними і включають в себе невелику кількість функцій. Мета архітектури в тому, щоб кожен сервіс можна реалізувати, тестувати, розгортати та підтримувати незалежно від інших.

Слабке зв’язування сервісів зменшує кількість залежностей і їх складність, окремо взятий сервіс може не враховувати зміни в інших сервісах, якщо протокол взаємодії з цими сервісами залишається незмінним. Таким чином, це дозволяє швидко розробляти нові сервіси, розширювати функціонал системи без зміни існуючих сервісів. Але слабке зв'язування призводить до інших складнощів, які потрібно вирішувати. Інтерфейси сервісів повинні бути ретельно розроблені та розглядатися як публічне API, реалізовувати механізми зворотньої сумісності щоб оновлення одного сервіса не впливали на роботу системи загалом.

Організація системи як сукупності мікросервісів надає можливість для проведення горизонтального масштабування системи.

Горизонтальне масштабування означає підключення більшої кількості фізичних серверів для виконання якоїсь спільної задачі, в той час як вертикальне масштабування означає покращення вже існуючого сервера. Вертикальне масштабування є простішим, але тільки до моменту коли ніякі покращення вже не можуть збільшити його швидкодію. Особливо переваги мікросервісів та горизонтального масштабування проявляються в серверах що виконують складні операції що потребують багато обчислювальних ресурсів.

У цьому розділі було розглянуто актуальність і способи вирішення задачі розпізнавання і класифікації об’єктів на аерофотознімках. Згорткові нейромережі архітектури U-Net або SegNet є одними з найкращих способів вирішення цієї задачі через їх можливість адаптуватись до різних наборів

вхідних даних, можливість працювати із зображеннями нижчої якості, стійкість до змін кутів повороту, зміщень об’єктів на зображеннях. Також було розглянуто технологію HTTP та мікросервісну архітектуру, які дозволяють створювати розподілені сервіси, горизонтально масштабувати процес обробки зображень для збільшення швидкодії аналізу.

2 МАТЕМАТИЧНЕ ПІДГРУНТЯ РОЗПІЗНАВАННЯ ОБ'ЄКТІВ НА ЗНІМКАХ МІСЦЕВОСТІ

2.1. Нейрон

Кожен нейрон являє собою елемент з n входів, їх кількість може бути від 1 і до довільно обраного в залежності від специфіки задачі, на кожен з яких подається деяка частина масиву вхідних даних. Вектор вхідних значень x, має довжину n і фактично містить значення ознак з деякого зображення із навчального набору даних. Кожен із входів має свій набір параметрів, які формують вектор w — вектор стовпців ваг(від англ. weight – вага) і вектор b — зміщення (від англ. bias — зміщення), які підлаштовуються під час процесу навчання. На кожній ітерації нейрон обчислює середнє зважене значення вектора x на основі його поточного вагового вектора w і зміщення b. Нарешті, результат цього обчислення передається через деяку функцію активації g. Обчислення вихідного значення нейрона обчислюється за формулою 2.1:

$$z = w_1 x_1 + w_2 x_2 + w_3 x_3 + \ldots + w_n x_n \qquad (2.1)$$

2.2. Функції активації

Функції активації є одним із ключових елементів нейронної мережі. Без них нейронна мережа є комбінацією лінійних функцій, тож сама мережа є лінійною. Така модель мала б обмежену можливість для використання на різних вхідних даних, і була б не кращою в цьому показнику ніж звичайна логістична регресія. Нелінійність забезпечує більшу гнучкість і створення складних функцій під час процесу навчання. Функція активації також має значний вплив на швидкість навчання та роботи мережі, що є одним з основних критеріїв їх вибору. Наразі найпопулярнішим рішенням для застосування в прихованих шарах є ReLU. Але традиційні функції, як наприклад сигмоїда, знаходять своє застосування в деяких задач. У випадку сигмоїди вона часто застосовується в

задачах бінарної класифікації, де є всього два можливих вихідних значення, 0 та 1. На рисунку 2.1 наведено графіки та формули деяких функцій активації.

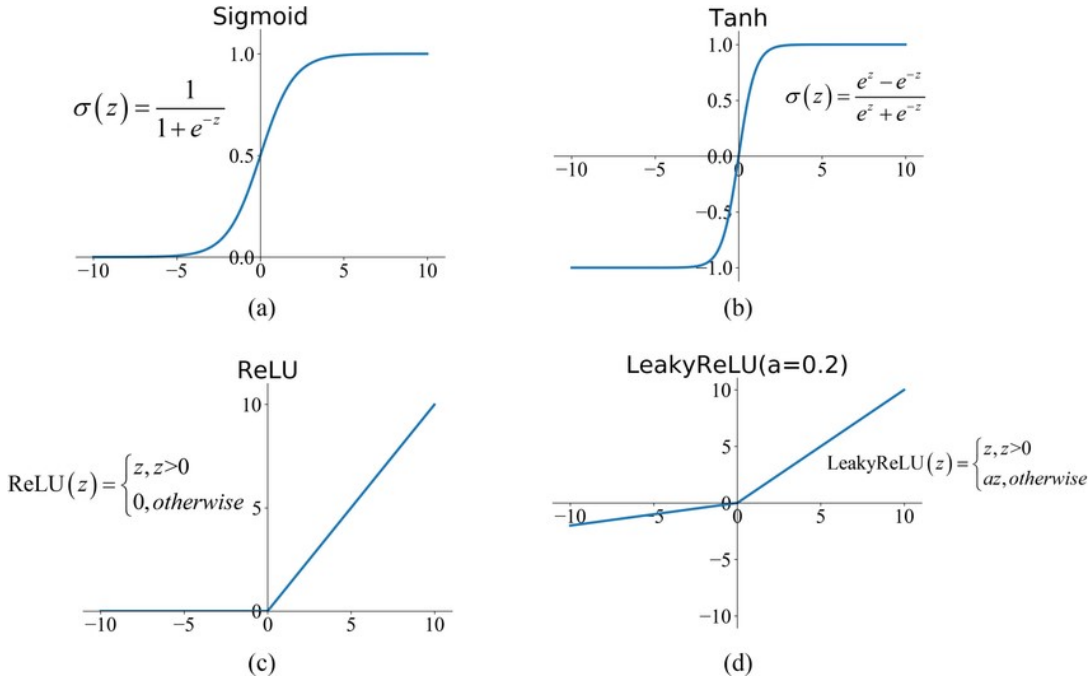


Рисунок 2.1 — Графіки найбільш часто використовуваних функції активації

2.3. ReLU

Функція випрямленого лінійного вузла або ReLU(англ. Rectified Linear Unit) є кусочно-лінійною функцією, значенням якої є значення переданого аргументу, якщо він є позитивним, якщо значення аргументу менше або дорівнює нулю то і її значення дорівнює нулю:

$$f(x) = max(0, x)$$

Вона стала функцією активації за замовчуванням для багатьох типів нейронних мереж, оскільки моделі з нею швидше тренуються та показують високу швидкодію.

При навчанні методом стохастичного градієнтного спуску із зворотним поширенням помилок для навчання нейронних мереж, потрібна функція

активації, яка веде себе як лінійна функція, але насправді є нелінійною функцією, що дозволяє формувати складні взаємозв'язки в досліджуваних даних.

Оскільки ReLU є майже лінійною, вона зберігає багато їх властивостей, які дозволяють легко оптимізувати лінійні моделі за допомогою методів на основі градієнтного спуску. Вони також зберігають багато властивостей, завдяки яким лінійні моделі добре узагальнюються.

2.4. Функція втрати

Основним джерелом інформації про хід процесу навчання є значення функції втрат. Функція втрат підібрана таким чином щоб показати, наскільки поточне рішення «не дотягує» до ідеального рішення. Найчастіше для функції втраст застосовується функція бінарної перехресної ентропії, але в залежності від вирішуваної проблеми, можуть застосовуватися і інші функції.

Нижче наведено формулу для обчислення бінарної перехресної ентропії:

$$L = -\frac{1}{S_o}\sum_{i=1}^{S_o} y_i * log * \widehat{y_i} + (1 - y_i) * log(1 - y_i)$$

де $\widehat{y_i}$— це і-те скалярне значення набору вихідних значень моделі;

y_i— відповідне еталонне значення;

S_o— кількість вихідних значень моделі.

2.5. Процес навчання нейромережі

Зміст процес навчання навчання мережі полягає у підлаштуванні значень ваг і зміщень нейронів таким чином, щоб значення функції втрат було мінімальним. Щоб досягти цього використовують метод градієнтного спуску для знаходження мінімумів функції.

На кожній ітерації навчання обчислюється значення часткових похідних функції втрат по кожному з параметрів нейронної мережі. Завдяки обчисленням

похідної можна зробити висновок, про те як налаштовувати параметри мережі, щоб з кожним етапом тренування зменшувати значення похідної. Але нейронна мережа являє собою дуже складну і обширну структуру в рамках якої обчислення цих похідних є складною задачею. Щоб вирішити цю задачу використовують метод зворотного поширення помилки.

Метод зворотного поширення помилки — це алгоритм, який дозволяє розрахувати дуже складні градієнти при навчанні. За цим методом величина, на яку будуть змінюватись ваги розраховується за формулою:

$$\Delta\omega_{i,j} = -\eta \frac{\partial L}{\partial \omega_{i,j}}$$

де η — швидкість навчання що є одним з найголовніших параметрів мережі;

$\omega_{i,j}$ — значення ваги, яке підлаштовується.

У наведених вище рівняннях η представляє швидкість навчання — метапараметр мережі, який дозволяє керувати величиною підлаштування за один етап навчання. Вибір швидкості навчання має велике значення — якщо встановити її значення як занизьке, мережа буде вчитися дуже повільно, якщо взяти зависоке значення, буде важко підібрати оптимальне значення мінімуму і як результат сформована модель не буде оптимальною. Зазвичай швидкість навчання регулюється в процесі навчання, на початку процесу навчання встановлюється висока швидкість навчання, щоб швидше знайти більш-менш оптимальне рішення, яке потім буде тонко регулюватись з меншою швидкістю навчання.

2.6. Операція згортки

В згорткових нейромережах для формування фільтрів ознак використовується так звана операція згортки. Фільтри дозволяють виявляти різні ознаки, межі об’єктів і фігури різної складності.

Згортка є операцією над двома функціями f(t) і g(t), в результаті якої утворюється третя функція. Можна вважати що ця третя функція є зсунутою і накладеною сумою двох інших функцій. Обчислюється за формулою:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(t - \tau)g(\tau)d\tau$$

На рисунку 2.2 показано приклад графіку функції згортки для двох прямокутних функцій.

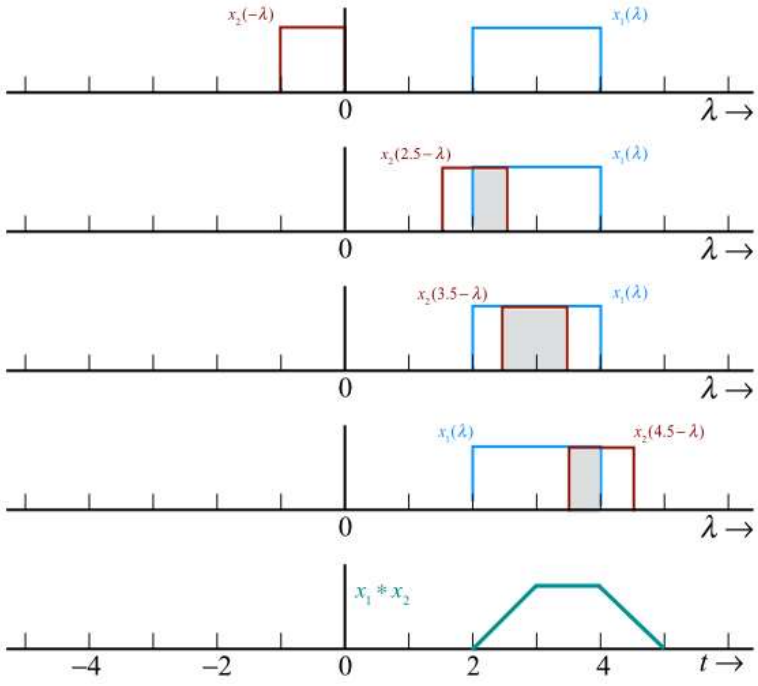


Рисунок 2.2 — Графік згортки двох прямокутних функцій

Операцію згортки можна розглядати як «схожість» однієї функції з віддзеркаленою та зсунутою копією іншої. Поняття згортки узагальнюється для функцій, визначених на вимірних просторах, і може розглядатися як особливий вид інтегрального перетворення.

У випадку дискретних значень двох функцій згортка відповідає сумі значень з коефіцієнтами, що відповідають зміщеним значенням.

Згортковий шар нейромережі в спрощеному вигляді записується наступним чином:

$$x_j^l = f(\sum_i x_i^{l-1} * k_j^l + b_j^l)$$

де x_j^l — карта ознак j шару l;

f() — функція активації;

b_j^l — коефіцієнт зсуву для j-тої карти ознак;

k_j^l — j-те ядро згортки.

2.7. Субдискретизація і збільшення розмірності карти ознак

Шар субдискртезицації виконує операцію зменшення розмірності вхідної карти ознак. Найчастіше використовуваним методом субдискретизації є розбиття карти ознак на матриці розміром n*n елементів, n зазвичай береться рівним 2 і вибір з кожної матриці максимального значення, адже саме воно в найбільшій мірі описує цю матрицю. Шар субдискретизації можна описати таким чином(2.6):

$$x^l = f(a^l * subsample(x^{l-1}) + b^l)$$

де x^l — вихід шару

f() — функція активації

a,b — коефіцієнти субдискретизації

Шар збільшення розмірності операцію обернену до субдискретизації, а саме збільшення карти ознак в деяку кількість раз, цей коефіцієнт збільшення повинен відповідати коефіцієнту з яким відбувалась субдискретизація. Тобто, якщо шар субдискретизації зменшував карту ознак вдвічі, то шар збільшення розмірності збільшує її вдвічі. Зазвичай операція представляє собою просте дублювання цільового значення як матриці розміром n*n, тобто значення x після операції буде представлено як матрицю де всі значення дорівнюють x. Описати шар збільшення розмірності можна аналогічно до шару субдискретизації:

$$x^l = f(a^l * upsample(x^{l-1}) + b^l)$$

де x^l — вихід шару;

f() — функція активації;

a,b — коефіцієнти.

2.8. Оцінка результатів розпізнавання

В задачі семантичної сегментації зображення кожному його пікселю присвоюється клас об’єкта, частиною якого він є. Множина пікселів одного типу формує на зображенні певну фігуру. Щоб оцінити точність роботи нейромережі можна порівняти наскільки параметри цих фігур, такі як площа чи розміри і форма їх сторін є подібними для зображення з тренувального набору, та цього самого зображення, обробленого нейромережею.

Таку оцінку дозволяє зробити обчислення коефіцієнта Жаккара(також відомий як «intersection over union») для зображення тренувального набору, та зображення після обробки. Візуальне представлення наведено на рисунку 2.3.

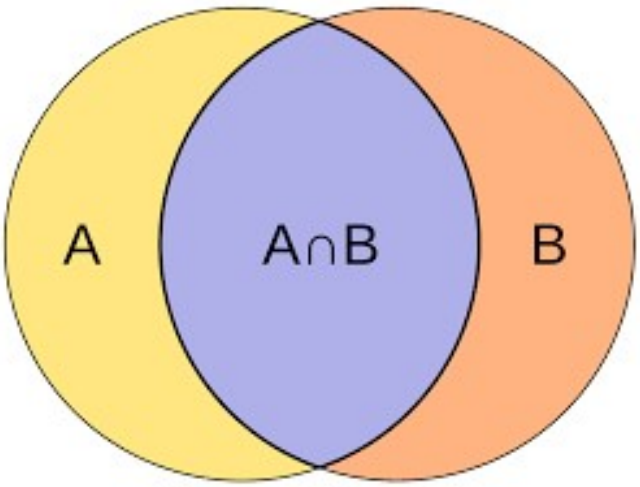


Рисунок 2.3 — Візуальне представлення коефіцієнта Жаккара

Цей коефіцієнт дозволяє визначити міру подібності або перетину двох множин(в нашому випадку розглядається множина пікселів).

Коефіцієнт обчислюється за формулою(2.9):

$$J(A,B) = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

де A — перша множина, в нашому випадку це буде еталонна фігура з тренувального набору;

B — друга множина, в нашому випадку це фігура отримана на виході обробки нейромережі.

З формули видно, що мінімальне значення коефіцієнту дорівнює 0, а максимальне — 1, тобто $0 \leq J(A,B) \leq 1$. Ціллю процесу тренування нейромережі є максимізація значення коефіцієнта Жаккара.

2.9. Вибір архітектури нейромережі

Для вирішення задачі семантичної сегментації використовуються декілька архітектур нейромереж, таких як: SegNet, DeconvNet, FCN, U-Net. За результатами дослідження[14] де було проведено тестування семантичної сегментації на прикладі дорожньої сцени, мережа SegNet показує кращі результати на більшій частині класів розпізнавання та найбільше значення усередненої загальної точності з усіх представлених архітектур. Також наведено порівняння архітектур в задачі сегментації інтер'єрної сцени, в якому SegNet знову показує одні з найкращих результатів, лише трохм поступаючись архітектурі DeepLabv1. SegNet показує кращі результати для великих за розміром об'єктів, але дещо гірші для менших об'єктів. В задачі розпізнавання об'єктів на аерофотознімках це може означати гіршу якість розпізнавання для малих об'єктів, таких як малі будівлі, автомобілі, поодинокі невеликі дерева.

SegNet є менш швидкодіючою ніж більшість інших архітектур через наявність декодера, але швидшою за DeconvNet через відсутність повнозв'язних шарів. Важливою перевагою SegNet є набагато менше споживання оперативної пам'яті(у випадку використання відеоадаптера — менше використання пам'яті

відеоадаптера). SegNet може використовувати майже в два рази менше оперативної пам’яті чи пам’яті відеоадаптера.

Також можна зазначити що натренована модель SegNet займає менше місця на диску ніж найближчі аналоги. У дослідженні модель SegNet мала об'єм в 117МБ що трохи більше ніж 83 у DeepLab, але значно краще ніж 539 у FCN, чи 877 у DeconvNet. Ця особливість дозволить зменшити розмір інсталяції на диску

На рисунку 2.4 наведено візуалізацію архітектури використаної нейронної мережі.

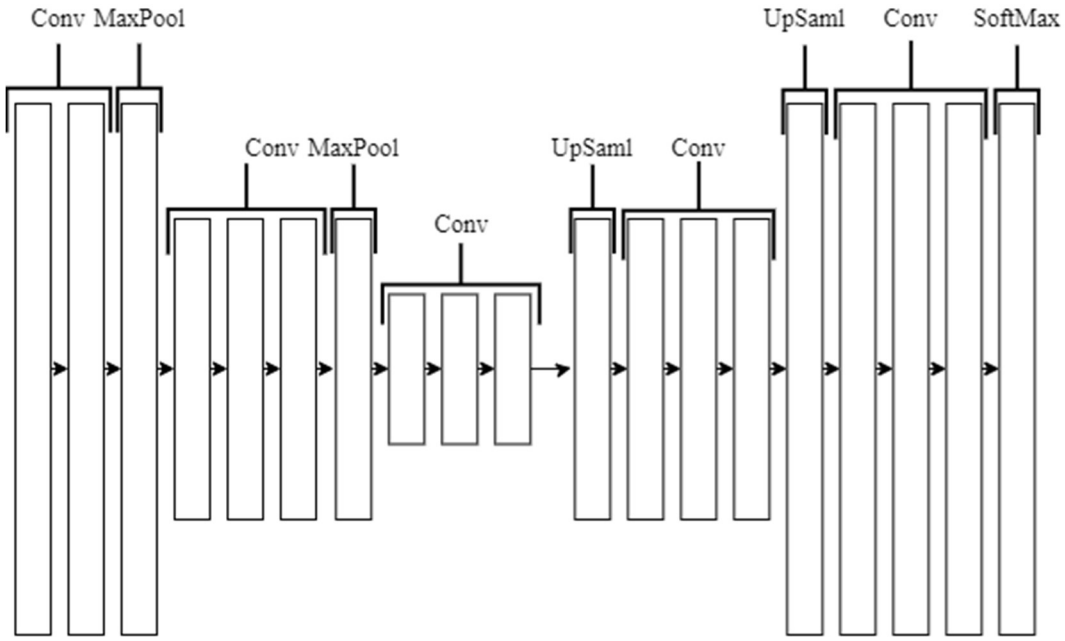


Рисунок 2.4 — Візуалізація архітектури використаної нейронної мережі

Враховуючи наведене порівняння як модель нейромережі було обрано архітектуру SegNet. Її основними компонентами є так звані енкодер та декодер.

Енкодер — це послідовність згортових шарів та шарів субдискретизації, з кожним шаром карта ознак зменшується вдвічі.

Декодер — послідовність згорткових шарів та шарів збільшення розмірності, на кожен шар енкодера припадає один шар декодера з такою ж розмірністю, також ці шари є пов'язаними між собою для обміну інформації. Наявність зв'язку(англ. pooling index) між відповідними шарами енкодера і

декодера, а також між їх послідовними шарами дозволяє враховувати при аналізі зображення одночасно і більші елементи зображення і менші, але тим не менш теж важливі для розуміння контексту зображення.

В даному розділі було розглянуто основні принципи функціонування та поняття теорії нейромереж. Розглянуто специфічні для згорткових нейромереж поняття субдискретизації та збільшення розмірності карти ознак. Для оцінювання якості роботи нейромережі обрано коефіцієнт Жаккара, також відомий як intersection over union. Розглянуто архітектури мереж для виконання семантичної сегментацію, для реалізації задачі було обрано архітектуру SegNet.

3 РОЗРОБКА КОМП'ЮТЕРНОЇ СИСТЕМИ

3.1. Вхідні та тестові дані

Для побудови системи розпізнавання та класифікації об'єктів на аерофотознімках було обрано набір даних Dstl Satellite Imagery Feature Detection зі змагання на платформі Kaggle[15]. Знімки було отримано з використанням супутника WorldView-3.

WorldView-3 входить до групи з 6 супутників компанії DigitalGlobe, Сумарно ця група має здатність фотографувати близько 4млн км^2 земної поверхні за добу. Набір даних включає знімки, кожен з яких покриває площу приблизно в 1км^2. Супутник веде зйомку в трьох режимах[16].

VNIR (Visible and Near Infrared) — мультиспектральний видимий та ближній інфрачервоний діапазон — 8 каналів, 6 з них — видимого діапазону (фіолетовий, синій, зелений, жовтий, червоний, крайній червоний) та 2 — ближнього інфрачервоного (Ближній-ІЧ1 , Ближній-ІЧ2).

SWIR (Shortwave Infrared) — середній інфрачервоний діапазон, дозволяє вести зйомку крізь туман, смог, пил, дим, туман та хмари(8 каналів), що дозволить зменшити вплив погодних умов на якість фото і як наслідок на процес розпізнавання.

CAVIS (clouds, aerosols, vapors, ice, snow – хмари, аерозолі, пари, лід, сніг) дозволяє проводити корекцію атмосферних спотворень(12 каналів), цей діапазон також дозволяє зменшити вплив на якість фото

В таблиці 3.1 детально описано залежність роздільної здатності зйомки та роздільної здатності файлів в залежності від діапазону зйомки. Всього в наборі даних зібрано знімки 450 зон в режимах VNIR і SWIR. 25 з цих зображень є розміченими, тобто для них вказано які об'єкти і на якій позиції розташовані на знімку. Ці зображення будуть тренувальним набором даних, саме на них нейромережа буде вчитись розпізнавати об'єкти на знімку. Решта 425 зображень таким чином може бути використана для перевірки нейромережі після процесу навчання.

Таблиця 3.1 — Залежність роздільної здатності зображень тренувального набору від режиму зйомки супутником WorldView-3

Назва	Довжина хвилі, мкм	Роздільна здатність, м	Роздільна здатність файлів, пікселі
Панхроматичний	450-800	0.31	3396*3348
Червоний (RGB)	630-690		
Зелений (RGB)	510-580		
Синій (RGB)	450-510		
Фіолетовий	400-450	1.24	849*837
Синій	450-510		
Зелений	510-580		
Жовтий	585-625		
Червоний	630-690		
Крайній червоний	705-745		
Ближній-ІЧ1	770-895		
Ближній-ІЧ2	860-1040		
SWIR-1	1195-1225	7.5	136*134
SWIR-2	1550-1590		
SWIR-3	1640-1680		
SWIR-4	1710-1750		
SWIR-5	2145-2185		
SWIR-6	2185-2225		
SWIR-7	2235-2285		
SWIR-8	2295-2365		

Типи об’єктів, які зображені на знімках були обрані організаторами змагання і включають в себе наступні:

— будівлі — різноманітні будівлі;

— структури — невеликі рукотворні об’єкти(переважно паркани);

— дороги;

— грунтові дороги, стежки;

— дерева;

— сільськогосподарські угіддя;

— річки;

— озера, ставки;

— великий транспорт — автобуси, вантажівки;

— малий транспорт — легкові автомобілі, двоколісний транспорт;

Гістограму розподілення об’єктів за типами відносно до загальної площі наведено на рисунку 3.1.

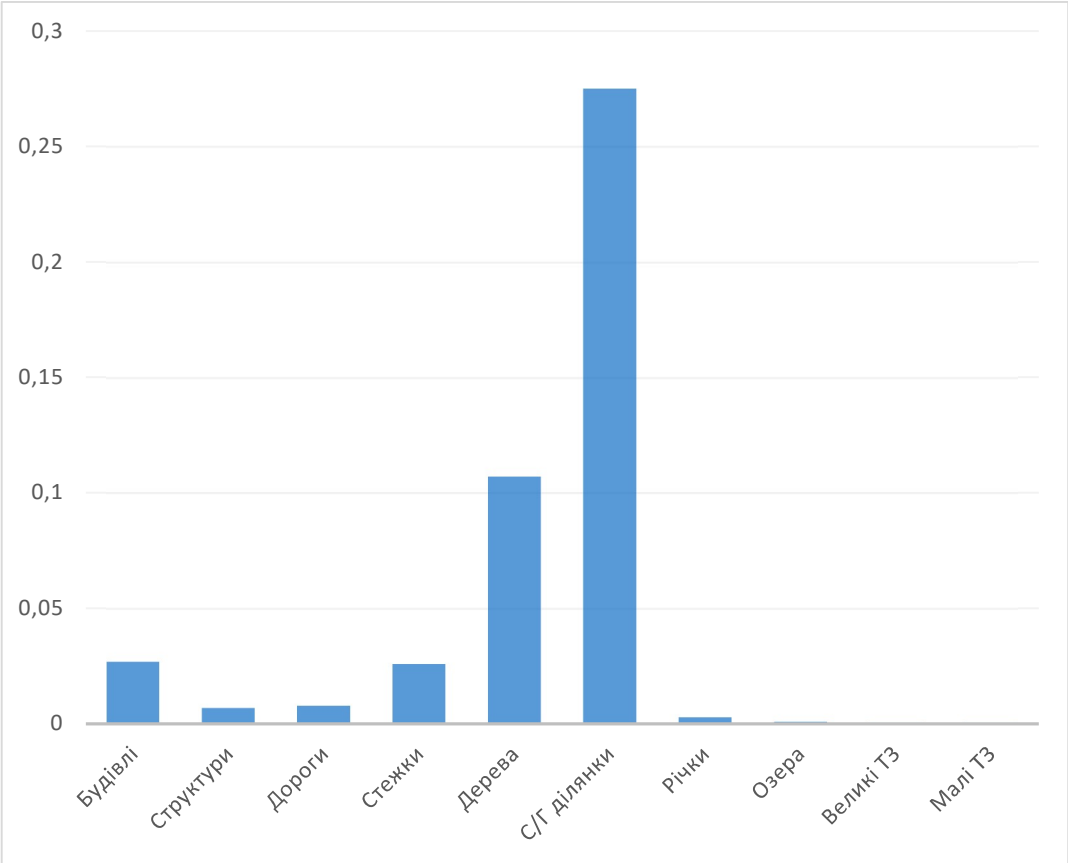


Рисунок 3.1 — Гістограма розподілення класів за площею у тренувальному наборі даних

Зрозуміло, що не всі типи об’єктів в наборі зустрічаються з однаковою частотою. Найбільше на більшості знімків будуть зустрічатись дерева, або сільськогосподарські угіддя. В меншій мірі будуть представлені будівлі та

дороги. Будівлі зустрічаються приблизно на половині зображень тренувального набору. Зображення водойм зустрічаються доволі рідко. Найменш часто можна побачити транспортні засоби.

На рисунку 3.2 наведено приклад зображення з набору даних у спектрі RGB, а на рисунку 3.3 наведено розмітку для цього самого зображення. Відтінками сірого позначено будівлі, дороги та інші рукотворні споруди. Зеленим позначені дерева, а блакитним — річки та інші водойми.

Рисунок 3.2 — Приклад зображення з тренувального набору даних

Рисунок 3.3 — Приклад розмітки зображення з тренувального набору даних

3.2. Використані інструменти

Для побудови системи розпізнавання об'єктів на аерофотознімках як основну мову програмування було використано мову Python, яка є де-факто галузевим стандартом для створення подібних систем Вона є простою у використанні, має простий синтаксис, є простою в читанні та надає широкі можливості для реалізації рішень в різних предметних областях. Мова є дуже популярною, що означає наявність[10] великої кількості бібліотек для вирішення різних задач таких як аналіз даних, автоматизація процесів, робота із зображеннями, створення HTTP-серверів та багато іншого. Програми на Python

легко та швидко відлагоджувати через наявність потужних інструментів відлагодження та інтерпретовану природу мови. Також варто зазначити, що при виникненні проблеми буде простіше знайти її рішення, тому що дуже ймовірно що інші розробники вже стикались з подібною проблемою.

Зображення тренувального набору перед навчанням мережі необхідно особливим чином підготувати, щоб привести їх у формат з яким вона може працювати. Для цього було обрано бібліотеку OpenCV що є однією з найбільших та найбільш підтримуваних бібліотек роботи із зображеннями, комп'ютерного зору та великою кількістю інших алгортимів. OpenCV доступна для ряду найпопулярніших мов програмування, що тільки підвищує її актуальність.

Для побудови моделі нейромережі було обрано інтерфейс Keras з бекендом Tensorflow. Keras є бібліотекою з відкритим кодом, що надає велику кількість реалізацій типових конструкцій нейромереж, таких як шари(в тому числі згорткові), активаційні функції, оптимізатори. Keras є одним з найпопулярніших фреймворків для роботи з нейромережами.

3.3. Реалізація нейромережі мовою Python

Основними шарами в мережі архітектури є згорткові, обрана архітектура має в своєму складі загалом 14 згорткових шарів. Після послідовності згорткових шарів в енкодері слідує шар субдискртетизації, таких шарів в мережі є 2, тобто розмір карти ознак зменшується двічі. В декодері навпаки відбувається збільшення карти ознак двічі, тобто в мережі присутні 2 шари збільшення розмірності. В лістингу 3.1 наведено приклад створення згорткового шару в Keras.

Лістинг 3.1

```
x = Conv2D (64, (3, 3), activation='relu', padding='same', kernel_initializer = tf.keras.initializers.he_normal(seed= 23))(x)
```

Клас Conv2D являє собою тип що представляє собою шар двовимірної згортки. Першим параметром є кількість каналів згортки, в даному випадку 64.

Після кожної субдискретизації кількість фільтрів буде збільшуватись вдвічі. Кортеж (3, 3) задає розміри фільтра згортки, тобто даний фільтр буде мати висоту і ширину в 3 одиниці. Функцією активації виступає випрямлений лінійний вузол, також відомий як “ReLU”. Для запобігання втрати інформації на краях зображення, що спричинено особливостями роботи згорткового фільтра, застосовується так званний зсув, або padding, суть якого полягає у вставці на краях зображення нульових значень. Візуалізацію зсуву представлено на рисунку 3.4.

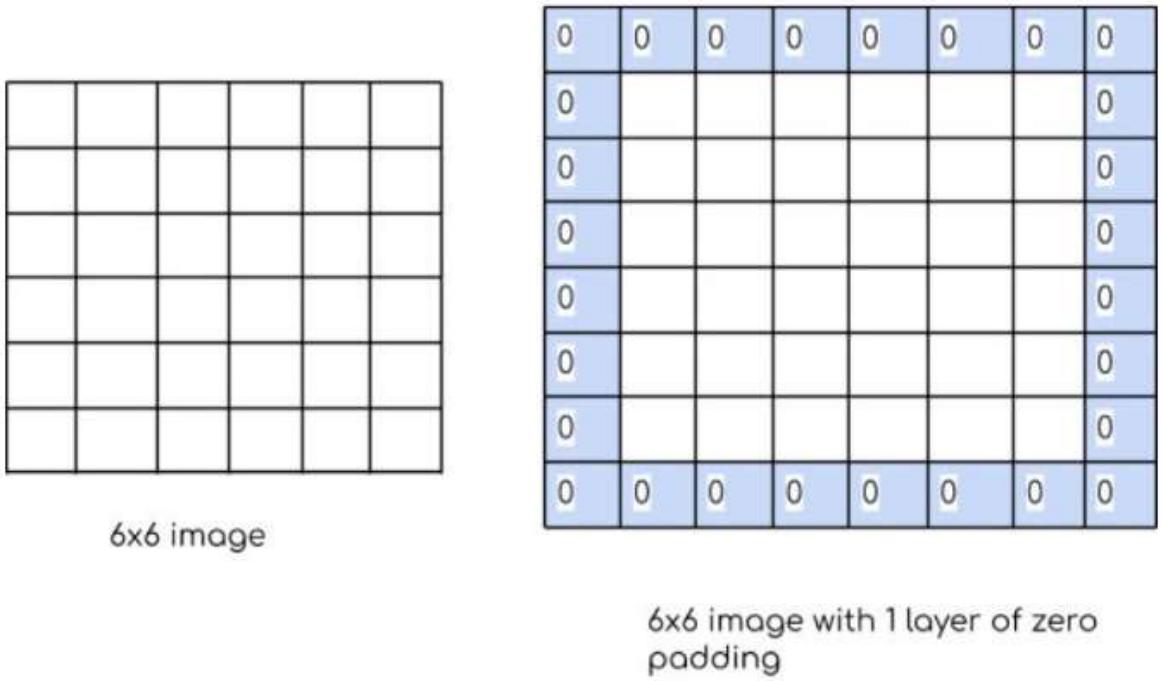


Рисунок 3.4 — Візуалізація зсуву

Режим зсуву регулюється параметром «padding» що має значення «same». Це призведе до розширення карти ознак по краям значеннями, що не впливають на результат роботи мережі, найчастіше використовується значення 0.

Після згорткового шару можна додати шар так званої Batch Normalization, для приведення вихідних значень до значень з центром в 0 і дисперсією в 1. Така зміна проводити навчання мережі швидше та має позитивний ефект на точність роботи мережі. Автори методики Batch Normalization досягли в 15 разів більшої

швидкості навчання при збереженні точності на деяких типах мереж[13]. На лістингу 3.2 приведено приклад створення шару Batch Normalization в Keras.

Лістинг 3.2

```
x = BatchNormalization()(x)
```

Також для того щоб запобігти оверфітингу мережі та ще більше підвищити швидкість навчання мережі можливо застосувати шар так званого виключення, або дропауту. Його суть полягає в тому що з деякою(зазвичай явно заданою) ймовірністю на кожному етапі тренування деякі випадково вибрані нейрони мережі не приймають в ньому участі, на наступному етапі ці нейрони вмикаються, з вагами які були знайдені раніше, до попереднього етапу навчання. Дропаут веде до невеликої втрати інформації мережею, що призводить до того, що зв’язки між різними парами нейронами дещо послаблюються. Так досягається створення більше надійних ознак розпізнавання, що робить мережу більш точною при роботі з даними, до яких вона не мала доступу під час навчання. На лістингу 3.3 наведено приклад створення шару виключення в Keras. Параметром тут є ймовірність з якою потрібно виключити нейрони з процесу навчання в межах від 0 до 1, тут використано значення 0.5.

Лістинг 3.3

```
x = Dropout(0.5)(x)
```

Одними з основних шарів SegNet є шари субдискретизації, які приймають на вході карту ознак та повертають її зменшену копію. На лістингу 3.4 наведено код створення шару субдискретизації в Keras.

Лістинг 3.4

```
x = MaxPooling2D((2, 2), strides=(2, 2))(x)
```

Параметрами тут є розмір вікна субдискретизації, тобто тут воно буде мати розмір 2 на 2 пікселі вхідного зображення. Параметр strides визначає крок вікна згортки на кожній ітерації процесу субдискретизації. Розмір кроку є таким самим як розмір вікна, тобто на першому кроці буде оброблено одну ділянку зображення розміром 2 на 2 пікселі, для наступного кроку вікно зміститься на 2

пікселі праворуч і проведе операцію субдискретизації для наступних 4 пікселів. Коли поточний рядок пікселів висотою в 2 пікселі буде оброблено, почнеться оброблюватись наступний. Цей процес буде повторюватись до закінчення обробки всього зображення. На рисунку 3.5 наведено візуалізацію процесу субдискретизації для матриці розміром 4 на 4, з розміром вікна і його кроком 2 на 2, що співпадає з параметрами вікна, застосованого в мережі.

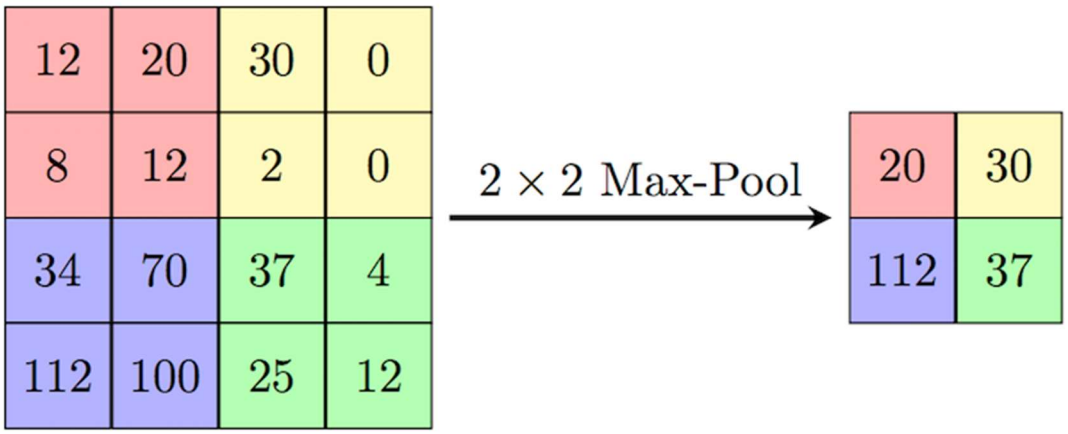


Рисунок 3.5 — Візуалізація процесу субдискретизації

В декодері мережі відбувається операція, протилежна субдискретизація — операція збільшення розмірності карти ознак. В лістингу 3.5 показано як створити шар збільшення дискретизації. Параметром тут є кортеж, значення якого вказують в скільки разів потрібно збільшити розмір вхідної матриці.

Лістинг 3.5

```
x = UpSampling2D(size=(2, 2))(x)
```

Тут буде відбуватись збільшення в 2 рази по обох осях зображення. В шарі субдискретизації в нас відбувалось зменшення вхідної матриці в 2 рази, в шарі збільшення розмірності матриця збільшується також в 2 рази, тобто в результаті після проходження шару субдискретизації і шару збільшення розмірності розмір зображення залишається незмінним. На рисунку 3.6 показано як відбувається збільшення розмірності матриці в 2 рази.

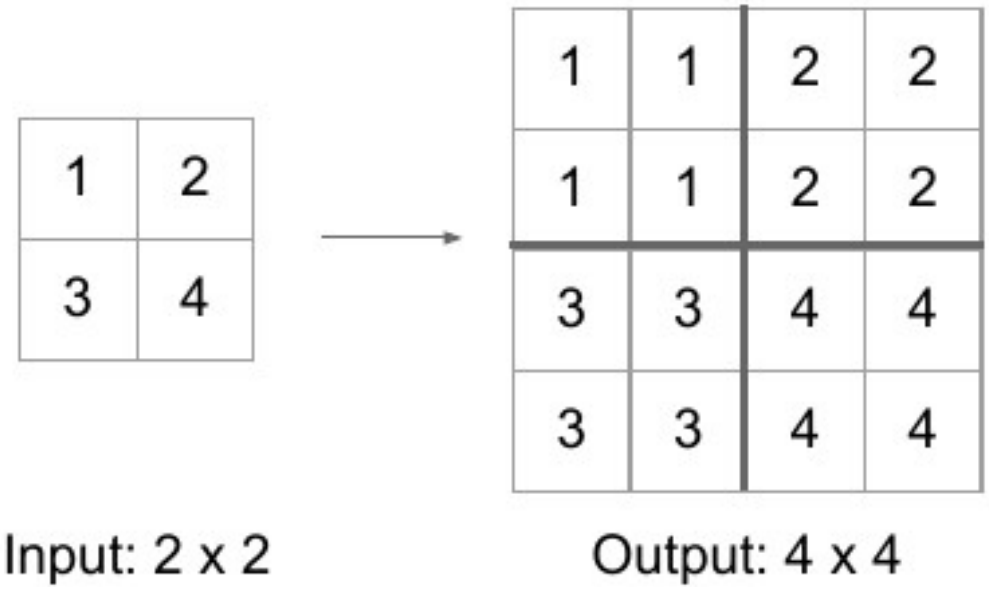


Рисунок 3.6 — Візуалізація операції збільшення розмірності

3.4. Навчання моделі

Процес навчання моделі було проведено на комп’ютері з наступними характеристиками:

— Центральний процесор — Intel i5-7600k;

— Оперативна пам’ять — 16ГБ DDR4;

— Відеокарта — Geforce GTX1060 6GB;

Для навчання можливо використовувати обчислювальні ресурсу центрального процесора, але більш ефективно проводити навчання з використанням графічного процесора з використанням набору інструментів CUDA[11]. При використанні відеокартии швидкість навчання може зрости в декілька, а в деяких випадках і в десятки разів[9]. Тому основну роль в процесі навчання відіграє обрана відеокарта. Аналіз зображень на готовій моделі також краще проводити з використанням відеокарти.

Використана модель не є моделлю нового покоління і є відеокартою з середніми характетистиками. Згідно бенчмарка Geekbench 5 GTX 1060 має[12] швидкодію при використанні CUDA на рівні 33740 очок. Найбільш продуктивними за результатами бенчамарку є моделі Nvidia GeForce RTX 3090 з результатом в 238937 очок, та Nvidia A100-SXM4-40GB з 238424 очками.

Відповідно і ціна подібних графічних процесорів є набагато більшою. На листопад 2021 року ціна RTX 3090 в залежності від модифікації складає близько 110 тисяч гривень, а A100-SXM4-40GB коштує від 250 тисяч гривень. Використану відеокарту було придбано в 2017 році за ціну близько 10 тисяч гривень.

Навчання проводилось на протязі 18 годин та зайняло 100 епох, динаміку зростання точності роботи моделі наведено на таблиці 3.2.

Таблиця 3.2 — Динаміка навчання моделі

Епоха	Індекс Жаккара на тренувальних даних	Індекс Жаккара на тестових даних
10	0,124	0,06
20	0,265	0,213
30	0,305	0,255
40	0,319	0,279
50	0,351	0,331
60	0,354	0,342
70	0,368	0,337
80	0,377	0,349
90	0,386	0,353
100	0,395	0,372

На рисунку 3.7 наведено графік залежності індексу Жаккара на тренувальних і тестових даних під час різних епох навчання. З нього можна бачити що точність роботи моделі швидко зростає на початку процесу навчання і поступово зменшується на пізніших епохах.

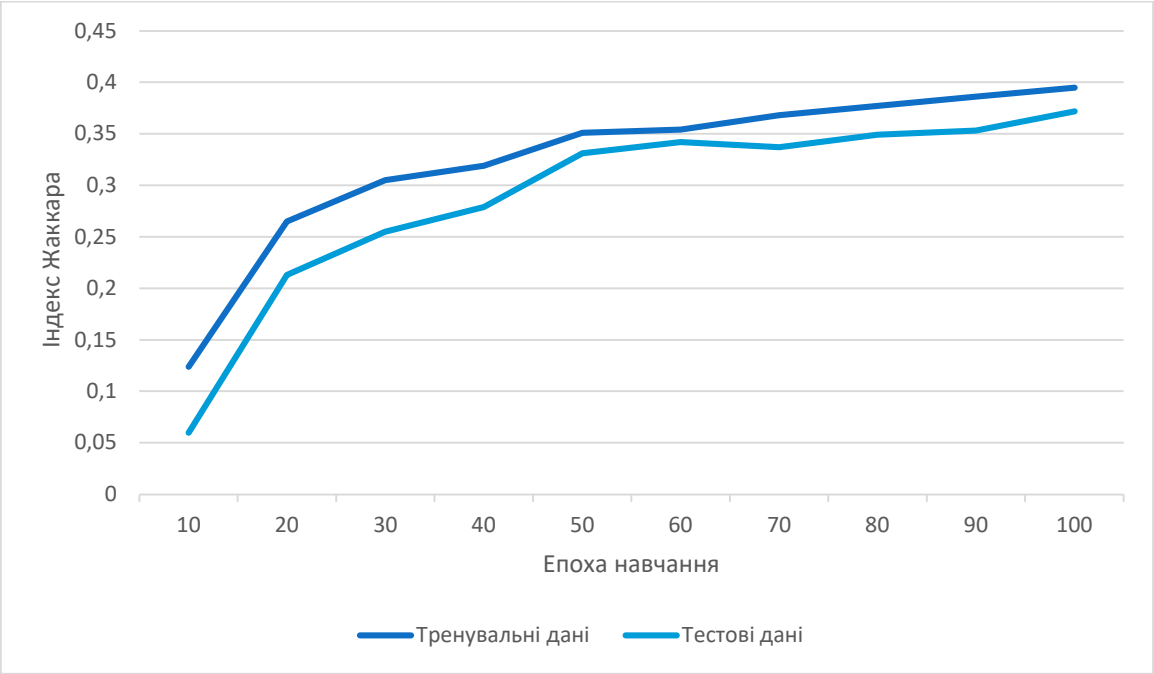


Рисунок 3.7 — Залежність індексу Жаккара на тренувальних і тестових даних в залежності від епохи навчання

Це є ознакою того що значення ваг нейронів мережі стабілізувались і подальше навчання не призведе до значного підвищення точності роботи і може навіть призвести до оверфітингу моделі, тобто ситуації коли модель є занадто пристосованою для роботи з тренувальними даними, а на даних які модель раніше не зустрічала буде працювати гірше.

3.5. Створення мікросервісу розпізнавання

Для реалізації взаємодії з системою розпізнавання об'єктів, було обрано варіант реалізації HTTP-серверу, який буде надавати функціонал завантаження зображень для обробки, їх проводити аналіз, та повертати результати аналізу у вигляді об'єктів JSON. Дане рішення дозволить надати системі широкі можливості для горизонтального масштабування, адже якщо обчислювальних ресурсів одного серверу є недостатньо для обробки даних з необхідною пропускною здатністю, то можна просто розгорнути ще один або декілька серверів, які будуть проводити обробку паралельно з першим. Для реалізації HTTP-серверу було обрано фреймворк Flask.

Flask — це мінімалістичний фреймворк призначений для створення веб-додатків, для мови Python. Його класифікують як мікрофреймворк, тому що він надає тільки найбільш базові компоненти веб-додатків. Наприклад функціонал роботи з базою даних, різні типи валідацій та інші необхідні компоненти звичайного веб-додатку не включаються в Flask, але в ньому існує можливість додавати розширення, які дозволяють використовувати в Flask додатковий функціонал. Це дозволяє встановлювати тільки необхідні компоненти для реалізації потрібного функціоналу, що дозволяє використовувати менше системних ресурсів, таких як оперативна пам'ять або пам'ять на диску, та зменшити навантаження на центральний процесор сервера.

Сервер розпізнавання містить одну кінцеву точку — /encoding. Для того щоб створити кінцеву точку в Flask, необхідно створити екземпляр серверу, створити методи, що будуть реалізовувати функціонал кінцевих точок, помітити їх декоратором @app.route з вказанням налаштувань кінцевої точки, та запустити зконфігурований екземпляр серверу. Приклад налаштування та запуску серверу наведено в лістингу 3.6.

Лістинг 3.6 — Код сервера з використанням Flask.

```
app = Flask(__name__)

@app.route('/encoding', methods=['POST'])
def predict_image():
    … # код функції розпізнавання

if __name__ == "__main__":
    … # код ініціалізації серверу(завантаження моделі, встановлення
необхідних параметрів)
app.run(host='127.0.0.1', port=3000, debug=True)
```

З лістингу можна побачити що було створено одну кінцеву точку яка знаходиться за адресою http://<адреса сервера>:<порт сервера>/encoding. Ця кінцева точка має HTTP метод POST. Цей вибір зумовлений тим що POST запит

дозволяє передавати на сервер бінарні файли, якими в нашому випадку є зображення.

Також необхідно відмітити, що даний код запустить сервер не на порту HTTP за замовчуванням яким є порт 80, а на порту 3000.

Для запуску аналізу зображень необхідно виконати запит на кінцеву точку /encoding з передачею зображень, які необхідно проаналізувати. Відповіддю на запит буде JSON масив, що містить по одному JSON об’єкту на кожен клас розпізнавання, тобто в нашому випадку масив буде містити 10 елементів. Кожен з цих об’єктів містить два поля: “Type”, що вказує на тип об’єкту, та “PolygonsWKT” що містить інформацію про розпізнані об’єкти в форматі WKT(Well-known text). WKT дозволяє працювати з різними системами координат[17]. Приклад опису об’єктів через формат WKT наведено на рисунку 3.8.

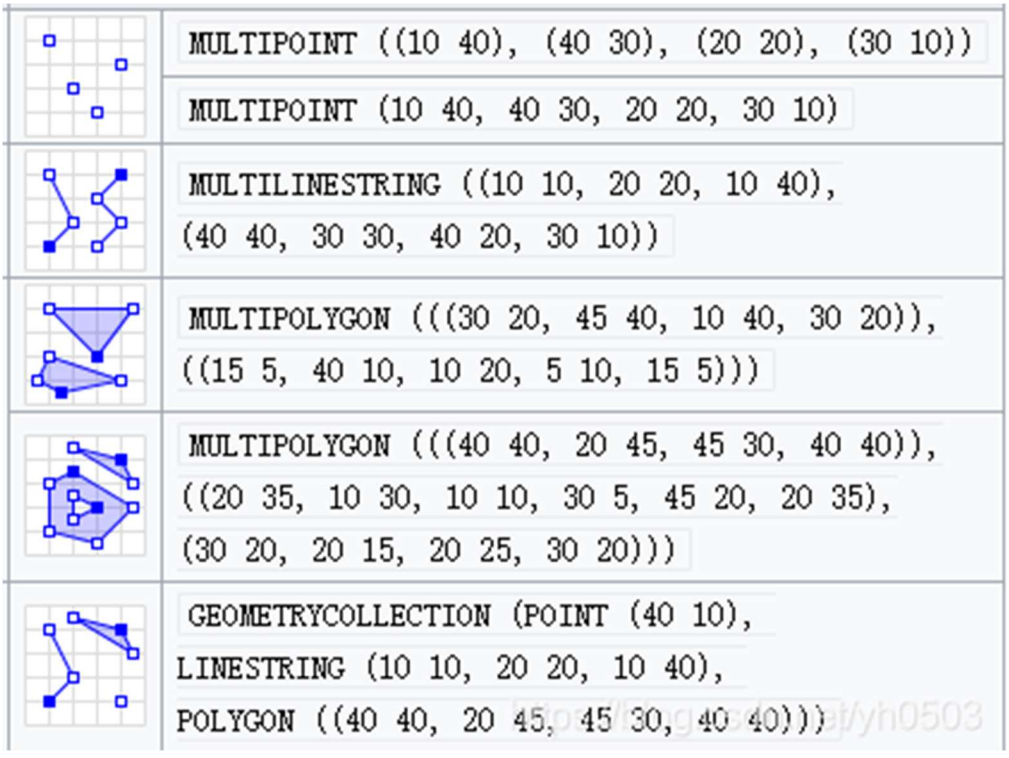


Рисунок 3.8 — Приклад опису геометричних об’єктів засобами WKT

В нашому випадку використовується система з початком координат в лівій нижній точці зображення, відповідно це буде точка з координатами (0 -1), x = 1

для точок на правому краю зображення, а y = 0 для точок на верхньому краю зображення.

В лістингу 3.7 наведено приклад фрагменту відповіді на запит розпізнавання зображення. Тут вказані результати для трьох типів об’єктів: будівель, малих архітектурних форм та для доріг, але реальний JSON відповіді буде завжди містити інформацію для всіх класів, навіть якщо об’єктів цих класів не було знайдено на зображенні.

Лістинг 3.7 — Приклад фрагменту відповіді на запит розпізнавання зображення


```
[
  {
    "Type": "Building",
    "PolygonsWKT": "GEOMETRYCOLLECTION EMPTY"
  },
  {
    "Type": "Structure",
    "PolygonsWKT": "GEOMETRYCOLLECTION EMPTY"
  },
  {
    "Type": "Road",
    "PolygonsWKT": "MULTIPOLYGON (((0.0080080881863371 -
0.008408175293387, 0.0079999197885139 0.008421675885066, 0.00800863818633
71 0.0084216718875066, 0.0080085381863371 -0.0084081712933387)))"
  }…
```


В лістингу можна бачити що на аналізованому зображенні не було знайдено будівель, тому поле "PolygonsWKT" має значення "GEOMETRYCOLLECTION EMPTY" що свідчить про відсутність об’єктів класу, але було виявлено дороги, тому значення поля містить параметри цих об’єктів.

В цьому розділі було розглянуто та обрано інструменти реалізації задачі. Основною мовою програмування було обрано мову Python. Було проведено навчання моделі з використанням Keras і Tensorflow. Для надання інтерфейсу роботи з функціональністю розпізнавання об’єктів було створено HTTP сервіс з використанням бібілотеки Flask. Сервіс дозволяє завантажити зображення, які необхідно обробити, провести аналіз засобами нейромережі і повернути результат розпізнавання у вигляді об’єктної нотації JSON та нотації опису геометричних об’єктів WKT.

4 ЕКСПЕРИМЕНТАЛЬНА ПЕРЕВІРКА СТВОРЕНОЇ КОМП'ЮТЕРНОЇ СИСТЕМИ ТА РОЗРОБКА ПРОГРАМНОЇ ДОКУМЕНТАЦІЇ

4.1. Результати розпізнавання

Для тестування роботи системи було вибрано зображення з набору даних Dstl Satellite Imagery Feature Detection які не були задіяні в процесі навчання мережі, щоб показати роботу системи з даними, з якими система ніколи не зустрічалась.

На рисунку 4.1 зображено результати результат роботи мережі при розпізнаванні будівель та інших рукотворних споруд. Зліва зображено результат обробки зображення, справа — початкове RGB зображення. Можна бачити що будівлі розпізнаються з гарною точністю, чітко виділені контури і форма будівель. Якщо взяти інші структури, то через їх менший розмір вони розпізнаються з гіршою якістю, але тим не менш задовільно, також є можливим донавчання мережі з використанням спеціалізованого набору даних.

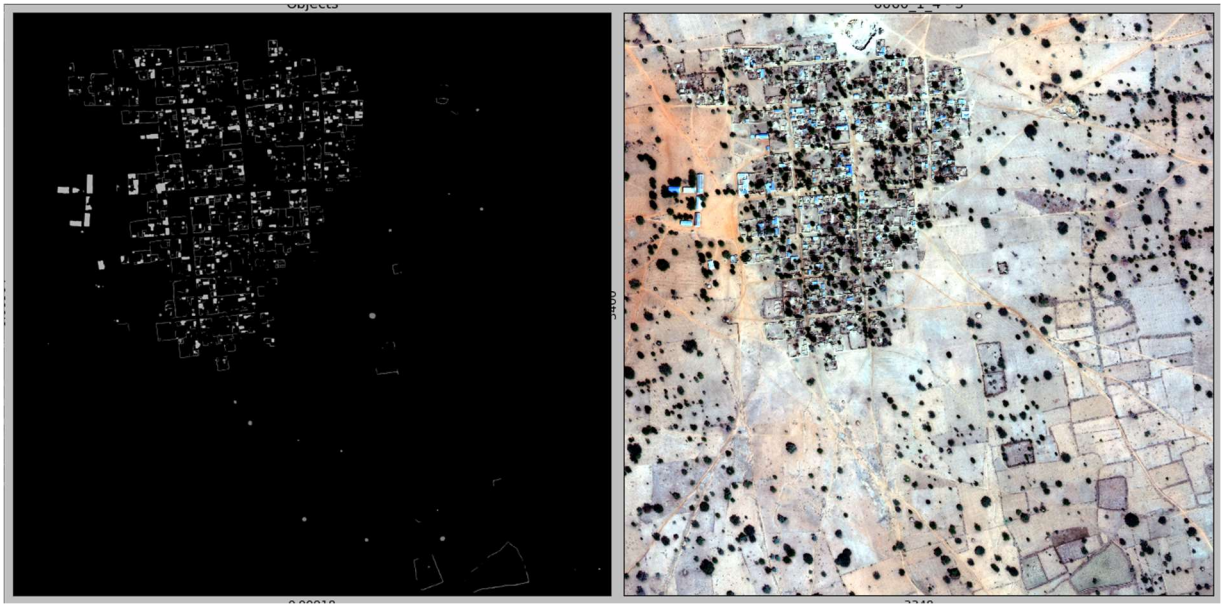


Рисунок 4.1 — Результат розпізнавання будівель і інших рукотворних споруд

На рисунку 4.2 зображено результати розпізнавання доріг та стежок. Дороги виділено світлим відтінком сірого, стежки — більш темним. Якість

розпізнавання доріг є доволі високою, стежки в загальному є менш помітними і можуть розпізнаватись не так якісно.

На рисунку 4.3 зображено результати розпізнавання ділянок сількогосподарського призначення. Подібні ділянки мають великий розмір і чіткі характерні особливості, тому їх не важко розпізнавати на знімках.

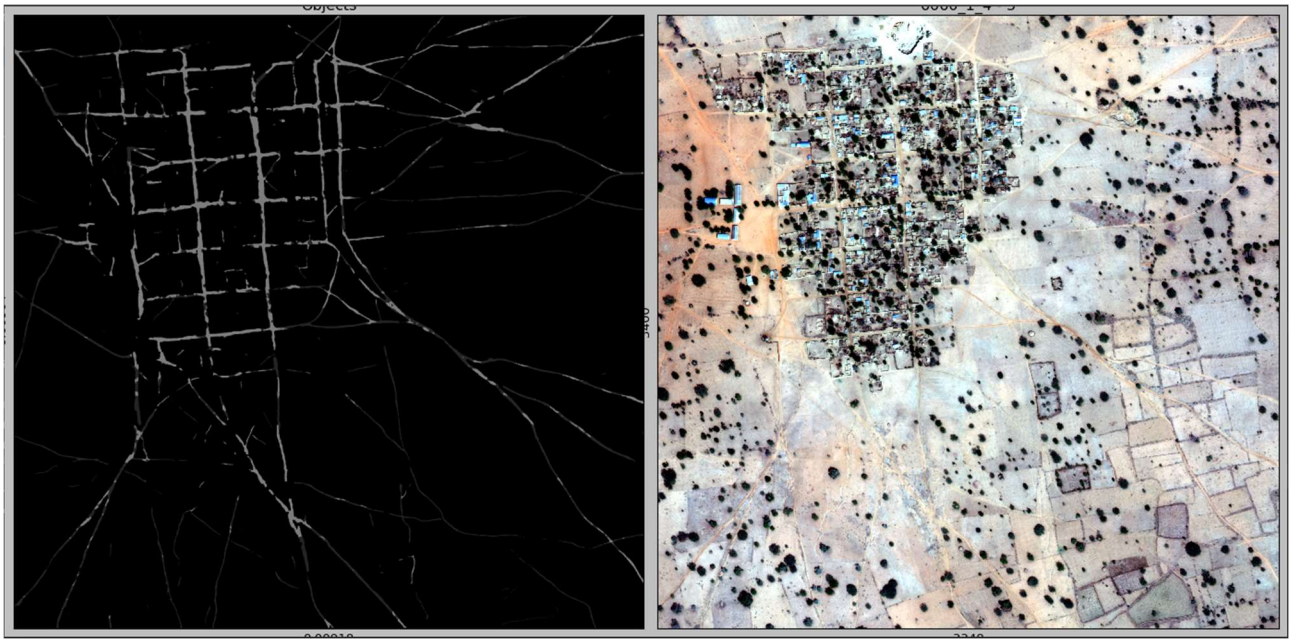


Рисунок 4.2 — Результат розпізнавання доріг

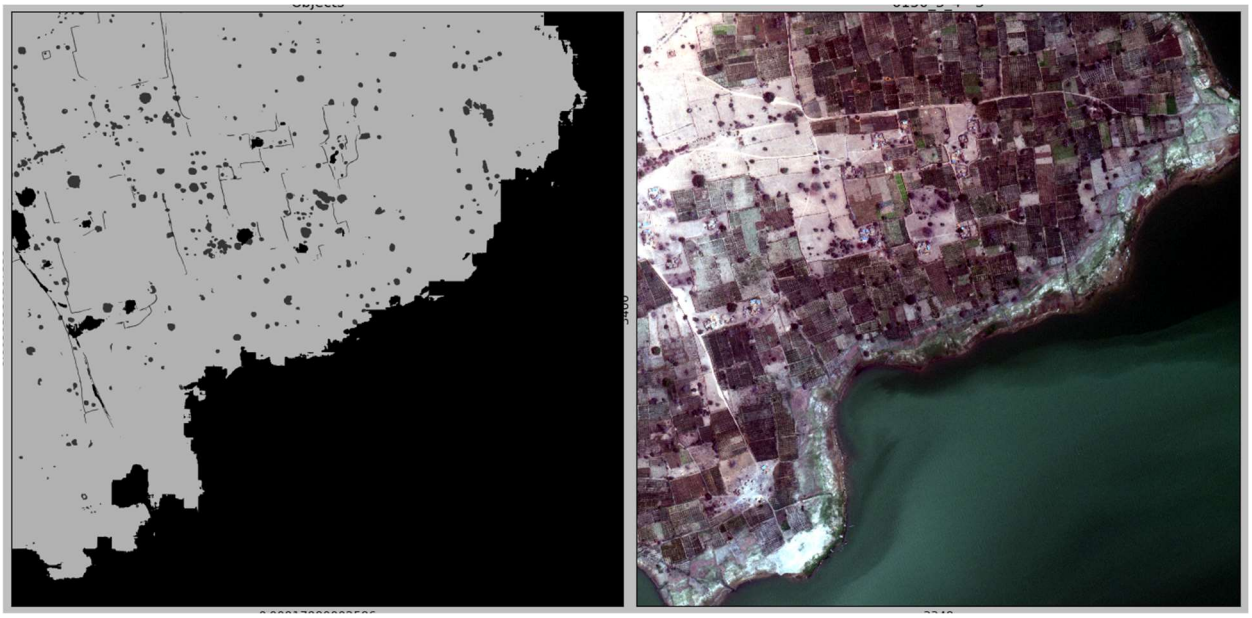


Рисунок 4.3 — Результат розпізнавання ділянок сільськогосподарського використання

На рисунку 4.4 зображено результат розпізнавання поодиноких дерев, а на рисунку 4.5 зображено результат розпізнавання лісових масивів. Окремі дерева нейромережа розпізнає з високою точністю, а у випадку густих лісових масивів не є можливим виділяти окремі дерева, тому вони розпізнаються як одне ціле. В залежності від області застосування це може розцінюватись як недолік системи, але для більшості випадків така особливість роботи не становить якихось проблем.

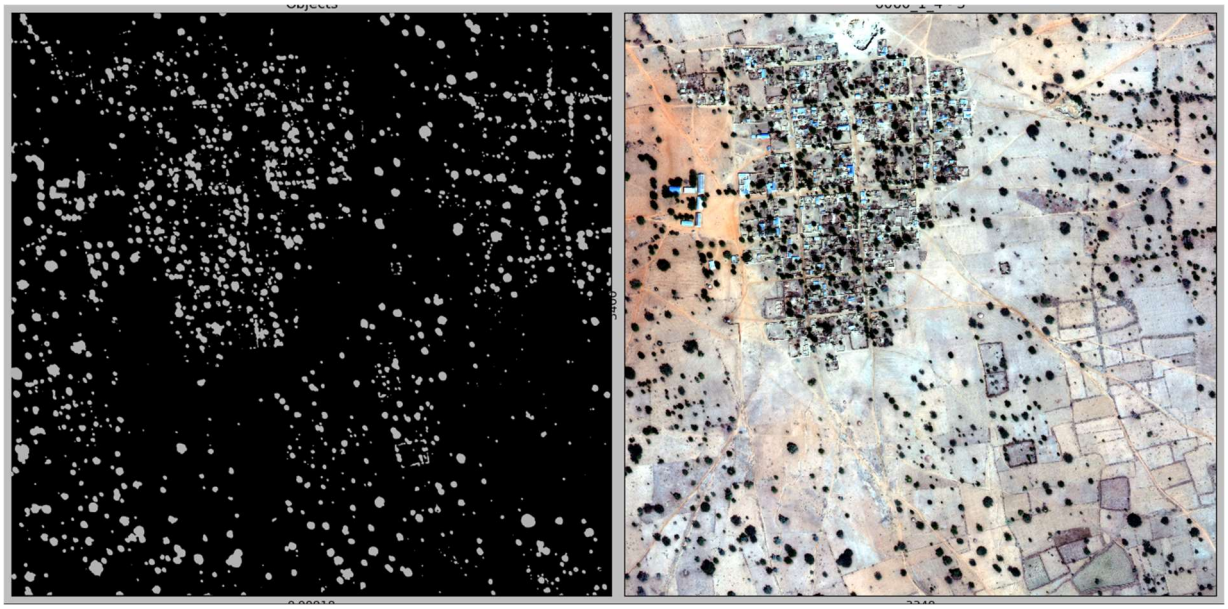


Рисунок 4.4 — Результат розпізнавання окремих дерев

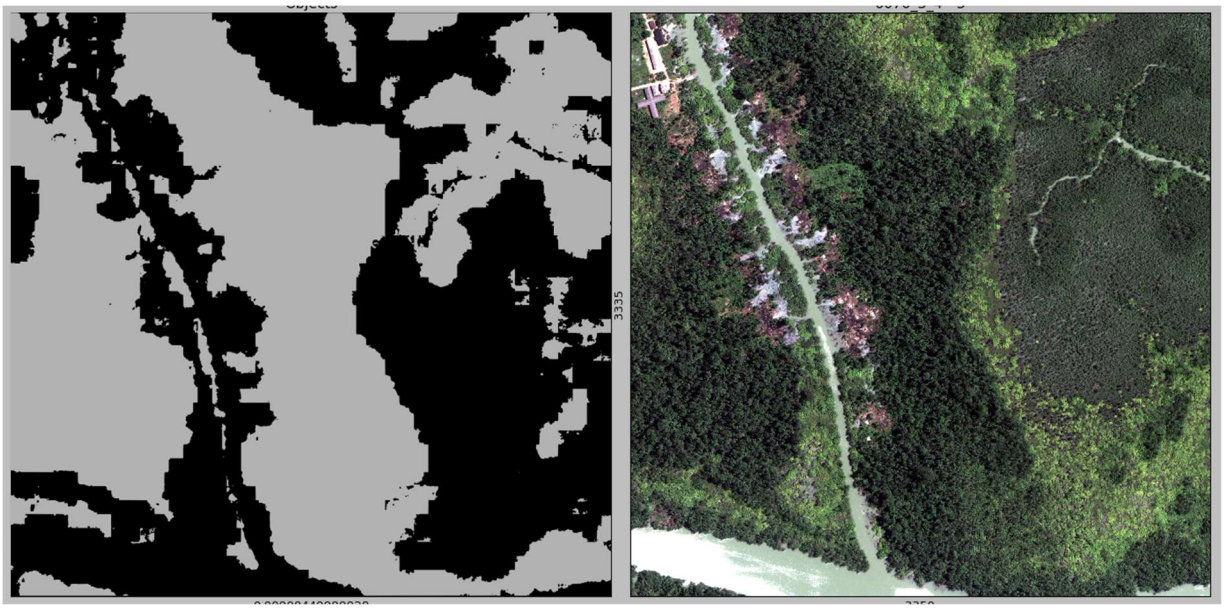


Рисунок 4.5 — Результат розпізнавання лісових масивів

Останньою категорією розпізнаваних об'єктів є водойми. На рисунку 4.6 наведено приклад розпізнавання струмка, що впадає в більшу річку, а на рисунку 4.7 наведено приклад розпізнавання великої водойми.

Можна бачити що річки і струмки розпізнаються якісно, мережа розпізнає невеликі притоки і інші малі водойми. Великі водойми теж розпізнаються якісно, є невеликі області переходу від суші до води, де мережа справилась не так якісно, але загалом показала гарний результат.

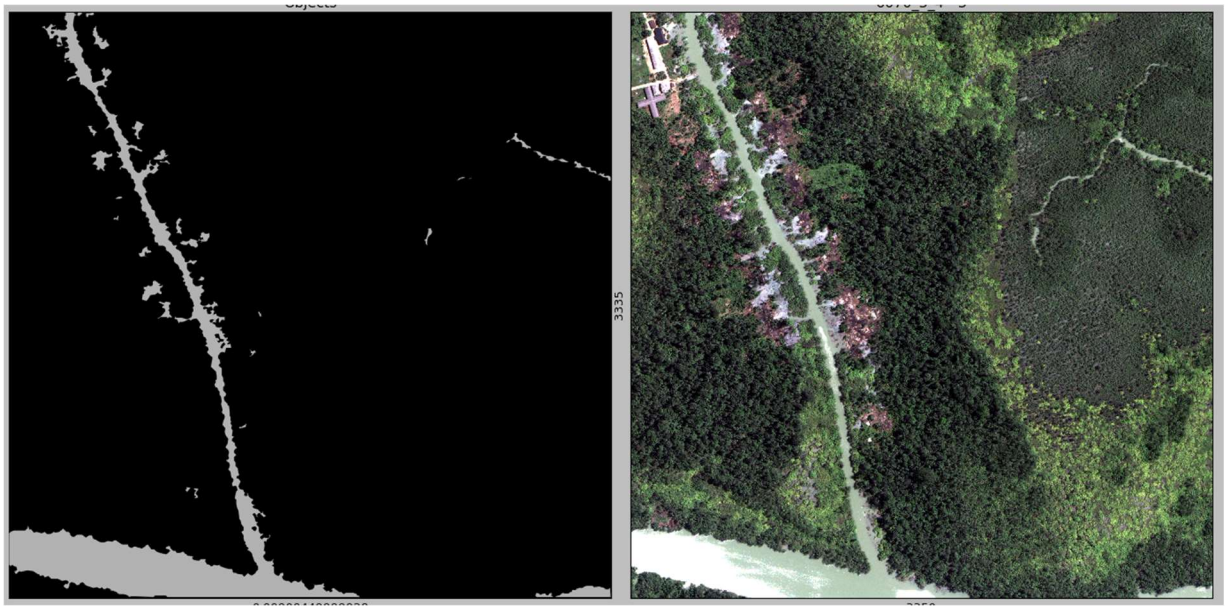


Рисунок 4.6 — Результат розпізнавання річок

Рисунок 4.7 — Результат розпізнавання великих водойм

Підсумувавши, можна зробити висновок, що створена нейромережа показує добрі результати розпізнавання на різних типах об'єктів і має потенціал до покращення характеристик точності і швидкодії розпізнавання.

Також важливо мати на увазі що навчання нейромережі і показані результати були отримані з використанням знімків з ідеальними погодніми умовами, тому що наприклад взимку буде майже неможливо розпізнавати нечищені дороги, важко розпізнвати будівлі. Розпізнвання водойм взимку теж може бути ускладнено. В період з середини осені до середини весни дуже важко розпізнавати дерева, адже основною ознакою за якою нейромережа розпізнає дерева — це крона дерева з листям. Точність розпізнавання ділянок сільськогосподарського значення в залежності від обраного набору тренувальних даних теж може помітно відрізнятись.

4.2. Тестування HTTP-сервісу розпізнавання

Для тестування створеного сервера було використано програму Postman яка дозволяє відправляти HTTP-запити з довільними параметрами. На рисунку 4.8 наведено інтерфейс створення запиту з вказанням адреси, методу та параметрів тестового запиту. Параметрами запиту є три зображення, зроблені в різних частотних спектрах. На рисунку 4.9 наведено дані розпізнавання що були отримані в результаті виконання запиту.

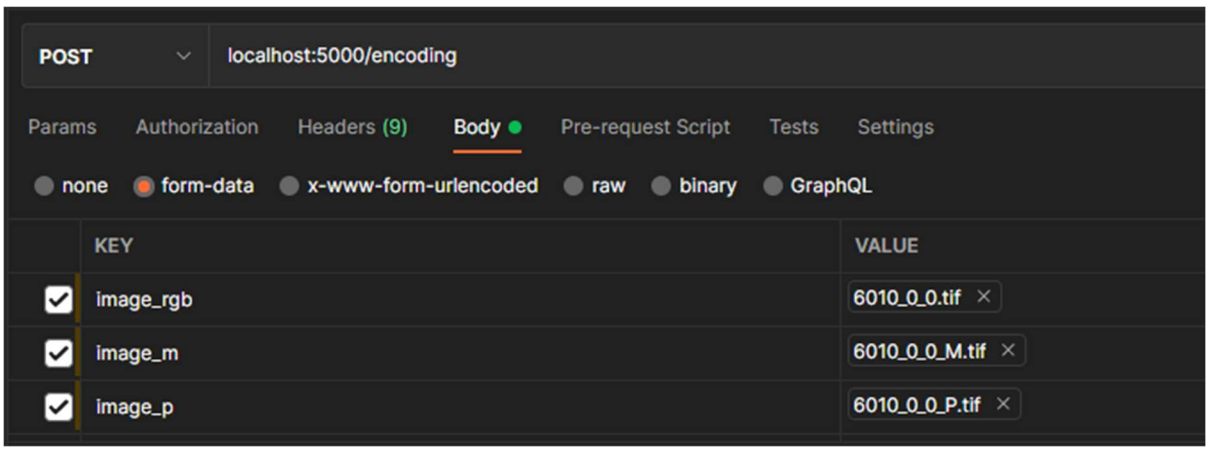
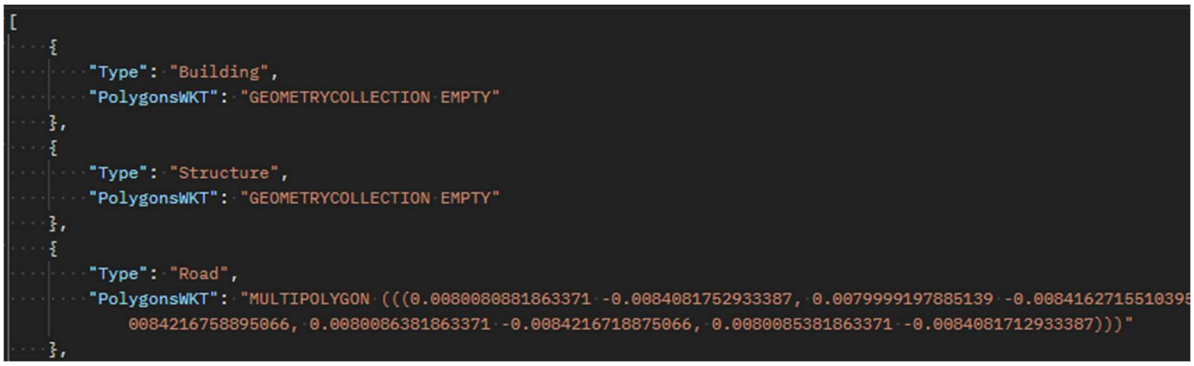


Рисунок 4.8 — Створення запиту в Postman



```
[
    {
        "Type": "Building",
        "PolygonsWKT": "GEOMETRYCOLLECTION EMPTY"
    },
    {
        "Type": "Structure",
        "PolygonsWKT": "GEOMETRYCOLLECTION EMPTY"
    },
    {
        "Type": "Road",
        "PolygonsWKT": "MULTIPOLYGON (((0.0080080881863371 -0.0084081752933387, 0.0079999197885139 -0.008416271551039
            0084216758895066, 0.0080086381863371 -0.0084216718875066, 0.0080085381863371 -0.0084081712933387)))"
    },
```


Рисунок 4.9 — Отримані в результаті виконання запиту дані

4.3. Інструкція запуску та системні вимоги системи

Системні вимоги системи розпізнавання:

— центральний процесор — Intel Core I5 6-го покоління або кращий;

— оперативна пам’ять — 4ГБ і більше, споживання сервісу — до 2 ГБ;

— відеоадаптер — Geforce GTX 10-ої серії або кращий;

— місце на жорсткому диску — 1ГБ;

Для запуску сервісу на комп’ютері необхідно мати встановленим інтерпретатор Python версії 3. Для самого запуску потрібно запустити файл recognition_service.exe для операційної системи Windows, або файл recognition_service для операційної системи Linux. Після запуску виконуваного файлу буде запущено екземпляр сервісу на порту за замовчуванням, який буде готовий для того щоб приймати запити на обробку зображень.

В даному розділі було проведено тестування роботи нейромережі та HTTP сервісу. Загалом мережа показала хороші результати розпізнавання, для деяких класів об’єктів результати є кращими ніж для інших, але загальна точність роботи є високою. Також надано інструкцію для запуску сервісу розпізнавання.

5 ЕКОНОМІЧНА ЧАСТИНА

Науково-технічна розробка має право на існування та впровадження, якщо вона відповідає вимогам часу, як в напрямку науково-технічного прогресу та і в плані економіки. Тому для науково-дослідної роботи необхідно оцінювати економічну ефективність результатів виконаної роботи.

Магістерська кваліфікаційна робота за темою «Система розпізнавання та класифікації об’єктів на аерофотознімках з використанням нейромереж» відноситься до науково-технічних робіт, які орієнтовані на виведення на ринок (або рішення про виведення науково-технічної розробки на ринок може бути прийнято у процесі проведення самої роботи), тобто коли відбувається так звана комерціалізація науково-технічної розробки. Цей напрямок є пріоритетним, оскільки результатами розробки можуть користуватися інші споживачі, отримуючи при цьому певний економічний ефект. Але для цього потрібно знайти потенційного інвестора, який би взявся за реалізацію цього проекту і переконати його в економічній доцільності такого кроку.

Для наведеного випадку нами мають бути виконані такі етапи робіт:

— проведено комерційний аудит науково-технічної розробки, тобто встановлення її науково-технічного рівня та комерційного потенціалу;

— розраховано витрати на здійснення науково-технічної розробки;

— розрахована економічна ефективність науково-технічної розробки у випадку її впровадження і комерціалізації потенційним інвестором і проведено обґрунтування економічної доцільності комерціалізації потенційним інвестором.

5.1. Проведення комерційного та технологічного аудиту науково-технічної розробки

Метою проведення комерційного і технологічного аудиту дослідження за темою «Система розпізнавання та класифікації об’єктів на аерофотознімках з використанням нейромереж» є оцінювання науково-технічного рівня та рівня

комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності.

Оцінювання науково-технічного рівня розробки та її комерційного потенціалу рекомендується здійснювати із застосуванням 5-ти бальної системи оцінювання за 12-ма критеріями, наведеними в табл. 5.1[18].

Таблиця 5.1 — Рекомендовані критерії оцінювання науково-технічного рівня і комерційного потенціалу розробки та бальна оцінка

Бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Технічна здійсненність концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено працездатність продукту в реальних умовах
Ринкові переваги (недоліки)					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в	Технічні та споживчі влас-тивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в	Технічні та споживчі властивості продукту значно кращі, ніж в
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає

Закінчення таблиці 5.1

Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуютьс я у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь- які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання науково-технічного рівня та комерційного потенціалу науково-технічної розробки потрібно звести до таблиці.

Таблиця 5.2 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки експертами

Критерії	Експерт (ПІБ, посада)		
	1	2	3
	Бали:		
1. Технічна здійсненність концепції	4	5	5
2. Ринкові переваги (наявність аналогів)	3	2	2
3. Ринкові переваги (ціна продукту)	3	4	3
4. Ринкові переваги (технічні властивості)	3	3	3
5. Ринкові переваги (експлуатаційні витрати)	1	2	2
6. Ринкові перспективи (розмір ринку)	3	3	3
7. Ринкові перспективи (конкуренція)	4	4	4
8. Практична здійсненність (наявність фахівців)	4	5	4
9. Практична здійсненність (наявність фінансів)	3	4	4
10. Практична здійсненність (необхідність нових матеріалів)	4	4	4
11. Практична здійсненність (термін реалізації)	4	4	4
12. Практична здійсненність (розробка документів)	4	4	3
Сума балів	40	44	41
Середньоарифметична сума балів $СБ_c$	41,7		

За результатами розрахунків, наведених в таблиці 5.2, зробимо висновок щодо науково-технічного рівня і рівня комерційного потенціалу розробки. При цьому використаємо рекомендації, наведені в табл. 5.3 [18].

Таблиця 5.3 – Науково-технічні рівні та комерційні потенціали розробки

Середньоарифметична сума балів СБ ,розрахована на основі висновків експертів	Науково-технічний рівень та комерційний потенціал розробки
41…48	Високий
31…40	Вище середнього
21…30	Середній
11…20	Нижче середнього
0…10	Низький

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Система розпізнавання та класифікації об’єктів на аерофотознімках з використанням нейромереж» становить 41,7 бала, що, відповідно до таблиці 4.3, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки високий).

5.2. Оцінювання рівня новизни розробки

Виводячи на ринок новинку виробник вважає, що тієї новизни, якою наділена нова розробка є достатньо для того, щоб вона була сприйнята споживачем як нова. Але це не завжди так, в силу того, що споживач і виробник неоднозначно визначать її рівень новизни. Тому доцільним є визначення рівня новизни розробки отриманої в результаті досліджень за темою «Система розпізнавання та класифікації об’єктів на аерофотознімках з використанням нейромереж».

Саме визначення рівня і ступеня інтегральної новизни є найбільш актуальним, оскільки її рівень визначає ступінь однакового позитивного сприйняття новизни розробки як виробником, так і споживачем, а отже і ринком в цілому, а це, у свою чергу, є гарантією того, що новинка знайде своє місце на ринку, користуватиметься попитом у споживачів і забезпечить відшкодування витрат, зазнаних товаровиробником під час розроблення та виробництва технічної розробки [19].

Рівень новизни нової продукції розраховуємо експертним методом шляхом протиставлення нової продукції та її аналогів, що існують в даний час на ринку, за чинниками що визначають її значення, в системі «краще-гірше». Рівень новизни встановлюємо відносно рівня аналога (або продукту, що досить близький до аналога).

Для визначення *i*-го виду новизни, застосуємо чинники, які впливають на її рівень. Кожен чинник *i*-го виду новизни розраховуємо в балах. Більша кількість набраних балів свідчить про більший рівень новизни. Для оцінювання рівня новизни використаємо думки експертів, які встановлюють визначені бали відповідним чинникам. Бал відповідності проставляється в діапазоні від -5 — значно гірше аналога до +5 — значно краще аналога. Результати попереднього оцінювання зведемо до відповідного листа оцінювання (таблиця 5.4).

Таблиця 5.4 – Лист оцінювання рівня новизни експертами

Види та чинники		Бали та експерти		
		Експерт 1	Експерт 2	Експерт 3
1		*2*	*3*	*4*
Споживча новизна	Питома вага 0,225	Максимальний бал $Б_{i\ max}$		25
1. Зміна поведінкових звичок споживача		4	3	3
2. Ступінь задоволення потреб і запитів		2	3	3
3. Спосіб задоволення потреби		2	2	2
4. Формування нової потреби		0	0	0
5. Формування нового споживача		0	0	0
Середній бал експертів $Б_{i\ отр}$		8		
Товарна новизна	Питома вага 0,217	Максимальний бал $Б_{i\ max}$		30
1. Параметричні зміни показників продукції				
1.1. Якісні		3	3	3
1.2. Технічні		4	4	3
1.3. Економічні		3	3	3
1.4. Сервісні		4	4	4
2. Якість продукції по відношенню до конкурентів		3	3	3
3. Функціональні зміни		3	3	3
Середній бал експертів $Б_{i\ отр}$		20		
Виробнича новизна	Питома вага 0,042	Максимальний бал $Б_{i\ max}$		25
1. Рівень унікальності товару для підприємства		5	5	5
2. Рівень унікальності для галузі		3	3	3
3. Рівень унікальності товару для країни		0	0	0
4. Зміна виробничої системи		4	4	4
5. Відносно існуючого асортименту		2	2	2
Середній бал експертів $Б_{i\ отр}$		14		
Прогресивна новизна	Питома вага 0,179	Максимальний бал $Б_{i\ max}$		25
1. Зміна технології виготовлення		4	4	4

Продовження таблиці 5.4

2. Рівень застосування нових компонентів і матеріалів		1	2	1
3. Зміна технологічного принципу дії виробу		1	2	1
4. Зміна конструктивного виконання		3	2	3
5. Рівень застосування інновацій		2	2	2
Середній бал експертів $Б_{i\ отр}$		11		
Ринкова новизна	Питома вага 0,12	Максимальний бал $Б_{i\ MAX}$		20
1. Новий виріб на новому ринку		0	0	0
2. Новий виріб на відомому ринку		2	2	2
3. Модернізований виріб		2	2	2
4. Нова модель		1	2	2
Середній бал експертів $Б_{i\ отр}$		6		
Екологічна новизна	Питома вага 0,035	Максимальний бал $Б_{i\ MAX}$		20
1. Рівень екологічної чистоти технології виробництва		5	5	5
2. Рівень впровадження мало- та безвідходних технологій		5	5	5
3.Рівень екологічно небезпечних режимів експлуатації продукції		5	5	5
4. Рівень забруднення навколишнього середовища		5	5	5
Середній бал експертів $Б_{i\ отр}$		20		
Соціальна новизна	Питома вага 0,036	Максимальний бал $Б_{i\ MAX}$		20
1. Використання нового товару приводить до покращення стану здоров’я нації		0	0	0
2. Використання нового товару приводить до зростання доходів населення		0	0	0
3. Виробництво нового товару приводить до збільшення (зменшення) кількості робочих місць на підприємстві		4	5	4

Закінчення таблиці 5.4

4. Виробництво нового товару приводить до підвищення кваліфікації персоналу		3	3	3
Середній бал експертів $Б_{i\ отр}$		7		
Маркетингова новизна	Питома вага 0,146	Максимальний бал $Б_{i\ MAX}$		20
1. Нові методи маркетингових досліджень		0	0	0
2. Вживання нових стратегій сегментації ринку		1	1	1
3. Вибір нової маркетингової стратегії обхвату і розвитку цільового сегмента		2	3	2
4. Побудова нових каналів збуту		2	2	2
Середній бал експертів $Б_{i\ отр}$		5		

Значення *i*-го виду новизни розрахуємо за формулою [19]:

$$I_i = \frac{Б_{i\,отр}}{Б_{i\,MAX}}, \tag{4.1}$$

де $Б_{i\ отр}$ – отримана кількість балів за шкалою оцінок чинників, що визначають *i*-й вид новизни;

$Б_{i\ MAX}$ – максимальна кількість балів, що може бути отримана за *i*-м видом новизни.

Загальний рівень інтегральної новизни розраховуємо шляхом перемноження отриманого значення *i*-го виду новизни на її вагомість, причому вагомість *i*-го виду новизни визначаємо експертним методом, за формулою [19]:

$$N_{інт} = \sum_{i}^{n} W_i \cdot I_i\ , \tag{4.2}$$

де $N_{інт}$ – рівень інтегральної (сукупної) новизни;

W_i – вагомість (питома вага) *i*-го виду новизни;

n – загальна кількість видів новизни.

$$N_{інт} = (0{,}225 \cdot 8/25)+(0{,}217 \cdot 20/30)+(0{,}042 \cdot 14/25)+(0{,}179 \cdot 11/25)+(0{,}12 \cdot 6/20)+ (0{,}035 \cdot 20/20)+(0{,}036 \cdot 7/20)+(0{,}146 \cdot 5/20) = 0{,}440.$$

Отримане значення інтегрального рівня новизни зіставляємо зі шкалою, що наведена в табл. 5.5[18].

Таблиця 5.5 – Рівні новизни нового товару та їхня характеристика

Рівні новизни товару	Значення інтегральної новизни	Характеристика товару	Вид нового товару
Найвища	1,00	Абсолютно новий товар	Новий товар, що наділений ознаками інноваційності (інноваційний товар)
Висока	0,8…0,99	Товар, який не має аналогів	
Значуща	0,6…0,79	Принципова зміна споживчих властивостей товару	
Достатня	0,4…0,59	Принципова технологічна модифікація товару	
Незначна	0,2…0,39	Кардинальна зміна параметрів	Новий товар
Помилкова	0,00…0,19	Малоістотна модифікація	

Згідно таблиці 5.5 розробка відповідає рівню при значенні інтегральної новизни 0,440 — достатня новизна; за характеристикою: принципова технологічна модифікація товару; вид розробки — новий товар, що наділений ознаками інноваційності (інноваційний товар).

5.3. Визначення рівня конкурентоспроможності розробки

В процесі визначення економічної ефективності науково-технічної розробки також доцільно провести прогноз рівня її конкурентоспроможності за сукупністю параметрів, що підлягають оцінюванню.

Одиничний параметричний індекс розраховуємо за формулою [18]:

$$q_i = \frac{P_i}{P_{баз\,i}}. \qquad (4.3)$$

де q_i – одиничний параметричний індекс, розрахований за *i*-м параметром;

P_i – значення *i*-го параметра виробу;

$P_{базі}$ – аналогічний параметр базового виробу-аналога, з яким проводиться порівняння.

Загальні технічні та економічні характеристики розробки представлено в таблиці 5.6.

Таблиця 5.6 – Основні техніко-економічні показники аналога та розробки, що проектується

Показники (параметри)	Одиниця вимірю-вання	Аналог	Проектований пристрій	Відношення параметрів нової розробки до аналога	Питома вага показника
Оперативна пам'ять	ГБ	4	2	2	0,3
Місце на диску	ГБ	3	1	3	0,12
Точність розпізнавання	%	85	90	1,06	0,15
Вимоги до CPU	бал	7	8	0,87	0,25
Захищеність БД від зовнішнього впливу (до 10 балів)	бал	7	7	1	0,18
Експлуатаційні витрати	грн	320	250	0,78	0,45
Ціна	грн	3000	2600	0,87	0,55

Нормативні параметри оцінюємо показником, який отримує одне з двох значень: 1 – пристрій відповідає нормам і стандартам; 0 – не відповідає.

Груповий показник конкурентоспроможності за нормативними параметрами розраховуємо як добуток частинних показників за кожним параметром за формулою [18]:

$$I_{НП} = \prod_{i=1}^{n} q_i \ , \qquad (4.4)$$

де $I_{нп}$ – загальний показник конкурентоспроможності за нормативними параметрами;

q_i – одиничний (частинний) показник за i-м нормативним параметром;

n – кількість нормативних параметрів, які підлягають оцінюванню.

За нормативними параметрами розроблюваний пристрій відповідає вимогам ДСТУ, тому $I_{нп} = 1$.

Значення групового параметричного індексу за технічними параметрами визначаємо з урахуванням вагомості (частки) кожного параметра [18]:

$$I_{ТП} = \sum_{i=1}^{n} q_i \cdot \alpha_i \,, \tag{4.5}$$

де $I_{тп}$ – груповий параметричний індекс за технічними показниками (порівняно з виробом-аналогом);

q_i – одиничний параметричний показник i-го параметра;

α_i – вагомість i-го параметричного показника, $\sum_{i=1}^{n} \alpha_i = 1$;

n – кількість технічних параметрів, за якими оцінюється конкурентоспроможність.

Проведемо аналіз параметрів згідно даних таблиці 5.4.

$$I_{тп} = 2 \cdot 0{,}3 + 3 \cdot 0{,}12 + 1{,}06 \cdot 0{,}15 + 0{,}87 \cdot 0{,}25 + 1 \cdot 0{,}18 = 1{,}52.$$

Груповий параметричний індекс за економічними параметрами розраховуємо за формулою[18]:

$$I_{ЕП} = \sum_{i=1}^{m} q_i \cdot \beta_i \,, \tag{4.6}$$

де $I_{ЕП}$ – груповий параметричний індекс за економічними показниками;

q_i – економічний параметр i-го виду;

β_i – частка *i*-го економічного параметра, $\sum_{i=1}^{m} \beta_i = 1$;

m – кількість економічних параметрів, за якими здійснюється оцінювання.

Проведемо аналіз параметрів згідно даних таблиці:

$$I_{ЕП}=0{,}78 \cdot 0{,}45 + 0{,}87 \cdot 0{,}55 = 0{,}83.$$

На основі групових параметричних індексів за нормативними, технічними та економічними показниками розрахуємо інтегральний показник конкурентоспроможності за формулою [18]:

$$K_{ІНТ} = I_{НП} \cdot \frac{I_{ТП}}{I_{ЕП}}, \quad (4.7)$$

$$K_{ІНТ} = 1 \cdot 1{,}52 / 0{,}83 = 1{,}83.$$

Інтегральний показник конкурентоспроможності $К_{ІНТ} > 1$, отже розробка переважає відомі аналоги за своїми техніко-економічними показниками.

5.4. Розрахунок витрат на проведення науково-дослідної роботи

Витрати, пов’язані з проведенням науково-дослідної роботи на тему «Система розпізнавання та класифікації об’єктів на аерофотознімках з використанням нейромереж», під час планування, обліку і калькулювання собівартості науково-дослідної роботи групуємо за відповідними статтями.

До статті «Витрати на оплату праці» належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно-технічним працівникам, конструкторам, технологам, креслярам, копіювальникам, лаборантам, робітникам, студентам, аспірантам та іншим працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифними ставками згідно з чинними в організаціях системами оплати праці.

Основна заробітна плата дослідників.

Витрати на основну заробітну плату дослідників ($З_о$) розраховуємо у відповідності до посадових окладів працівників, за формулою [18]:

$$З_о = \sum_{i=1}^{k} \frac{M_{ni} \cdot t_i}{T_p}, \quad (4.8)$$

де k – кількість посад дослідників залучених до процесу досліджень;

M_{ni} – місячний посадовий оклад конкретного дослідника, грн;

t_i – число днів роботи конкретного дослідника, дн.;

T_p – середнє число робочих днів в місяці, T_p=22 дні.

$$З_о = 13650{,}00 \cdot 56 / 22 = 34745{,}45 \; грн.$$

Проведені розрахунки зведемо до таблиці.

Таблиця 5.7 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Число днів роботи	Витрати на заробітну плату, грн
Керівник проекту	13650,00	620,45	56	34745,45
Інженер-розробник програмного забезпечення	12850,00	584,09	42	24531,82
Консультант (фахівець топограф)	11780,00	535,45	14	7496,36
Лаборант	6540,00	297,27	21	6242,73
Всього				73016,36

Витрати на основну заробітну плату робітників ($З_р$) за відповідними найменуваннями робіт НДР на тему «Система розпізнавання та класифікації

об’єктів на аерофотознімках з використанням нейромереж» розраховуємо за формулою:

$$З_р = \sum_{i=1}^{n} C_i \cdot t_i \,, \tag{4.9}$$

де C_i – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год;

t_i – час роботи робітника при виконанні визначеної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду C_i можна визначити за формулою:

$$C_i = \frac{M_M \cdot K_i \cdot K_c}{T_p \cdot t_{зм}} \,, \tag{4.10}$$

де M_M – розмір прожиткового мінімуму працездатної особи, або мінімальної місячної заробітної плати (в залежності від діючого законодавства), приймемо $Мм$=2379,00 грн;

K_i – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду (табл. Б.2, додаток Б) [18];

K_c – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих об’єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати.

T_p – середнє число робочих днів в місяці, приблизно T_p = 22 дн;

$t_{зм}$ – тривалість зміни, год.

$$C_1 = 2379{,}00 \cdot 1{,}10 \cdot 1{,}65 \,/\, (22 \cdot 8) = 24{,}53 \text{ } грн.$$

$$З_{р1} = 24{,}53 \cdot 9{,}20 = 225{,}71 \text{ } грн.$$

Таблиця 5.8 – Величина витрат на основну заробітну плату робітників

Найменування робіт	Тривалість роботи, год	Розряд роботи	Тарифний коефіцієнт	Погодинна тарифна ставка, грн	Величина оплати на робітника грн
Установка обладнання для розробки ПЗ	9,20	2	1,10	24,53	225,71
Підготовка робочого місця розробника програмного забезпечення	4,30	2	1,10	24,53	105,49
Інсталяція програмного забезпечення для розпізнавання зображення	5,60	5	1,70	37,92	212,33
Компіляція програмних блоків	6,30	4	1,50	33,45	210,76
Налагодження програмних блоків	4,80	6	2,00	44,61	214,11
Підбір тренувальних даних	25,00	2	1,10	24,53	613,34
Попереднє тренування нейромережі	15,00	2	1,10	24,53	368,00
Всього					1949,74

Додаткова заробітна плата дослідників та робітників

Додаткову заробітну плату розраховуємо як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_{дод} = (З_о + З_р) \cdot \frac{Н_{дод}}{100\%}, \quad (4.11)$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати. Приймемо 11%.

$З_{дод}$ = (73016,36 + 1949,74) · 11 / 100% = 8246,27 грн.

Нарахування на заробітну плату дослідників та робітників розраховуємо як 22% від суми основної та додаткової заробітної плати дослідників і робітників за формулою:

$$З_н = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%} \quad (4.12)$$

де $Н_{зп}$ – норма нарахування на заробітну плату. Приймаємо 22%.

Зн = (73016,36 + 1949,74 + 8246,27) · 22 / 100% = 18306,72 грн.

До статті «Сировина та матеріали» належать витрати на сировину, основні та допоміжні матеріали, інструменти, пристрої та інші засоби і предмети праці, які придбані у сторонніх підприємств, установ і організацій та витрачені на проведення досліджень за темою «Система розпізнавання та класифікації об'єктів на аерофотознімках з використанням нейромереж».

Витрати на матеріали ($М$), у вартісному вираженні розраховуються окремо по кожному виду матеріалів за формулою:

$$M = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j - \sum_{j=1}^{n} В_j \cdot Ц_{вj}, \qquad (4.13)$$

де $Н_j$ – норма витрат матеріалу j-го найменування, кг;

n – кількість видів матеріалів;

$Ц_j$ – вартість матеріалу j-го найменування, грн/кг;

$К_j$ – коефіцієнт транспортних витрат, ($К_j$ = *1,1 … 1,15*);

$В_j$ – маса відходів j-го найменування, кг;

$Ц_{вj}$ – вартість відходів j-го найменування, грн/кг.

$М_1$ = 4,00 · 112,00 · 1,1 - 0,000 · 0,00 = 492,80 грн.

Проведені розрахунки зведемо до таблиці.

Таблиця 5.9 – Витрати на матеріали

Найменування матеріалу, марка, тип, сорт	Ціна за 1 кг, грн	Норма витрат, кг	Величина відходів, кг	Ціна відходів, грн/кг	Вартість витраченого матеріалу, грн
Офісний папір ECOSOFT Ultra Plus	112,00	4,00	0,000	0,00	492,80
Папір для записів Parers ECOSOFT Light A5	72,00	3,00	0,000	0,00	237,60
Органайзер офісний OFFICE ECOSOFT	210,00	2,00	0,000	0,00	462,00

Закінчення таблиці 5.9

Канцелярське приладдя (набір офісного працівника)	174,00	4,00	0,000	0,00	765,60
Картридж для принтера Erixon EZ2500	708,00	2,00	0,000	0,00	1557,60
Диск оптичний NewOPtice CD-RW	12,20	4,00	0,000	0,00	53,68
Flesh-пам'ять Kingston 32 GB	243,00	1,00	0,000	0,00	267,30
Тека для паперів BOSS PAPERS BOX	87,00	3,00	0,000	0,00	287,10
Всього					4123,68

Витрати на комплектуючі *($К_в$)*, які використовують при проведенні МКР на тему «Система розпізнавання та класифікації об’єктів на аерофотознімках з використанням нейромереж» відсутні.

До статті «Програмне забезпечення для наукових (експериментальних) робіт» належать витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення, (програм, алгоритмів, баз даних) необхідних для проведення досліджень, також витрати на їх проектування, формування та встановлення.

Балансову вартість програмного забезпечення розраховуємо за формулою:

$$B_{прг} = \sum_{i=1}^{k} Ц_{iпрг} \cdot C_{прг.i} \cdot K_i \ , \qquad (4.15)$$

де $Ц_{iпрг}$ – ціна придбання одиниці програмного засобу даного виду, грн;

$C_{прг.i}$ – кількість одиниць програмного забезпечення відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує інсталяцію, налагодження програмного засобу тощо, ($K_i = 1,10…1,12$);

k – кількість найменувань програмних засобів.

$B_{прг} = 27500,00 \cdot 1 \cdot 1,1 = 29799,00$ *грн.*

Отримані результати зведемо до таблиці:

Таблиця 5.11 – Витрати на придбання програмних засобів по кожному виду

Найменування програмного засобу	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
Середовище розробки Pycharm	1	27500,00	29799,00
Всього			29799,00

В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо, розраховуємо з використанням прямолінійного методу амортизації за формулою:

$$A_{обл} = \frac{Ц_б}{T_в} \cdot \frac{t_{вик}}{12}, \quad (4.16)$$

де $Ц_б$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{вик}$ – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$T_в$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

$A_{обл} = (24800,00 \cdot 3) / (2 \cdot 12) = 3100,00$ *грн.*

Проведені розрахунки зведемо до таблиці 5.12.

Таблиця 5.12 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
Персональний комп'ютер розробника	24800,00	2	3	3100,00
Робоче місце інженера-програміста	8250,00	5	3	412,50
Пристрої передачі даних	6540,00	4	3	408,75

Закінчення таблиці 5.12

Оргтехніка	7850,00	4	3	490,63
Приміщення лабораторії розробки інформаційних систем	205000,00	25	3	2050,00
ОС Windows 11	7910,00	2	3	988,75
Прикладний пакет Microsoft Office 2019	6450,00	2	3	806,25
Електронно-обчислювальний комплекс обробки зображень на базі QUBE QB i9 10900KF RTX 3060 TI 8GB 1642	68900,00	2	3	8612,50
Всього				16869,38

Витрати на електроенергію (B_e) розраховуємо за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_e \cdot K_{вni}}{\eta_i}, \qquad (4.17)$$

де W_{yi} – встановлена потужність обладнання на визначеному етапі розробки, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год;

$Ц_e$ – вартість 1 кВт-години електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії), приймемо $Ц_e$ = 4,50 грн;

$K_{вni}$ – коефіцієнт, що враховує використання потужності, $K_{вni} < 1$;

η_i – коефіцієнт корисної дії обладнання, $\eta_i < 1$.

$$B_e = 0,45 \cdot 400,0 \cdot 4,50 \cdot 0,95 / 0,97 = 810,00 \text{ грн.}$$

Проведені розрахунки зведемо до таблиці.

Таблиця 4.13 – Витрати на електроенергію

Найменування обладнання	Встановлена потужність, кВт	Тривалість роботи, год	Сума, грн
Персональний комп'ютер розробника програмного забезпечення	0,45	400,0	810,00
Робоче місце інженера-програміста	0,10	400,0	180,00
Пристрої передачі даних	0,03	32,0	4,32
Оргтехніка	0,65	10,0	29,25
Електронно-обчислювальний комплекс обробки зображень на базі QUBE QB i9 10900KF RTX 3060 TI 8GB 1642	0,50	400,0	900,00
Всього			1923,57

До статті «Службові відрядження» дослідної роботи на тему «Система розпізнавання та класифікації об’єктів на аерофотознімках з використанням нейромереж» належать витрати на відрядження штатних працівників, працівників організацій, які працюють за договорами цивільно-правового характеру, аспірантів, зайнятих розробленням досліджень, відрядження, пов’язані з проведенням випробувань машин та приладів, а також витрати на відрядження на наукові з’їзди, конференції, наради, пов’язані з виконанням конкретних досліджень.

Витрати за статтею «Службові відрядження» розраховуємо як 20…25% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{св} = (З_о + З_р) \cdot \frac{H_{св}}{100\%}, \qquad (4.18)$$

де $H_{св}$ – норма нарахування за статтею «Службові відрядження», приймемо $Н_{св} = 20\%$.

$$В_{св} = (73016,36 + 1949,74) \cdot 20 / 100\% = 14993,22 \text{ грн.}$$

Витрати за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації» розраховуємо як 30…45% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{сп} = (З_о + З_р) \cdot \frac{Н_{сп}}{100\%}, \quad (4.19)$$

де $Н_{сп}$ – норма нарахування за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації», приймемо $Н_{сп}$= 30%.

$$B_{сп} = (73016,36 + 1949,74) \cdot 30 / 100\% = 22489,83 \text{ грн.}$$

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «Інші витрати» розраховуємо як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$I_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \quad (4.20)$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати», приймемо $Н_{ів}$ = 52%.

$$I_в = (73016,36 + 1949,74) \cdot 52 / 100\% = 38982,37 \text{ грн.}$$

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов’язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов’язані з набором робочої сили; витрати на оплату послуг банків;

витрати, пов’язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін.

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуємо як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}\ , \qquad (4.21)$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати», приймемо $Н_{нзв}$ = 107%.

$$В_{нзв} = (73016,36 + 1949,74) \cdot 107 / 100\% = 80213,73 \text{ грн.}$$

Витрати на проведення науково-дослідної роботи на тему «Система розпізнавання та класифікації об’єктів на аерофотознімках з використанням нейромереж» розраховуємо як суму всіх попередніх статей витрат за формулою:

$$B_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + B_{спец} + B_{прг} + A_{обл} + B_е + B_{св} + B_{сп} + I_в + B_{нзв}. \qquad (4.22)$$

$$В_{заг} = 73016,36 + 1949,74 + 8246,27 + 18306,72231 + 4123,68 + 0,00 + 34364,00 + 29799,00 + 16869,38 + 1935,99 + 14993,22 + 22489,83 + 38982,37 + 80213,73 = 345290,30 \text{ грн.}$$

Загальні витрати *ЗВ* на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховується за формулою:

$$ЗВ = \frac{B_{заг}}{\eta}, \qquad (4.23)$$

де η — коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи, приймемо η=0,9.

$$ЗВ = 345290,30 / 0,9 = 383655,88 \text{ грн.}$$

5.5. Розрахунок економічної ефективності науково-технічної розробки при її можливій комерціалізації потенційним інвестором

В ринкових умовах узагальнюючим позитивним результатом, що його може отримати потенційний інвестор від можливого впровадження результатів тієї чи іншої науково-технічної розробки, є збільшення у потенційного інвестора величини чистого прибутку.

Результати дослідження передбачають комерціалізацію протягом 4-х років реалізації на ринку.

В цьому випадку майбутній економічний ефект буде формуватися на основі таких даних:

ΔN – збільшення кількості споживачів продукту, у періоди часу, що аналізуються, від покращення його певних характеристик;

Показник	1-й рік	2-й рік	3-й рік	4-й рік
Збільшення кількості споживачів, осіб	1250	2650	3500	3000

N – кількість споживачів які використовували аналогічний продукт у році до впровадження результатів нової науково-технічної розробки, приймемо 10400 осіб;

$Ц_б$ – вартість програмного продукту у році до впровадження результатів розробки, приймемо 2300,00 грн;

$\pm\Delta Ц_о$ – зміна вартості програмного продукту від впровадження результатів науково-технічної розробки, приймемо 300,95 грн.

Можливе збільшення чистого прибутку у потенційного інвестора $\Delta П_i$ для кожного із 4-х років, протягом яких очікується отримання позитивних результатів від можливого впровадження та комерціалізації науково-технічної розробки, розраховуємо за формулою [18];

$$\Delta П_i = (\pm \Delta Ц_o \cdot N + Ц_o \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \quad (4.24)$$

де λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість. У 2021 році ставка податку на додану вартість складає 20%, а коефіцієнт $\lambda = 0,8333$;

ρ – коефіцієнт, який враховує рентабельність інноваційного продукту). Приймемо $\rho = 40\%$;

ϑ – ставка податку на прибуток, який має сплачувати потенційний інвестор, у 2021 році $\vartheta = 18\%$;

Збільшення чистого прибутку 1-го року:

$\Delta П_1 = (300,95 \cdot 10400,00 + 2600,95 \cdot 1250) \cdot 0,83 \cdot 0,4 \cdot (1\text{-}0,18/100\%) = 1737181,82$ грн.

Збільшення чистого прибутку 2-го року:

$\Delta П_2 = (300,95 \cdot 10400,00 + 2600,95 \cdot 3900) \cdot 0,83 \cdot 0,4 \cdot (1\text{-}0,18/100\%) = 3613600,78$ грн.

Збільшення чистого прибутку 3-го року:

$\Delta П_3 = (300,95 \cdot 10400,00 + 2600,95 \cdot 7400) \cdot 0,83 \cdot 0,4 \cdot (1\text{-}0,18/100\%) = 6091889,98$ грн.

Збільшення чистого прибутку 4-го року:

$\Delta П_4 = (300,95 \cdot 10400,00 + 2600,95 \cdot 10400) \cdot 0,83 \cdot 0,4 \cdot (1\text{-}0,18/100\%) = 8216137,8$ грн.

Приведена вартість збільшення всіх чистих прибутків *ПП*, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \quad (4.25)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн;

T – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ=0,12;

t – період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

$$ПП = 1737181,82/(1+0,12)^1+3613600,78/(1+0,12)^2+6091889,98/(1+0,12)^3+$$
$$+8216137,86/(1+0,12)^4=1551055,19+2880740,42+4336086,96+5221504,15=$$
$$=13989386,72 \text{ грн.}$$

Величина початкових інвестицій PV, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки:

$$PV = k_{інв} \cdot ЗВ, \qquad (4.26)$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію, приймаємо $k_{інв}$=1,5;

$ЗВ$ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, приймаємо 383655,88 грн.

$$PV = k_{інв} \cdot ЗВ = 1,5 \cdot 383655,88 = 575483,83 \text{ грн.}$$

Абсолютний економічний ефект $E_{абс}$ для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$E_{абс} = ПП - PV \qquad (4.27)$$

де $ПП$ – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, 13989386,72 грн;

PV – теперішня вартість початкових інвестицій, 575483,83 грн.

$E_{абс} = ПП - PV$ = 13989386,72 - 575483,83 = 13413902,89 грн.

Внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$E_в = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1, \quad (4.28)$$

де $E_{абс}$ – абсолютний економічний ефект вкладених інвестицій, 13413902,89 грн;

PV – теперішня вартість початкових інвестицій, 575483,83 грн;

$T_{ж}$ – життєвий цикл науково-технічної розробки, тобто час від початку її розробки до закінчення отримування позитивних результатів від її впровадження, 4 роки.

$$E_в = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1 = (1+13413902,89/575483,83)^{1/4} = 1,22.$$

Мінімальна внутрішня економічна дохідність вкладених інвестицій $\tau_{мін}$:

$$\tau_{мін} = d + f, \quad (4.29)$$

де d — середньозважена ставка за депозитними операціями в комерційних банках в 2021 році в Україні, d =0,09;

f – показник, що характеризує ризикованість вкладення інвестицій, приймемо 0,3.

$$\tau_{мін} = 0,09+0,3 = 0,39$$

Отриманий коефіцієнт має значення менше ніж 1.22 свідчить про те, що внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки вища мінімальної внутрішньої дохідності. Тобто інвестувати в науково-дослідну роботу за темою «Система розпізнавання та класифікації об'єктів на аерофотознімках з використанням нейромереж» доцільно.

Період окупності інвестицій $T_{ок}$ які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$T_{ок} = \frac{1}{E_в}, \tag{4.30}$$

де $E_в$ – внутрішня економічна дохідність вкладених інвестицій.

$T_{ок}$ = 1 / 1,22 = 0,82 р.

$T_{ок} <$ 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

В цьому розділі було проведено оцінку економічної доцільності розробки при впровадженні її у використання. Згідно проведених досліджень рівень комерційного потенціалу становить 41,7 бала, що, свідчить про комерційну важливість проведення даних досліджень. При оцінюванні рівня конкурентоспроможності, згідно узагальненого коефіцієнту конкурентоспроможності розробки, науково-технічна розробка переважає існуючі аналоги приблизно в 1,83 рази.

Також термін окупності становить 0,82 р., що менше 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок, що дозволяє зробити висновок про доцільність впровадження розробки у комерційне використання.

ВИСНОВКИ

У першому розділі було розглянуто актуальність і способи вирішення задачі розпізнавання і класифікації об’єктів на аерофотознімках. На сьогоднішній день згорткові нейромережі архітектури U-Net або SegNet є найкращим способом вирішення цієї задачі через їх можливість адаптуватись до різних наборів вхідних даних, можливість працювати із зображеннями нижчої якості, стійксть до змін кутів повороту, зміщень об’єктів на зображеннях. Також було розглянуто технологію HTTP та мікросервісну архітектуру, які разом дозволяють будувати розподілені сервіси, горизонтально масштабувати процес обробки зображень для збільшення швидкодії аналізу.

В другому розділі було розглянуто основні принципи функціонування нейромереж, основні поняття теорії нейромереж, такі як нейрон, функція активації, процес навчання і його особливості. Також розглянуто специфічні для згорткових нейромереж поняття субдискретизації та збільшення розмірності карти ознак. Як метрику оцінювання якості роботи нейромережі обрано коефіцієнт Жаккара також відомий як intersection over union. Розглянуто архітектури мереж для виконання семантичної сегментацію, для реалізації задачі було обрано архітектуру SegNet.

В третьому розділі було вибрано інструменти реалізації задачі. Основною мовою програмування було обрано мову Python та фреймворк Keras з використаннями бекенду Tensorflow. Вибір обумовлено широкою підтримкою цих інструментів, їх популярністю для виконання подібних задач, та загальною простотою використання. Було проведено навчання моделі з використанням Keras і Tensorflow, тренувальні і тестові дані було взято з набору даних Dstl Satellite Imagery Feature Detection. Для надання інтерфейсу роботи з функціональністю розпізнавання об’єктів було створено HTTP сервіс з використанням бібілотеки Flask. Сервіс дозволяє завантажити зображення, які необхідно обробити, провести аналіз засобами нейромережі і повернути

результат розпізнавання у вигляді об’єктної нотації JSON та нотації опису геометричних об’єктів WKT.

В четвертій частині було наведено результати тестування роботи нейромережі та HTTP сервісу. В цілому мережа показала хороші результати розпізнавання, але для деяких класів об’єктів вона працює краще ніж для інших, але загальна точність роботи є високою. Також надано інструкцію для запуску сервісу розпізнавання.

В п’ятому розділі було проведено оцінку економічної доцільності розробки при впровадженні її у використання. Згідно проведених досліджень рівень комерційного потенціалу становить 41,7 бала, що, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки високий).

При оцінюванні рівня конкурентоспроможності, згідно узагальненого коефіцієнту конкурентоспроможності розробки, науково-технічна розробка переважає існуючі аналоги приблизно в 1,83 рази.

Також термін окупності становить 0,82 р., що менше 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок, що дозволяє зробити висновок про доцільність впровадження розробки у комерційне використання.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ярмощук Д. О. Система розпізнавання та класифікації об’єктів на аерофотознімках з використанням нейромереж Д. О. Ярмощук // Молодь в науці: дослідження, проблеми, перспективи (МН-2022). Тез. доп. - Вінниця, 2021. [Електронний ресурс] — Режим доступу: https://conferences.vntu.edu.ua/index.php/mn/mn2021/paper/view/13144/11053.
2. Very Deep Convolutional Networks For Large-Scale Image Recognition [Електронний ресурс] – Режим доступу: https://arxiv.org/pdf/1409.1556.pdf
3. Как превратить спутниковые снимки в карты. Компьютерное зрение в Яндексе [Електронний ресурс] — Режим доступу: https://habr.com/ru/company/yandex/blog/431108/
4. U-Net: Convolutional Networks for Biomedical Image Segmentation [Електронний ресурс] – Режим доступу: https://arxiv.org/pdf/1505.04597.pdf
5. Mahinda Pathegama & Ö Göl (2004): "Edge-end pixel extraction for edge-based image segmentation", Transactions on Engineering, Computing and Technology, vol. 2, 2004.
6. Matusugu, Masakazu; Katsuhiko Mori; Yusuke Mitari; Yuji Kaneda. Subject independent facial expression recognition with robust face detection using a convolutional neural network, 2003.
7. Hypercolumns for Object Segmentation and Fine-grained Localization [Електронний ресурс] – Режим доступу: http://home.bharathh.info/pubs/pdfs/BharathCVPR2015.pdf
8. Р. Дуда / Распознавание образов и анализ сцен, 2013. – 508 с.
9. Порівняння швидкодії роботи нейромережі на центральному процесорі та відеоадаптері [Електронний ресурс] — Режим доступу: https://datamadness.github.io/TensorFlow2-CPU-vs-GPU.

10. Дослідження популярності мови Python [Електронний ресурс] — https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3426281
11. Інструментарій CUDA [Електронний ресурс] — https://developer.nvidia.com/cuda-zone
12. Порівняльна таблиця швидкодії різних відеоадаптерів [Електронний ресурс] — https://browser.geekbench.com/cuda-benchmarks
13. Batch Normalization [Електронний ресурс] — https://arxiv.org/pdf/1502.03167.pdf
14. Огляд архітектури SegNet [Електронний ресурс] — https://towardsdatascience.com/review-segnet-semantic-segmentation-e66f2e30fb96
15. Дані для тренування нейромережі [Електронний ресурс] — https://www.kaggle.com/c/dstl-satellite-imagery-feature-detection/data
16. Режими роботи супутника WorldView-3 [Електронний ресурс] — https://space.oscar.wmo.int/instruments
17. Стандарт WKT [Електронний ресурс] — https://www.ogc.org/standards/sfs
18. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. – Вінниця : ВНТУ, 2021. – 42 с.
19. Кавецький В. В. Економічне обґрунтування інноваційних рішень: практикум / В. В. Кавецький, В. О. Козловський, І. В. Причепа – Вінниця : ВНТУ, 2016. – 113 с.

ДОДАТОК А

Міністерство освіти і науки України

Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра обчислювальної техніки

ЗАТВЕРДЖУЮ

Завідувач кафедри ОТ

Азаров О. Д.

«____» _____2021 року

ТЕХНІЧНЕ ЗАВДАННЯ

на виконання магістерської кваліфікаційної роботи на тему:

«Система розпізнавання та класифікації об'єктів на аерофотознімках з використанням нейромереж»

08-23.МКР.015.00.000 ТЗ

Науковий керівник: _____ Ткаченко О. М.

(підпис)

студент групи _____ Ярмощук Д. О.

(підпис)

Вінниця 2021

1 Підстава для виконання магістерської кваліфікаційної роботи (МКР):

— актуальність розробки системи полягає у тому що аналіз аерофотознімків є важливою задачею в багатьох галузях де потрібно отримувати поточну інформацію про стан місцевості;

— наказ про затвердження теми магістерської кваліфікаційної роботи.

2 Мета і призначення МКР:

Мета роботи полягає у розробці системи, що дозволить автоматизувати та пришвидшити процес аналізу аерофотознімків місцевості.

3 Джерела розробки МКР:

Інтегроване середовище розробки JetBrains PyCharm для програмування мовою Python з використанням бібіліотек Keras та Tensoflow.

4 Технічні вимоги до виконання МКР:

— наявність набору зображень для обробки;

— наявність програми-сервісу для отримання запитів обробки зображень та відправки відповіді на ці запити.

5 Етапи МКР та очікувані результати наведені в таблиці А.1.

Таблиця А.1 — Етапи виконання роботи

№ етапу	Назва етапу	Термін виконання		Очікувані результати
		початок	кінець	
1	Огляд і аналіз джерел інформації	01.09.2021	03.10.2021	Аналітичний огляд літературних джерел
2	Розробка технічного завдання	03.10.2021	10.10.2021	Огляд задачі дослідження
3	Аналіз сучасного стану технологій	11.10.2021	24.10.2021	Розділ 1
4	Обгрунтування вибраних методів	25.10.2021	7.11.2021	Розділ 2
5	Реалізація і тестування системи	8.11.2021	21.11.2021	Розділ 3-4
6	Розрахунок економічної частини	22.11.2021	3.12.2021	Розділ 5
7	Оформлення пояснювальної записки та презентації	4.12.2021	20.12.2021	Пояснювальна записка, презентація

6 Матеріали, що подаються до захисту МКР: пояснювальна записка МКР, графічні та ілюстративні матеріали, протокол попереднього захисту МКР на кафедрі, відгук наукового керівника, відгук рецензента, анотації до МКР українською та іноземною мовами, відповідність оформлення МКР діючим вимогам.

7 Порядок контролю виконання та захисту МКР.

Виконання етапів графічної та розрахункової документації МКР контролюється науковим керівником згідно зі встановленими термінами. Захист МКР відбувається на засіданні Державної екзаменаційної комісії, затвердженою наказом ректора.

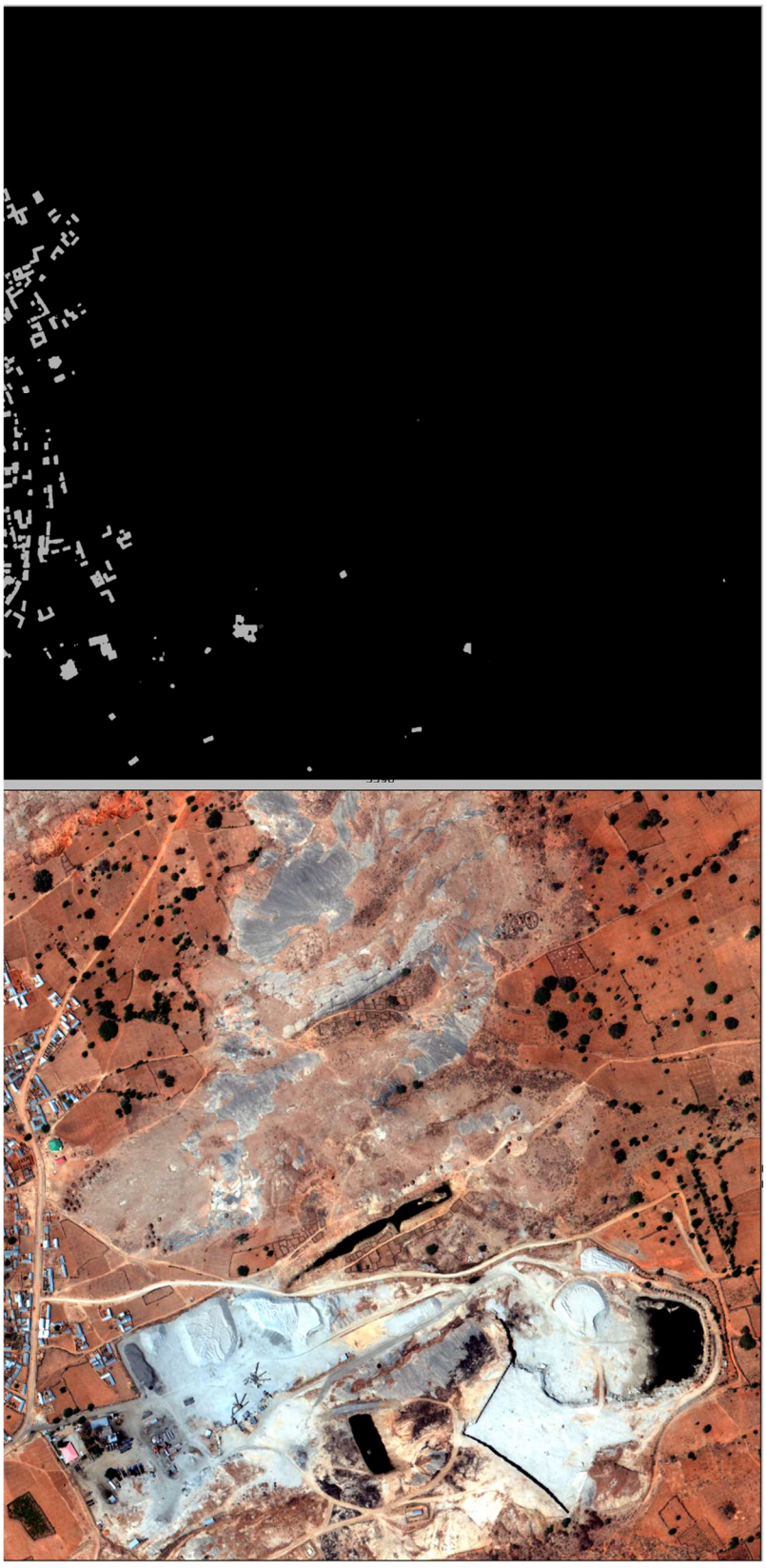
8 Вимоги до оформлення МКР

Оформлення магістерської кваліфікаційної роботи відбувається згідно «Положень про кваліфікаційні роботи» та ДСТУ 3008-2015.

Технічне завдання до виконання прийняв _____

ДОДАТОК Б

Фото результату тестового розпізнавання будівель на зображенні



Риснуок Б.1 — фото результату тестового розпізнавання будівель на зображенні

ДОДАТОК В

Фото результату тестового розпізнавання доріг і стежок на зображенні

Рисунок В.1 — фото результату тестового розпізнавання доріг і стежок на зображенні

ДОДАТОК Г

Фото результату тестового розпізнавання водойм на зображенні

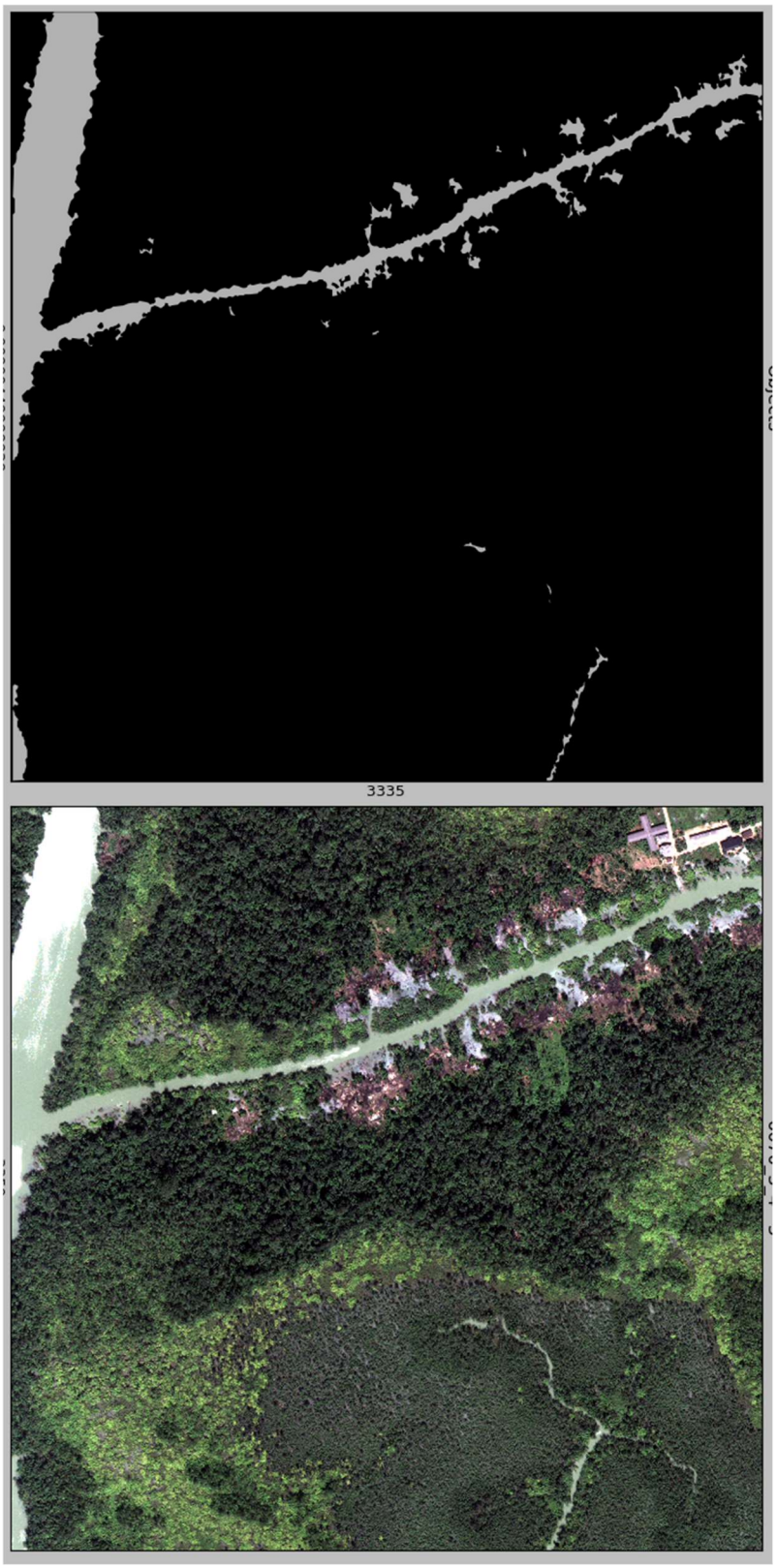


Рисунок Г.1 — фото результату тестового розпізнавання водойм на зображенні

ДОДАТОК Д

Лістинг HTTP-сервісу для аналізу зображень

```
import json
import os
from collections import defaultdict

import cv2
import numpy as np
import pandas as pd
import shapely
import tensorflow as tf
import tifffile as tiff
from flask import Flask, request, Response
from keras import backend as K
from shapely.geometry import MultiPolygon, Polygon
from skimage.transform import resize
from tensorflow.keras.layers import *
from tensorflow.keras.models import *
from tensorflow.keras.optimizers import *
from tifffile import imwrite

classes_len = 10
size = 160
smooth = 1e-12
UPLOAD_FOLDER = /home/upload'



def calculate_scale(h, w, x_max, y_min):
    w_ = w * (w / (w + 1))
```

```
    h_ = h * (h / (h + 1))
    return w_ / x_max, h_ / y_min


def fix_polygons(polygons):
    if not polygons.is_valid:
        polygons = polygons.buffer(0)
        if polygons.type == 'Polygon':
            polygons = MultiPolygon([polygons])
    return polygons


def get_contours(contours, epsilon):
    return [cv2.approxPolyDP(cnt, epsilon, True) for cnt in contours]


def find_parent(hierarchy, contours, min_area):
    children = defaultdict(list)
    child_contours = set()
    assert hierarchy.shape[0] == 1

    for idx, (_, _, _, parent_idx) in enumerate(hierarchy[0]):
        if parent_idx != -1:
            child_contours.add(idx)
            children[parent_idx].append(contours[idx])

    all_polygons = []
    for idx, cnt in enumerate(contours):
        if idx not in child_contours and cv2.contourArea(cnt) >= min_area:
            assert cnt.shape[1] == 1
```

```
            holes = [c[:, 0, :] for c in children.get(idx, []) if cv2.contourArea(c) >=
min_area]
            contour = cnt[:, 0, :]

            poly = Polygon(shell=contour, holes=holes)

            if poly.area >= min_area:
                all_polygons.append(poly)

    return all_polygons


def mask_to_polygons(mask, epsilon=1.0, min_area=10.0):
    # first, find contours with cv2: it's much faster than shapely
    contours, hierarchy = cv2.findContours(((mask == 1) * 255).astype(np.uint8),
cv2.RETR_CCOMP, cv2.CHAIN_APPROX_TC89_KCOS)

    # create approximate contours to have reasonable submission size
    if epsilon != 0:
        approx_contours = get_contours(contours, epsilon)
    else:
        approx_contours = contours

    if not approx_contours:
        return MultiPolygon()

    polygons = find_parent(hierarchy, approx_contours, min_area)
    polygons = MultiPolygon(polygons)
    polygons = fix_polygons(polygons)
    return polygons
```

```
def mask_to_polygon(predicted_mask, threashold, x_scale, y_scale):
    polygons = mask_to_polygons(predicted_mask[0] > threashold, epsilon=0,
min_area=5)
    polygons = shapely.affinity.scale(polygons, xfact=1.0 / x_scale, yfact=1.0 /
y_scale, origin=(0, 0, 0))
    return MultiPolygon(polygons.buffer(2.6e-5))


def fix_contrast(bands, lower_percent=2, higher_percent=98):
    out = np.zeros_like(bands).astype(np.float32)
    n = bands.shape[2]
    for i in range(n):
        a = 0  # np.min(band)
        b = 1  # np.max(band)
        c = np.percentile(bands[:, :, i], lower_percent)
        d = np.percentile(bands[:, :, i], higher_percent)
        t = a + (bands[:, :, i] - c) * (b - a) / (d - c)
        t[t < a] = a
        t[t > b] = b
        out[:, :, i] = t

    return out.astype(np.float32)


def resize_image(image, model):
    if image.shape == (837, 837, 8):
        return image
```

```
    else:
        resized_data = resize(image, (837, 837, 8))
        imwrite('resized.tif', resized_data, planarconfig='CONTIG')
        return tiff.imread("resized.tif")


def predict(img_path, model):
    rgb_img = tiff.imread(img_path)
    img = np.rollaxis(rgb_img, 0, 3)

    # resize the image according to model architecture
    img = resize_image(img, model)

    # adjust the contrast of the image
    x = fix_contrast(img)

    cnv = np.zeros((960, 960, 8)).astype(np.float32)
    prd = np.zeros((classes_len, 960, 960)).astype(np.float32)
    cnv[:img.shape[0], :img.shape[1], :] = x

    for i in range(0, 6):
        line = []
        for j in range(0, 6):
            line.append(cnv[i * size:(i + 1) * size, j * size:(j + 1) * size])

        x = 2 * np.transpose(line, (0, 1, 2, 3)) - 1

        tmp = model.predict(x, batch_size=4)
        tmp = np.transpose(tmp, (0, 3, 1, 2))
```

```
        for j in range(tmp.shape[0]):
            prd[:, i * size:(i + 1) * size, j * size:(j + 1) * size] = tmp[j]

    # thresholds for each class
    trs = [0.4, 0.1, 0.4, 0.3, 0.3, 0.5, 0.3, 0.6, 0.1, 0.1]
    for i in range(classes_len):
        prd[i] = prd[i] > trs[i]
    p = prd[:, :img.shape[0], :img.shape[1]]
    return p


app = Flask(__name__, static_url_path="/static")


@app.route('/encoding', methods=['POST'])
def predict_image():
    uploaded_image = request.files['image']
    if not uploaded_image:
        return Response(json.dumps({}).encode(), mimetype="application/json")

    mask = predict(uploaded_image, model)
    class_list = ["Building", "Structure", "Road", "Track", "Tree", "Crop",
"RunningWater", "StandingWater", "LargeVehicle", "SmallVehicle"]

    result = []
    for i in range(classes_len):
        x_sc, y_sc = calculate_scale(mask.shape[1], mask.shape[2], 1, 1)
        result += [(class_list[i], mask_to_polygon(mask[i], 0.3, x_sc, y_sc))]

    data_frame = pd.DataFrame(result, columns=['Type', 'PolygonsWKT'])
```

```
    return Response(json.dumps(json.loads(data_frame.to_json(orient="records",
default_handler=str))).encode(), mimetype="application/json")


def SegNet():
    tf.random.set_seed(32)
    classes = 10
    img_input = Input(shape=(size, size, 8))
    x = img_input

    # Encoder

    x = Conv2D(64, (3, 3), activation='relu', padding='same',
kernel_initializer=tf.keras.initializers.he_normal(seed=23))(x)
    x = BatchNormalization()(x)
    x = Conv2D(64, (3, 3), activation='relu', padding='same',
kernel_initializer=tf.keras.initializers.he_normal(seed=43))(x)
    # x = BatchNormalization()(x)
    x = MaxPooling2D((2, 2), strides=(2, 2))(x)
    x = Dropout(0.25)(x)

    x = Conv2D(128, (3, 3), activation='relu', padding='same',
kernel_initializer=tf.keras.initializers.he_normal(seed=32))(x)
    x = BatchNormalization()(x)
    x = Conv2D(128, (3, 3), activation='relu', padding='same',
kernel_initializer=tf.keras.initializers.he_normal(seed=41))(x)
    # x = BatchNormalization()(x)
    x = Conv2D(128, (3, 3), activation='relu', padding='same',
kernel_initializer=tf.keras.initializers.he_normal(seed=33))(x)
```

```
    x = BatchNormalization()(x)
    x = MaxPooling2D((2, 2), strides=(2, 2))(x)
    x = Dropout(0.5)(x)

    x = Conv2D(256, (3, 3), activation='relu', padding='same',
kernel_initializer=tf.keras.initializers.he_normal(seed=35))(x)
    x = BatchNormalization()(x)
    x = Conv2D(256, (3, 3), activation='relu', padding='same',
kernel_initializer=tf.keras.initializers.he_normal(seed=54))(x)
    x = BatchNormalization()(x)
    x = Conv2D(256, (3, 3), activation='relu', padding='same',
kernel_initializer=tf.keras.initializers.he_normal(seed=39))(x)
    x = BatchNormalization()(x)
    x = Dropout(0.5)(x)

    # Decoder

    x = UpSampling2D(size=(2, 2))(x)
    x = Conv2D(128, kernel_size=3, activation='relu', padding='same',
kernel_initializer=tf.keras.initializers.he_normal(seed=45))(x)
    # x = BatchNormalization()(x)
    x = Conv2D(128, kernel_size=3, activation='relu', padding='same',
kernel_initializer=tf.keras.initializers.he_normal(seed=41))(x)
    x = BatchNormalization()(x)
    x = Conv2D(128, kernel_size=3, activation='relu', padding='same',
kernel_initializer=tf.keras.initializers.he_normal(seed=49))(x)
    x = BatchNormalization()(x)
    x = Dropout(0.25)(x)

    x = UpSampling2D(size=(2, 2))(x)
```

```
    x = Conv2D(64, kernel_size=3, activation='relu', padding='same',
kernel_initializer=tf.keras.initializers.he_normal(seed=18))(x)
    x = BatchNormalization()(x)
    x = Conv2D(64, kernel_size=3, activation='relu', padding='same',
kernel_initializer=tf.keras.initializers.he_normal(seed=21))(x)
    x = BatchNormalization()(x)
    x = Conv2D(classes, kernel_size=3, activation='relu', padding='same',
kernel_initializer=tf.keras.initializers.he_normal(seed=16))(x)
    x = Dropout(0.25)(x)

    x = Activation("softmax")(x)

    model = Model(img_input, x)

    model.compile(optimizer=Adam(lr=1e-4), loss='binary_crossentropy',
metrics=[jaccard_coef])
    return model


def jaccard_coef(y_true, y_pred):
    intersection = K.sum(y_true * y_pred, axis=[0, -1, -2])
    total = K.sum(y_true + y_pred, axis=[0, -1, -2])
    union = total - intersection

    jac = (intersection + smooth) / (union+ smooth)

    return K.mean(jac)


if __name__ == "__main__":
```

```
model = SegNet()
model.load_weights("/home/my_trained_model.hdf5")

print('Model loaded')

app.run(host='127.0.0.1', port=5000, debug=True)

predict_image()
```

ДОДАТОК Е

Лістинг коду навчання нейромережі

```
import os
import warnings

import matplotlib.pyplot as plt
import numpy as np
import tifffile as tiff
from keras import backend as K
from keras.layers import *
from keras.models import *

warnings.filterwarnings("ignore")
from tifffile import imwrite
from skimage.transform import resize
import tensorflow as tf
import random as rn
from tensorflow.keras.optimizers import *



class Dataloder(tf.keras.utils.Sequence):
    def __init__(self, dataset, batch_size=1, shuffle=False):
        self.dataset = dataset
        self.batch_size = batch_size
        self.shuffle = shuffle
        self.indexes = np.arange(len(dataset))

    def __getitem__(self, i):

        # collect batch data
```

```
        start = i * self.batch_size
        stop = (i + 1) * self.batch_size
        data = []
        for j in range(start, stop):
            data.append(self.dataset[j])

        batch = [np.stack(samples, axis=0) for samples in zip(*data)]

        #print(len(batch))
        return tuple(batch)

    def __len__(self):
        return len(self.indexes) // self.batch_size



class Dataset:

    def __init__(self, images_dir, mask_dir):

        self.ids = images_dir
        self.images_fps = images_dir
        self.masks_fps  = mask_dir

    def __getitem__(self, i):

        # read data
        image = np.load(self.images_fps[i])
        mask  = np.load(self.masks_fps[i])
```

```
        image = np.stack(image, axis=-1).astype('float')
        mask = np.stack(mask, axis=-1).astype('float')

        #image = np.transpose(image, (1,0,2))
        #mask = np.transpose(mask, (1,0,2))

        image = np.transpose(image, (0,2,1))
        mask = np.transpose(mask, (0,2,1))

        return image, mask

    def __len__(self):
        return len(self.ids)



x_test_data = np.load("/home/content")
y_test_data = np.load("/home/content")
test_data = Dataset(x_test_data, y_test_data)
test_dataloader = Dataloder(test_data, batch_size=1)

a = rn.randint(0,len(test_dataloader))
image = test_dataloader[a][0]
mask =  test_dataloader[a][1]

classes_num = 10
size = 160
smooth = 1e-12

def fix_contrast(bands, lower_percent=2, higher_percent=98):
    out = np.zeros_like(bands).astype(np.float32)
```

```
    n = bands.shape[2]
    for i in range(n):
        a = 0  # np.min(band)
        b = 1  # np.max(band)
        c = np.percentile(bands[:, :, i], lower_percent)
        d = np.percentile(bands[:, :, i], higher_percent)
        t = a + (bands[:, :, i] - c) * (b - a) / (d - c)
        t[t < a] = a
        t[t > b] = b
        out[:, :, i] = t

    return out.astype(np.float32)

def jaccard_coef(y_true, y_pred):
    intersection = K.sum(y_true * y_pred, axis=[0, -1, -2])
    total = K.sum(y_true + y_pred, axis=[0, -1, -2])
    union = total - intersection

    jac = (intersection + smooth) / (union+ smooth)

    return K.mean(jac)



np.random.seed(42)
tf.random.set_seed(32)
rn.seed(12)

def SegNet():

    classes= 10
```

```
    img_input = Input(shape=(size, size, 8))
    x = img_input

    # Encoder

    x = Conv2D(64, (3, 3), activation='relu', padding='same', kernel_initializer =
tf.keras.initializers.he_normal(seed= 23))(x)
    x = BatchNormalization()(x)
    x = Conv2D(64, (3, 3), activation='relu', padding='same',  kernel_initializer =
tf.keras.initializers.he_normal(seed= 43))(x)
   # x = BatchNormalization()(x)
    x = MaxPooling2D((2, 2), strides=(2, 2))(x)
    x = Dropout(0.25)(x)

    x = Conv2D(128, (3, 3), activation='relu', padding='same', kernel_initializer =
tf.keras.initializers.he_normal(seed= 32))(x)
    x = BatchNormalization()(x)
    x = Conv2D(128, (3, 3), activation='relu', padding='same', kernel_initializer =
tf.keras.initializers.he_normal(seed= 41))(x)
   # x = BatchNormalization()(x)
    x = Conv2D(128, (3, 3), activation='relu', padding='same', kernel_initializer =
tf.keras.initializers.he_normal(seed= 33))(x)
    x = BatchNormalization()(x)
    x = MaxPooling2D((2, 2), strides=(2, 2))(x)
    x = Dropout(0.5)(x)

    x = Conv2D(256, (3, 3), activation='relu', padding='same', kernel_initializer =
tf.keras.initializers.he_normal(seed= 35))(x)
    x = BatchNormalization()(x)
    x = Conv2D(256, (3, 3), activation='relu', padding='same', kernel_initializer =
```

```
tf.keras.initializers.he_normal(seed= 54))(x)
    x = BatchNormalization()(x)
    x = Conv2D(256, (3, 3), activation='relu', padding='same', kernel_initializer =
tf.keras.initializers.he_normal(seed= 39))(x)
    x = BatchNormalization()(x)
    x = Dropout(0.5)(x)

    #Decoder

    x = UpSampling2D(size=(2, 2))(x)
    x = Conv2D(128, kernel_size=3, activation='relu', padding='same',
kernel_initializer = tf.keras.initializers.he_normal(seed= 45))(x)
   # x = BatchNormalization()(x)
    x = Conv2D(128, kernel_size=3, activation='relu', padding='same',
kernel_initializer = tf.keras.initializers.he_normal(seed= 41))(x)
    x = BatchNormalization()(x)
    x = Conv2D(128, kernel_size=3, activation='relu', padding='same',
kernel_initializer = tf.keras.initializers.he_normal(seed= 49))(x)
    x = BatchNormalization()(x)
    x = Dropout(0.25)(x)

    x = UpSampling2D(size=(2, 2))(x)
    x = Conv2D(64, kernel_size=3, activation='relu', padding='same', kernel_initializer
= tf.keras.initializers.he_normal(seed= 18))(x)
    x = BatchNormalization()(x)
    x = Conv2D(64, kernel_size=3, activation='relu', padding='same', kernel_initializer
= tf.keras.initializers.he_normal(seed= 21))(x)
    x = BatchNormalization()(x)
    x = Conv2D(classes, kernel_size=3, activation='relu', padding='same',
kernel_initializer = tf.keras.initializers.he_normal(seed= 16))(x)
```

```
    x = Dropout(0.25)(x)

    x = Activation("softmax")(x)

    model = Model(img_input, x)

    model.compile(optimizer=Adam(lr=1e-4),loss='binary_crossentropy',
metrics=[jaccard_coef])
    return model

def resize_image(image, model):
  if image.shape == (837,837,8):
    return image

  else:
    resized_data = resize(image, (837,837,8))
    imwrite('resized.tif', resized_data, planarconfig='CONTIG')
    return tiff.imread("resized.tif")



def predict(id, model):
    rgb_img = os.path.join( 'sixteen_band', '{}_M.tif'.format(id))
    rgb_img = tiff.imread(rgb_img)
    img = np.rollaxis(rgb_img, 0, 3)

    img = resize_image(img, model)

    x = fix_contrast(img)

    cnv = np.zeros((960, 960, 8)).astype(np.float32)
```

```
    prd = np.zeros((classes_num, 960, 960)).astype(np.float32)
    cnv[:img.shape[0], :img.shape[1], :] = x

    for i in range(0, 6):
        line = []
        for j in range(0, 6):
            line.append(cnv[i * size:(i + 1) * size, j * size:(j + 1) * size])



        x = 2 * np.transpose(line, (0, 3, 1, 2)) - 1

        tmp = model.predict(x, batch_size=4)

        for j in range(tmp.shape[0]):
            prd[:, i * size:(i + 1) * size, j * size:(j + 1) * size] = tmp[j]

    # thresholds for each class
    trs = [0.4, 0.1, 0.4, 0.3, 0.3, 0.5, 0.3, 0.6, 0.1, 0.1]
    for i in range(classes_num):
        prd[i] = prd[i] > trs[i]

    return prd[:, :img.shape[0], :img.shape[1]]

def test_1(id):
    rgb_img = os.path.join('/home/dataset/sixteen_band', '{}_M.tif'.format(id))
    rgb_img = tiff.imread(rgb_img)
    image = np.rollaxis(rgb_img, 0, 3)
    print(image.shape)
    #load the trained model
    model = SegNet()
```

```
    model.load_weights("/home/content/weights-59-0.4575.hdf5")
    #predict the mask input image id
    mask = predict(id, model)
    class_list = ["Building", "Structure", "Road", "Track", "Tree", "Crop",
"RunningWater", "StandingWater", "LargeVehicle", "SmallVehicle"]

    img = np.zeros((image.shape[0],image.shape[1],3))
    img[:,:,0] = image[:,:,4]
    img[:,:,1] = image[:,:,2]
    img[:,:,2] = image[:,:,1]

    for i in range(classes_num):
        plt.figure(figsize=(25,25))
        ax1 = plt.subplot(131)
        ax1.set_title('image ID:6120_2_0')
        ax1.imshow(fix_contrast(img))
        ax2 = plt.subplot(132)
        ax2.set_title("predict "+ class_list[i] +" pixels")
        ax2.imshow(mask[i], cmap=plt.get_cmap('gray'))
        plt.show()

test_1("6120_2_0")

def test_2(data_point, data_point_label):
    image = data_point
    mask = data_point_label

    model = SegNet()
    model.load_weights("home/content/my_trained_model.hdf5")
```

```
    predicted_mask = model.predict(image)
    print(image.shape, predicted_mask.shape)

    score = jaccard_coef(mask, predicted_mask)
    return score

score = test_2(image, mask)

print("The jaccard score on a test data point is", score.numpy())
```

ДОДАТОК Ж
ПРОТОКОЛ ПЕРЕВІРКИ НАВЧАЛЬНОЇ (КВАЛІФІКАЦІЙНОЇ) РОБОТИ

Назва роботи: Система розпізнавання та класифікації об’єктів на аерофотознімках з використанням нейромереж._____
Тип роботи:_____ кваліфікаційна робота _____
(кваліфікаційна роботи, курсовий проект (робота), реферат, аналітичний огляд, інше (зазначити))

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

Науковий керівник _____к.т.н., доц. Ткаченко О. М._____
(прізвище, ініціали, посада)

Показники звіту подібності

Plagiat.pl (StrikePlagiarism)		Unicheck	
КП1			
КП2		Оригінальність	95,8
Тривога/Білі знаки	/	Схожість	4,2

Аналіз звіту подібності (відмітити подібне)

- ✓ Запозичення, виявлення у роботі, оформлені коректно і не містять ознак плагіату.
- □ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності і відсутності самостійності її автора. Робот направити на доопрацювання.
- □ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Заявляю, що ознайомлений(-на) з повним звітом подібності, який був згенерований Системою щодо роботи (додається)

Автор _____ _____Ярмощук Д. О._____
(підпис) (прізвище, ініціали)

Опис прийнятого рішення

_____
_____

Особа, відповідальна за перевірку_____ _Захарченко С.М.__
(підпис) (прізвище, ініціали)

Експерт _____ _____

(за потреби) (підпис) (прізвище, ініціали)