Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
Кафедра Комп'ютерних інтелектуальних технологій та систем

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Виявлення морського сміття за допомогою комерційних супутникових зображень і глибокого навчання

(тема)

Виконав:

Студент 2 курсу, групи КІТм-21-1

Личман М.С.

(прізвище, ініціали)

Спеціальність 123 «Комп'ютерна інженерія»

Тип програми освітньо-професійна

Освітня програма Комп'ютерні інтелектуальні технології

Керівник ст. викладач, Ілюнін О.О.

(посада, прізвище, ініціали)

Допускається до захисту

(підпис)

Зав. кафедри О. Г. Руденко

(підпис)

2022 р.

Харківський національний університет радіоелектроніки

Факультет	Комп'ютерної інженерії та управління
Кафедра	Комп'ютерних інтелектуальних технологій та систем
Рівень вищої освіти	другий (магістерський)
Спеціальність	123 Комп'ютерна інженерія
Тип програми	освітньо-професійна
Освітня програма	Комп'ютерні інтелектуальні технології

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____»_____2022р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Личману Микиті Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Виявлення морського сміття за допомогою комерційних супутникових зображень і глибокого навчання.

затверджена наказом по університету від «___» _____ 2022р. № _____

2. Термін подання студентом роботи до екзаменаційної комісії _____

3. Вихідні дані до роботи (проекту)

1) Мова програмування Python
2) Бібліотеки Tensorflow, Keras, NumPy, Kivy
3) Утіліта Label Maker
4) Документація по Tensorflow Object API, Kivy та NumPy
5) Супутникові зображення морського сміття PlanetScope
6) Веб-сервіс Nasa ImageLabeler
7) Література

4. Перелік питань, що потрібно опрацювати в роботі

1) Аналіз предметної області
2) Використання згорткових нейронних мереж для розпізнавання образів
3) Створення набору даних для глибинного навчання
4) Опис технологій розробки та інструментальні засоби. Розробка додатку

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням кафедри)

Слайд-презентація – 15 слайдів

6. Консультанти розділів роботи (п.6 включається до завдання за наявністю консультантів згідно до наказу, зазначеному у п.1)

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

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Ознайомлення з літературою	07.11.22-08.11.22	вик
2	Аналіз публікацій за напрямком кваліфікаційної	09.11.22-10.11.22	вик
	роботи		
3	Огляд існуючих рішень	11.11.22-13.11.22	вик
4	Дослідження технологій комп'ютерного зору для	14.11.22-18.11.22	вик
	розпізнавання образів		
5	Дослідження шуму зображення	21.11.22-23.11.22	вик
6	Вибір технологій розробки та інструментальних	24.11.22-25.11.22	вик
	засобів		
7	Розробка програми	28.11.22-30.11.22	вик
8	Розробка інструкції користувача	01.12.22-02.12.22	вик
9	Оформлення матеріалів кваліфікаційної роботи	05.12.22-09.12.22	вик

Дата видачі завдання _____

Студент _____
(підпис)

Керівник роботи _____ ст. викладач, к.т.н. Ілюнін О.О.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка магістерської кваліфікаційної роботи: 16 аркушів презентації, 64 стор., 8 рисунків, 5 таблиць, 21 джерело посилань.

НЕЙРОННА МЕРЕЖА, CONVNET, СУПУТНИКОВІ ЗНІМКИ, МОРСЬКЕ СМІТТЯ, NASA, MARINE DEBRIS, МІКРОПЛАСТИК, PYTHON, TENSORFLOW, KERAS, KIVY, NUMPY, SSD, FPN, RESNET.

Метою кваліфікаційної роботи є створення глибинної нейронної мережі для аналізу зображень з комерційних супутникових знімків.

В ході виконання кваліфікаційної роботи було виконане наступне:

- обґрунтовано актуальність обраної теми, оглянуто наукові публікації, які стосуються теми кваліфікаційної роботи та існуючі рішення на ринку;
- розглянуто та проаналізовано технології нейронних мереж та глибинного навчання;
- розглянуто існуючі рішення з розпізнавання об’єктів, бібліотеки та фреймворки, що були використані під час дослідження;
- описано технології розробки та інструментальні засоби, що використовувалися для створення глибинної нейронної мережі;
- зібрано набір даних для навчання глибинної нейронної мережі;
- створено кросплатформенну утиліту для аналізу фотографій користувача навченою моделью на наявність на ній морського сміття;

ABSTRACT

Master's qualification work contains 64 pages, which contains 8 figures and 5 tables. The list of references contains 21 titles.

NEURAL NETWORK, CONVNET, SATELLITE IMAGERY, MARINE DEBRIS, NASA, MICROPLASTIC, PYTHON, TENSORFLOW, KERAS, KIVY, NUMPY, SSD, FPN, RESNET.

The purpose of the qualification work is the creation of a deep neural network for image analysis from commercial satellite images.

In scope of the qualification work, the following was performed:

- the relevance of the chosen topic is substantiated, scientific publications related to the topic of the qualification work and existing solutions on the market are reviewed;

- considered and analyzed the technologies of neural networks and deep learning;

- apparently existing object recognition solutions, libraries and frameworks that were used during the research;

- development technologies and tools used to create a deep neural network are described;

- the dataset was collected for training a deep neural network;

- a cross-platform utility was created to analyze the user's photos with a trained model for the presence of marine debris;

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп’ютерної інженерії та управління
Кафедра Комп’ютерних інтелектуальних технологій та систем

АНОТАЦІЯ

КВАЛІФІКАЦІЙНОЇ РОБОТИ

Рівень вищої освіти другий (магістерський)

Виявлення морського сміття за допомогою комерційних супутникових зображень і глибокого навчання
(тема)

Виконав:
Студент 2 курсу, групи КІТм-21-1
Личман М.С.
(прізвище, ініціали)

Спеціальність 123 «Комп’ютерна інженерія»

Тип програми освітньо-професійна

Освітня програма Комп’ютерні інтелектуальні технології

Керівник ст. викладач, к.т.н. Ілюнін О.О.
(посада, прізвище, ініціали)

2022 р.

АНОТАЦІЯ

Личман М.С. Тема магістерської кваліфікаційної роботи – Виявлення морського сміття за допомогою супутникових зображень і глибокого навчання.

У магістерській кваліфікаційній роботі вирішено актуально проблему виявлення морського сміття за допомогою комерційних супутникових зображень і глибокого навчання.

Метою кваліфікаційної роботи є дослідження згорткових моделей класифікації об’єктів.

Об’єктом дослідження цієї роботи є процес глибинного навчання обраної моделі.

Предмет дослідження – глибинна нейронна мережа для класифікації об’єктів.

Плавуче морське сміття є глобальною проблемою забруднення, яка призводить до втрати морського та наземного біорізноманіття. Великі смуги морського сміття також становлять небезпеку для навігації океанських суден. Використання даних спостереження Землі та методів штучного інтелекту може революціонізувати виявлення плаваючого морського сміття на супутникових зображеннях і прокласти шлях до глобальної системи моніторингу для контролю та запобігання накопиченню морського сміття в океанах.

Результатом написання **першого розділу** магістерської кваліфікаційної роботи є огляд літератури з теми дослідження, наукових публікацій та існуючих рішень.

Встановлено, що в останні десятиріччя увагу наукових установ, світових лідерів та й людства в цілому дедалі більше захоплюють проблеми забруднення планети. Тисячі статей та досліджень на цю тему з’являються

щороку, і ситуація виглядає так, що особливої альтернативи (як-то колонізація Марса) наразі не існує.

Наприкінці 20 – на початку 21 століття науковий прогрес значно розширив можливості людства щодо дослідження планети. Ключовим елементом в цій схемі стало розміщення на орбіті тисяч супутників, які наразі збирають всі доступні різновиди інформації. Паралельний розвиток комп'ютерної техніки й алгоритмів обробки даних дозволили використовувати і утилізувати величезні масиви даних, накопичені із самого початку освоєння космосу.

Також встановлено, що в останні десятиріччя увагу наукових установ, світових лідерів та й людства в цілому дедалі більше захоплюють проблеми забруднення планети. Тисячі статей та досліджень на цю тему з'являються щороку, і ситуація виглядає так, що особливої альтернативи (як-то колонізація Марса) наразі не існує.

Встановлено, що тенденції до зростання населення планети, всі глобальні проблеми матимуть тенденцію до поглиблення. Це стосуватиметься й проблеми світового забруднення, тому питання переробки та утилізації сміття ще неодноразово ставатиме на порядку денному. Саме тому найбільш ефект для майбутнього дадуть ті заходи, що будуть розпочаті якнайшвидше.

Другий розділ роботи присвячений дослідженню сучасних рішень, пов'язаних з обробкою супутникових даних та досліджень в галузі створення нейронних мереж, що здатні ідентифікувати морське сміття.

Встановлено, що нейронні мережі є функціональною одиницею глибокого навчання і, як відомо, імітують поведінку людського мозку на вирішення складних завдань, керованих даними.

Аналіз супутникових знімків разом із моделюванням ходу подій у Світовому океані може стати проривним дослідженням саме тому, що раніше можливості людства щодо спостереження за планетою були обмежені аерофотозйомкою та спостереження команд кораблів, що ходили в рейси,

спостерігаючи лише мізерну долю океанічної поверхні. Обробка супутникових наборів даних з використанням розроблених моделей вирішує одразу дві проблеми. Перша – величезні об’єми даних можуть бути оброблені так швидко, як дозволяє апаратна частина системи. Друга – результати можуть бути використані для подальших досліджень без додаткової обробки, якщо модель була навчена необхідним чином.

Далі встановлено, що у межах роботи доцільно розглядати роботу компаній-гігантів у даному напрямку. Існує дуже велика кількість наукових досліджень від NASA, Amazon, які описують роботу з супутниковими даними та забрудненям світового океану.

Визначено, що як основу для роботи буде використано роботу команди NASA-IMPACT і їх проекту, присвяченому пошуку морського сміття у воді. Також будуть запроваджені ідеї для покращення описаної ними моделі.

В **третьому розділі** роботи представлено дослідження згорткових нейронних мереж, проаналізовані їх недоліки та переваги в межах поставленої задачі.

Визначено, що для задач комп’ютерного зору найбільш доцільними вважаються згорткові нейронні мережі, які у свою чергу поділяються на різні типи.

Далі встановлено, що для задач класифікації сміття на супутниковому зображенні доцільно очікувати, що модель здатна швидко працювати з маленькими по розміру об’єктами. Було обрано модель Single Shot Detector та впроваджені деякі покращення у її архітектурі, зокрема розглянуто Feature Pyramid Network та її переваги у роботі з дрібними об’єктами.

В **четвертому розділі** роботи представлена реалізація утиліти додатку, процеси створення навчального датасету, опису конфігурації моделі та власне процес глибинного навчання і його результату.

Визначено, що для завдання створення моделі гарним рішенням буде використовувати Tensorflow Object Detection API. За допомогою платформи

можна контролювати кожен процес роботи з моделью та зручним чином модифікувати існуючі моделі під свої задачі.

Наукова новизна роботи лежить у наступному. В якості вдосконалення моделі при конфігуруванні SSD в якості змін архітектури застосовані інструкції, які зазвичай використовуються в FPN, що надало змогу обробляти дрібні об’єкти без значних втрат швидкості обробки зображень, не використовуючи більш тривалі: R-CNN, F-CNN, YOLO. Таким чином алгоритм набув адаптивності, що важливо для загальної швидкості виконання завдань обробки зображень.

Важливою перевагою використання платформи Tensorflow, бібліотек Keras, NumPy та Kivy є зручність у використанні та легкість в описанні, завдяки мові програмування Python.

Проведені тести додатку наочно демонструють, що модель, навчана з розміром партії 12 досить непогано справляється із задачею класифікації морського сміття. Глибинні методики навчання були впроваджені та розглянуті у роботі.

Програма-утіліта була протестована на декількох системах, що показує надійність обраних інструментів та бібліотек. Експериментально встановлена ефективність використання глибинної нейронної мережі Single Shot Detector для класифікації морського сміття на супутникових зображеннях.

СУПУТНИКОВІ ЗОБРАЖЕННЯ, КЛАСИФІКАЦІЯ, ЗГОРТКОВА НЕЙРОННА МЕРЕЖА, ГЛИБИННЕ НАВЧАННЯ, NASA, КРОСПЛАТФОРМНЕ ПО, МОРСЬКЕ СМІТТЯ.

Використані в роботі публікації здобувача, керівника та співробітників кафедри:

1. Бессонов А.А. Интеллектуальная система распознавания лиц с использованием сверточной нейронной сети / А.А. Бессонов, Ю.Ю. Белов //

В сб. тез. докл. по материалам 4-ой междунар. научно-технич. конф. «Проблеми інформатизації», Черкаси-Баку-Бельсько-Бяла-Полтава, 2016. - С. 21.

……..

ЗМІСТ

ВСТУП

Плавуче морське сміття є глобальною проблемою забруднення, яка призводить до втрати морського та наземного біорізноманіття. Великі смуги морського сміття також становлять небезпеку для навігації океанських суден. Використання даних спостереження Землі та методів штучного інтелекту може революціонізувати виявлення плаваючого морського сміття на супутникових зображеннях і прокласти шлях до глобальної системи моніторингу для контролю та запобігання накопиченню морського сміття в океанах.

Метою написання магістерської дисертації є дослідження згорткових моделей класифікації об’єктів.

При написанні роботи були поставлені наступні задачі:

1. Виконати огляд літератури з теми дослідження. Розглянути задачі аналізу супутникових зображень та існуючі підходи.

2. Дослідити особливості застосування нейронних мереж для класифікації об’єктів.

3. Розробити додаток-утиліту, який буде використовувати навчену глибинну нейронну мережу для виявлення морського сміття на супутникових зображеннях.

Об’єктом дослідження магістерської дисертації є процес глибинного навчання обраної моделі.

Предметом дослідження є глибинна нейронна мережа для класифікації об’єктів.

При написанні роботи були застосовані методи системного аналізу, порівняння, штучних нейронних мереж, модельного експерименту.

Практична значимість отриманих результатів полягає у можливості використання розробленої глибинної нейронної мережі для виявлення морського сміття на супутникових зображення та вирішення аналогічних задач по дослідженню засмічення ландшафту.

1 АКТУАЛЬНІСТЬ ЗАДАЧІ ТА АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Актуальність обраної теми

Давнє китайське прислів’я каже: «Найкращий момент, аби посадити дерево, минув 20 років потому. Коли ж прийде наступний найкращий момент? Зараз». В середині 20 сторіччя, коли людство почало нарощувати виробництво пластмас, проблема глобального забруднення ще не стояла на повістці денній. З того часу виробництво пластику сягнуло планетарних масштабів, так як промисловість потребувала все більше й більше цього «диво-матеріалу». Не слід забувати й про те, що населення Землі за останні 70 років зросло втричі, а споживання промислових товарів – на порядок. Власне, чому саме пластик? Причин є безліч. Бо це недорого. Бо це універсально й зручно. Бо його хімічні властивості дозволяють застосовувати цей матеріал в багатьох сферах виробництва. Тим не менш, солідний список переваг є двосічним мечем, адже надійність та довговічність пластику й особливості його молекулярної структури призводять до того, що він розкладається надзвичайно повільно.

Значним поштовхом до вивчення проблеми пластмас стало відкриття так званого «Великої сміттєвої плями» у Тихому океані [1]. Точний розмір цієї плями, яку ще називають «сміттєвим континентом», складає близько 1 мільйона квадратних кілометрів, тобто в півтора рази більше ніж площа України [2]. За кілька років до відкриття існування «плями» було передбачене на основі більш локальних спостережень (на Алясці [3]), яке виявило, що течії відносять сміття від американського узбережжя до центру Тихого океану. Пізніші дослідження показали, що сміття з річок східної Азії, таких як Янцзи, Хуанхе, Амур, також із часом опиняється в області «плями». Варто зазначити, що «сміттєвий континент» більш точно описується як скупчення мікропластику. Це означає, що пластикове забруднення можна побачити,

узявши проби океанічної води, а також із палуби корабля, що пропливає поруч. Великі уламки пластику тут є винятком, а не правилом (рис.1.1).

Рисунок 1.1 – Знімок «Великої сміттєвої плями» у Тихому океані

Минає щонайменше декілька років від потрапляння пластику в океан до моменту, коли цей пластик опиняється в плямі. Значна частина сміття протягом цього часу частково розкладається під дією сонячних променів, а також механічних сил океанічних хвиль та зіткнень з іншими шматками пластику. Досліджуючи маршрути прямування пластику від точки, де його викинули, до «континенту», вчені можуть отримати більш точне уявлення про океанічні течії, котрі в контексті пластикового сміття нагадують блендер.

Інший цікавий епізод, пов’язаний із пластмасовими виробами та течіями, увійшов в історію як «Friendly floatees» [4] (з англ. – дружелюбні пловці).

У січні 1992 року контейнеровоз, на борту якого були контейнери із пластиковими іграшками, потрапив в ураган на півдорозі між Гонконгом та Західним узбережжям США. В результаті частину контейнерів змило в океану, і частина пластикових іграшок розпочали своє багаторічне турне Світовим океаном. Іграшки досягли Аляски та Камчатки у 1995 році, Каліфорнії – у 1996, Східного узбережжя США (через Північний океан) – у 2000, а у 2003 та 2007 роках іграшки були знайдені на пляжах Великобританії. За неповних десять років пластикове сміття з одного контейнеру побачило половину планети.

Ймовірно, проблемі пластмас не присвячували б стільки досліджень, якби не значний вплив на довколишнє середовище. Окрім деяких шкідливих видів пластику, які виділяють токсичні речовини під дією сонячних променів, він також стає частиною харчових ланцюгів морських та океанічних видів. Тварини та риби можуть плутати планктон та частинки мікропластику, через що він опиняється в їх організмі, повільно піднімаючись все вище і вище за ланцюгом, на вершині котрого зараз стоїть людина [5]. За інформацією ВООЗ, наразі немає жодних підстав для того, аби бити на сполох через отруєння людства пластиком. Проте, за їх словами, ця тема все ще недостатньо досліджена, щоб робити остаточні висновки. С іншого боку, зростання пластикового забруднення до небезпечного для людини рівня є лише питанням часу, якщо не докласти зусиль для досягнення протилежного ефекту.

Зрештою, можна сказати, що людство не впоралось із тим, щоб «посадити дерево 20 років тому». С іншого боку, в той період технічні можливості людства було значно обмеженіші, і методи, котрі з легкістю можуть бути застосовані сьогодні – були недоступними. Саме завдяки технічному прогресу, що відбувся за останні півстоліття, можливо реалізувати ідею даної наукової роботи, а саме – відстеження морського сміття за допомогою супутникових знімків. В контексті проблеми пластикового забруднення цей крок є лише першим із багатьох: так, склавши таймлайн зображень певної території, можна відслідкувати кількість сміття та його маршрут. Це дозволить поглибити знання стосовно океанічних течій, а також відслідковувати та відловлювати сміття

значно раніше, ніж воно опиниться посередині океану у напіврозкладеному стані.

Проте в масштабах планети не можна просто взяти супутникові знімки та вручну шукати на них сміття та зіставляти зображення між собою. Чимось подібним могли б займатися головні герої детективу у жанрі нуар, але не ми. Дякуючи лавиноподібному збільшенню можливостей комп'ютерної техніки в останні десятиріччя, сьогодні можна створити програмне забезпечення, яке буде самостійно виявляти сміття на супутникових знімках, використовуючи нейронні мережі для глибокого навчання алгоритму. Роботи у подібних напрямках уже деякий час ведуться провідними світовими інституціями, такими як NASA, зі своєю базою супутникової інформації, а також науково-технічного потенціалу. Наступний найкращий момент – зараз.

1.2 Огляд наукових публікацій та існуючих рішень

Історично склалося, що лідируючу роль у науково технічному-прогресі обіймають країни Західної Європи, а також США. Вчені саме цих країн першими почали звертати увагу на проблеми всесвітнього забруднення та шукати можливі шляхи розв'язання цієї проблеми.

Найбільше кроків у дослідженні планети, в тому числі з космосу, зробили США руками космічної агенції – NASA. Завдяки даним, що були накопичені і продовжують з'являтися щодня з використанням супутників, найбільш помітні роботи в області морського сміття (англ. – «marine debris») були виконані американськими вченими на технічній базі NASA.

Перше дослідження, що заслуговує на увагу, було виконано вченими Мічиганського університету. Вони використали супутникові дані, які були зроблені системою CYGNSS [6] – глобальна супутникова система, що займається прогнозуванням ураганів. Супутники, задіяні в цій системі, мають спеціальні радари, котрі замірюють ocean roughness – нерівності поверхні океану, тобто хвилі. Увагу вчених привернув той факт, що в деяких регіонах

океану даний параметр відрізняється від очікуваного значення, рівні вітру та хвиль не накладалися одне на одного так, як в інший регіонах [7].

Порівнявши перелік регіонів з картою відомих накопичень морського сміття, вони прийшли до висновку, що причиною такого відхилення можуть бути пластикові відходи, що вкривають собою поверхню океану. Це дослідження може стати першим кроком у складенні карти забруднення океану, що дозволить виявити найбільш «пластикові» області океану та виконати їх точкове очищення.

Наступне дослідження, котре варто розглянути, є складовою програми NASA IMPACT [8], метою якого є всебічне використання супутникової інформації, зібраної різноманітними супутниками NASA. Мета дослідження полягає у дослідженні поверхні океану на предмет будь-яких об’єктів, що можуть там знаходитися і котрі видно на супутникових знімках. Такими об’єктами можуть бути кораблі, хвилі, а також власне морське сміття. Також не слід забувати про хмари, що спотворюють чисте зображення моря, й власне артефакти супутникових знімків.

Варто зазначити, що обидві роботи наразі є лише дослідними прототипами. Пошук найліпшого застосування для них є важливою задачею, так само як і даної роботи. Потрібно створити ще не один десяток програмних комплексів, перш ніж можна буде отримати такий продукт, котрий напряму допомагатиме вирішувати проблему пластикового забруднення моря.

2 РОЗПІЗНАВАННЯ ОБРАЗІВ З СУПУТНИКОВИХ ЗОБРАЖЕНЬ

2.1 Коротка історія комп’ютерного зору

Спроби навчити комп’ютер «бачити» у звичному нам розумінні важко уявити, якщо не мати уявлення про те, як влаштований зір у живих істот. Тому не дивно, що однією із перших робіт, присвячених нейронним мережам, є дослідження на тему того, як мозок кішки розрізняє предмети у просторі. Авторами роботи були нейропсихологи Девід Хубель та Торстен Візель, які навряд чи здогадувалися, що їх дослідження є піонером саме в області комп’ютерного зору.

Тим не менш, це дослідження було здебільшого теоретичним в контексті саме комп’ютерного зору. Основу та базу заклали дві події: першою є створення комп’ютерного перцептрона [9], в другою – створення Лоуренсом Робертсом системи розпізнавання форм предметів [10]. Чому це настільки важливо? Комп’ютер, щоб «бачити», використовує додаткові пристрої – наприклад, камеру. Камера може надати комп’ютерові двовимірне зображення, яке є лише «зрізом» об’єктивної тривимірної реальності. Робота Лоуренса поклала початок перетворення фотографії у 3-D представлення, котре давало б машині вичерпну інформацію про ту частину навколишнього світу, куди направлена камера.

Однак і це було лише одним із перших кроків, так як технічні можливості комп’ютерів в середині 60-х років XX сторіччя не дозволяли виконання достатньо складних розрахунків. Лише 20 років потому з’явилися роботи, які могли створювати для себе адекватну картину навколишнього середовища та виконувати в ньому певні дії, зазвичай специфічні для конкретного робота та закладеної в нього програми.

Найбільш критичним параметром надійності системи комп’ютерного зору було те, наскільки достовірну картину навколишнього світу «бачить»

комп’ютер. Лише наприкінці 90-х років минулого століття вдалося створити перші комерційні системи автоматичної навігації автомобілів, що було значим досягненням, зважаючи на потенційно можливу кількість ситуацій, що можуть скластися на дорозі.

Поряд із покращенням апаратури, створювалися також більше просунуті та потужні алгоритми, що дозволяли втілювати все більш і більш комплексні системи в реальність. Із сучасних нам прикладів[11] використання комп’ютерного зору можна згадати роботів-хірургів, які раніше страхували хірургів-людей, а зараз можуть самі виконувати операції під їх наглядом; Face ID від Apple, яка по-справжньому рознесла розпізнавання обличь в маси; Amazon Alexa, яка не лише розпізнає голос власника, а й може підтримувати співбесіду на будь-яку тему.

2.2 Комп’ютерний зір та напрямки його застосування

Відповідаючи на запитання «в яких саме сферах людського життя можна застосувати комп’ютерний зір», кращою відповіддю буде інше запитання: «А в яких не можна?». Справді, будь-де можна знайти простір для автоматизації та оптимізації, полегшивши тим самим життя усіх учасників. Тим не менш, коли технології розпізнавання лише починали знаходити своє застосування, імплементація подібного рішення могла бути дуже дорогою і мати сумнівну ефективність.

Але вибір є не завжди. Наприклад, під час Холодної війни[12] немає іншого вибору, окрім як використовувати супутника для відслідковування дій ворога. Додаймо той факт, що ворог не геть дурний і буде маскувати свої пускові шахти, аеродроми та місця зберігання техніки – і отримаємо нетривіальну задачу, котра не завжди може бути розв’язана людиною.

Іншим прикладом задачі, з якою машина впорається ліпше – аналіз відеоматеріалів. Наприклад, якщо потрібно продивитися матеріали з сотень камер спостереження у стислі терміни, намагаючись знайти певний номер

машини, чи певне обличчя, або й просто людину з характерною зовнішністю. Очевидно, можливості не обмежені лише пошуком людини – це може бути будь-який об’єкт, який система вміє розпізнавати. Подібні напрацювання стали основою для роботів, котрі працюють на складах Amazon, під наглядом людей, а подекуди – й замінюючи їх.

Значне застосування комп’ютерний зір знаходить в медицині. Окрім вже згаданих роботів-хірургів [14], активно використовується комп’ютерна система для допомоги в діагностиці, здатна за набором вхідних даних (аналізи, кардіограма, УЗД та інше) підтвердити, або уточнити діагноз лікаря, або зробити власну пропозиції щодо можливого діагнозу та лікування. Приклад використання комп’ютерного зору можна побачити на рисунку 2.1.

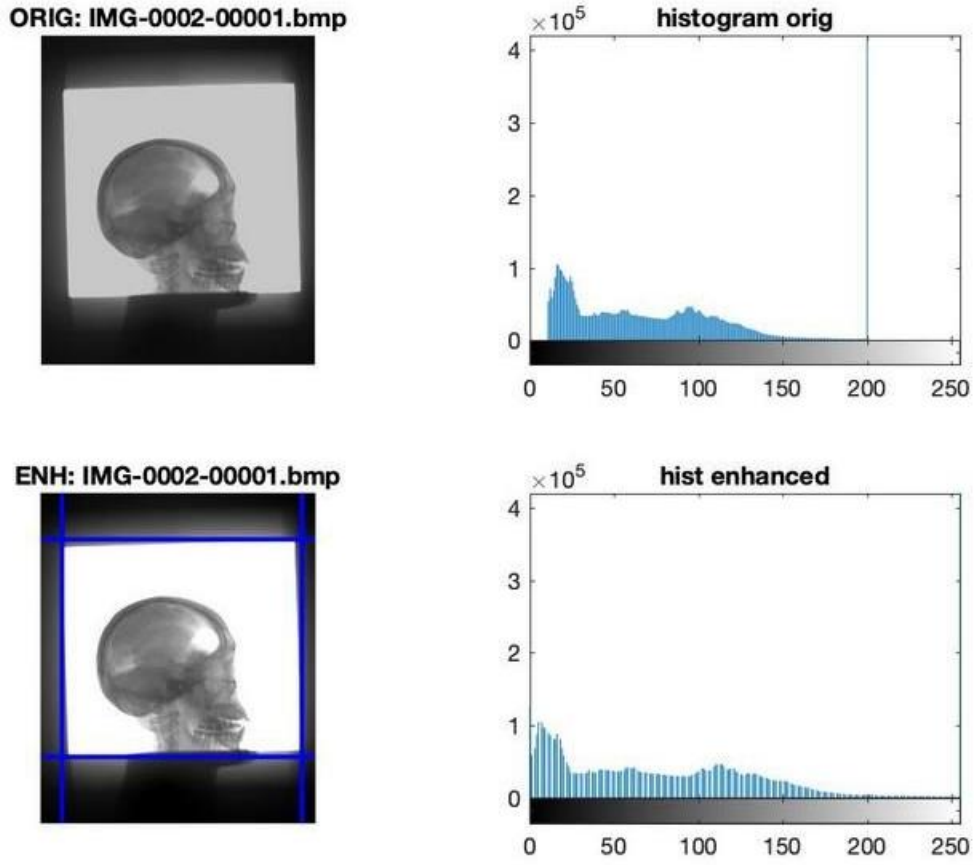


Рисунок 2.1 – Приклад використання згорткової нейронної мережі для аналізу вад мозку людини

На заводах вже тривалий час застосовують промислових роботів [15], здатних виготовляти деталі із заданою конфігурацією та алгоритмом, а також збирати цілісні механізми з машинною точністю. Саме ця область, ймовірно, найбільш близька до початкових досліджень комп'ютерного зору, де метою було створення достовірного тривимірного зображення. Завдяки цьому зібраний автомобіль не розвалиться на швидкості 200 кілометрів на годину, а турбіна на ГЕС не зламається під час роботи.

В розвинених країнах комп'ютерний зір знаходить застосування в сільському господарстві [16]. Завдяки широкому наборові датчиків, які здатні відслідковувати усі параметри, що так чи інакше впливають на ріст та дозрівання сільгосппродукції, можна майже повністю автоматизувати цикл вирощування рослин, забезпечивши кожній культурі ідеальні умови для зростання.

В умовах, що складаються у 2022 році, не можна обійти стороною військове застосування [17] систем комп'ютерного зору. В ситуації, коли бойові дії без повітряної розвідки неможливі в принципі, ідентифікація об'єктів у реальному часі значно скоротить час на прийняття рішень. Безпілотник, який під час розвідки зразу «підсвічує» свої та ворожі позиції, ідентифікує солдатів та техніку за номенклатурою і при необхідності може самостійно знищити ту чи іншу ціль або скинути міну, може сильно змінити співвідношення сил на полі бою. В парі із супутниковою розвідкою безпілотники з системою комп'ютерного зору здатні покривати потреби як оперативної, так і тактичної розвідки. Додаймо до цього рівняння ідентифікацію ворожих безпілотників та створення карти того, що «бачить» ворог, і хід бою стане в рази більш передбачуваним.

Окрім розвідки, значна частина армійських систем має інструменти для розпізнавання конкретних цілей (зокрема, сюди можна віднести так звані «розумні» снаряди, а також системи ППО та ПРО). Застосування якісних алгоритмів розпізнавання, що дозволять звести нанівець маскування та механізми, спрямовані проти ППО, дронів та артилерії, спроможні значно

підсилити потенціал кожної дії, що має бойовий ефект, з використанням подібних систем.

В підсумку можна зробити висновок, що комп’ютерний зір може бути застосований майже усюди. Але в умовах обмежених ресурсів необхідно чітко виставляти пріоритет і застосовувати системи розпізнавання саме там, де це дасть найбільш ефект, незалежно від контексту застосування.

1.3 Застосування комп'ютерного зору для аналізу супутникових зображень.

Неможливо переоцінити практичну корисність комп’ютерного зору в контексті його використання для глибокого навчання систем штучного інтелекту. Завдяки широкій доступності багатьох бібліотек, як-то TensorFlow, PyTorch, OpenCv, Fastai, а також публічно доступно датасетів (CIFAR-0, Imagenet, IMDB etc), наразі дуже легко знайти всі необхідні інструменти для розв’язання задач у даному напрямі.

В залежності від задачі, необхідно обрати дані потрібного спектру, відшукавши відповідний датасет, та, за необхідності, додатково обробити його. Це може бути власне робота із зображенням, підгін розмірів зображень під певний шаблон, зрозумілий для нейронної мережі, що навчається, а також відсіювання заздалегідь некоректних даних.

Після підготовки датасету, перша задача, яку потрібно вирішити – це знайти або створити механізм для розпізнавання потрібних елементів на супутникових знімках. Вони можуть бути специфічними для задачі, наприклад, для дослідження врожаїв з космосу ми б використали інформацію про те, якого кольору набуває поле певної рослини залежно від стадії вегетації. Наступною задачею буде аналіз границь областей, котрі наш алгоритм вирізнить серед множини пікселів знімку. Знайдені границі допоможуть моделі сформувати полігони, в області яких і будуть знаходитися об’єкти, що нас цікавлять, як то коти, ліси або кораблі. Для оптимізації роботи моделі можна застосувати

додаткові механізми, як-то маніпуляції з різкістю, контрастністю та сатурацією зображення.

Останнім кроком буде створення навчальної вибірки, яка б могла «розказати» нашій моделі, що є що на кожній окремо взятій картинці. Для цього слід використати вже існуючі датасети, які можна знайти в мережі інтернет та в спеціалізованих джерелах. Маючи з чим порівнювати, створена нами модель зможе сама опрацьовувати зображення, вирізняючи на ньому певні елементи та вказуючи їх приналежність. На основі даного результату, можна (із певною імовірністю та похибкою) виконати класифікацію зображень, які модель отримає на аналіз.

Таким чином, працюючи з нейронними мережами та супутниковою фотографією, можна створювати корисні для бізнесу та прикладних проектів моделі, здатні обробляти значні набори даних в автоматичному режимі. Тим не менш, весь процес від початку до кінця потребує неабияких навичок, а подекуди – доступу до комерційної інформації та пропрієтарного програмного забезпечення, тому виконання подібних проектів з нуля може бути неабияким викликом навіть для технічно підкованого спеціаліста.

2.4 Оптичне розпізнавання морського сміття

В задачі зменшення забруднення моря першим пунктом стоїть задача його ідентифікації. Можна скільки завгодно аналізувати його наявність у Світовому океані та можливості очищення, проте, як кажуть «чисто там, де не смітять». Отже важливою проблемою постає пошук джерел сміття.

Дана робота базується на виключно супутниковому рішенні, тому можна розглянути три основних підходи до ідентифікації сміття. Перший – поки що теоретичний – це збільшення роздільної здатності супутникової зйомки на порядок, що дозволило би ідентифікувати сміття за формою, назвою чи навіть штрих-кодом на упаковці. Наразі такого рівня розпізнавання можна досягти лише за допомогою безпілотників, або авіації, що не є вигідним рішенням,

якщо врахувати потенційні затрати. Єдиною умовою є відносна цілісність сміття, тобто воно ще не встигло розкластися. Наразі реалізувати такий підхід технічно неможливо, проте із часом все може змінитися.

Другий підхід, застосований у схожій роботі вченими з Мічиганського університету, не є оптичним, проте також може бути використаний як елемент майбутньої системи. Він полягає у аналізі макропоказників Світового океану, котрі можна збирати за допомогою супутників зі специфічним функціоналом. Тобто дослідження виконується методом непрямого аналізу, подібно до того, як відбуваються дослідження областей глибокого космосу.

Третій підхід, який можна розглядати як логічне продовження даної роботи, полягає у дослідженні прибережних областей, особливо в областях активного проведення рибної ловлі, а також дельт крупних річок. Кольорова гама в таких регіонах буде значно відрізнятися під інших морських регіонів, зокрема через забруднення відходами людської діяльності. Тому аналіз супутникових знімків дозволить збирати відносно «свіжу» інформацію, аналізувати її, робити прогнози та приймати заходи щодо зниження забруднення.

Будь-який метод, окрім другого, потенційно може бути покращеним за допомогою інструменту, розробленого китайськими дослідними[18]. Його ідея полягає у збиранні інформації з вуличних камер, які виходять на пляжі, по усьому узбережжю Східної Азії. На таких камерах часто можна побачити сміття, і відповідно навчена комп’ютерна модель може як з’ясувати звідки сміття з’явилось на даному пляжі, так і відслідкувати збільшення або зменшення інтенсивності забруднення. Нейронна мережа формує тривимірну модель сцени, и намагається знайти та ідентифікувати сміття за наявними зразками.

3 ГЛИБОКЕ НАВЧАННЯ

3.1 Історія глибинного навчання

Глибоке навчання – метод машинного навчання, що має на меті вибудовування загальних правил у вигляді штучної нейронної мережі під час процесу навчання [19]. В основі його лежить імітація роботи мозку як природної нейронної мережі, а процес називається глибоким, оскільки структура штучної нейронної мережі має в своєму складі декілька вхідних, вихідних та прихованих шарів.

Глибоке навчання є підкласом машинного навчання, і основна різниця полягає саме у використанні нейронної мережі. Завдяки нейронним мережам процес і можна назвати глибоким, тоді як алгоритми машинного навчання здебільшого є спеціалізованими під певну задачу.

Яскравим прикладом реалізації штучного інтелекту саме з глибоким навчання є система AlphaGo [20], створена компанією Google DeepMind, і яка змогла обіграти професійного гравця в гру «го» у серії з рахунком 4:1. Специфічною особливістю AlphaGo було те, що вона була побудована на загальних принципах машинного навчання, не застосовуючи специфічні для даної гри прийоми. Саме ж навчання проводилось на значній вибірці матчів професійних гравців.

Отже, глибоке навчання дозволяє обмежитися використанням загальних принципів Machine Learning, щоб створити модель, яка, в свою чергу, буде розв’язувати конкретну задачу. Одним із прикладів задачі, що можна розв’язати таким чином, є розпізнавання зображень. Для нейронної мережі немає різниці, хочемо ми розпізнавати котів чи фрикадельки. Нам лише треба навчити мережу на потрібних даних, і отримана модель вже буде «бачити» на зображенні те, що нам треба. Важливим допоміжним інструментом є сегментація зображень, коли на картинці ми виділяємо певні блоки, які можуть становити інтерес для нас і

нейронної мережі, котра навчається. Тобто перш ніж виконувати аналіз зображення, мережа виділяє певні блоки і вже їх пропускає через себе, видаючи відповідь на задане питання.

Всі базові елементи нейронних мереж у звичному вигляді дослідив Френк Розенблатт, якого іноді називають «батьком глибинного навчання» [21]. Перший загальний робочий алгоритм керованого навчання багаторівневої мережі перцептронів було опубліковано у 1965р. вченими О. Івахненком та В. Лапою. В праці 1971р. [22] ними було описано нейронну мережу з 8 шарами, навченими методом групового урахування аргументу. Інші архітектури глибинного навчання, зокрема побудовані зі штучних нейронних мереж, беруть свій початок з неокогнітрону, запровадженого Куніхіко Фукусімою [23].

Власне термін «глибинне навчання» був запропонований спільноті машинного навчання Ріною Дехтер у 1986 році. У 1989 році Ян ЛеКун зміг застосувати до нейронної мережі стандартний алгоритм зворотнього поширення [24], з метою розпізнавання рукописних поштових індексів на поштових відправленнях. Алгоритм був робочим, проте тривалість навчання була занадто тривалою.

У 1994 році Андре де Карвальо, разом із Майком Фейрхерстом та та Девідом Біссетом, опублікував експериментальні результати багатошарової булевої нейронної мережі, також відомої як невагома нейронна мережа, що складалася із трирівневої модуля нейронної мережі для виділення ознак, а згодом модуль нейронної мережі багаторівневої класифікації. Кожен модуль пройшов незалежне навчання, і отримував об’єкти із наростаючою складністю відносно попереднього шару.

У 1995 році Брендан Фрей продемонстрував, що протягом двох днів можна навчити мережу, що має шість повністю з’єднаних шарів і декілька сотень прихованих юнітів. Для цього він використав алгоритм сну-неспання, розроблений разом із Пітером Даяном та Джеффрі Хінтоном [25]. Є багато факторів, що впливають на швидкість навчання мереж, включаючи проблему зникаючого градієнта, яку аналізував З. Хохрайтер у 1991 році.

Глибоке навчання почало набирати популярність в середині 2000-х років, коли комп'ютери стали достатньо потужними, а набори даних стало достатньо об'ємними, щоб навчання мереж мало сенс. Разом із цим мали місце прориви в область алгоритмізації – статті Хінтона, Осіндеро й Те, а також Бенджио, в котрих [26] автори показали, що можна застосувати попередню підготовку до багатошарових нейронних мереж, якщо проводити навчання кожного шару окремо, використовуючи обмежену машину Больцмана, а пізніше виконувати додаткове навчання методом зворотнього поширення.

В 2012 році команда під керівництвом Джорджа Е. Даля виграла конкурс «Merck Molecular Activity Challenge» [27], використавши багатозадачну нейронну мережу для прогнозування біомолекулярної мішені одного препарата. У 2014 році група Хохрайтера використала глибинне навчання для виявлення нецільових і токсичних ефектів хімічних речовин, присутніх у навколишньому середовищі, поживних речовинах та продуктах домашнього вжитку, й виграла «Tox21 Data Chellenge» від Національного університету охорони здоров'я США.

Різке підвищення продуктивності нейронних мереж припало на період 2011-2012 років, коли дослідники почали використовувати GPU для навчання згорткових нейронних мереж. У 2011 році такий підхід вперше дозволив підвищити продуктивність нейронних мереж на декілька порядків. У тому ж 2011 році цей метод виграш конкурс рукописного вводу ICDAR, а у 2012 – конкурс сегментації зображень ISBI [28].

До 2011 року згорткові нейронні мережі не грали провідної ролі, проте у 2012 Циресан у доповіді показав, що максимальна інтеграція згорткових нейронних мереж та GPU може покращити велику кількість бенчмарків. Того ж року аналогічна система, розроблена Крижевським, виграла масштабний конкурс ImageNet зі значним відривом від опонентів [29]. Деякі дослідники вважають, що саме ця перемога поклала початок революції в області глибокого навчання, яка докорінно змінила індустрію штучного інтелекту.

В 2015 році у вільному доступі з’явився пакет TensorFlow, що розроблявся Google для внутрішніх продуктів починаючи з 2011. В його розробці вагому роль зіграв Джеффрі Хінтон, під керівництвом якого раніше було створено узагальнений метод зворотнього поширення й ряд інших покращень, що суттєво зменшили ймовірність помилок при розпізнаванні мови. Основний конкурент TensorFlow – бібліотека PyTorch, була розроблена AI групою компанії Facebook та вперше випущена у 2016 році.

3.2 Принцип роботи оптичного розпізнавання образів

Згоркові нейронні мережі (CNN) є найпопулярнішою моделлю глибинної нейронної мережі, яка використовується для задачі класифікації зображень. Ідея CNN полягає в тому, що місцеве розуміння зображення є достатньо добрим. Практична перевага полягає в тому, що менша кількість параметрів значно скорочує час, необхідний для навчання, а також зменшує кількість даних, необхідних для навчання моделі. Замість повністю пов’язаної мережі вагових коефіцієнтів кожного пікселя, CNN має вагові коефіцієнти, достатні для перегляду невеликої ділянки зображення. На рисунку 3.1 зображена схема CNN-подібних нейроних мереж.

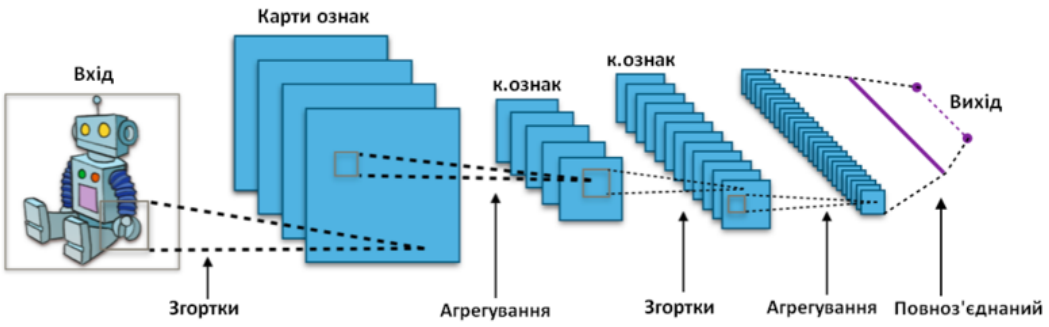


Рисунок 3.1 – Схема роботи згорткових нейронних мереж

Алгоритм CNN базується на різних модулях, структурованих у певному робочому процесі:

1) Шар розгортки (Convolution Layer) – використовується як навігатор по зображенню, який поступово фільтрує пікселі, роблячи кожен піксель пікселем у новому зображенні.
2) Блок активації (Activation Unit) – слідує шару розгортки, а його власна функція активації використовується для встановлення від'ємних значень у вхідних даних на 0.
3) Шар об'єднання (Pooling Layer) – зменшує кількість параметрів і зберігає найбільш критичні параметри.
4) Зрівняльний шар (Flattering Layer) – перетворює останню матрицю на один вектор і переносить її в нашу штучну нейронну мережу через наші нейрони.

Штучні нейронні мережі – це прості математичекі моделі, які визначають функцію $f: X \to Y$ або розподіл по X і Y. Математична функція нейронної мережі $f(x)$ визначається як композиція інших функцій $g_i(x)$. Широко використовується такий тип композиції, як нелінійна зважена сума:

$$f(x) = K\left(Z_i w_i g_i(x)\right),$$

(3.1)

де K – функція активації;

Z_i – безліч значень або градація критерію $f(x)$;

набір функцій $g_i(x)$ визначається, як вектор $g = (g_1, g_2, \dots, g_n,)$;

w_i – маса призначена для визначення важливості критеріїв.

Існує також випадкова величина $F = f(x)$, яка в свою чергу залежить від випадкової величини $G = g(H)$, яка також залежить від величини $H = h(x)$, і в результаті всі величини виявляються залежними від випадкової змінної (X).

В згортковій нейронній мережі, окрім функції активації, також використовуються формули функціонування нейронів в згортковому та підвиборочному шарах. Формула функціонування нейрона згорткового шару мережи:

$$(X)$$

$$y_k^{(i,j)} = b_k + \sum\sum w_{k,s,t}\, x^{((i-1)+s,(j+t))},$$

де $y_k^{(i,j)}$ – нейрон k-й площині згорткового шару, b_k – нейронное зміщення k-й площини загорткового шару, K – розмір рецептивної області нейрона, $w_{k,s,t}$ – матриця синаптичних коефіцієнтів, x – виходи нейронів попереднього шару.

Формула функціонування нейрона підвиборного шару:

$$y_k^{(i,j)} = b_k + ¼ w_k \sum\sum x^{((i-1)+s,(j+t))}$$

Нейронні мережі не програмуються у звичному значенні слова, вони навчаються. Однією з найважливіших переваг нейронних мереж над іншими алгоритмами розпізнавання є можливість навчання. Це навчання полягає у знаходженні коефіцієнтів зв'язків між нейронами. У згортковій нейронній мережі як спосіб навчання використовується алгоритм зворотного поширення помилки. Для вимірювання якості розпізнавання використовується функція середньоквадратичної помилки:

$$E_p=\sum_j (t_{pj}+o_{pj})^2/2$$

$$E_p = \sum_j \frac{(t_{pj}+o_{pj})^2}{2}, \qquad (3.1x)$$

де E_p – величина функції помилки образу p;

t_{pj} – бажаний вихід нейрона j для образу p,;

o_{pj} – дійсний вихід нейрона j для образу p.

У ході навчання нейронна мережа може визначати складні залежності між вхідними та вихідними даними, та виконувати узагальнення. В результаті успішного навчання, нейронна мережа зможе дати правильний результат навіть за неповних або спотворених даних. Для успішного навчання мережі потрібно наблизити вихід мережі до бажаного виходу, а щоб зменшити величину функції помилки, слід налаштовувати ваги нейронів.

3.3 Сучасні згорткові нейронні мережи для розпізнавання образів

Існує розширення CNN, яке використовується саме з метою розпізнавання образів. Мова йде про R-CNN (Region Based Convolutional Neural Network). R-CNN — це регіональна згортка нейронної мережі, яка поєднує пропозиції областей знизу вгору з багатьма функціями, обчисленими згортковою нейронною мережею. Варто відзначити, що R-CNN виділяє на зображенні кілька ділянок і перевіряє, чи дійсно якась із них відповідає об’єкту. Спочатку зображення ділиться приблизно на 2000 рекомендованих регіонів, а потім CNN застосовується для кожного регіону відповідно. Визначається розмір областей і правильна область вставляється в штучну нейронну мережу.

Найбільша проблема цього методу – тривалість навчання та виконання предбачення. Оскільки кожен регіон на зображенні застосовано CNN окремо, час навчання є сталим і становить приблизно 84 години, а час прогнозу – приблизно 47 секунд. Саме тому в межах цієї роботи було розглянуто більш швидші імплементації такої мережі.

Усунення недоліків R-CNN привело до появи покращеної модіфікації моделі. Її назвали Fast R-CNN і вона показує трохи більшу точність та великий прогрес у часі оброки. У Fast R-CNN зображення подається на вхід згорткової нейронної мережі і обробляється Selective Search.

У результаті, маємо карту ознак та регіони потенційних об'єктів. Координати регіонів потенційних об'єктів перетворюються на координати на карті ознак. Отримана карта ознак із регіонами передається шару Polling Layer. І шарі на кожен регіон накладається сітка розміром HxW, а потім застосовується MaxPolling зменшення розмірності. Так, усі регіони потенційних об'єктів мають однакову фіксовану розмірність. Отримані ознаки подаються на вхід повнозв'язкового шару (Fully-conectedlayer), який передається двом іншим повнозв'язковим шарам. Перший з функцією активацією softmax визначає можливість приналежності класу, другий — межі

(зміщення) регіону потенційного об'єкта. Однак все одно через використання достатньо ресурсномісткого Selective Search цей варіант не є ідеальним з точки зору швидкості.

Основна мета спільноти тепер полягала у тому, щоб винайти спосіб локалізувати об’єкти більш дешевшим способом, ніж у попередніх імплементаціях R-CNN. Вже через рік з’явилась Faster R-CNN, яка використовувала RPN (Region Proporsal Network) і могла створювати регіональні пропозиції. Це має деякі переваги:

1. Регіональні пропозиції тепер створюються за допомогою мережі, яку можна навчити та налаштувати відповідно до завдання виявлення.
2. Створюються кращі пропозиції регіонів порівняно з загальними методами, такими як Selective Search і EdgeBoxes.
3. RPN обробляє зображення за допомогою тих самих згорткових шарів, які використовуються в мережі виявлення Fast R-CNN. Таким чином, RPN не вимагає додаткового часу для створення пропозицій порівняно з такими алгоритмами, як вибірковий пошук.
4. Завдяки спільному використанню однакових згорткових рівнів можна об’єднати RPN і Fast R-CNN в одну мережу. Таким чином, навчання проводиться тільки один раз.

Використовуючи еталонні прив’язки (Anchor Boxes), використовується одне зображення в одному масштабі, але є можливість запропонувати масштабно-інваріантні детектори об’єктів, оскільки прив’язки існують у різних масштабах. Це дозволяє уникнути використання кількох фільтрів або зображень. Багатомасштабні прив’язки є ключовими для спільного використання функцій через RPN і мережу виявлення Fast R-CNN. Підсумовуючи, Faster R-CNN справляється трохи гірше з локалізацією, але працює швидше за Fast R-CNN.

Усі попередньо досліджені алгоритми виявлення об’єктів використовують області для локалізації об’єкта на зображенні. Мережа не дивиться на повне зображення, а натомість — на частини зображення, які

мають високу ймовірність містити об’єкт. YOLO (You Only Look Once) — це алгоритм виявлення об’єктів, який значно відрізняється від алгоритмів на основі регіону, розглянутих вище. У YOLO єдина згорточна мережа передбачає обмежувальні рамки та ймовірності класу для цих рамок.

Принцип роботи YOLO полягає в тому, що ми беремо зображення та розбиваємо його на сітку S x S, у межах кожної сітки ми беремо m обмежувальних рамок. Для кожної обмежувальної рамки мережа виводить значення ймовірності класу та зміщення для обмежувальної рамки. Обмежувальні рамки, імовірність класу яких перевищує порогове значення, вибираються та використовуються для визначення місцезнаходження об’єкта на зображенні.

YOLO достатньо швидкий, щоб використовуватися в межах задачі, проте має значний недолік, — через просторові обмеження алгоритму дрібні об’єкти обробляються дуже нестабільно.

З найбільш популярних рішень залишається SSD (Single-Shot Detector). Швидкість навчання та час прогнозу цієї мережі достатній, щоб використовувати її у режимі реального часу. SSD складається з двох компонентів (рис.3.1): опорної моделі (Backbone Model) та голови (Head).

Базовою моделлю зазвичай є попередньо навчена мережа класифікації зображень як екстрактор ознак. Зазвичай це мережа на кшталт ResNet, навчена на ImageNet, з якої було видалено остаточний, повністю підключений, рівень класифікації. Таким чином, у нас залишається глибока нейронна мережа, яка здатна витягувати семантичне значення із вхідного зображення, зберігаючи при цьому просторову структуру зображення, хоча й із меншою роздільною здатністю.

Замість використання Sliding Window, як в Faster-RCNN, SSD ділить зображення за допомогою сітки, і кожна клітинка сітки відповідає за виявлення об’єктів у цій області зображення. Виявлення об’єктів означає просто передбачення класу та розташування об’єкта в цьому регіоні. Якщо жодного об’єкта немає, ми розглядаємо його як фоновий клас, а розташування

ігноруємо. Кожній клітинці сітки в SSD можна призначити кілька прив’язок/попередніх полів. Ці прив’язки попередньо визначені, і кожна з них відповідає за розмір і форму в клітинці сітки. SSD використовує фазу зіставлення під час навчання, щоб узгодити відповідну рамку прив’язки з обмежувальними рамками кожного наземного об’єкта істинності в зображенні. По суті, блок прив’язки з найвищим ступенем перекриття з об’єктом відповідає за прогнозування класу цього об’єкта та його розташування. Ця властивість використовується для навчання мережі, прогнозування виявлених об’єктів та їхнього розташування після того, як мережу було навчено. На практиці кожен блок прив’язки визначається співвідношенням сторін і рівнем масштабування.

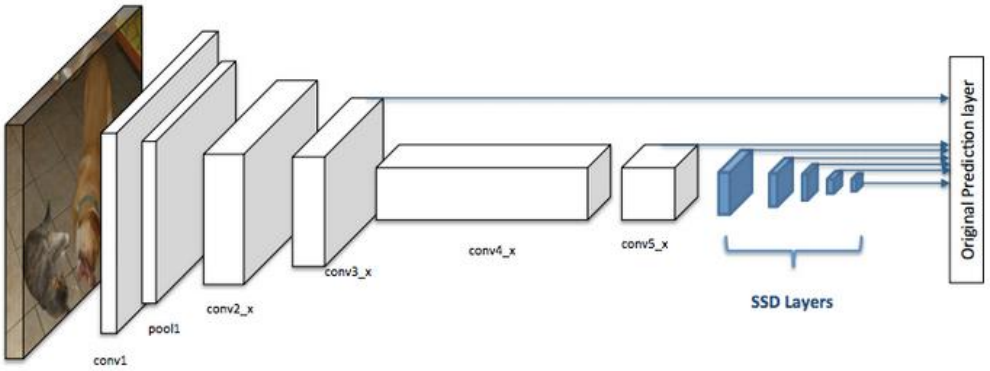


Рисунок 3.1 – Приклад архітектури згорткової нейронної мережі з Single Shot Detector

Оцінивши недоліки і переваги різних загорткових нейронних мереж, було прийняте рішення використовувати Single-Shot Detector як основу для побудови глибинної нейронної мережі.

3.4 Сучасні згорткові нейронні мережи для розпізнавання образів

Розпізнавання об’єктів у дуже різних масштабах є фундаментальною проблемою комп’ютерного зору. Виявлення об’єктів у різних масштабах є складним завданням, особливо для малих об’єктів. Існує можливість

використовувати піраміду з шарів того самого зображення в різному масштабі, однак обробка зображень у кількох масштабах займає багато часу, а потреба в пам’яті є надто високою для одночасного навчання від кінця до кінця. Піраміди ознак, або піраміди зображень із ознаками (Feature Pyramid Network, FPN), складають собою засіб вилучення функцій, розроблений для такої концепції піраміди з урахуванням точності та швидкості. FPN замінює екстрактор ознак таких детекторів, як Faster R-CNN, і генерує кілька шарів карти функцій з інформацією кращої якості, ніж звичайна піраміда функцій для виявлення об’єктів.

Оскільки SSD не повністю використовує багаторівневі функції для об’єднання низької розподільної здатності та високої семантичної функції для отримання задовільного результату виявлення об’єктів, то доцільно буде розглянути її поєднання з ідеями FPN.

Суть полягає в тому, що в ідеалі піраміда у стилі SSD повторно використовувала б багатомасштабні карти функцій з різних шарів, обчислені в прямому проході, і таким чином була б безкоштовною. Але щоб уникнути використання низькорівневих функцій, SSD уникає повторного використання вже обчислених рівнів і натомість будує піраміду, починаючи з високого рівня в мережі, а потім додає кілька нових рівнів.

При конфігуруванні SSD в *якості змін архітектури* застосовані інструкції, які зазвичай використовуються в FPN, що надало змогу обробляти дрібні об’єкти без значних втрат швидкості обробки зображень, не використовуючи більш тривалі (R-CNN, F-CNN, YOLO). Це важливо для загальної швидкості виконання завдань обробки зображень, алгоритм набуває адаптивності.

Таким чином, Single-Shot Detector втрачає можливість повторно використовувати карти з вищою роздільною здатністю ієрархії функцій, які являються важливими для виявлення малих об’єктів.

Мережа пірамідальних функцій є екстрактором функцій, який приймає одномасштабне зображення довільного розміру як вхідні дані та виводить карти

функцій пропорційного розміру на кількох рівнях у повністю згортковий спосіб. Цей процес не залежить від магістральних згорткових архітектур. Тому він діє як загальне рішення для побудови пірамід функцій у глибоких згорткових мережах для використання в таких завданнях, як виявлення об’єктів. Побудова піраміди включає шляхи знизу вгору та шляхи зверху вниз. Принцип роботи мережи пірамідальних функцій зображено на рисунку 3.2.

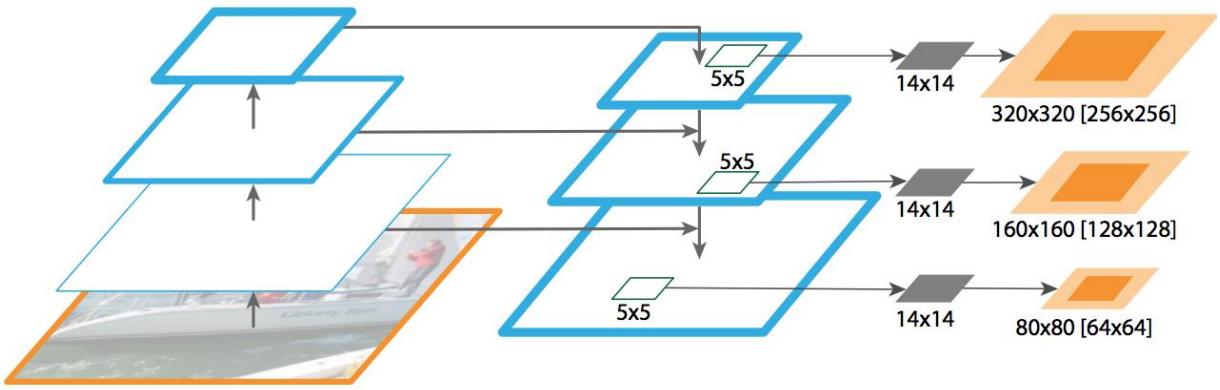


Рисунок 3.2 – Архітектура Feature Pyramid Network

Шлях «знизу вгору» — це обчислення прямого зв’язку магістральної мережі ConvNet, яка обчислює ієрархію функцій, що складається з карт функцій у кількох масштабах із кроком масштабування 2. Для піраміди функцій визначено один рівень піраміди для кожного етапу. Результат останнього шару кожного етапу використовується як еталонний набір карт функцій. Для ResNet (Residual neural network) ми використовуємо дані про активацію функції, виведені останнім залишковим блоком кожного етапу.

Зворотній шлях оцінює об’єкти з вищою роздільною здатністю шляхом підвищення дискретизації просторово більш грубих, але семантично сильніших карт об’єктів з вищих рівнів піраміди. Ці функції потім посилюються функціями «знизу вгору» через бічні з’єднання. Кожне таке з’єднання об’єднує карти характеристик однакового просторового розміру від шляху «знизу вгору» та шляху «зверху вниз». Карта функцій «знизу вгору» має семантику нижчого

рівня, але її активації локалізовані точніше, оскільки її підвибірку проводили менше разів.

Тепер, коли було розглянуто модель глибинної нейронної мережі SSD та способи покращити її в межах задачі роботи можна переходити до програмної реалізації та навчання такої моделі на актуальному набору даних.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ

4.1 Технології розробки та інструментальні засоби

Для реалізації моделі, яка буде автоматично розпізнавати морське сміття на супутниковому знімку було обрано Tensorflow 2 та Tensorflow Object Detection API.

Для підготовки набору даних у готовому форматі для глибинного навчання моделі використовувались пакет супутникових знімків Planetscope та продукт NASA ImageLabeler.

Для уявлення вхідних зображень, які будуть оцінюватись моделлю було обрано бібліотеку NumPy та Keras.

Для реалізації кросплатформленого додатку-утіліти було обрано фреймворк Python Kivy.

Для реалізації програми буде використано кросплатформове середовище розробки PyCharm для мови програмування Python.

4.1.1 Платформа Tensorflow

Tensorflow — це бібліотека з відкритим кодом для чисельних обчислень і великомасштабного машинного навчання, збирає дані, навчальні моделі, обслуговує прогнози та уточнює майбутні результати. Tensorflow об’єднує моделі та алгоритми машинного та глибокого навчання. Він використовує Python як зручний інтерфейс і ефективно працює на оптимізованому C++.

Tensorflow дозволяє розробникам створювати графік обчислень для виконання. Кожен вузол на графіку представляє математичну операцію, а кожне з’єднання – дані. Таким чином, замість того, щоб мати справу з низькими деталями, такими як з’ясування правильних способів зв’язати вихідні

дані однієї функції з вхідними даними іншої, розробник може зосередитися на загальній логіці програми.

У дослідницькій групі глибокого навчання штучного інтелекту в Google Google Brain у 2015 році розробила TensorFlow для внутрішнього використання Google. Дослідницька група використовує цю бібліотеку програмного забезпечення з відкритим кодом для виконання кількох важливих завдань.

TensorFlow на даний момент є найпопулярнішою програмною бібліотекою. Є кілька реальних програм глибокого навчання, які роблять TensorFlow популярним. Будучи бібліотекою з відкритим кодом для глибокого та машинного навчання, TensorFlow відіграє важливу роль у текстових програмах, розпізнаванні зображень, голосовому пошуку та багато іншого. DeepFace, система розпізнавання зображень Facebook, використовує для розпізнавання зображень TensorFlow. Він використовується Siri від Apple для розпізнавання голосу. Кожна програма Google добре використовує TensorFlow, щоб покращити ваш досвід.

Усі обчислення, пов’язані з TensorFlow, передбачають використання тензорів.

Тензор — це вектор/матриця n-вимірів, що представляють типи даних. Значення в тензорі містять ідентичні типи даних із відомою формою, і ця форма є розмірністю матриці:

- Вектор — одновимірний тензор;
- Матриця - це двовимірний тензор;
- Скаляр — це нульвимірний тензор.

TensorFlow Object Detection API — це платформа комп’ютерного бачення з відкритим кодом для побудови моделей виявлення об’єктів і сегментації зображень, які можуть локалізувати кілька об’єктів на одному зображенні. Фреймворк працює як для першої версії TensorFlow, так і для другої.

Перевага цього рішення полягає у тому, що моделі можна завантажити з TensorFlow 2 Detection Model Zoo. Це дає можливість практичним шляхом спробувати досить різні підходи для вирішення конкретної задачі. Завдяки

влаштованому файлу конфігурації користувач-програміст може дуже гнучким способом змінювати налаштування мереж на свій розсуд.

4.1.2 Продукти Planetscope та NASA ImageLabeler

PlanetScope — це веб-сайт з набором приблизно 130 комерційних та відкрито доступних супутників, здатних щодня знімати зображення всієї поверхні Землі (щоденна потужність збору 200 мільйонів км²/день). Зображення PlanetScope мають роздільну здатність приблизно 3 метри на піксель.

Угруповання супутників PlanetScope складається з кількох запусків супутників Dove. Потужність на орбіті постійно вдосконалюється як у можливостях, так і в кількості, а технологічні вдосконалення впроваджуються швидкими темпами. Кожен супутник має форм-фактор CubeSat 3U.

Продукти PlanetScope доступні для пошуку та завантаження через API, інтерфейси користувача та інтеграції Planet у формі продуктів Basic Scene, Ortho scenes і OrthoTile, які доступні через платформу як набір типів елементів і типів активів.

Image Labeler — це швидкий і масштабований веб-інструмент від NASA, який сприяє швидкому створенню наборів даних про явища науки про Землю на основі зображень, щоб допомогти застосуванню глибокого навчання та автоматизованій класифікації/виявленню зображень. Image Labeler побудовано на основі сучасних веб-технологій, щоб максимально збільшити масштабованість і доступність платформи. Він має зручний інтерфейс, який дозволяє відносно швидко додавати теги до кількох зображень. По суті, Image Labeler удосконалює існуючі методи, надаючи дослідникам доступне джерело позначених тегами наукових зображень для всіх їхніх потреб у машинному навчанні.

4.1.3 Мова програмування Python

Python — це відома високорівнева, універсальна, динамічно типізована та інтерпретована мова програмування. Інтерпретована частина цього визначення означає, що Python (та інші інтерпретовані мови) не компілюється безпосередньо в машинну мову чи інструкції перед виконанням комп'ютером. Замість цього Python зчитується та виконується іншою програмою – або інтерпретатором – який потім перекладає код на машинну мову, яку може зрозуміти процесор комп'ютера.

Є деякі переваги та недоліки мови програмування, що підпадає під категорію інтерпретованих. Зокрема, інтерпретовані мови програмування, такі як Python, не залежать від платформи, тобто вони можуть працювати на будь-якій операційній машині, системі чи платформі. Ці типи мов кодування також зазвичай менші за розміром і мають такі функції, як динамічний тип.

Незважаючи на свою простоту та невелику криву навчання, Python надзвичайно потужний і універсальний з точки зору типів програм, які можно створювати.

В роботі майже всі бібліотеки та фреймворки обирались серед екосистеми Python, так як Machine Learning спільнота цієї мови програмування вважається найширшою з усіх.

4.2 Підготовка навчальних даних

Навчання обранної моделі потребує відповідного датасету, який повинен складатися з власне супутникових знімків та міток, які є обмежувальними рамками актуальної області місцезнаходження морського сміття.

Відповідно до досліджених джерел та архівів супутникових знімків було прийняте рішення використовувати оптичні зображення PlanetScope. Більшість обраних знімків оточує берегові лінії островів Бей в Гондурасі, хоча невеликий

відсоток набору даних належить прибережним районам Гани та Греції. Усього було зібрано приблизно 700 знімків розміром 256 на 256 пікселів, а сама просторова розподільна здатність супутникових зображень приблизно складає 3 метри. Також у цьому наборі даних морське сміття складається з плаваючих об’єктів на поверхні океану, які можуть належати до одного або кількох класів, а саме пластик, водорості, саргассум, дерево та інші штучні предмети.

Особливою перевагою супутників PlanetScope є те, що їх зображення має чотири смуги, а саме червону, зелену, синю та ближню інфрачервону область. Поєднання досить високої просторової роздільної здатності, високої часової роздільної здатності, наявності ближнього інфрачервоного каналу та глобального покриття берегових ліній зробили це зображення досить вигідним для цілей цього дослідження. Завдяки цим специфікаціям зображень, а також категоріям розміру пластику та розміру примарних рибальських сіток, передбачається, що навчанна модель зможе виявляти накопичене сміття, а також деякі мегапластики, включаючи привидні рибальські сіті середнього та великого розміру.

За допомогою програми Planet Explorer для обраного навчального набору даних було вибрано конкретні сцени зображення, що складаються з видимих плям морського сміття. Цей крок передбачав ручне дослідження сцен PlanetScope та перевірку наявності морського сміття. Для цього початкового дослідження було вирішено зосередити зусилля на виявленні морського сміття за допомогою зображень оптичних (червоних, зелених, синіх) каналів. Було проведено початкове дослідження користі каналу ближнього інфрачервоного діапазону Planetscope, і в подальшій роботі буде інтегровано канал ближнього інфрачервоного діапазону.

Щоб вручну оцифрувати анотації обмежувальної рамки для спостережуваного сміття на оптичних зображеннях PlanetScope було використано Image Labeler. Загалом на сценах зображень було позначено 703 обмежувальні рамки. Це стало початковим набором даних для навчання, тестування та перевірки для моделювання виявлення об’єктів.

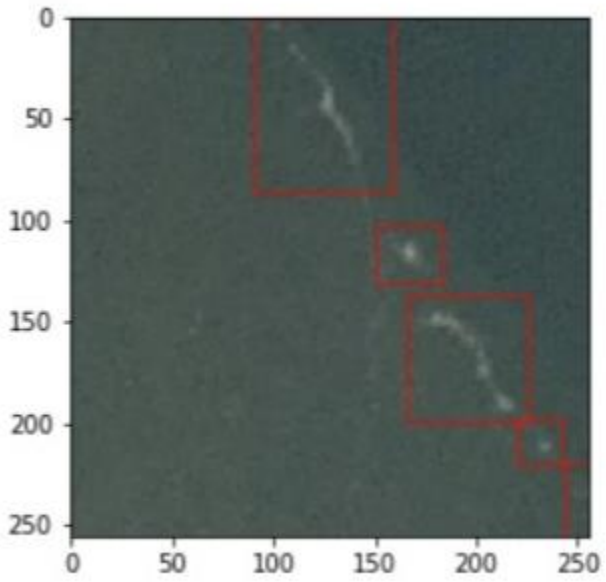
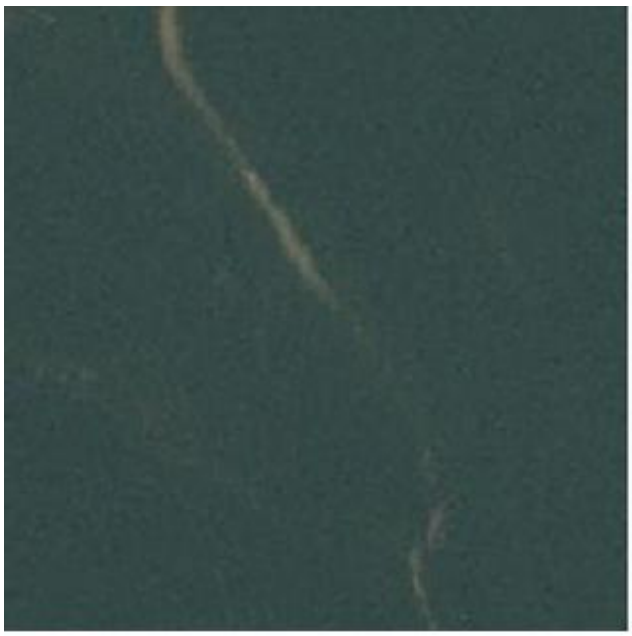


Рисунок 4.1 — Приклад ручної анотації обмежувальних рамок морського сміття

Наступним завданням було підготувати набір даних у готовому до моделі форматі, що передбачало розбиття сцен зображення на менші кадри та кодування обмежувальних рамок у масиви координат із числовими ідентифікаторами класів. Потреба в мозаїці зображення випливає з обчислювальної ефективності під час виконання моделі. Для виконання цих завдань була використовувана утіліта Label Maker. Рівень масштабування було обрано 16, оскільки він найбільше наближає до рідної просторової роздільної здатності зображень Planetscope. Після успішної обробки даних утилітою був отриманий набір даних у форматі стисненого масиву, який подалі використовувався для створення двійкових наборів даних TensorFlow Records.



```
array([[ 36,   0, 160, 139,   1],
       [148, 175, 218, 255,   1]])

array([[[63, 75, 71],
        [64, 76, 72],
        [64, 76, 72],
        ...,
        [54, 74, 72],
        [48, 68, 66],
        [46, 66, 64]],

       [[63, 75, 71],
        [64, 76, 72],
        [64, 76, 72],
        ...,
        [55, 75, 73],
        [52, 72, 70],
        [52, 72, 70]],

       [[65, 77, 73],
        [67, 79, 75],
        [67, 79, 75],
        ...,
        [54, 74, 72],
        [52, 72, 70],
        [55, 75, 73]], dtype=uint8)
```

Рисунок 4.2 — Координати анотації обмежувальної рамки [xmin, ymin, xmax, ymax] та ідентифікатор класу в форматі файла масива NumPy

Tensorflow API підтримує різні формати файлів. Формат файлу TFRecord — це простий орієнтований на запис бінарний формат, який використовують багато програм TensorFlow. Завдяки утиліті utils_convert_tfrecords.py було перетворено створений через Label Maker файл у потрібні нам TFRecord файли train.record, val.record і test.record у папці під назвою tf_records у робочому каталозі. Кожен файл запису містить різні та неперекриваючі розділи даних (86,7,7 відсотків відповідно).

4.3 Конфігурація моделі

Для рішення задачі по виявленню морського сміття в роботі використовується Tensorflow Object Detection API. Спочатку потрібно знайти придатну модель, яка буде використовуватись як контрольна точка навчання. Екосистема Tensorflow має спеціальне середовище, де розташовані різноманітні моделі – Tensorflow Model Zoo. Переглянувши SSD рішення було обрано ssd_resnet_50_v1_fpn.

Після завантаження моделі потрібно її сконфігурувати. Почнемо з того, що наш набір даних складається з картинок розміром 256x256 пікселів та одного конкретного об’єкта, який потрібно класифікувати при наявності, тому налаштуємо влаштований image_shape_resizer на стале значення нашого набору:

```
num_classes: 1
image_resizer {
  fixed_shape_resizer {
    height: 256
    width: 256
  }
}
```

Далі необхідно описати сам екстрактор функцій. Сама модель вже навчалась на моделі ResNet FPN, тому додаткових конфігурацій робити не треба.

Серед основних функцій регуляризаторів звичайно розглядають дві техники регуляризації: L1, називається Lasso Regression, а модель, яка використовує L2, називається Ridge Regression. Ключова відмінність між цими техніками полягає в тому, що Lasso зменшує коефіцієнт менш важливої функції до нуля, таким чином усуваючи деякі функції. Цей варіант доречно використовувати, коли є величезна кількість функцій. Проте в роботі було обрано функцію L2 регуляризатора, щоб уникнути проблеми з надмірною посадкою.

В якості функції активації у прихованому шарі було обрано ReLU6. У сучасних мережах зазвичай розглядають ReLU та ReLU6, а їх основна відмінність полягає в тому, що ReLU дозволяє дуже високі значення на позитивній стороні, тоді як ReLU6 обмежено значенням 6 на позитивній стороні. Будь-яке вхідне значення, яке дорівнює 6 або перевищує 6, буде обрізано значенням 6 (звідси й назва).

Інші параметри залишаються сталими, отже конфігурація екстрактору функцій:

```
feature_extractor {
  type: "ssd_resnet50_v1_fpn_keras"
  depth_multiplier: 1.0
  min_depth: 16
  conv_hyperparams {
    regularizer {
      l2_regularizer {
        weight: 0.00039999998989515007
      }
    }
    initializer {
      truncated_normal_initializer {
        mean: 0.0
        stddev: 0.029999999329447746
      }
    }
    activation: RELU_6
    batch_norm {
      decay: 0.996999979019165
      scale: true
      epsilon: 0.0010000000474974513
    }
  }
```

В якості блока кодувальника використовувався підхід Faster R-CNN. Мережа передбачає зміни для кожного блоку прив’язки. Тобто для кожного опорного блоку він передбачає зміщення для позиції x, y, а також ширини та висоти.

Ми повинні згенерувати блоки прив’язки для кожного підрозділу, який є на карті функцій. Для кожного блоку ми згенеруємо блоки прив’язки, які дорівнюють number_of_apect_ratios * number_of_scales. Все це описано в файлі конфігурації:

```
box_coder {
  faster_rcnn_box_coder {
    y_scale: 10.0
    x_scale: 10.0
    height_scale: 5.0
    width_scale: 5.0
  }
}
anchor_generator {
```

```
    multiscale_anchor_generator {
      min_level: 3
      max_level: 7
      anchor_scale: 4.0
      aspect_ratios: 1.0
      aspect_ratios: 2.0
      aspect_ratios: 0.5
      scales_per_octave: 2
    }
  }
```

В якості блока предиктору використовувався схожа з екстрактором функцій з власним значенням глибини 256. Використовані значення було залишено сталими і вони дорівнюють нормам моделей типу Single Shot Detector:

```
  box_predictor {
    weight_shared_convolutional_box_predictor {
      conv_hyperparams {
        regularizer {
          l2_regularizer {
            weight: 0.00039999998989515007
          }
        }
        initializer {
          random_normal_initializer {
            mean: 0.0
            stddev: 0.009999999776482582
          }
        }
        activation: RELU_6
        batch_norm {
          decay: 0.996999979019165
          scale: true
          epsilon: 0.0010000000474974513
        }
      }
      depth: 256
      num_layers_before_predictor: 4
      kernel_size: 3
      class_prediction_bias_init: -4.599999904632568
    }
  }
```

Постпроцесор обробляє прогнозовані та анотаційні дані після визначення моделі та перед обчисленням метрики. На відміну від адаптерів, постпроцесори працюють не з необробленим виводом моделі, а з певним форматом

представлення. Видаляє блоки, які мають високе перекриття перетину, яке також регулюється параметром iou_threshold через об’єднання з попередньо вибраними блоками. Обмежувальні прямокутники надаються як [*y1, x1, y2, x2*], де (*y1, x1*) і (*y2, x2*) є координатами будь-якої діагональної пари кутів прямокутників, а координати можна надати як нормалізовані (які знаходяться в інтервал [0, 1]) або абсолютні. Цей алгоритм не залежить від того, де знаходиться початок у системі координат, а також він інваріантний до ортогональних перетворень і трансляцій системи координат; таким чином трансляція або відображення системи координат призводять до того, що алгоритм вибирає однакові прямокутники. В якості оціночної функції використовувалась функція SOFTMAX. Її перевага перед SIGMOID у тому, що вона краще підходить нашій задачі, так як ми маємо лише один клас для класифікації, а в свою чергу SIGMOID навпаки буде доречнішою у випадку, коли існує декілька класів і кожен із них може бути класифікований одночасно.

```
post_processing {
  batch_non_max_suppression {
    score_threshold: 0.1
    iou_threshold: 0.6000000238418579
    max_detections_per_class: 50
    max_total_detections: 50
    use_static_shapes: true
  }
  score_converter: SIGMOID
}
```

Для оцінки якості моделі існують функції втрат класифікації та локалізації. Функції оптимізації намагаються зменшити ці значення втрат, доки сума втрат не досягне точки, коли результат не буде вважитися задовільним. Для функції втрат локалізації була використовувана функція Lasso Regression, яка зменшує коефіцієнт менш важливої функції до нуля, а як функція класифікації використовувалась SIGMOID_FOCAL, суть якої полягає у тому, вона являється дуже актуальною, коли більшість пікселів на зображені є фоном і лише невелика кількість пікселів усередині мають розглядатися.

```
normalize_loss_by_num_matches: true
loss {
  localization_loss {
    weighted_smooth_l1 {
    }
  }
  classification_loss {
    weighted_sigmoid_focal {
      gamma: 2.0
      alpha: 0.25
    }
  }
  classification_weight: 1.0
  localization_weight: 1.0
}
```

4.4 Навчання моделі

Тепер, коли була описана конфігурація для моделі, є потреба сконфігурувати її навчання. Отже почнемо з найважливіших параметрів — розмір партії. Розмір партії (batch_size) — це кількість зразків, оброблених перед оновленням моделі. Розмір партії має бути більше або дорівнювати одиниці та менше або дорівнювати кількості зразків у навчальному наборі даних. Немає магічних правил, як налаштувати ці параметри, тому в межах будь якої задачі треба спробувати різні значення та знайти ті, що найкраще підходить для досліджуваної проблеми. Було протестовано кілька параметрів batch_size від 1 до 16 і якість моделі дуже сильно змінювалась, як і час на проходження заданої кількості кроків. Експерементальним шляхом було прийняте рішення залишити це значення рівним 12, як найоптимальніше для обраного датасету.

Тепер варто зупинитись на дуже важливій властивості глибинних нейронних мереж. Обраний нами набір містить у собі не дуже велику кількість даних для аналізу моделью. Звичайно, якщо є багато різноманітних типів морського сміття, потрібно буде показати моделі машинного навчання пропорційну кількість прикладів, щоб отримати хорошу продуктивність. Отже, щоб отримати більше даних, потрібно внести незначні зміни в наш існуючий набір даних. Незначні зміни, такі як повороти, переклади чи

повороти. Наша нейронна мережа все одно подумає, що це різні зображення.

Для цього в Tensorflow Object Detection API вже влаштовані фукції збільшення кількості навчальних даних. В роботі було використано лише 5 з них, але великі моделі мають у собі дуже велику кількість таких правил. Опишемо кожну з них у таблиці.

Таблиця 4.1 – Функції розширення даних у Tensorflow

Спосіб розширення даних	Опис
`random_horizontal_flip`	Функція випадкового горизонтального повороту зображення
`random_vertical_flip`	Функція випадкового вертикального повороту зображення
`random_adjust_brightness`	Функція випадкового значення яскравості зображення
`random_adjust_contrast`	Функція випадкового значення контрасту зображення
`random_crop_image`	Функція випадкового обрізання зображення

Фрагмент лістингу наведення влаштованих та описаних вище функцій збільшення кількості наведених даних:

```
data_augmentation_options {
  random_horizontal_flip {
  }
}
data_augmentation_options {
  random_vertical_flip {
  }
}
data_augmentation_options {
  random_adjust_brightness {
  }
```

```
  }
  data_augmentation_options {
    random_adjust_contrast {
    }
  }
  data_augmentation_options {
    random_crop_image {
      min_area: 0.75
      max_area: 1.0
    }
  }
```

Наступним конфігураційним параметром буде швидкість навчання, яке контролює наскільки вагові коефіцієнти оновлюються відповідно до оціненої помилки. При виборі занадто малого значення, модель буде тренуватися некоректно, а занадто великої швидкості навчання, модель може пропустити оптимальний набір ваг під час тренування. Ідея оптимізації полягає в тому, щоб почати з малого — скажімо, з 0,0001 і збільшувати значення кожен раз під час певної кількості кроків виконання. Під час навчання моделі, значення 0,0001 як learning_rate_base вистачило, щоб модель могла відпрацьовувати коректно. Щодо кількості кроків виконання — це значення цільком пов’язане з швидкістю навчання і його доцільно обирати експериментальним шляхом. Під час навчання моделі було виявлено, що 25000 кроків було замало для того, щоб зменшити втрати моделі, тому воно було збільшено вдвічі.

```
learning_rate {
        cosine_decay_learning_rate {
          learning_rate_base: 0.0001
          total_steps: 50000
          warmup_learning_rate: 0.0000001
          warmup_steps: 1000
        }
}
```

Зібравши в купу конфігурації моделі та її навчання можна приступати до самого процесу тренування моделі. Кожен раз, як буде проходити певна кількість кроків — Tensorflow буде автоматично зберегати точку зберігання(Checkpoint), в якій буде описаний поточний стан моделі. Саме ці

точки зберігання і будуть використані далі, як нейронна мережа, яка здатна оцінювати супутниковий знімок води на предмет наявності у ній морського сміття.

Дана модель навчалася приблизно 25 годин і за цей час було перевірено багато проміжних точок зберігання (рис.4.3).

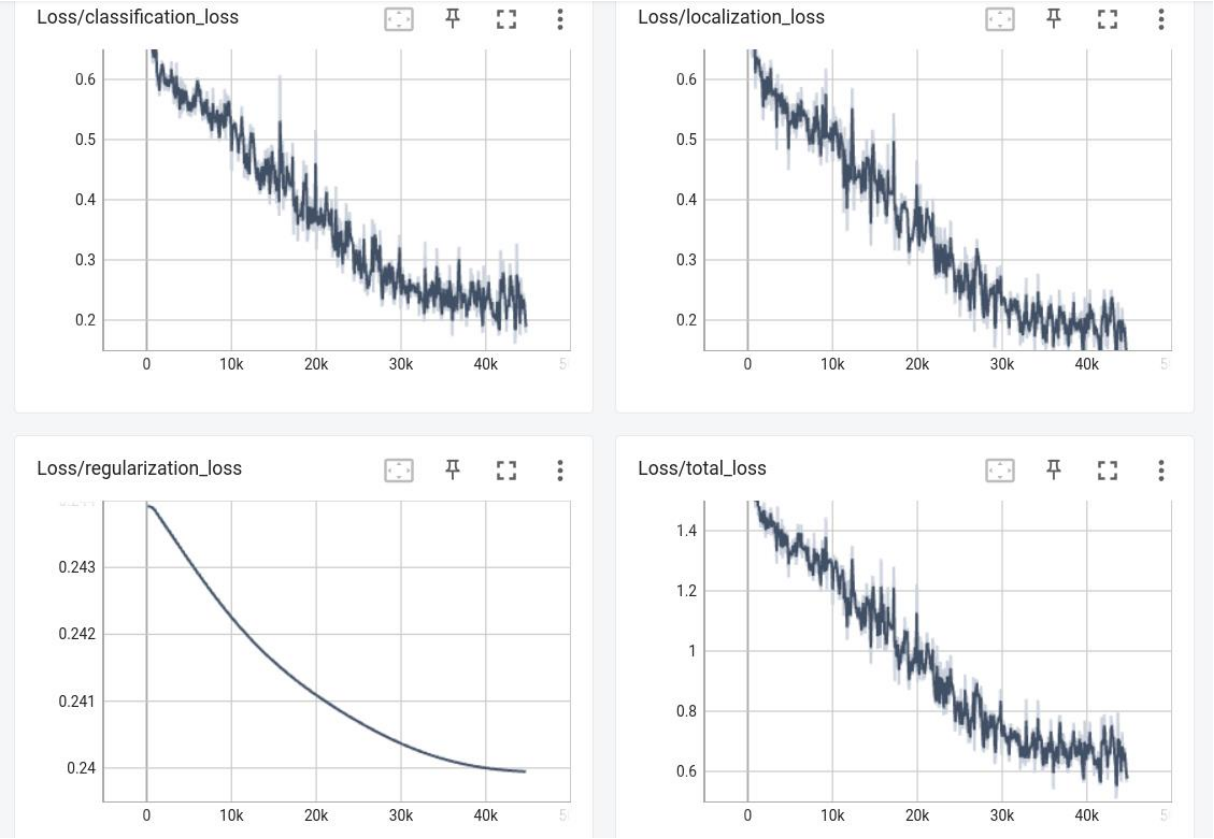


Рисунок 4.3 – Значення функції втрат для задачі класифікації

Значення втрат класифікації та локалізації моделі з часом зменшувались, а результати передбачання більш наближалися до реальних. На рис.4.3. видно, що хоч ці значення і становилися меньшими у процесі навчання — цей результат не є ідеальним. В якості вдосконалювання моделі у майбутньому пропонується збільшити параметр швидкості навчання та кількість кроків виконання, це може дати поштовх для більш об’єктивних значень втрат, проте навчання такої моделі займе немало часу.

Таблиця 4.2 – Приблизне значення функції втрат для задачі класифікації

Step	Classification Loss
1000	0.872
2000	0.79
5000	0.6631
10000	0.535
25000	0.29851
50000	0.177

Таблиця 4.3 – Приблизне значення функції втрат для задачі локалізації

Step	Localization Loss
1000	1.1
2000	0.8946
5000	0.612
10000	0.519
25000	0.241
50000	0.149

Таблиця 4.4 – Приблизне значення показнику перенавчання моделі

Step	Regularization Loss
1000	0.77
2000	0.69
5000	0.42
10000	0.29
25000	0.24
50000	0.23

Таблиця 4.5 – Приблизне значення показнику тотальних втрат

Step	Total Loss
1000	2.123
2000	1.952
5000	1.63
10000	1.25
25000	0.8125
50000	0.5219

4.5 Реалізація додатку

Маючи навчену модель класифікації морського сміття на супутникових знімках доцільно використати цю модель для написання кросплатформного додатку-утіліти. В задачі такої програми має входити можливість проглядати файли на системі користувача, підключати нейронну модель та аналізувати вхідне зображення завдяки моделі.

Виконуючи проект в екосистемі Python було прийняте рішення дивитися в сторону його фреймворків, які дозволяють писати такі програми. Один з таких фреймворків — Kivy.

Kivy — це безкоштовний фреймворк Python з відкритим вихідним кодом для розробки мобільних додатків та іншого мультисенсорного програмного забезпечення з природним інтерфейсом користувача. Він поширюється згідно з умовами ліцензії MIT і може працювати на Android, iOS, Linux, macOS і Windows.

Фреймворк містить усі елементи для створення програми, такі як:

- Розширена підтримка введення для миші, клавіатури, TUIO та специфічних для ОС мультисенсорних подій.
- Графічна бібліотека, що використовує лише OpenGL ES 2 і базується на Vertex Buffer Object і шейдерах.
- Широкий вибір віджетів, які підтримують мультитач.
- Проміжна мова Kv, яка використовується для легкого створення власних віджетів.

Варто почати з імпорту моделі та її зчитування. Для цього був використаний напряму Tensorflow та останній чекпоінт моделі:

```
configs =
config_util.get_configs_from_pipeline_file(PATH_TO_CFG)
model_config = configs['model']
detection_model = model_builder.build(model_config=model_config,
is_training=False)



# Restore checkpoint
ckpt = tf.compat.v2.train.Checkpoint(model=detection_model)
ckpt.restore(PATH_TO_CKPT).expect_partial()
```

Саме таким чином, тепер у змінній model_config знаходиться наша модель з врахуванням специфіки FPN, і тепер можна робити передбачення. Насамперед опишемо функцію зчитування обраного користувачем файла у граф tensorflow, для цього використаємо бібліотеку NumPy:

```
def load_image_into_numpy_array(path):
    return np.array(Image.open(path))

# Where filename[0] is the choosed file by user
image_np = load_image_into_numpy_array(filename[0])
```

Надалі потрібно обробити саме зображення навченною згортковою моделью та зробити саме передбачення. Отримані дані запишемо в окремий масив:

```
detections = detect_fn(detection_model, input_tensor)
num_detections = int(detections.pop('num_detections'))
detections = {key: value[0, :num_detections].numpy()
    for key, value in detections.items()}
detections['num_detections'] = num_detections
viz_utils.visualize_boxes_and_labels_on_image_array(
    image_np_with_detections,
    detections['detection_boxes'],
    detections['detection_classes'] + label_id_offset,
    detections['detection_scores'],
    category_index,
    use_normalized_coordinates=True,
    max_boxes_to_draw=200,
    min_score_thresh=.88,
    agnostic_mode=False)
```

Параметр min_score_thresh відповідає за найменшу можливу оцінку об’єкту нейронною мережою для подальшого запису у масив. Після цих кроків — все, що залишається, це нанесення “зони інтересу” моделі на сам знімок та її відображення у програмі:

```
plt.figure()
plt.savefig(TEMP_PATH + '/' + random_name)
self.ids.my_image.source(_PATH + '/' + random_name)
```

Інтерфейс віджету автоматично згенерується завдяки влаштованому в бібліотеку Kivy файловим менеджером та галереї при запуску програми. У кінці отримуємо структуру нашого проекту, зображену на рисунку 4.4.

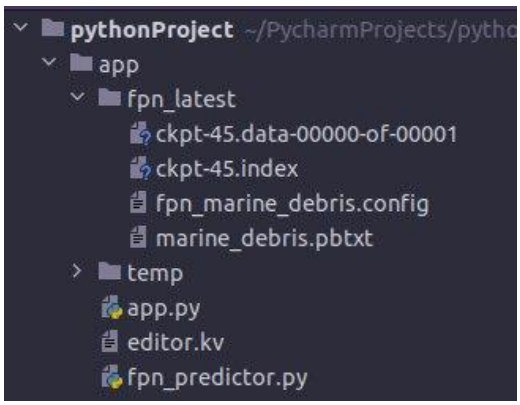


Рисунок 4.4 – Загальний вигляд кінцевого результату проекта

Запустивши утиліту бачимо вспливаюче вікно, у якому є файловий менеджер. Відкривши будь-яке зображення, путь до нього буде передано у модель та буде зроблено передбачення. У випадку, якщо нейронна мережа “впевнена” у класифікації об’єкта на зображені більш ніж на 88% — буде відображено результат у вигляді тієї ж самої картинки, з доданою на неї зеленою рамкою та підписом “marine_debris”. Результат роботи програми можна побачити на рисунку 4.5.

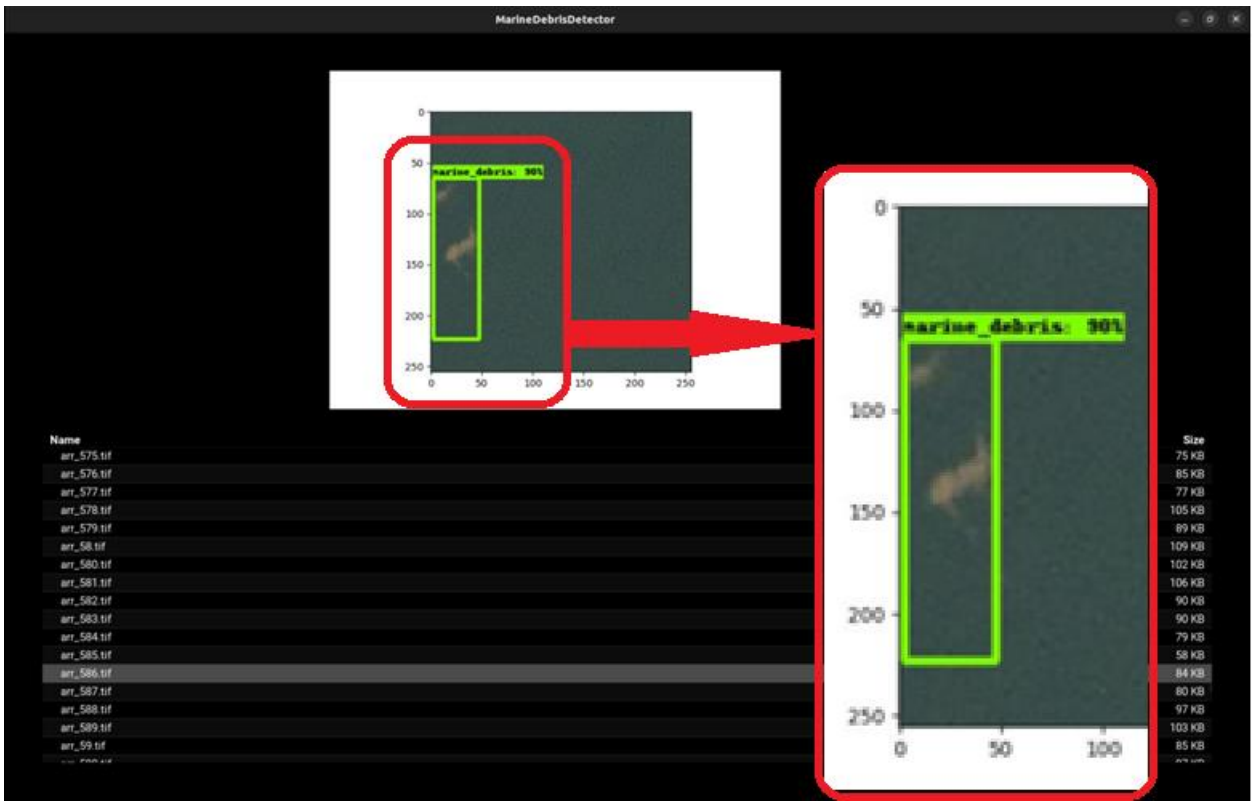


Рисунок 4.5 – Кінцевий вигляд програми з вдалим передбаченням саргасу у

морі

ВИСНОВКИ

В результаті впконання магістерської роботи було проведене дослідження методів виявлення морського сміття на супутникових зображеннях за допомогою глибинного навчання.

На основі огляду існуючих згорткових моделей і існуючих рішень їх застосування для класифікації об’єктів було встановлено, що актуально застосовувати модель нейронної мережі Single Shot Detector. В якості вдосконалення моделі при конфігуруванні SSD в якості змін архітектури застосовані інструкції, які зазвичай використовуються в FPN, що надало змогу обробляти дрібні об’єкти без значних втрат швидкості обробки зображень, не використовуючи більш тривалі: R-CNN, F-CNN, YOLO. Таким чином алгоритм набув адаптивності, що важливо для загальної швидкості виконання завдань обробки зображень.

В експериментальній частині було створено набір навчальних даних та реалізовано програму-утіліту, яка спроможна по навченій моделі класифікувати у воді різні класи морського сміття.

Проведені тести наочно демонструють, що для навчання загорткової моделі найбільш важливими конфігураційними показниками є розмір партії, швидкість навчання та функції розширення даних.

Експериментально встановлено, що застосовування отриманої моделі нейронної мережі для класифікації морського сміття на супутникових зображеннях є працездатним.

В результаті написання магістерської дисертації було вирішено наступні завдання:

1. Здійснено огляд літератури з теми дослідження. Розглянуто задачі нейромережевої класифікації даних та існуючі підходи.

2. Досліджено особливості описання глибинних нейронних мереж та їх навчання.

3. Розблено додаток-утиліту, який використовує навчанну модель для задач виявлення морського сміття.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Oliver J. Dameron; Michael Parke; Mark A. Albins; Russell Brainard (April 2007). "Marine debris accumulation in the Northwestern Hawaiian Islands: An examination of rates and processes". Marine Pollution Bulletin. 54 (4): 423–33.

2. Plastics, pelagic tar, and other litter. In E. D. Goldberg (editor), Strategies for marine pollution monitoring, p. 77-89. Wiley, N.Y. 27 p.

3. Hohn, Donovan, Moby-Duck: The True Story of 28,800 Bath Toys Lost at Sea and of the Beachcombers, Oceanographers, Environmentalists, and Fools, Including the Author, Who Went in Search of Them. Viking, New York, NY 2011

4. Curtis C. Ebbesmeyer and W. James Ingraham Jr. (October 1994). "Pacific Toy Spill Fuels Ocean Current Pathways Research". Earth in Space. 7 (2): 7–9, 14.

5. Susan D. Richardson and Thomas A. Ternes . Water Analysis: Emerging Contaminants and Current Issues. Analytical Chemistry 2018, 90 (1) , 398-428.

6. Bates, S. (2021). Scientists Use NASA Satellite Data to Track Ocean Microplastics From Space. NASA Earth Science News Team. June 28, 2021.

7. Evans, M. C. & Ruf, C. S. (2021). Toward the Detection and Imaging of Ocean Microplastics With a Spaceborne Radar. IEEE Transactions on Geoscience and Remote Sensing.

8. Shah, A., Thomas, L. & Maskey, M. (2021). Marine Debris Dataset for Object Detection in Planetscope Imagery, Version 1.0, Radiant MLHub

9. Aizerman, M. A. and Braverman, E. M. and Lev I. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. Automation and Remote Control, 25:821–837, 1964.

10. Collins, M. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with the perceptron algorithm in Proceedings of the *Conference on Empirical Methods in Natural Language Processing* (EMNLP '02).

11. LeCun, Yann; Bengio, Yoshua; Hinton, Geoffrey (2015). "Deep Learning". Nature. 521 (7553): 436–444.

12. Ніколенко С. Глибоке навчання, Київ, КНЕУ, 2019. 412 с.

13. Хайкін С. Нейронні мережі: повний курс. Запоріжжя, ЗНУ, 2019. 868 с.

14. Малов Р. Нейронні мережі: Короткий довідник. Київ, КНЕУ, 2017. 288 с.

15. Субботін С.О. Нейронні мережі: теорія та практика. Житомир: О.О. Євенок, 2020. 184 с.

16. C. Chandhok. A Novel Approach до Image Segmentation using Artificial Neural Networks and K–Means Clustering. International Journal of Engineering Research and Applications (IJERA), 2 (3), 2012. pp. 274-279.

17. Z. Qing, Y. Guanhui, G. Tingling, Z. Hong, L. Junxiao. Fabric Defect Segmentation Based on Region Growing PCNN Model. Computer application and software, 28(11), 2011. 46 p.

18. Vásquez-Martín, R., Núnez, P., Bandera, A. & Sandoval, F. Curvature-based environment description for robot navigation using laser range sensors. Sensors 9, 5894–5918 (2009).

19. V.B. Nemirovsky, A.K. Stoyanov. Multi–step segmentation of images by means of a recurrent neural network. Proc. of the 7th Intern. forum on strategic technology (IFOST–2012), Tomsk, 1, 2012. pp. 557–560.

20. Ribic, C. A., Sheavly, S. B., Rugg, D. J. & Erdmann, E. S. Trends and drivers of marine debris on the Atlantic coast of the United States 1997-2007. Mar. Pollut. Bull. 60, 1231–1242 (2010).

21. Bengio, Yoshua; Lamblin, Pascal; Popovici, Dan; Larochelle, Hugo (2007). Greedy layer-wise training of deep networks (PDF). Advances in neural information processing systems. pp. 153–160.

22. Cireşan, Dan; Meier, Ueli; Masci, Jonathan; Schmidhuber, Jürgen (August 2012). "Multi-column deep neural network for traffic sign classification". Neural Networks. Selected Papers from IJCNN 2011. 32: 333–338.

23. G. Marcus., "Is "Deep Learning" a Revolution in Artificial Intelligence?" The New Yorker, 25 November 2012.

24. Socher, Richard; Lin, Cliff; Ng, Andrew Y.; Manning, Christopher D. Parsing Natural Scenes and Natural Language with Recursive Neural Networks. The 28th International Conference on Machine Learning (ICML 2011).

25. Alexander Mordvintsev; Christopher Olah; Mike Tyka (17 червня 2015). Inceptionism: Going Deeper into Neural Networks. Google Research Blog.

26. J. Weng, "Why Have We Passed `Neural Networks Do not Abstract Well'?, " Natural Intelligence: the INNS Magazine, vol. 1, no.1, pp. 13-22, 2011.

27. Hinton, Geoffrey; Salakhutdinov, Ruslan (2006). Reducing the Dimensionality of Data with Neural Networks. Science 313: 504–507.

28. J. Gao, X. He, L. Deng (2014) "Deep Learning for Natural Language Processing: Theory and Practice (Tutorial), " CIKM.

29. Socher, Richard (2013). Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. EMNLP 2013.

30. Бессонов А.А. Интеллектуальная система распознавания лиц с использованием сверточной нейронной сети / А.А. Бессонов, Ю.Ю. Белов // В сб. тез. докл. по материалам 4-ой междунар. научно-технич. конф. «Проблеми інформатизації», Черкаси-Баку-Бельсько-Бяла-Полтава, 2016. - С. 21.