Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

РОЗРОБКА МЕТОДУ ВИДАЛЕННЯ РУКОПИСНИХ НАПИСІВ НА ФОТОГРАФІЇ
(тема)

Виконав:
студент 4 курсу, групи ІТІНФ-19-1

Ріпний В. В.
(прізвище, ініціали)

Спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник доц. Кобилін О.А.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Кобилін О.А.
(підпис) (прізвище, ініціали)

2023 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Ріпному Владиславу Вячеславовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка методу видалення рукописних написів на фотографії

затверджена наказом університету від 15 травня 2023 року № 474 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 02 червня 2023 р.

3. Вихідні дані до роботи науково-методична та науково-технічна література, матеріали конференцій, дані інтернет-мережі, бібліотеки комп'ютерного зору з відкритим кодом Tensorflow/Keras.

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд методів розпізнавання та видалення рукописних написів на фотографії.
2. Моделювання та розробка датасету та програмного забезпечення.
3. Тестування програмного забезпечення та оцінка подальших перспектив розвитку роботи.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Актуальність проблеми наявності рукописних написів на фотографії, постановка задачі, сучасні методи розпізнавання рукописних написів, датасет для навчання моделі, вибір мережі для навчання, результати розпізнавання.

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

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Консультант з дотримання діючих стандартів та норм	Доцент Творошенко І.С.		

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	10.04.2023	
2	Аналіз завдання, підбір літератури	11.04.23-17.04.23	
3	Аналіз літератури з досліджуваної проблеми	18.04.23-20.04.23	
4	Аналіз технічних засобів	21.04.23-30.04.23	
5	Розробка методу розпізнавання написів	01.05.23-14.05.23	
6	Програмна реалізація	15.05.23-23.05.23	
7	Оформлення пояснювальної записки	24.05.23-26.05.23	
8	Перевірка на плагіат	27.05.23	
9	Рецензування	28.05.23	
10	Підготовка презентації та доповіді	29.05.23-30.05.23	
11	Занесення роботи в електронний архів	31.05.23	
12	Попередній захист кваліфікаційної роботи	11.06.23	

Дата видачі завдання 10 квітня 2023 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Кобилін О.А.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 59 с., 2 табл., 23 рис., 42 джерела.

КОМП'ЮТЕРНИЙ ЗІР, РОЗПІЗНАВАННЯ РУКОПИСНИХ НАПИСІВ НА ФОТО, МАШИННЕ НАВЧАННЯ, ЗГОРТКОВІ НЕЙРОННІ МЕРЕЖІ, ДАТАСЕТ.

Об'єктом роботи є набір зображень з друкованим текстом та рукописними помітками.

Метою роботи є розробка методу розпізнавання та видалення рукописних написів із зображень з друкованим текстом.

Використано методи машинного навчання. Проведено дослідження існуючих рішень проблеми. Розроблений власний тренувальний датасет із зображень та маско сегментації. Досліджено та програмно реалізовано різні архітектури нейронних мереж та їх тренування.

У результаті роботи розроблений програмний продукт для тестування різних архітектур нейронних мереж, а також створений аутентичний датасет для тренування моделей.

COMPUTER VISION, RECOGNITION OF HANDWRITTEN INSCRIPTIONS ON IMAGES, MACHINE LEARNING, CONVOLUTIONAL NEURAL NETWORKS, DATASET.

The object of the study is a set of images with printed text and handwritten notes.

The aim of the study is to develop a method for recognizing and removing handwritten labels from images with printed text.

Machine learning methods are used. A study of existing solutions to the problem was conducted. An own training dataset of images and mask segmentation was developed. Different neural network architectures and training were studied and implemented in software.

As a result of the work, a software product for testing different neural network architectures was developed, and an authentic dataset for training models was created.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ШІ – штучний інтелект

OCR – Optical Character Recognition (оптичне розпізнавання символів)

HCR – Handwritten Character Recognition (розпізнавання рукописних символів)

ICR – Intelligent Character Recognition (інтелектуальне розпізнавання символів)

HMM – Hidden Markov Model (прихована модель Маркова)

SVM – Support Vector Machine (метод опорних векторів)

FCNN – Fully Convolutional Neural Network (повністю згорткова нейронна мережа)

CRF – Conditional Random Field (умовне випадкове поле)

PSPNET – Pyramid Scene Parsing Network (пірамідальна мережа розбору сцен)

GNN – Graph Neural Network (графова нейронна мережа)

HED – Hausdorff Edit Distance (відстань Гаусдорфа)

ВСТУП

Процес оцифрування по своїй суті є досить складним та комплексним. На початку, потрібно добре знати той тип даних, який буде підлягати оцифруванню, дані мають бути приведені до єдиного формату та розміру. Також важливим аспектом є те, щоб вони були позбавлені від максимуму шумів, були чіткими та з мінімумом зайвих артефактів. Прикладом такого процесу є оцифрування робіт науковців, які жили в часи далекі до сучасної діджиталізації. Одним з проєктів, який займається тим, що збирає, оцифровує, обробляє та систематизує такі старі записи науковців, які відносяться іноді до середини минулого тисячоліття, є проєкт «digispecies» [1]. Важливим аспектом для успішного оцифрування є етап передобробки зображень, а саме очищення їх від різних шумів та артефактів. Одним з найпоширеніших видів артефактів, який зустрічається на сторінках робіт та публікацій – це рукописні помітки, які робили минулі читачі. Ці помітки мають різний вигляд: підкреслення обведення окремих слів, речень та абзаців, стрілки, написи на полях та між рядками тексту. Для кращого запам’ятовування та структурування прочитаної інформації, такі помітки є корисним інструментом, який дозволяє виділяти головне в тексті та при поверненні через деякий час до матеріалу одразу вказує на важливі місця. Втім для процесу оцифрування, даний вид поміток є однією з перешкод. Є декілька варіантів вирішення такої проблеми. Перший – ручне видалення артефактів з кожного скану/фото сторінки в графічному редакторі. Другий – створення системи, яка б могла автоматизувати цей процес. Метою даної роботи є проєктування та створення саме такої системи.

Комп’ютерний зір – це галузь штучного інтелекту (ШІ), яка дозволяє комп’ютерам і системам отримувати значущу інформацію з цифрових зображень, відео та інших візуальних даних і виконувати дії або давати рекомендації на основі цієї інформації, зазначено на офіційному сайті IBM [2].

Комп’ютерний зір допомагає машинам виконувати функції людського зору, але так, щоб ці операції виконувалися за набагато менший час та за допомогою камер, даних і алгоритмів, а не сітківки, зорових нервів і зорової кори головного мозку. Оскільки система, навчена перевіряти продукцію або спостерігати за виробничими активами, може аналізувати тисячі продуктів або процесів за хвилину, помічаючи непомітні дефекти або проблеми, вона може швидко перевершити людські здібності.

Говорячи про актуальність даної теми, слід також зазначити той факт, що за прогнозами, світовий ринок комп’ютерного зору, ймовірно, досягне 26,11 мільярдів доларів США при середньорічному темпі зростання 7,3% в прогнозованому періоді до 2033 року [3].

1 ОГЛЯД ОБЛАСТІ РОЗПІЗНАВАННЯ СИМВОЛІВ ТА ІСНУЮЧИХ РІШЕНЬ

1.1 Розпізнавання рукописного тексту

1.1.1 OCR – оптичне розпізнавання символів

Оптичне розпізнавання символів – популярний напрямок досліджень в епоху високих технологій. По суті, це процес машинного навчання, під час якого машину навчають читати, як людина. OCR корисний у багатьох відношеннях. Його можна використовувати для збереження літератури (наприклад, старовинних рукописів), яка знаходиться на папері, але не оцифрована, для розпізнавання номерів автомобілів із зображень камер відеоспостереження тощо. Для розпізнавання текстів було зроблено багато роботи і дослідження все ще тривають для досягнення кращих результатів. Виходячи з методів збору даних, діяльність з розпізнавання можна розділити на дві основні категорії: Оптичне розпізнавання символів в режимі онлайн та оптичне розпізнавання символів в режимі офлайн [4]. Офлайн-метод можна розділити на дві підкатегорії – друкований і рукописний.

Розпізнавання символів онлайн: це динамічна система, що працює в режимі реального часу. Вона обробляє символ під час його створення. Зовнішні фактори, такі як положення стилусу, тиск, швидкість письма, нанесення штрихів тощо, відіграватимуть важливу роль в ідентифікації символів [5]. Деталі, пов’язані з рухом пера для створення кривих і штрихів, також допоможуть у цьому процесі. Ідентифікація тексту, написаного за допомогою стилусу на портативних пристроях, є прикладом онлайн розпізнавання символів

Офлайн-розпізнавання символів: в режимі офлайн існуючий документ оцифровується, зберігається, а потім обробляється [5]. Таким чином, він не має жодної зовнішньої інформації, що ускладнює процес розпізнавання. Офлайн-

розпізнавання можна використовувати для друкованих або рукописних документів:

– друковані документи: позитивним фактором для друкованого матеріалу є наявність символів єдиного стилю та розміру. Це полегшує розпізнавання;

– рукописні документи: система розпізнавання рукописних символів (HCR) дуже важка і складна. Письмові символи відрізняються у різних авторів. Навіть написані однією людиною символи можуть відрізнятися в різні моменти часу в межах слова, речення, абзацу або документу.

Нижче наведено основні етапи процесу оптичного розпізнавання символів (рис. 1.1).

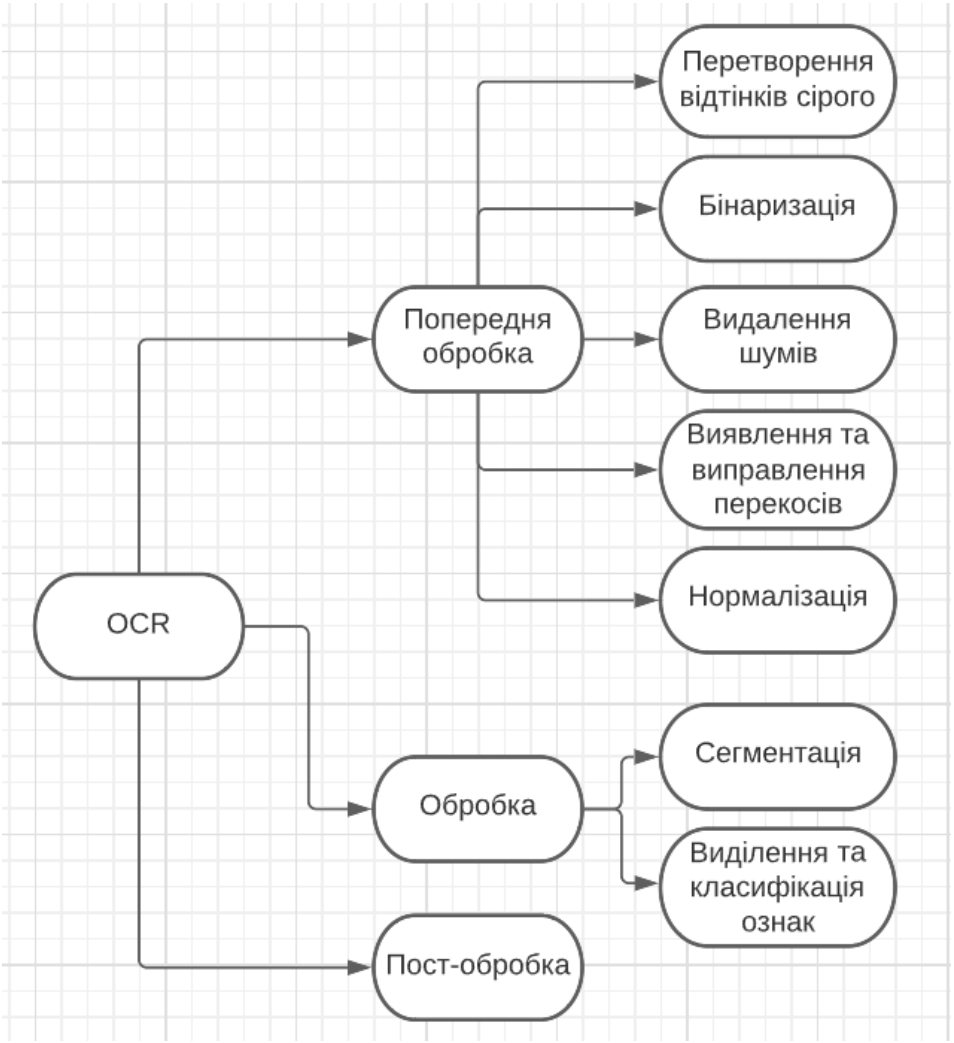


Рисунок 1.1 – Основні етапи OCR

1.1.2 Вплив машинного навчання на розпізнавання рукописного тексту та можливі застосування

Нещодавні досягнення в галузі глибокого навчання, такі як впровадження трансформаторних топологій, прискорили прогрес у розпізнаванні рукописних символів. Інтелектуальне розпізнавання символів (ICR) – це слово, яке використовується для опису процесу розпізнавання рукописного контенту. Алгоритми, що використовуються для вирішення задач ICR, вимагають набагато більше інтелекту, ніж ті, що використовуються для вирішення задач звичайного OCR. Далі буде детальніше описано про завдання ідентифікації рукописного тексту, прикладні задачі, їх складнощі та про те, як їх можна вирішити за допомогою методів машинного та глибокого навчання [6].

Охорона здоров’я та фармацевтика: у галузі охорони здоров’я/фармацевтики оцифрування ліків для пацієнтів є серйозною проблемою. Наприклад, компанія «Roche» щодня обробляє мільйони петабайт медичних PDF-файлів [7]. Запис пацієнтів і оцифрування форм – інші сфери, де розпізнавання рукописного тексту має значний вплив. Лікарні та фармацевтичні компанії можуть значно покращити якість обслуговування клієнтів, додавши аналіз рукописного тексту до свого набору послуг.

Страхування: середня крупна страхова компанія отримує понад 20 мільйонів документів на день, і затримка в обробці заяв може мати значний вплив на організацію. Документ може містити різні стилі почерку, і покладання лише на людський фактор при обробці заяв може значно сповільнити процес.

Онлайн бібліотеки: величезні обсяги історичних знань оцифровуються і стають доступними для всього світу шляхом завантаження сканів зображень. Однак ці зусилля будуть неефективними, якщо текст на фотографіях не буде ідентифікований, проіндексований, переглянутий і процитований. Ідентифікація почерку необхідна для того, щоб повернути до життя

середньовічні документи, листівки та дослідницькі роботи, написані в середньовіччі та XX столітті.

Методи машинного навчання, такі як приховані моделі Маркова (HMM), SVM (метод опорних векторів) та інші, використовувалися в ранніх спробах вирішити проблему розпізнавання рукописного тексту. Після попередньої обробки початкового тексту використовується вилучення ознак для виявлення важливої інформації про кожен символ, такої як петлі, переломні моменти, співвідношення сторін тощо. Для отримання результатів ці створені ознаки передаються в класифікатор, такий як HMM. Через ручний етап виділення ознак та їхню низьку здатність до навчання, продуктивність алгоритмів машинного навчання досить обмежена. Оскільки етап вилучення ознак відрізняється для кожної мови, він не є масштабованим. З впровадженням глибокого навчання швидкість розпізнавання рукописного тексту значно покращилася.

Методи обробки зображень використовуються в усіх сферах досліджень. Ці методи дозволяють отримати додаткову інформацію, краще зрозуміти об’єкт, який вивчається. Серед сфер використання методів обробки зображень медицина посідає особливе місце. Біомедичні дані дозволяють оцінити стан здоров’я людини, виявити захворювання на ранніх стадіях. Зображення клітинних структур цитологічних препаратів є одним із прикладів біомедичних даних. На основі методів аналізу зображень можна виділити різні компоненти клітинних структур цитологічних препаратів. Для цього застосовуються методи вейвлет-аналізу [8] для різних колірних компонентів вхідного зображення. Застосовуючи морфологічний аналіз, ми можемо ідентифікувати окремі клітинні структури. Результати показано на прикладі зображень клітинних структур цитологічних препаратів.

В роботі Гороховатського В. [9] представлено результати розробки нового швидкісного методу класифікації зображень з використанням структурного підходу. В основу методу покладено систему ієрархічних ознак, що базується на побітовому розподілі даних для множини дескрипторів опису

зображень. Експерименти показали, що час обчислення релевантності для двох описів за їх розподілами приблизно в 1000 разів менший, ніж для традиційної процедури голосування, для якої порівнюються набори дескрипторів. Введення системи ієрархічних ознак дозволяє додатково скоротити час обчислень у 2-3 рази при забезпеченні високої ефективності класифікації. За результатами досліджень встановлено, що граничним ступенем ієрархії ознак для надійної класифікації зі стандартним відхиленням шуму менше 30 є 8-розрядний розподіл. Обчислювальні витрати пропорційно зростають зі зменшенням розрядності. Метод може бути використаний для прикладних задач, де час ідентифікації об'єктів є критичним.

1.1.3 Проблеми розпізнавання рукописного тексту

Основними проблемами при розпізнаванні рукописного тексту є наступне:

– штрихи мають велику варіативність і неоднозначність від людини до людини;

– стиль почерку користувача також час від часу змінюється і є нерівномірним;

– деградація вихідного тексту з часом призвела до погіршення якості;

– людям не потрібно писати рядок тексту в одному напрямку на білому папері, проте текст у друкованих документах розташовується по прямій лінії;

– розділення та розпізнавання символів у скорописному почерку ускладнене;

– у порівнянні з друкованим текстом, де весь текст розташований прямо, рукописний текст може мати різний нахил вправо;

– порівняно з синтетичними даними, отримання добре розміченого набору даних для навчання є недоступним.

1.2 Прості методи розпізнавання та видалення тексту з фотографії

Існує множина онлайн та офлайн застосунків, які дозволяють автоматично редагувати фотографії, покращувати їх якість та видаляти зайві деталі. Такі рішення особливо стали популярними, коли трапився значний скачок в розробках штучного інтелекту в останні роки. Прикладом такого може служити сервіс «cleanup.pictures» [10]. Приклад роботи по видаленню тексту з зображення можна побачити на рисунку 1.2.

Аналогічним за суттю та метою є сервіс «cutout.pro» [11]. Результат його роботи відображено на рисунку 1.3.

а) б)

Рисунок 1.2 – Робота сервісу «cleanup.pictures» по видаленню тексту з фото: а) зображення до обробки; б) зображення після обробки

а) б)

Рисунок 1.3 – Робота сервісу «cutout.pro» по видаленню тексту з фото: а) зображення до обробки; б) зображення після обробки

1.3 Згорткові нейронні мережі в розпізнаванні рукописного тексту

1.3.1 Опис архітектури мереж

В своїй роботі [12] Andreas K¨olsch описує та порівнює роботу декількох популярних згорткових нейронних мереж по розпізнаванню рукописного тексту.

FCN-8s – це багатопотокова архітектура FCNN, яка була запропонована Лонгом та ін. у 2015 році [13] і набирає середній бал 65,4% на завданні сегментації Pascal Visual Object Classes (VOC) Challenge. Архітектура базується на відомій архітектурі VGG-16, яка показала найкращі результати в задачі локалізації на конкурсі ImageNet Large Scale Visual Recognition Competition (ILSVRC) [14] у 2014 році. Як і VGG-16, FCN-8 використовує п’ять стеків згорткових шарів з додаванням max-pooling.

Кожен з цих шарів об’єднання зменшує висоту і ширину карт ознак у 2 рази. Таким чином, карти ознак після п’ятого шару об’єднання мають розмір $(1/32 * H) \times (1/32 * W)$, де H і W позначають висоту і ширину вхідного зображення, відповідно (рис. 1.4) [12]. П’ять згорткових стеків зменшують розмір зображення в 2 рази кожен У той час як VGG-16 подає ознаки останнього шару об’єднання до стеку з трьох повністю з’єднаних шарів для класифікації, карти ознак у FCN-8 обробляються шаром згортки 1×1, який створює по одній карті ознак на клас, тобто оцінки класу.

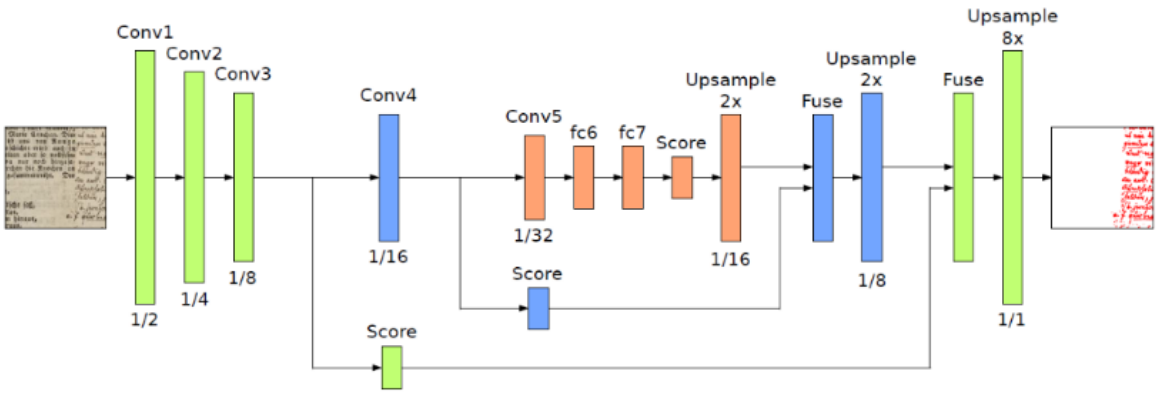


Рисунок 1.4 – Сегментація зображення за допомогою багатопотокової мережі FCN-8

Deeplab v2 [15] – це FCNN, яка базується на архітектурі ResNet-101 [16] і був запропонований Ченом та ін. у 2016 році. Використовуючи атомарну згортку, багатомасштабні зображення та умовні випадкові поля (CRF) [17], метод показує 79,7% середнього IOU на задачі сегментації Pascal VOC Challenge. Модель розширених згорток, також відома як розширена згортка, має практичну перевагу над звичайними згортками та шарами з максимальним об’єднанням, оскільки просторова роздільна здатність карти ознак може бути збережена без будь-якої додаткової дискретизації.

Пірамідальна мережа розбору сцен (PSPNET). Як і DeepLab v2, PSPNET [18] – це FCNN-архітектура, яка базується на ResNet з розширеними згортками. Вона була запропонована у різних версіях, побудованих на основі ResNet-50, ResNet-101, ResNet-152 та ResNet-269. На Pascal VOC Challenge метод показав 85,4% середнього IOU.

Інші запропоновані підходи та методи до вирішення задачі обробки та класифікації зображень. У роботі [19] представлено результати розв'язання задач класифікації зображень у системах комп'ютерного зору з використанням структурних методів. Розроблено моделі оцінювання статистичних даних під час визначення релевантності та класифікації зображень. В статтях [20, 21] розглянуто методологію обробки зображень при дослідженні полімерних композицій. Показано необхідність застосування методології обробки зображень при дослідженні полімерних композицій. Однією з перших задач при обробці зображення, є визначення його границь [22]. У роботі [23] говориться про методологію обробки зображень, яка полягає в компенсації різних геометричних спотворень вхідного зображення, отриманого в результаті реєстрації тестового зображення і передачі його в систему інтелектуального аналізу даних, в порівнянні з деяким еталонним зображенням. Про кластеризацію об'єктів на зображенні, що дозволяє реалізувати різні процедури сегментації зображень з подальшим аналізом, який призведе до прийняття відповідних рішень, говориться в роботі [24] (в контексті дослідження хвороб риб за допомогою методики сегментації

кольорових зображень). У статті [25] запропоновано адаптивні модифікації методів нечіткої кластеризації для розв'язання задачі інтелектуального аналізу потоків даних в онлайн-режимі, а також розглянуто задачу кластеризації-сегментації коротких часових рядів з нерівномірно розподіленими спостереженнями (одночасно у всіх вибірках). Запропонований підхід до адаптивної нечіткої кластеризації потоку даних є досить простим у чисельній реалізації та характеризується високою швидкістю обробки інформації. В роботі [26] говориться про метод кольорової сегментації, який було використано для аналізу структури мікроскопічних зображень крові. Результати показали, що сегментація зображення збільшується при використанні методу нечіткого маскування, що, в свою чергу, призводить до збільшення та покращення аналізу мікроскопічних зображень крові. В статті [27] обговорюються проблеми інтелектуальної обробки багатовимірних даних у системах комп'ютерного зору. Запропоновано метод генерування стисненого структурного опису зображення в термінах гранулювання значень ознак у вигляді скінченновимірних векторів. Такий підхід дозволяє мінімізувати час обчислень і зменшити кількість помилкових відповідностей. У роботі [28] запропоновано метод пошуку об'єктів на зображенні, заснований на ідентифікації кластерного представлення описів запиту і поточного зображення вікна з обчисленням міри релевантності.

1.3.2 Результати тестування

Провелося навчання та оцінювання FCNN, описаних в розділі вище. Таким чином, було досліджене навчання з перенесенням, бінаризовані зображення та методи аугментації даних. Всі експерименти виконано на графічному процесорі NVIDIA Titan X. Наводиться середній бал IOU для всього тестового набору. Для кожного класу оцінка IOU дорівнює

$$\frac{TP}{(FP+TP+FN)}, \tag{1.1}$$

де TP – істино позитивні результати;

FP – хибно позитивні результати;

FN – хибно негативні результати.

Середній показник IOU – це просто середнє значення по всіх класах. Обчислення IOU проводиться як у роботі Лонга та ін. [12]

$$mean\ IoU = \frac{1}{n_{cl}} \times \sum_i \frac{n_{ii}}{(t_i+\sum_i n_{ji}-n_{ii})}, \tag{1.2}$$

де n_{ii} – кількість класів;

n_{ji} – кількість пікселів класу i, які за прогнозом належать до класу j;

t_i – загальна кількість пікселів класу i.

Для оцінки навчених мереж, було пропущено через мережу ділянки розміром 512×512 пікселів з тестових зображень за принципом ковзного вікна з перекриттям 128 пікселів. Результати, отримані різними мережами, наведено в таблиці 1.1 [12]. Таблиця 1.1 показує, що ані бінаризація зображень, ані попереднє навчання на DIVA-HisDB не дає хороших результатів. У випадку навчання і тестування на бінарних зображеннях багато інформації про пікселі втрачається вже під час попередньої обробки. Хоча кольоровий піксель може мати 224 різних значення, на бінарних зображеннях він може бути лише чорним або білим. Різноманітність набору даних також може пояснити, чому попереднє навчання на DIVA-HisDB призвело до низького середнього показника IOU, який становив лише 70%. Після навчання на DIVA-HisDB FCN-8s досягає середнього значення IOU понад 90% на тестовому наборі DIVA-HisDB, що можна порівняти з найсучаснішим рівнем. Однак, якщо ці ваги використовувати як початкові для навчання на нашому наборі даних, мережа не здатна узагальнювати нові типи документів, оскільки вона занадто схильна до однорідних зображень DIVA-HisDB.

Таблиця 1.1 – Продуктивність мереж на новому наборі даних з різним попереднім навчанням та аугментацією даних

	Попередня підготовка	**Аугментація даних**	**Значення IOU**
FCN-8s	ILSVRC	Бінаризація	63,0%
FCN-8s	DIVA-HisDB	Випадкове обрізання	70,0%
PSPNET	ILSVRC	Випадкове обрізання	87,7%
DeepLab v2	ILSVRC	Випадкове обрізання	89,2%
FCN-8s	ILSVRC	Випадкове обрізання	91,3%
FCN-8s	ILSVRC	Inception	95,6%

Попереднє навчання мережі на наборі даних ILSVRC дає загалом хороші результати, оскільки навчені згорткові фільтри є дуже надійні і добре узагальнюють завдяки великій кількості навчальних даних.

1.4 Постановка задачі

Таким чином, розробка методу автоматичного розпізнавання та видалення рукописних написів на фотографіях є актуальним та перспективним завданням для спрощення процесу оцифровування документів/книг. Тому ставиться завдання проєктування програмного застосунку на базі нейронної мережі по обробці зображень з рукописними написами.

Об’єктом роботи є набір зображень з друкованим текстом та рукописними помітками.

Метою роботи є розробка методу розпізнавання та видалення рукописних написів із зображень з друкованим текстом.

Для досягнення мети необхідно вирішити такі завдання:

– провести аналіз існуючих методів розпізнавання написів на зображеннях;

– на основі аналізу обрати тип нейронної мережи для виконання роботи;

– створити датасет для навчання нейронної мережі;

– спроєктувати та написати програмний застосунок з реалізацією обраної моделі;

– провести навчання моделі на створеному датасеті.

2 ТЕОРЕТИЧНІ ТА МАТЕМИТИЧНІ АСПЕКТИ МЕТОДІВ РОЗПІЗНАВАННЯ ТА ВИДАЛЕННЯ РУКОПИСНИХ НАПИСІВ ІЗ ЗОБРАЖЕННЯ

2.1 Теоретичні аспекти згорткових нейронних мереж

2.1.1 Загальний опис згорткових нейронних мереж

Згорткова нейронна мережа, також відома як CNN або ConvNet – це клас нейронних мереж, який спеціалізується на обробці даних, що мають сітчасту топологію, таких як зображення. Цифрове зображення – це двійкове представлення візуальних даних (рис. 2.1) [29]. Воно містить серію пікселів, розташованих у вигляді сітки, яка містить значення пікселів для позначення яскравості та кольору кожного пікселя.

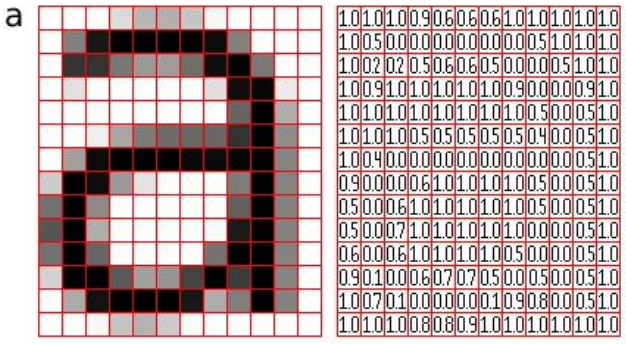


Рисунок 2.1 – Представлення зображення у вигляді сітки пікселів

Людський мозок обробляє величезну кількість інформації в ту секунду, коли ми бачимо зображення. Кожен нейрон працює у своєму власному рецептивному полі і пов’язаний з іншими нейронами таким чином, що вони охоплюють все поле зору. Подібно до того, як кожен нейрон реагує на стимули лише в обмеженій області зорового поля, що називається рецептивним полем у біологічній системі зору, кожен нейрон у CNN також обробляє дані лише у своєму рецептивному полі. Шари розташовані таким чином, що спочатку вони

виявляють простіші патерни (лінії, криві тощо), а далі – складніші (обличчя, об’єкти тощо). Використовуючи CNN, можна надати комп’ютерам можливість «бачити».

2.1.2 Архітектура згорткової нейронної мережі

CNN складається з вхідного шару, вихідного шару та багатьох прихованих шарів між ними (рис. 2.2) [30].

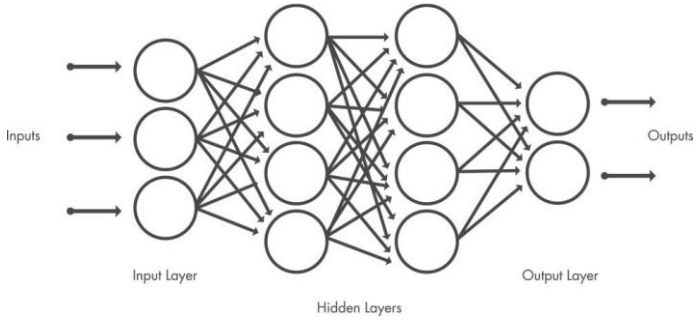


Рисунок 2.2 – Загальна архітектура CNN

Якщо відійти від загального вигляду архітектури CNN, то більш детально вона має три шари: шар згортки, шар об’єднання та повністю з’єднаний шар (рис. 2.3) [29]. Далі про кожен з цих шарів детальніше.

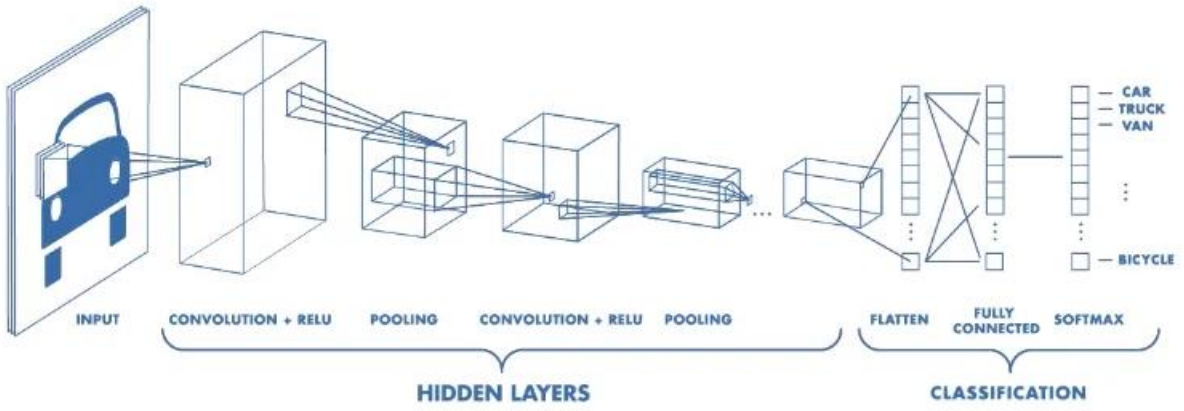


Рисунок 2.3 – Детальніший вигляд архітектури CNN

2.1.3 Згортковий шар

Як говориться в статті [31], згортковий шар – це перший шар, який витягує ознаки з вхідного зображення і працює шляхом накладання фільтра на масив пікселів. Цей шар зберігає кореляцію між пікселями, вивчаючи особливості зображення, використовуючи невеликі квадрати вхідних даних. Це, по суті, математична операція, яка потребує двох вхідних даних, включаючи матрицю зображення і фільтр, що призводить до створення карти особливостей.

По суті, карта особливостей є результатом одного фільтра, застосованого до попереднього шару. Даний фільтр наноситься на попередній шар, переміщуючись по одному пікселю за раз. Кожна позиція призводить до активації нейронів, а результати активації збираються на карті особливостей.

Якщо ми маємо вхідні дані розміром $W \times W \times D$ і $Dout$ кількість ядер з просторовим розміром F з кроком S, і кількістю пропусків P, то розмір вихідного об’єму може бути визначений за наступною формулою:

$$W_{out} = \frac{W-F+2*P}{S} + 1. \quad (2.1)$$

Деякі з переваг використання концепції згорткового шару викладено в статті [29]: згортка використовує три важливі ідеї, які мотивували дослідників комп’ютерного зору: розріджена взаємодія, спільне використання параметрів та еквівалентне представлення. Опишемо кожну з них детальніше.

Тривіальні шари нейронної мережі використовують множення матриці на матрицю параметрів, що описують взаємодію між вхідним і вихідним блоком. Це означає, що кожен вихідний елемент взаємодіє з кожним вхідним елементом. Однак згорткові нейронні мережі мають розріджену взаємодію. Це досягається за рахунок того, що ядро має менший розмір, ніж вхідні дані, наприклад, зображення може мати мільйони або тисячі пікселів, але при обробці його за допомогою ядра ми можемо виявити значущу інформацію, яка

складається з десятків або сотень пікселів. Це означає, що нам потрібно зберігати менше параметрів, що не тільки зменшує вимоги до пам’яті моделі, але й покращує статистичну ефективність моделі.

Якщо обчислення однієї функції в просторовій точці (x_1, y_1) є корисним, то воно також має бути корисним в іншій просторовій точці, скажімо, (x_2, y_2). Це означає, що для одного двовимірного зрізу, тобто для створення однієї карти активації, нейрони змушені використовувати один і той же набір ваг. У традиційній нейронній мережі кожен елемент вагової матриці використовується один раз і більше ніколи не повертається до нього, в той час як згорткова мережа має спільні параметри, тобто для отримання виходу ваги, застосовані до одного входу, є такими ж, як і ваги, застосовані до інших входів.

Завдяки спільному використанню параметрів, шари згорткової нейронної мережі матимуть властивість еквівалентності до перекладу. Це означає, що якщо ми змінили вхідні дані певним чином, то вихідні дані також будуть змінені таким же чином.

2.1.4 Шар об’єднання

Як зазначено на сайті [31], шар об’єднання зменшує розмір вибірки карти об’єктів і додатково пришвидшує обробку, оскільки зменшує загальну кількість параметрів, які потрібно обробити мережі. Результатом шару об’єднання є об’єднана карта об’єктів. Існує два способи створення об’єднаної карти ознак: максимальне об’єднання (max pooling) або середнє об’єднання (average pooling). Різниця між цими двома способами (рис. 2.4):

– average pooling відрізняється від максимального об’єднання тим, що зберігає менш важливу інформацію в об’єднаній карті об’єктів;

– max pooling відкидає ці менш важливі об’єкти, вибираючи максимальне значення об’єднаної карти об’єктів.

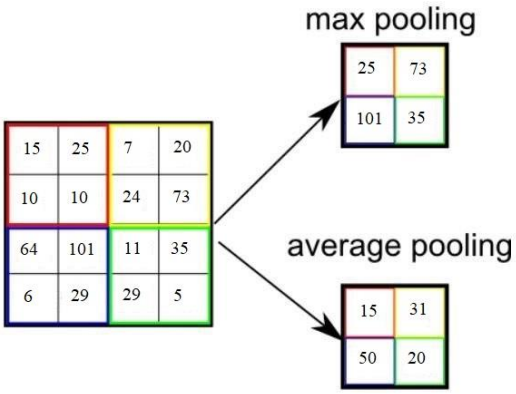


Рисунок 2.4 – Приклад роботи максимального та середнього об’єднань

2.1.5 Повністю з’єднаний шар

Нейрони в цьому шарі мають повний зв’язок з усіма нейронами в попередньому і наступному шарах, як у звичайному FCNN (Fully Connected Neural Network). Ось чому його можна обчислити, як зазвичай, шляхом множення матриці з наступним ефектом зміщення. Fully Connected layer допомагає відобразити представлення між входом і виходом.

2.1.6 Використання CNN у розпізнаванні символів

В своїй роботі, М. Фанаті пропонує наступний метод застосування CNN у поєднанні з SVM для розпізнавання рукописних символів (рис. 2.5) [32].

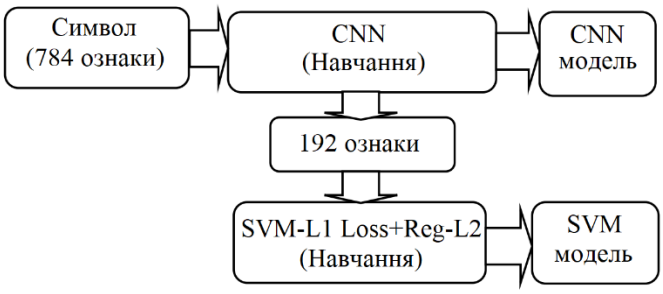


Рисунок 2.5 – Один із можливих сценаріїв поєднання CNN та SVM

В тому дослідженні [32] використовується набір даних NIST SD 19 2-го видання [33] як для навчання, так і для тестування. Він складається з чисел, великих і малих літер та злиття великих і малих літер. Початковий розмір набору даних – 128×128 пікселів. Деяка попередня обробка, така як обрізання та зміна розміру зображення, проведена на NIST, щоб видалити неінформативний фон зображення, дозволила змінити розмір зображень до 28×28 пікселів.

Запропонована архітектура CNN для вилучення ознак для розпізнавання рукописних символів показана на рисунку 2.6. Це стандартна архітектура CNN. Мережа містить п’ять шарів. Перші чотири – це два набори шарів згортки та субдискретизації, за якими слідує вихідний шар, який є повністю з’єднаним шаром.

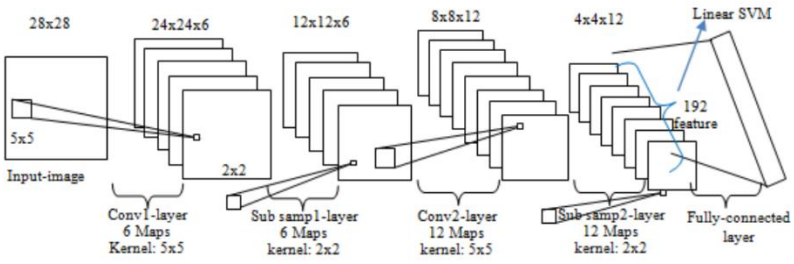


Рисунок 2.6 – Архітектура CNN для вилучення ознак

Вперше SVM була запропонована Вапніком [34]. Ця система широко використовується для класифікації та регресії. Основна ідея SVM полягає у знаходженні найкращої гіперплощини, яка максимізує запас гіперплощини. Гіперплощина з максимальним запасом виконує хороше узагальнення. SVM – це квадратичне програмування, де клас нових вхідних даних можна передбачити за допомогою

$$f(x_d) = \sum_{i=1}^{Ns} a_i x_i y_i x_d + b, \quad (2.2)$$

де x_i – опорний вектор;

Ns – деякий опорний вектор;

x_d – вхідні дані, які буть прогнозуватися.

У даному підході оцінюються оригінальні CNN та різновиди SVM підходу, такі як ядро SVM, лінійна SVM з використанням L1 регуляризації та лінійна SVM з використанням L2 функції втрат, як для первинної, так і для подвійної задачі, для порівняння з приведеним вище запропонованим методом. Кожен метод тестується на точність, достовірність та запам'ятовування. Також оцінено час обчислення кожного методу. Також була проведена десятикратна перехресна перевірка для підтвердження запропонованого методу.

У цьому дослідженні було проведено два сценарні експерименти. У першому експерименті оцінюється запропонований метод, використовуючи набір даних NIST як для навчання, так і для тестування. У другому експерименті тестується модель навчання, побудована в першому експерименті, використовуючи більш складний набір даних з документа форми. Випадковим чином вибирається 1000 зразків з набору даних NIST, де 80% для навчання і 20% для тестування. Модель CNN побудовано за допомогою інструментарію Deep Learning Matlab Toolbox. Лінійний SVM-класифікатор – за допомогою Liblinear Matlab Toolbox, а ядро SVM – за допомогою Libsvm Matlab Toolbox. Коефіцієнт точності, достовірність і запам'ятовування кожного методу наведено у таблиці 2.1 [32]. Показники точності, достовірність та запам'ятовування обчислюються у відсотках (%), а час обчислень – у секундах (с).

Виходячи з таблиці 2.1, запропонований метод досягає найкращої точності, достовірності та запам'ятовування, за винятком великих літер. Для великих літер CNN та SVM з регуляризацією L1 досягають кращої точності, ніж запропонований метод, але водночас, метод з регуляризацією L1 вимагає більше часу на навчання, ніж розглянутий вище метод.

Таблиця 2.1 – Точність розпізнавання рукописного тексту

Method	Numeral			Uppercase		
	Acc	Prec	Rec	Acc	Prec	Rec
CNN	98,30	98,30	98,32	92,33	92,33	92,39
CNN+SVM kernel	97,7	97,7	97,71	91,34	91,35	91,48
CNN+SVM L1 Reg-L2 Loss	98,65	98,65	98,66	**93,15**	**93,15**	93,19
CNN+SVM L1 Reg-L2 Loss (dual)	98,7	98,6	98,62	93,05	93,06	93,09
CNN+SVM L1 Reg-L2 Loss (primal)	98,78	98,78	98,77	93,05	93,06	93,09
CNN+SVM L2 Reg-L1 Loss (dual)	**98,85**	**98,85**	**98,86**	**93,05**	**93,06**	**93,08**
Method	Lowercase			Numeral + Uppercase		
	Acc	Prec	Rec	Acc	Prec	Rec
CNN	83,54	83,54	83,68	88,32	88,32	88,84
CNN+SVM kernel	82,21	82,21	82,49	89,12	89,13	89,82
CNN+SVM L1 Reg-L2 Loss	86,07	86,08	85,99	91,02	90,85	91,36
CNN+SVM L1 Reg-L2 Loss (dual)	86,21	86,21	86,13	90,86	90,86	91,35
CNN+SVM L1 Reg-L2 Loss (primal)	86,19	86,19	86,11	90,84	90,94	91,16
CNN+SVM L2 Reg-L1 Loss (dual)	**86,21**	**86,21**	**86,12**	**91,37**	**91,37**	**91,88**

2.2 Теоретичні аспекти графових нейронних мереж

2.2.1 Огляд та архітектура графових нейронних мереж

З 2006 року теорія графів тісно пов'язана з машинним навчанням завдяки новій концепції застосування графових нейронних мереж. Ми знайомі з деякими типами графових даних, наприклад, з соціальними мережами. Однак графи є надзвичайно потужним і багатоохоплюючим представленням даних, тому далі буде розглянуто на прикладі декількох типів даних (більш детально ніж в першому розділі), як інформація може бути змодельована у вигляді графів: зображення і текст. Хоча це і здається на перший погляд дещо контрінтуїтивним, але можна дізнатися більше про симетрію та структуру зображень і тексту, розглядаючи їх як графи, а також розвинути метод сприйняття інформації, який допоможе зрозуміти інші, менш сітчасті та графові типи даних [35].

Зображення як графи. Зазвичай ми думаємо про зображення як про прямокутні сітки з каналами зображення, представляючи їх як масиви (наприклад, 244×244×3). Інший спосіб розглядати зображення – як графи з регулярною структурою, де кожен піксель є вузлом і з’єднаний ребрами з сусідніми пікселями. Кожен неграничний піксель має рівно 8 сусідів, а інформація, що зберігається в кожному вузлі, є 3-вимірним вектором, що представляє RGB-значення пікселя.

Спосіб візуалізації зв’язності графа – це його матриця суміжності. Ми впорядковуємо вузли, в даному випадку кожен з 25 пікселів у простому зображенні смайлика 5×5, і заповнюємо матрицю $n_{nodes} \times n_{nodes}$ записом, якщо два вузли мають спільне ребро. Зауважимо, що кожне з цих трьох представлень нижче є різними поглядами на один і той самий фрагмент даних (рис. 2.7) [35].

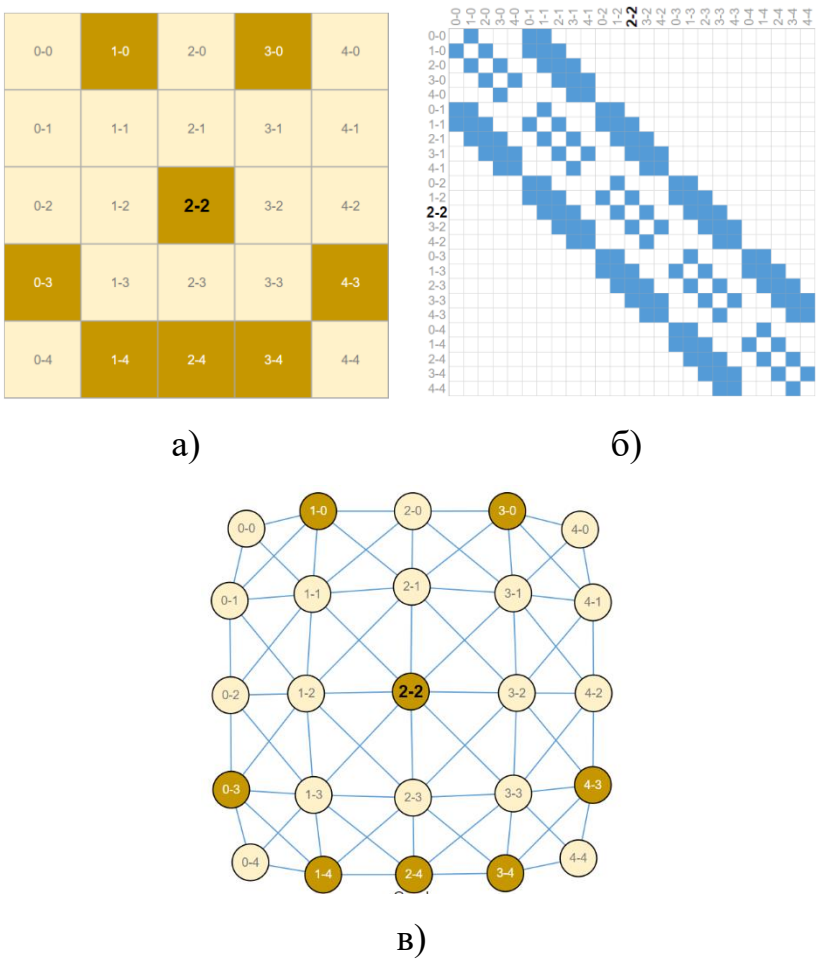


Рисунок 2.7 – Представлення простого зображення в різних форматах:
а) піксельне зображення; б) матриця суміжності; в) граф

Задача на рівні ребер. Однією з проблем прогнозування в графах є прогнозування ребер. Один з прикладів висновків на рівні ребер – це розуміння сцени зображення. Окрім ідентифікації об’єктів на зображенні, моделі глибокого навчання можуть бути використані для прогнозування взаємозв’язків між ними. Ми можемо сформулювати це як класифікацію на рівні ребер: маючи вузли, які представляють об’єкти на зображенні, ми хочемо передбачити, які з цих вузлів мають спільне ребро або яке значення цього ребра. Якщо ми хочемо виявити зв’язки між об’єктами, ми можемо вважати граф повністю зв’язним і на основі передбачених значень підрізати ребра, щоб отримати розріджений граф (рис. 2.8) [35].

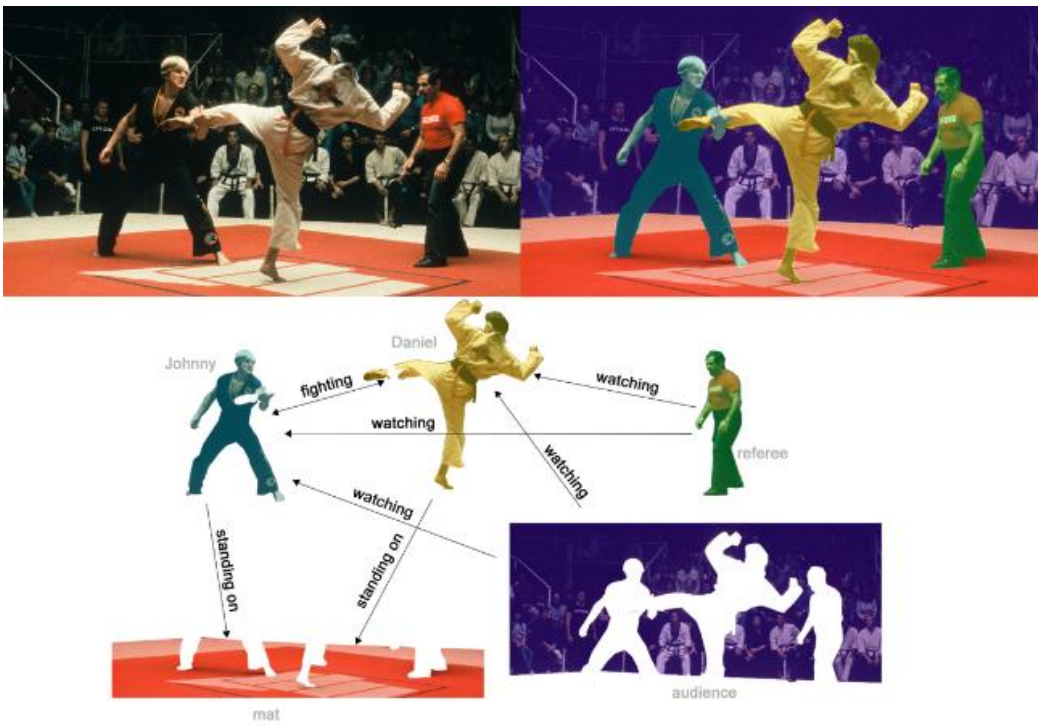


Рисунок 2.8 – Відношення об’єктів на зображенні один до одного у вигляді спрямованих ребер

2.2.2 Типи задач GNN

Вузловий рівень.

Класифікація вузлів є типовою задачею на рівні вузлів (рис. 2.9 а)) [36], яка може бути виконана за допомогою керованого навчання, некерованого навчання та напівкерованого навчання. Як найбільш поширений метод класифікації вузлів, напівконтрольоване навчання поєднує в собі характеристики контрольованого і неконтрольованого навчання. На відміну від навчання під контролем і навчання без контролю, напівконтрольоване навчання на графі витягує високорівневі представлення вершин за допомогою поширення інформації, яке не потребує маркування всіх вершин і ефективно використовує відому пов’язану інформацію. Це налаштування є потужним для завдання виведення асоціацій між об’єктами в біологічній мережі. Наприклад, Ioannidis та ін. [37] побудували множинні мережі білок-білкової взаємодії

(PPI) на основі білкової зв’язності для різних типів клітин і запропонували архітектуру графової залишкової нейронної мережі (GRNN) для напівконтрольованого навчання на мультиреляційних графах. Вплив різних зв’язків вимірювався за допомогою параметрів, що навчаються. Для прогнозування білкової функції в наборах даних про загальні клітини, клітини мозку і клітини кровообігу GRNN отримала макропоказник F1 на рівні 0,86, 0,77 і 0,80, що набагато краще, ніж у базової моделі.

Рівень класифікації ребер.

Основним задачами на рівні ребер є прогнозування зв’язків, коли на основі деяких графів навчається модель прогнозування ребер на основі особливостей вершин або ребер для прогнозування ймовірності зв’язності між парами вершин у цих графах або нових графах, як показано на рисунку 2.9 б) [36]. Задача прогнозування зв’язності привернула увагу різних дослідницьких галузей завдяки своїй широкій застосовності. Моделі GNN також ефективні для розв’язання задач прогнозування зв’язків. Чжан і Чен [38] запропонували модель SEAL (навчання на підграфах, вбудовуваннях і атрибутах для прогнозування зв’язків), засновану на поширенні інформації, яка використовує GNN для заміни повністю зв’язної нейронної мережі в традиційному методі нейронної машини Вайсфейлера-Лемана і вивчає загальні особливості структури графа з локальних підграфів. Його продуктивність на загальнодоступних наборах даних біологічних мереж, таких як дріжджі, Caenorhabditis elegans та Escherichia coli, була вищою, ніж у традиційних моделях вбудовування графів.

Графовий рівень.

На рівні графа основними завданнями є:

– генерація графів, що використовується при розробці ліків для створення нових правдоподібних молекул;

– еволюція графів (за заданим графом передбачити, як він буде розвиватися з часом), використовується у фізиці для прогнозування еволюції систем;

– прогнозування на рівні графів (завдання категоризації або регресії на основі графів), наприклад, прогнозування токсичності молекул.

Завдання рівня графів в основному пов’язані з генерацією графів (рис. 2.9 в)) [36].

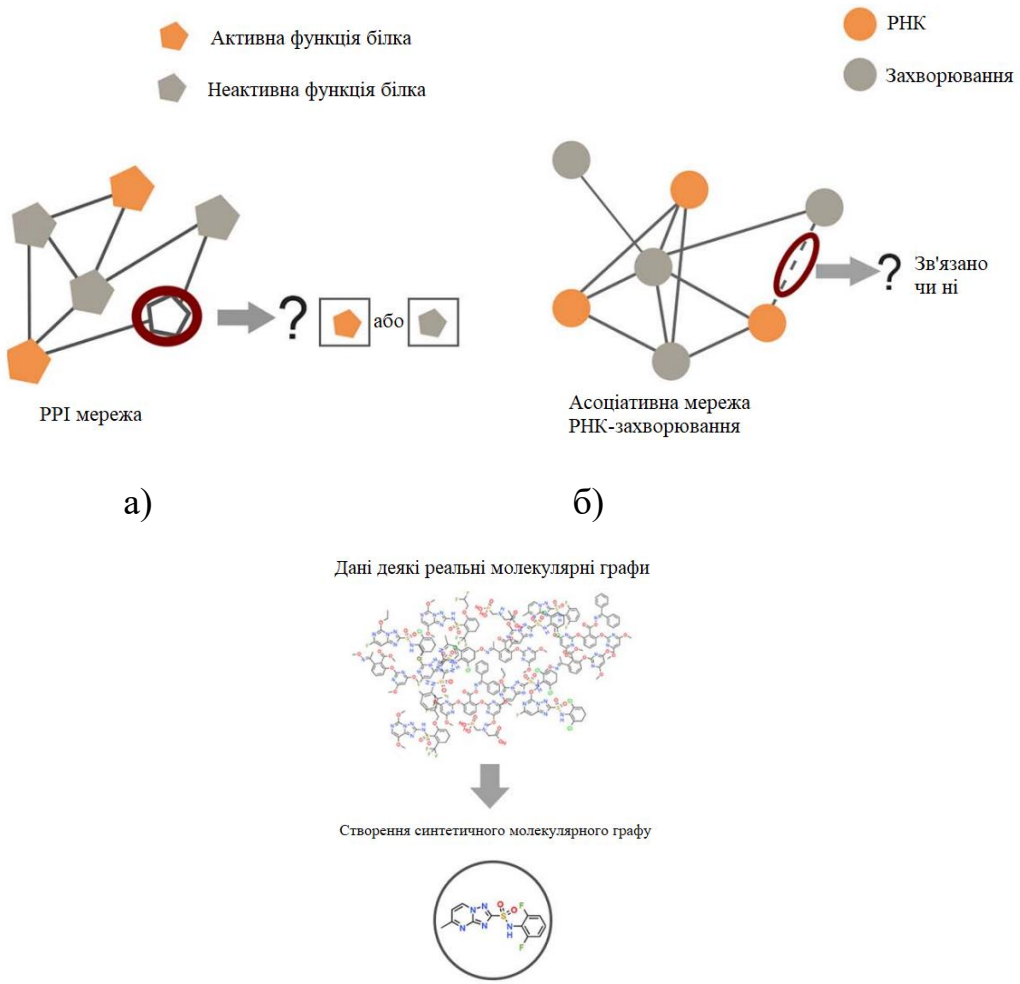


Рисунок 2.9 – Приклади завдань на аналіз графів у трьох рівнях: а) вузловий рівень: передбачення типу немічених білків через мічені білки в мережі взаємодії білок-білок; б) прогнозування невідомого зв’язку між РНК-вузлами; в) генерування синтетичних молекул шляхом фактичного вивчення молекулярних графів

Навчитися генерувати дані про структуру графа шляхом тренування на наборі репрезентативних даних є основою завдань генерації графів. Для

відкриття нових хімічних структур вперше була запропонована модель генерації графів на основі GNN з мотивацією генерації молекулярних графів. Шимановський і Комодакіс [39] об’єднали GNN і VAE, щоб запропонувати GraphVAE, який був використаний для генерації молекулярних графів малого масштабу. Експерименти на базі даних QM9 та ZINC довели, що GraphVAE має вищу точність, ніж попередні методи.

2.2.3 GNN у розпізнаванні рукописних символів

За останні кілька десятиліть графи стали дуже ефективною та вагомою областю досліджень, і багато проблем розпізнавання образів вирішуються за допомогою графів, зокрема, аналіз документів. Хоча статистичні методи історично були більш ефективними з точки зору обчислень при вирішенні завдань аналізу зображень документів, методи, засновані на графах, в останні кілька років досягають еталонних показників. В аналізі документів графіки можуть відображати структурну інформацію форми (знаків, символів та іншого змісту) в документах. Графові методи сьогодні широко використовуються в аналізі документів не як альтернатива статистичним методам, а як доповнення до них. Статистичні методи менш складні і потребують відносно простих математичних операцій, але, з іншого боку, графіки здатні представляти як символьну, так і структурну інформацію, що може бути більш корисним при вирішенні проблем розпізнавання образів. Графи дуже ефективні при вимірюванні подібності в розпізнаванні образів, яке відоме як зіставлення графів. Важливою властивістю є вимірювання схожості або відстані між двома графами. Ця проблема в основному полягає у знаходженні мінімального спільного підграфа, щоб з’ясувати, чи існує ізоморфізм між графами.

Нещодавній успіх згорткових нейронних мереж (CNN) у комп’ютерному зорі та інших галузях також привернув увагу дослідників, які

зосередилися на використанні структурного розпізнавання образів, щоб поширити ці рамки на неевклідові структури, такі як множини та графи. Ці розширення в CNN зазвичай називають геометричним глибоким навчанням (Geometric Deep Learning, GDL). Серед багатьох інших методів, графові нейронні мережі (GNN) стають все більш відомими і досягають значних успіхів.

Для пошуку схожості між двома графами дуже ефективним методом є відстань редагування графів (Graph Edit distance (GED)). Це толерантний до помилок метод зіставлення графів, але основним недоліком GED є його висока обчислювальна складність. GED має експоненціальну часову складність по відношенню до кількості вузлів, що робить неможливим застосування цього методу в реальних умовах. Для подолання проблеми часової складності дослідниками було запропоновано багато алгоритмів, наприклад, в одному з таких запропоновано приблизну відстань редагування графа, яка є методом двостороннього зіставлення графів і базується на розв'язанні задачі про призначення [40]:

$$D(g_1,\ g_2) = \ min_{(e_1,\dots,e_k)\in\gamma(g_1,g_2)} \sum_{i=1}^{k} c(e_i). \qquad (2.3)$$

Рівняння (2.3) використовується для знаходження відстані. Воно використовує матрицю витрат з операціями редагування і дає верхню межу початкового GED. Відстань редагування за Хаусдорфом (HED) дає нижню межу порядку GED та базується на відповідності Хаусдорфа.

3 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ СИСТЕМИ

3.1 Алгоритмізація проєкту

Задача проєкту – це задача сегментації зображень, що являє собою підкатегорію навчання з учителем в області комп'ютерного зору. В даному випадку, необхідно навчити модель розпізнавати та виділяти рукописні написи на зображеннях.

План реалізації виглядає наступним чином:

– створення датасету для навчання моделі: знаходження та вибір «сирих» даних для подальшої обробки, ручне створення масок поміток, обробка зображень та приведення до єдиного типу;

– створення моделі: кодування моделі глибокого навчання для сегментації зображень. В даній роботі було використано та протестовано одні з найбільш популярних архітектур згорткових нейронних мереж: U-Net, ResNet50, FCN-8s;

– навчання моделі на підготованих даних;

– тестування моделі на тестових даних;

– за задовільного результату роботи моделі в розпізнаванні написів, проєктування методу їх видалення з фото.

Короткий огляд технологій для досягнення поставлених цілей.

Мова програмування – Python 3.10: Це остання версія Python на момент роботи над проєктом. Python є найкращою мовою для роботи з машинним навчанням і обробкою зображень через її простоту і велику кількість підтримуваних бібліотек.

TensorFlow/Keras: TensorFlow – це бібліотека машинного навчання з відкритим вихідним кодом, яка надає комплексний набір інструментів для створення і навчання моделей глибокого навчання. Keras – це високорівневий API для TensorFlow, який спрощує процес створення і навчання моделей.

OpenCV: це бібліотека з відкритим вихідним кодом для обробки зображень і комп’ютерного зору. Ми будемо використовувати її для завантаження, обробки та відображення зображень.

NumPy: бібліотека для роботи з масивами даних.

Matplotlib: бібліотека для візуалізації даних. Ми будемо використовувати її для відображення зображень.

Як альтернативу TensorFlow/Keras можна використовувати PyTorch, іншу популярну бібліотеку для машинного навчання. Однак, TensorFlow/Keras зазвичай вважаються простішими і мають більшу підтримку в спільноті розробників.

Важливо зазначити, що для виконання поставленого завдання потрібне досить потужне обладнання, зокрема, графічний процесор (GPU), який значно прискорює процес навчання моделей глибокого навчання а також певний набір оперативної пам’яті для завантаження навчального набору даних.

Варто зазначити, що попередньо проводилось тестування процесу навчання нейронних мереж на різних наборах даних, та виявилося, що про використанні досить об’ємних вхідних даних з високою роздільною здатністю, процес навчання є досить повільним та займає багато часу. Розробка програм в сфері машинного навчання по своїй суті є досить складним та ітеративним процесом, тобто вимагає постійного доналаштування системи та її параметрів після кожного запуску процесу навчання та оцінки результатів. Тобто швидкість навчання є однією з критичних властивостей системи, особливо в рамках обмеженості в часі. Тому, через те, що ресурси домашнього ПК обмежені (в даному випадку відеокарта NVIDIA GTX1650 (чотири гігабайти відеопам’яті) та вісім гігабайт ОЗУ) було прийнято рішення використовувати ресурси хмарних обчислень. Одними з найпопулярніших є Google Colab та Azure Virtual Machine. Для прискорення роботи, було прийнято рішення використовувати два дані ресурси паралельно.

Google Colaboratory – це безкоштовне інтерактивне хмарне середовище для роботи з кодом від Google. Сервіс потрібен, щоб писати код у jupyter

notebook. Він функціонує за принципом хмари, що дає змогу працювати над одним проєктом цілою командою та головне – перенести ресурсомісткі обчислення з локального ПК до хмари. При цьому сервіс безкоштовний, але є одне обмеження – через дванадцять годин дані видаляються. В рамках цих годин надається дванадцять и сім десятих гігабайти оперативної пам’яті, близько чотирнадцяти гігабайт відеопам’яті та близько сімдесяти восьми гігабайт пам’яті для накопичення файлів. Для того, щоб, по завершенню відведеного часу, дані проєкту не були втрачені, можна завантажити проєкт на гугл диск та підключити його в свою чергу Google Colab. Таким чином результат обчислень буде зберігатися у відповідну папку проєкту, яка в свою чергу буде знаходитись на окремому хмарному диску.

Віртуальна машина Microsoft Azure була створена для паралельної роботи над іншими версіями моделі системи. В даному випадку, віртуальна машина надавала тридцять два гігабайти оперативної пам’яті та чотири ядра віртуального центрального процесора. Втім, в рамках обраної підписки не надається потужностей відеочіпу, тому за рахунок більшого об’єму ОЗУ, потужність була порівнянною з Google Colab.

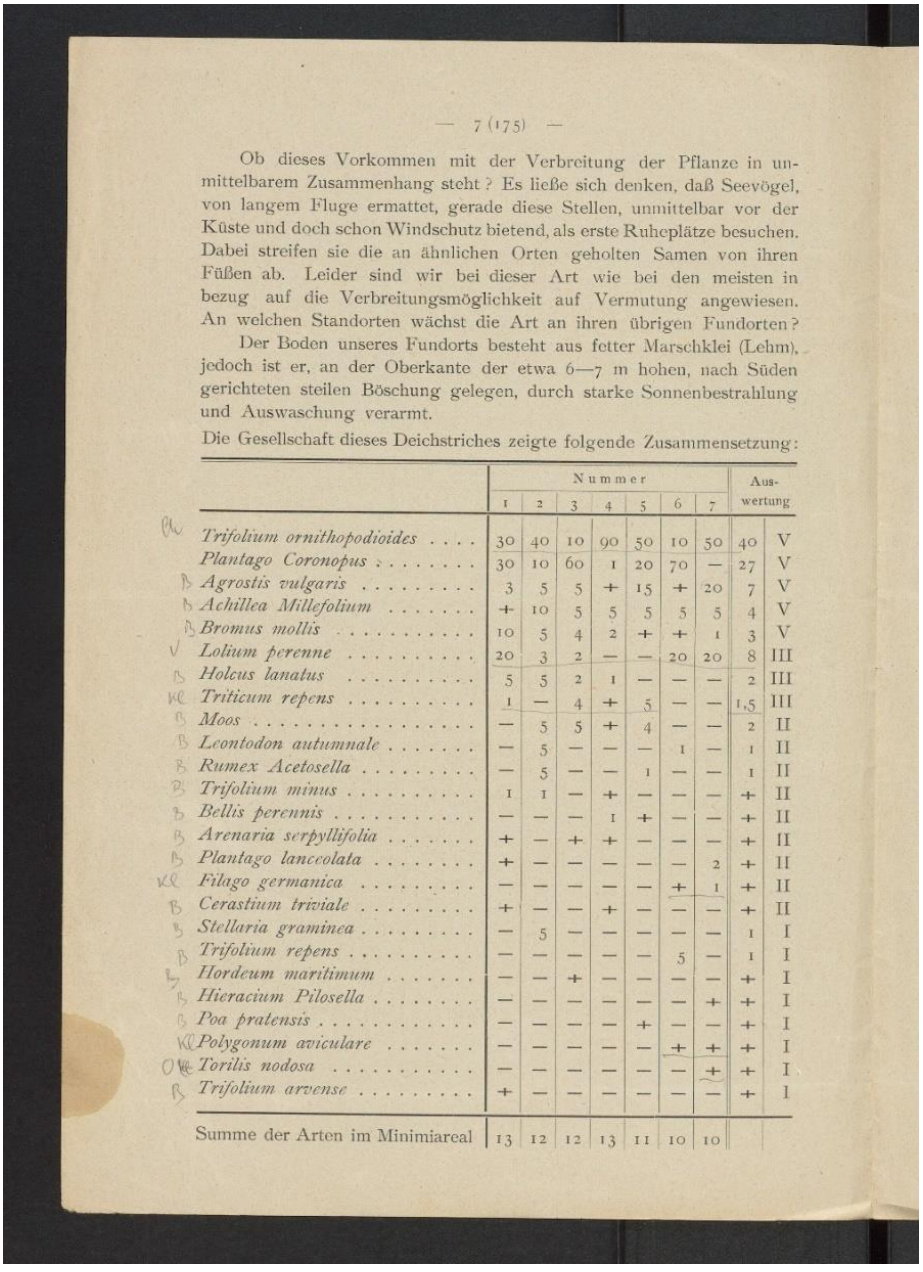
3.2 Створення датасету

Дані для датасету.

Результат роботи будь-якої нейронної мережі напряму залежить від якості набору даних, на яких вона була тренована.

В наборі даних важливо все – від типу даних до їх релевантності відносно поставленої мети. Тому важливо перед кодуванням будь-якої моделі створити відповідний датасет. В рамках даного проєкту має бути реалізована модель, яка могла б розпізнавати рукописні помітки та написи на зображеннях/сканах сторінок книжок з друкованим текстом. Тому і тренувальні дані повинні містити відповідний контент.

За основу тренувального набору були взяті скановані документи з кафедри геоботаніки університету Лейбніца. В основному вони стосуються рослин Нідерзаксену (район на північному заході Германії) [41], які збиралися приблизно на протязі останніх двохсот років. Даний набір сканів містить близько декількох тисяч зображень, не кожне з яких, зрозуміло, містить рукописні помітки. Тому з набору було відібрано сто п’ятдесят зображень, що задовольняють вимогам до датасету. Приклад такого зображення на рисунку 3.1.



— 7 (175) —

Ob dieses Vorkommen mit der Verbreitung der Pflanze in unmittelbarem Zusammenhang steht? Es ließe sich denken, daß Seevögel, von langem Fluge ermattet, gerade diese Stellen, unmittelbar vor der Küste und doch schon Windschutz bietend, als erste Ruheplätze besuchen. Dabei streifen sie die an ähnlichen Orten geholten Samen von ihren Füßen ab. Leider sind wir bei dieser Art wie bei den meisten in bezug auf die Verbreitungsmöglichkeit auf Vermutung angewiesen. An welchen Standorten wächst die Art an ihren übrigen Fundorten?

Der Boden unseres Fundorts besteht aus fetter Marschklei (Lehm), jedoch ist er, an der Oberkante der etwa 6—7 m hohen, nach Süden gerichteten steilen Böschung gelegen, durch starke Sonnenbestrahlung und Auswaschung verarmt.

Die Gesellschaft dieses Deichstriches zeigte folgende Zusammensetzung:

	Nummer							Auswertung	
	1	2	3	4	5	6	7		
Trifolium ornithopodioides	30	40	10	90	50	10	50	40	V
Plantago Coronopus	30	10	60	1	20	70	—	27	V
Agrostis vulgaris	3	5	5	+	15	+	20	7	V
Achillea Millefolium	+	10	5	5	5	5	5	4	V
Bromus mollis	10	5	4	2	+	+	1	3	V
Lolium perenne	20	3	2	—	—	20	20	8	III
Holcus lanatus	5	5	2	1	—	—	—	2	III
Triticum repens	1	—	4	+	5	—	—	1,5	III
Moos	—	5	5	+	4	—	—	2	II
Leontodon autumnale	—	5	—	—	—	1	—	1	II
Rumex Acetosella	—	5	—	—	1	—	—	1	II
Trifolium minus	1	1	—	+	—	—	—	+	II
Bellis perennis	—	—	—	1	+	—	—	+	II
Arenaria serpyllifolia	+	—	+	+	—	—	—	+	II
Plantago lanceolata	+	—	—	—	—	—	2	+	II
Filago germanica	—	—	—	—	—	+	1	+	II
Cerastium triviale	+	—	—	+	—	—	—	+	II
Stellaria graminea	—	5	—	—	—	—	—	1	I
Trifolium repens	—	—	—	—	—	5	—	1	I
Hordeum maritimum	—	—	+	—	—	—	—	+	I
Hieracium Pilosella	—	—	—	—	—	—	+	+	I
Poa pratensis	—	—	—	—	+	—	—	+	I
Polygonum aviculare	—	—	—	—	—	+	+	+	I
Torilis nodosa	—	—	—	—	—	—	+	+	I
Trifolium arvense	+	—	—	—	—	—	—	+	I
Summe der Arten im Minimiareal	13	12	12	13	11	10	10		

Рисунок 3.1 – Приклад одного з зображень для датасету

Як видно, на рисунку 3.1 є зображення сторінки з друкованим текстом, таблицею, а також рукописними написами у вигляді окремих літер зліва від таблиці та деякими підкресленнями на самій таблиці.

Маркування даних.

Для того, щоб нейронна мережа вміла розпізнавати об’єкти й патерни на зображенні, потрібно їй для початку «показати», що саме там знаходиться. Для цього і призначена розмітка.

Розмітка зображень (Image Annotation, анотування зображень, маркування зображень) є невід’ємною частиною розробки систем штучного інтелекту, і це одне з основних завдань у технології комп’ютерного зору. Анотовані зображення потрібні як вхідні дані для навчання нейронних мереж.

Ручна розмітка об’єктів на зображеннях – трудомістке і досить витратне завдання, особливо якщо потрібно розмітити великі набори даних. Також гостро стає питання вибору інструменту для маркування зображень, адже він має бути зручним, підтримувати різні формати та розмір зображень, швидко працювати з фото з великою роздільною здатністю та бажано бути безкоштовним. Під дані вимоги був обраний онлайн-інструмент розмітки зображень «supervisely» [42]. Інтерфейс представлений на рисунку 3.2.

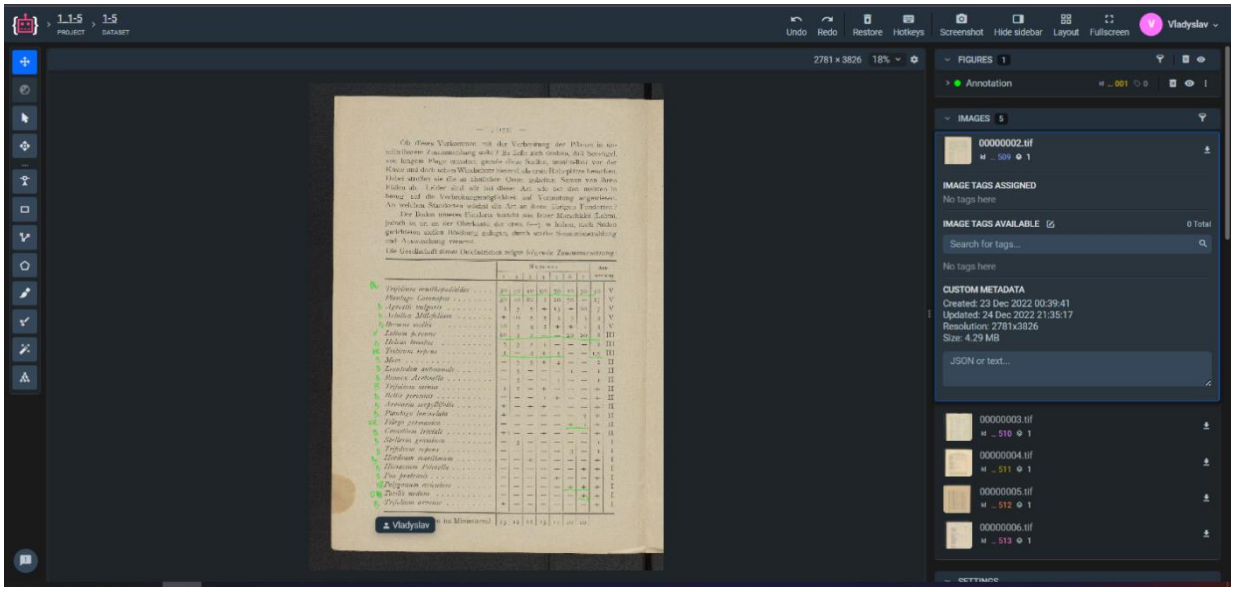
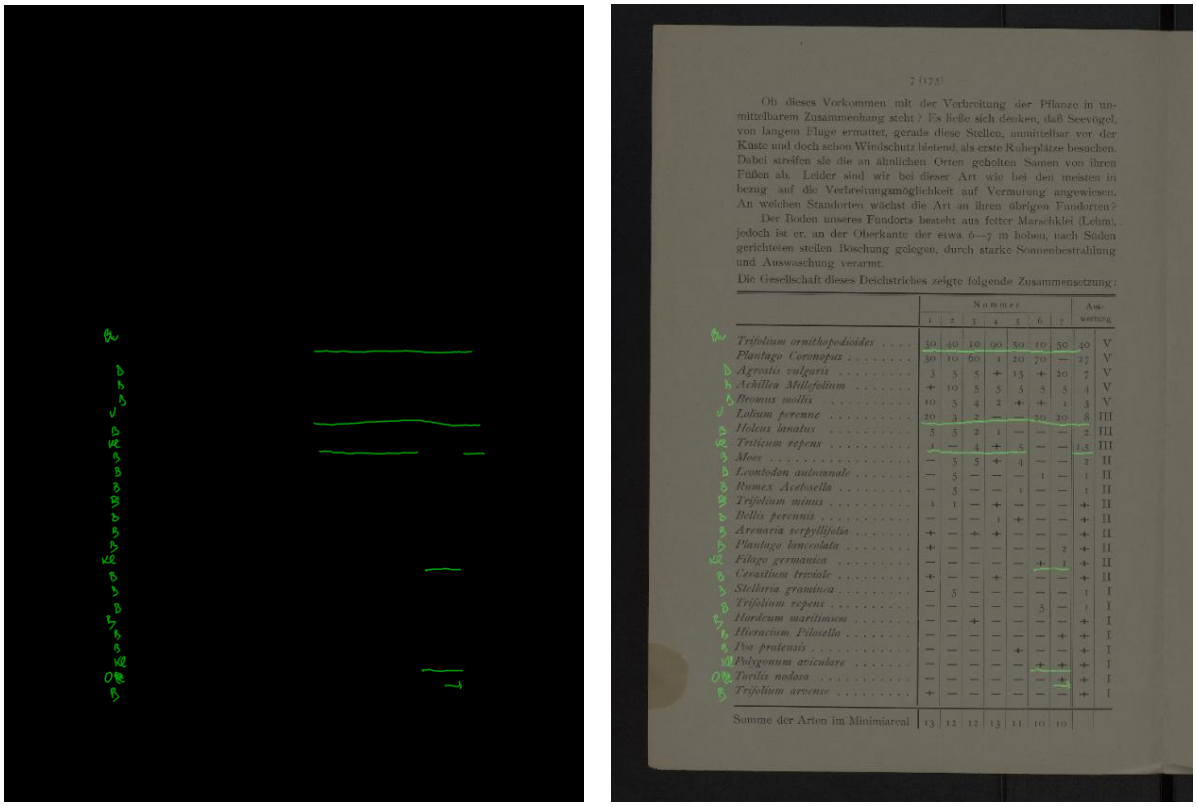


Рисунок 3.2 – Інтерфейс інструменту «supervisely» для розмітки зображень

Після завантаження необхідного набору фото на ресурс, процес роботи виглядав наступним чином: створюється клас поміток («Annotation»), обираються його параметри, такі як, колір виділення та товщина виділення. Далі на кожному зображенні, обравши на панелі інтерфейсу клас поміток, проводимо пензлем по самих рукописних написах. Таким чином створюється клас об’єктів «Annotation» для кожного фото. Хоч зображень не надто велика кількість, як для наборів для навчання мереж, їх роздільна здатність була досить високою (близько 2k), а також вони містили багато дрібних та крупних деталей. Тому даний етап роботи був найтривалішим в проєкті та зайняв близько чотирьох тижнів. Експорт вихідних даних, тобто масок поміток, здійснювався у форматі PNG. Приклад наведений на рисунку 3.3.



а) б)

Риснуок 3.3 – Вихідні дані:

а) маска рукописних написів; б) накладання маски на оригінальне зображення

На виході отримали 150 оригінальних зображень та 150 масок рукописних написів на них.

3.3 Технологія реалізації системи

Структура проєкту.

Розробка практичної та здатної до масштабування структури проєкту є одним з найважливіших про розробці будь-якого програмного продукту. Тому даному етапу слід приділити особливу увагу. В даній роботі використовується наступна структура проєкту (рис. 3.4).

```
project/
├── data/
│   ├── 512_512/
│   │   ├── test_images/
│   │   ├── masks/
│   │   └── images/
│   └── 1024_1024/
│       ├── test_images/
│       ├── masks/
│       └── images/
├── predictions/
├── models_weights/
├── utils/
│   ├── models/
│   │   ├── u-net.py
│   │   ├── resnet50.py
│   │   └── fcn-8s.py
│   └── generators/
│       ├── data_generator_u-net.py
│       ├── data_generator_resnet50.py
│       └── data_generator_fcn-8s.py
├── predict.py
└── main.py
```

Рисунок 3.4 – Структура проєкту

В структурі проєкту є наступні елементи:

– «project»: головна директорія проєкту;

– «data»: директорія де містяться всі набори даних проєкту;

– «512_512», «1024_1024»: директорії для розмежування даних за роздільною здатністю;

– «images»: папка де містяться фото/скани сторінок;

– «masks»: створені маски для навчання мережі;

– «test_images»: фото/скани для тестування мережі;

– «predictions»: папка, куди будуть зберігатися передбачення моделі;

– «models_weights»: папка, для зберігання вагів моделі;

– «utils/models»: директорія, де зберігаються файли різних мереж;

– «utils/generators»: директорія, де зберігаються файли обробки даних для кожної мережі;

– «main.py»: головний файл програми;

– «predict.py»: файл для тестування того, як модель може розпізнавати помітки на не знайомих даних.

Написання коду.

Перше за все створимо реалізацію завантаження та обробки даних проєкту.

Лістинг 3.1 Файл «data_generator_u-net.py» – конвертація зображень та масок до одноканального вигляду та їх нормалізація:

```
# Convert image to grayscale
img = Image.open(os.path.join(self.img_folder, filename)).convert('L')
# Convert mask to grayscale
mask = Image.open(os.path.join(self.mask_folder, filename)).convert('L')
img = np.array(img) / 255.  # Normalize pixel values to [0, 1]
mask = np.array(mask) / 255.  # Normalize pixel values to [0, 1]
```

Вхідні дані для навчання мають бути обробленими та підготованими для передачі моделі. Тому зображення та маски були приведені до однакової роздільної здатності, а в самій програмі переведені до одного каналу та нормалізовані. В якості величин роздільної здатності були обрані значення 512×512 та 1024×1024 пікселі. Спочатку буде проведене навчання на даних меншої роздільної здатності, тому що такі дані будуть оброблятися значно швидше, але, якщо результат навчання буде не задовільним, буде проведена спроба навчання на другому варіанті даних.

Навчання моделі відбувається за допомогою вбудованої, стандартної функції keras – *model.fit()*. В якості параметрів зазвичай передають *batch_size* – розмрі батчу, тобто кількість пар зображення/маска, яка буде братися для одного кроку під час епохи. Якщо поставити велике значення (наприклад 32 або 64) то навчання пройде швидко, але такий підхід потребує значних обчислювальних ресурсів та пам’яті, тому в нашому проєкті буде проводитись підбір оптимального значення експереметальним шляхом; *steps_per_epoch* – відповідає за кількість кроків навчання, які будуть виконуватися за одну епоху. Рекомендуєтсья, що добуток *batch_size* та *steps_per_epoch* давав загальну кількість файлів для навчання, оскільки після кожної епохи, генератор даних буде знову брати дані з директорій, і якщо значення добутку буде меншим за загальну кількість файлів, то деякі зображення та маски можуть взагалі не потрапити до процесу навчання. Втім, якщо зображень багато та вони великої роздільної здатності може статися переповнення пам’яті та примусове завершення програми. Тому, щоб такого не траплялося, можна використати підхід, коли генеретор даних після кожної епохи обирє випадкові пари зображення/маски. Тоді, навіть, якщо за одну епоху не будуть оброблені все дані, вони будь використані при наступних епохах (за умови, що кількість епох буде достатньо великою); *epoch* – власне, кількість епох навчання, після кожної епохи модель має зберігати свої ваги, для того, щоб в разі вильоту програми дані навчання були збережені.

Під час навчання моделі важилвим аспектом є оцінка її успішності в реальному часі. Є декілька метрик продуктивності моделі:

– accuracy (точність): частка правильних передбачень моделі відносно загальної кількості передбачень. Вона обчислюється як співвідношення кількості правильних передбачень до загальної кількості передбачень. Точність підходить для задач класифікації, коли класи збалансовані, тобто мають приблизно однакову кількість екземплярів кожного класу. Проте, якщо класи не збалансовані, точність може бути вводити в оману, оскільки модель може просто передбачати найбільш поширений клас і все одно досягати високої точності;

– precision (точність за класами) вимірює частку правильних передбачень моделі для певного класу відносно загальної кількості передбачень, які були віднесені до цього класу. Вона обчислюється як співвідношення кількості правильних передбачень класу до загальної кількості передбачень класу. Точність за класами корисна, коли вартість помилкового позитивного результату (неправильне визначення об’єкта як належного до класу) є високою. Наприклад, у завданні діагностики хвороби точність за класом «позитивний» може бути важливою для уникнення помилкових діагнозів;

– recall (повнота) вимірює частку правильних передбачень моделі для певного класу відносно загальної кількості екземплярів цього класу в даних. Вона обчислюється як співвідношення кількості правильних передбачень класу до загальної кількості екземплярів класу. Повнота корисна, коли вартість помилкового негативного результату (неправильне визначення об’єкта як не належного до класу) є високою. Наприклад, у завданні виявлення шахрайства повнота може бути важливою для мінімізації кількості пропущених шахрайських операцій;

– Intersection over Union (IoU), також відома як Jaccard Index, – це метрика, яка використовується для кількісної оцінки перекриття двох областей. Вона часто використовується в завданнях комп’ютерного зору,

таких як сегментація зображень і виявлення об’єктів, для оцінки того, наскільки добре передбачені області збігаються з істинними областями. IoU обчислюється як відношення перетину двох областей до їх об’єднання. У контексті сегментації зображень, «перетин» – це кількість пікселів, які одночасно є частиною істинної та передбаченої областей, а «об’єднання» – це загальна кількість пікселів, які є частиною або істинної, або передбаченої області (або обох). Для випадку нашого проєкту, важливими метрикам будуть саме precision, recall та IoU.

Створення моделі.

Наступним етапом йде реалізація архітектури моделі U-Net з використанням технології «tesnsorflow/keras». U-Net складається з двох частин: стискаючого шляху (або кодувальника) і розширювального шляху (або декодера).

Стискаючий шлях (кодування): Цей шлях складається з блоків, що повторюються, кожен з яких містить два згорткові шари, що йдуть за шаром максимального пулінгу. Згорткові шари використовуються для вивчення ознак на зображенні, а шари максимального пулінгу зменшують просторові розмірності зображення, що допомагає моделі стати більш стійкою до змін масштабу і положення об’єктів на зображенні.

Розширювальний шлях (декодер): Цей шлях складається з блоків, які підвищують просторові розмірності зображення і відновлюють його вихідний розмір. Кожен блок містить операцію апсемплінгу, наступну за згортковими шарами. Важливо зазначити, що в U-Net використовується конкатенація ознак з кодувальника і декодера на кожному рівні, що допомагає моделі краще локалізувати об’єкти на зображенні.

Модель в даному проєкті буде містити наступні шари:

Вхідний шар: цей шар приймає зображення заданого розміру.

Стискаючий шлях (кодування): цей шлях складається з 4 блоків, кожен з яких містить два згорткові шари і один шар максимального пулінгу. Згорткові шари використовуються для вивчення ознак на зображенні, а шари

максимального пулінгу зменшують просторові розмірності зображення, що допомагає моделі стати більш стійкою до змін масштабу і положення об’єктів на зображенні. Кількість фільтрів у згорткових шарах подвоюється з кожним блоком, починаючи з 64 і закінчуючи 512.

Середній блок: цей блок містить два згорткових шари і шар Dropout для запобігання перенавчання. Згорткові шари мають 1024 фільтри.

Розширювальний шлях (декодер): цей шлях складається з 4 блоків, кожен з яких містить операцію апсемплінгу, конкатенацію ознак з кодувальника та два згорткові шари. Кількість фільтрів у згорткових шарах зменшується вдвічі з кожним блоком, починаючи з 512 і закінчуючи 64.

Вихідний блок: цей блок містить два згорткових шари і один згортковий шар з одним фільтром і функцією активації сигмоїд, який генерує остаточну сегментаційну маску.

Загалом, модель містить 23 згорткових шари, 4 шари максимального пулінгу, 4 шари апсемплінгу, 4 шари конкатенації і 2 шари Dropout.

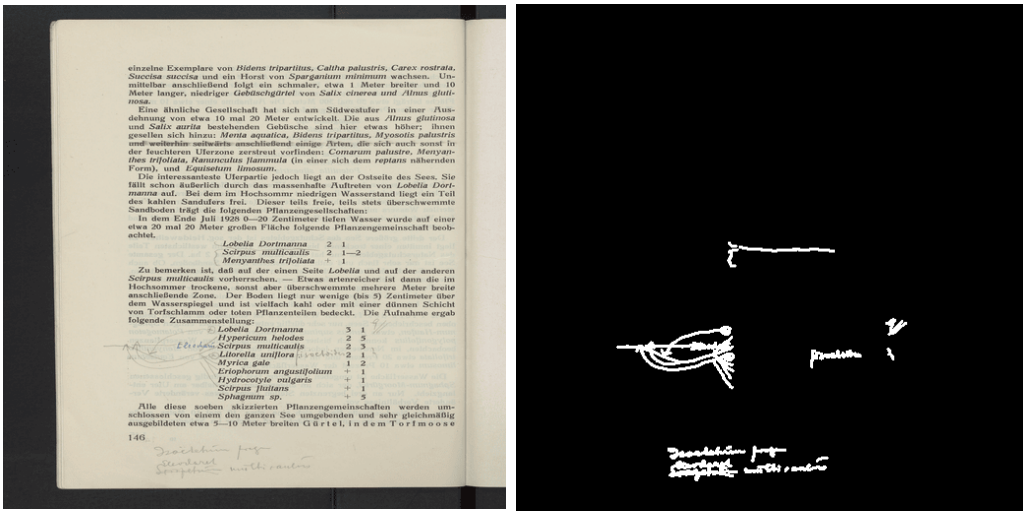
3.4 Проведення експериментів

При першій спробі навчання, виставимо наступні параметри навчання: *batch_size=16; steps_per_epoch=10; epochs=10*. Результат на рисунку 3.5.

```
Epoch 1/10
10/10 [==============================] - 25s 754ms/step - loss: 809.3420 - iou: 0.0103 - precision: 0.0172 - recall: 0.4140
Epoch 2/10
10/10 [==============================] - 8s 704ms/step - loss: 0.9737 - iou: 0.0081 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 3/10
10/10 [==============================] - 8s 717ms/step - loss: 0.9678 - iou: 0.0080 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 4/10
10/10 [==============================] - 7s 650ms/step - loss: 0.9790 - iou: 0.0083 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 5/10
10/10 [==============================] - 7s 664ms/step - loss: 0.9810 - iou: 0.0084 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 6/10
10/10 [==============================] - 7s 742ms/step - loss: 1.0402 - iou: 0.0101 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 7/10
10/10 [==============================] - 6s 585ms/step - loss: 0.9368 - iou: 0.0073 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 8/10
10/10 [==============================] - 7s 618ms/step - loss: 1.0051 - iou: 0.0092 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 9/10
10/10 [==============================] - 7s 588ms/step - loss: 0.9556 - iou: 0.0080 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 10/10
10/10 [==============================] - 7s 706ms/step - loss: 1.1438 - iou: 0.0130 - precision: 0.0000e+00 - recall: 0.0000e+00
```

Рисунок 3.5 – Результат першої спроби навчання

Бачимо що на першій епосі показники точностей були малими, але далі *IoU* фактично не змінилося, а *precision* та *recall* взагалі впали до нуля та не збільшилися. Втім, навіть з таким результатом була проведена спроба застосувати модель на тестових даних (рис. 3.6).



а) б)

в)

Рисунок 3.6 – Невдалий результат передбачення моделі на тестових даних: а) тестове зображення; б) очікуваний результат; в) фактичний результат

Як і можна було очікувати модель спрацювала дуже не вдало. Причин такого результату може бути декілька:

– недостатнє навчання: можливо моделі потрібно більше часу для навчання. Хоча якщо навіть після десяти епох метрики не покращились, то навряд чи далі це зміниться;

– складність моделі: можливо модель надто проста для такого складного завдання сегментації;

– помилки в тренувальних даних. Втім вірогідність цього сценарію мінімальна, адже дані неодноразово перевірялись;

– низька роздільна здатність зображень та масок;

Спробуємо натренувати модель з тими ж параметрами, але на даних більшої роздільної здатності в чотири рази (1024×1024 пікселі) (рис. 3.7).

```
Epoch 1/10
10/10 [==============================] - 24s 1s/step - loss: 620.9484 - iou: 0.0026 - precision: 0.0000e+00 - recall: 0.0000e+0
Epoch 2/10
10/10 [==============================] - 13s 1s/step - loss: 0.6495 - iou: 0.0042 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 3/10
10/10 [==============================] - 14s 1s/step - loss: 0.5775 - iou: 0.0025 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 4/10
10/10 [==============================] - 14s 1s/step - loss: 0.6943 - iou: 0.0031 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 5/10
10/10 [==============================] - 15s 1s/step - loss: 0.6830 - iou: 0.0047 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 6/10
10/10 [==============================] - 13s 1s/step - loss: 0.8415 - iou: 0.0032 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 7/10
10/10 [==============================] - 13s 1s/step - loss: 0.8190 - iou: 0.0039 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 8/10
10/10 [==============================] - 14s 1s/step - loss: 0.8508 - iou: 0.0048 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 9/10
10/10 [==============================] - 13s 1s/step - loss: 0.8778 - iou: 0.0056 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 10/10
10/10 [==============================] - 13s 1s/step - loss: 2.3385 - iou: 0.0050 - precision: 0.0000e+00 - recall: 0.0000e+00
```

Рисунок 3.7 – Метрики навчання на даних з більшою роздільною здатністю

Як бачимо показники не сильно відрізняються від першого результату, тому проводити передбачення на тестовому зображенні не має сенсу. Спробуємо провести навчання збільшивши параметр *steps_per_epoch* до 50 (рис. 3.8).

```
Epoch 1/10
50/50 [==============================] - 72s 1s/step - loss: 36112.2109 - iou: 0.0043 - precision: 0.0045 - recall: 0.0209
Epoch 2/10
50/50 [==============================] - 63s 1s/step - loss: 0.8284 - iou: 0.0046 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 3/10
50/50 [==============================] - 63s 1s/step - loss: 0.7564 - iou: 0.0031 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 4/10
50/50 [==============================] - 63s 1s/step - loss: 0.7920 - iou: 0.0045 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 5/10
50/50 [==============================] - 63s 1s/step - loss: 0.7556 - iou: 0.0040 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 6/10
50/50 [==============================] - 63s 1s/step - loss: 0.7639 - iou: 0.0045 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 7/10
50/50 [==============================] - 62s 1s/step - loss: 0.7433 - iou: 0.0043 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 8/10
50/50 [==============================] - 63s 1s/step - loss: 0.6980 - iou: 0.0036 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 9/10
50/50 [==============================] - 63s 1s/step - loss: 0.7414 - iou: 0.0048 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 10/10
50/50 [==============================] - 63s 1s/step - loss: 0.6935 - iou: 0.0040 - precision: 0.0000e+00 - recall: 0.0000e+00
```

Рисунок 3.8 – Метрики навчання на даних з більшою роздільною здатністю та більшим значенням параметру *steps_per_epoch*

Як бачимо показники не сильно відрізняються від першого результату, тому проводити передбачення на тестовому зображенні не має сенсу. Спробуємо провести навчання збільшивши параметр *steps_per_epoch* до 50 (рис. 3.8).

Наступним кроком, щоб покращити результат було вирішено ускладнити модель додаванням нових шарів. Загалом у новій моделі використовується 37 згорткових шарів, 4 шари MaxPooling, 2 шари Dropout, 4 шари UpSampling і 4 шари об’єднання. Метрики навчання нової моделі на рисунку 3.9.

```
Epoch 1/5
10/10 [==============================] - 28s 2s/step - loss: 227.3678 - iou: 0.0050 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 2/5
10/10 [==============================] - 17s 2s/step - loss: 0.8879 - iou: 0.0056 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 3/5
10/10 [==============================] - 17s 2s/step - loss: 0.8034 - iou: 0.0033 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 4/5
10/10 [==============================] - 17s 2s/step - loss: 64.4806 - iou: 0.0035 - precision: 0.0000e+00 - recall: 0.0000e+00
Epoch 5/5
10/10 [==============================] - 17s 2s/step - loss: 0.8196 - iou: 0.0040 - precision: 0.0000e+00 - recall: 0.0000e+00
```

Рисунок 3.9 – Метрики навчання ускладненої моделі

Дані вказують на те, що модель не навчається належним чином.

Для покращення результатів, було вирішено використати мережу ResNet50, яка є варіантом Residual Network (ResNet). Це тип згорткової нейронної мережі (Convolutional Neural Network, CNN), розроблений для розв’язання проблеми загасання градієнта, яка виникає під час навчання глибоких нейронних мереж. ResNet було презентовано 2015 року на конференції NeurIPS, і відтоді вона стала однією з найпопулярніших архітектур для задач комп’ютерного зору. ResNet50 містить 50 шарів і використовує концепцію «залишкових блоків» або «skip connections», які дають змогу пропускати один або кілька шарів у процесі навчання. Це допомагає запобігти загасанню градієнта і дає змогу мережі навчатися ефективніше. У даному проєкті використовується ResNet50 для завдання сегментації зображень. Починаємо з переднавченої моделі ResNet50, завантаженої з Keras, і потім донавчаємо її на наших даних. Це дає нам змогу

використовувати вже навчені на великій кількості даних ваги, що прискорює процес навчання і покращує продуктивність моделі. Заморожуємо всі шари в базовій моделі ResNet50, щоб вони не оновлювалися під час навчання. Потім додаємо кілька шарів Conv2D і Conv2DTranspose для відновлення вихідного розміру зображення і передбачення маски для кожного пікселя. Результат навчання даної моделі (*batch_size=16*; *steps_per_epoch=10*; *epochs=5*) на рисунку 3.10.

```
Epoch 1/5
10/10 [==============================] - 67s 6s/step - loss: 0.3251 - iou: 0.0027 - precision: 0.0027 - recall: 1.0000
Epoch 2/5
10/10 [==============================] - 57s 5s/step - loss: 0.1345 - iou: 0.0040 - precision: 0.0040 - recall: 1.0000
Epoch 3/5
10/10 [==============================] - 58s 6s/step - loss: 0.0550 - iou: 0.0035 - precision: 0.0035 - recall: 1.0000
Epoch 4/5
10/10 [==============================] - 57s 6s/step - loss: 0.0445 - iou: 0.0054 - precision: 0.0054 - recall: 1.0000
Epoch 5/5
10/10 [==============================] - 57s 6s/step - loss: 0.0340 - iou: 0.0040 - precision: 0.0040 - recall: 1.0000
```

Рисунок 3.10 – Метрики навчання ResNet50

Як можна побачити, показники метрик навчання вказують на те, що модель не навчається належним чином. Виконавши передбачення моделлю такої конфігурації на тестовому зображенні, в результаті було отримане зображення суцільно білого кольору, що свідчить про те, що модель сто відсотків пікселів віднесла до класу поміток. *Recall* дорівнює 1, що означає, що модель ідеально передбачає позитивний клас. Однак, враховуючи низькі значення *precision* та *IoU* – це, найімовірніше, означає, що модель передбачає позитивний клас для більшості або всіх пікселів, що пояснює, чому результат тестування – це повністю біле зображення.

3.5 Перспективи подальшої роботи

З результатів очевидно, що проблема скоріше за все криється в недостатній кількості тренувальних даних. Вирішення цієї проблеми є два: збільшити об’єм існуючого датасету; провести попереднє навчання моделі на

більш загальному, але значно більшому наборі даних, а потім провести донавчання на цільовому наборі. Перший варіант не є доцільним, оскільки датасет має бути збільшений на багато порядків, а ручне створення маркованих даних для навчання займе надмірно багато часу. Тому другий варіант вирішення проблеми є прийнятнішим. В такого підходу є назва – *Transfer Learning.* Такий метод використовувався і в роботі Andreas K¨olsch [12] та показав хороший результат. Andreas K¨olsch та ін. використовували в якості початкового об'ємного набору даних датасет ILSVRC [14], що містить близько 1,2 мільйони зображень з помітками класифікації об'єктів на зображенні. Втім, оскільки задача нашого проєкту в попіксельній сегментації, а набір ILSVRC містить помітки в вигляді прямокутників, що не завжди достатньо точно описують об'єкт, має доцільність використання набору даних, що призначений саме для тренування моделі для задачі сегментації. Таким набором є ImageNet-S [41]. Він базується на ILSVRC та також містить близько 1,2 мільйони зображень, але в якості навчального матеріалу містить помітки сегментації. Тренування на такому наборі даних потребує значних обчислювальних ресурсів та затрат часу. Втім, після донавчання моделі на цільовому наборі даних, очікується задовільний результат по розпізнаванню рукописних поміток на зображеннях. Таким чином, проведення попереднього навчання моделі FCN-8s на наборі ImageNet-S та подальше донавчання на цільовому наборі даних, що вже створений, є пріоритетними задачами для подальшого розвитку проєкту. При отриманні задовільних результатів в розпізнаванні, наступним кроком має стати розробка методу видалення рукописних поміток.

ВИСНОВКИ

У рамках кваліфікаційної роботи був створений та протестований власний датасет для тренування нейронних мереж в рамках задачі розпізнавання рукописних написів на зображенні. Також виконані поставлені задачі роботи, а саме:

– проведено аналіз існуючих рішень по розпізнаванню рукописних написів, а саме розглянуто застосування різних типів мереж, таких як згорткові нейронні мережі, графові нейронні мережі;

– розроблено та створено унікальний датасет зображень та масок сегментації (виділено в цільовий клас рукописні помітки);

– був створений програмний проєкт на мові Python з використанням таких бібліотек машинного навчання як TensorFlow та keras, в якому провелося тестування різних типів мереж;

– був зіставлений план по подальшому розвитку проєкту для покращення результатів.

Загалом, одним з головних прикладних здобутків даної роботи, є створення аутентичного датасету, який найкраще підходить для донавчання моделі нейронної мережі, яка вже була натренована на загальному, об’ємному наборі даних та потребує тонкого налаштування та навчання на цільових даних.

Результати роботи апробовано у вигляді тез доповідей під час IV Міжнародної науково-теоретичної конференції «Theoretical and practical scientific achievements: research and results of their implementation», 7 квітня 2023 р., Піза, Італія [42].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Digispecies - L3S Research Center. *L3S Research Center - Trustworthy AI & Digital Transformation*. URL: https://www.l3s.de/research-at-l3s/all-projects/digispecies/ (дата звернення: 04.05.2023).

2. The flow of improved BRISK algorithm. URL: https://www.researchgate.net/figure/the-flow-of-improved-BRISK-algorithm_fig1_328946366 (дата звернення 24.04.2023).

3. Future Market Insights Global and Consulting Pvt. Ltd. Global Computer Vision Market is likely to reach a worth of US$ 26.11 Billion, at a CAGR of 7.3% by the forecast period ending 2033 | Future Market Insights, Inc. GlobeNewswire News Room. URL: https://www.globenewswire.com/news-release/2023/04/27/2656617/0/en/Global-Computer-Vision-Market-is-likely-to-reach-a-worth-of-US-26-11-Billion-at-a-CAGR-of-7-3-by-the-forecast-period-ending-2033-Future-Market-Insights-Inc.html (дата звернення: 11.04.2023).

4. Indira, B., Qureshi, M. S., Shaik, M. S., Saqib, R. M., & Murthy, M. R. (2012). Devanagari Character Recognition: A Short Review. *International Journal of Computer Applications*, *59*(6).

5. Shah, M., & Jethava, G. B. (2013). A literature review on hand written character recognition.

6. Uses of Machine Learning Handwriting Recognition. *Analytics Insight*. URL: https://www.analyticsinsight.net/uses-of-machine-learning-handwriting-recognition/ (дата звернення: 12.04.2023).

7. Roche | About Roche. *Roche - Doing now what patients need next*. URL: https://www.roche.com/about/ (дата звернення: 11.05.2023).

8. Lyashenko, V., Kobylin, O., Ryazantsev, O., & Ryazantsev, I. (2019). Processing Technique for Biomedical Image Analysis.

9. Daradkeh, Y. I., Gorokhovatskyi, V., Tvoroshenko, I., & Al-Dhaifallah, M. (2022). Classification of images based on a system of hierarchical features. *Computers, Materials & Continua*, *72*(1), 1785-1797.

10. Cleanup.pictures - Remove objects, people, text and defects from any picture for free. *Cleanup.pictures - Remove objects, people, text and defects from any picture for free*. URL: https://cleanup.pictures (дата звернення: 15.04.2023).

11. Cutout.Pro - AI Photo Editing | Visual Content Generation Platform, best for image and video design. *Cutout.Pro - AI Photo Editing | Visual Content Generation Platform, best for image and video design*. URL: https://www.cutout.pro (дата звернення: 15.04.2023).

12. Kölsch, A., Mishra, A., Varshneya, S., Afzal, M. Z., & Liwicki, M. (2018, August). Recognizing challenging handwritten annotations with fully convolutional networks. In *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)* (pp. 25-31). IEEE.

13. Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3431-3440).

14. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, *115*, 211-252.

15. Chen, L. C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2017). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, *40*(4), 834-848.

16. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

17. Krähenbühl, P., & Koltun, V. (2011). Efficient inference in fully connected crfs with gaussian edge potentials. *Advances in neural information processing systems*, *24*.

18. Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017). Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2881-2890).

19. Kobylin, O. A., Gorokhovatskyi, V. O., Tvoroshenko, I. S., & Peredrii, O. O. (2020). The application of non-parametric statistics methods in image classifiers based on structural description components. *Telecommunications and Radio Engineering*, *79*(10).

20. Lyashenko, V., Lyubchenko, V., Mohammad, A., Alveera, K., & Kobylin, O. (2016). The methodology of image processing in the study of the properties of fiber as a reinforcing agent in polymer compositions.

21. Lyashenko, V., Mohammad, A., & Kobylin, O. (2015). Experiments with Fusion of Images with Use of Wavelet Transformation in Problems of the Text Information Analysis.

22. Kobylin, O., & Lyashenko, V. (2014). Comparison of standard image edge detection techniques and of method based on wavelet transform.

23. Lyashenko, V., Kobylin, O., & Ahmad, M. A. (2014). General methodology for implementation of image normalization procedure using its wavelet transform.

24. Lyubchenko, V., Matarneh, R., Kobylin, O., & Lyashenko, V. (2016). Digital image processing techniques for detection and diagnosis of fish diseases.

25. Bodyanskiy, Y., Vynokurova, O., Kobylin, I., & Kobylin, O. (2016). Adaptive fuzzy clustering of short time series with unevenly distributed observations in Data Stream Mining tasks. *Information Technology and Management Science*, *19*(1), 23-28.

26. Lyashenko, V., Matarneh, R., & Kobylin, O. (2016). Contrast modification as a tool to study the structure of blood components.

27. Gorokhovatskiy, V. A., Kobylin, O. A., & Kulikov, Y. A. (2015). Application of Granulation of Feature Descriptions in Structural Image Recognition. *Telecommunications and Radio Engineering*, *74*(6).

28. Gorokhovatskyi, V., Tvoroshenko, I., Kobylin, O., & Vlasenko, N. (2023). Search for Visual Objects by Request in the Form of a Cluster Representation for the Structural Image Description. *Advances in Electrical and Electronic Engineering*, *21*(1), 19-27.

29. Mishra M. Convolutional Neural Networks, Explained. *Medium*. URL: https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939 (дата звернення: 20.04.2023).

30. What Is a Convolutional Neural Network? | 3 things you need to know. *MathWorks - Makers of MATLAB and Simulink - MATLAB & Simulink*. URL: https://www.mathworks.com/discovery/convolutional-neural-network-matlab.html (дата звернення: 20.04.2023).

31. Ashley. An Overview on Convolutional Neural Networks. *Medium*. URL: https://medium.com/swlh/an-overview-on-convolutional-neural-networks-ea48e76fb186 (дата звернення: 25.04.2023).

32. Fanany, M. I. (2017, May). Handwriting recognition on form document using convolutional neural network and support vector machines (CNN-SVM). In *2017 5th international conference on information and communication technology (ICoIC7)* (pp. 1-6). IEEE.

33. Grother, P. J. (1995). Nist special database 19-hand-printed forms and characters database. *Technical Report, National Institute of Standards and Technology*.

34. Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*, 273-297.

35. A Gentle Introduction to Graph Neural Networks. *Distill*. URL: https://distill.pub/2021/gnn-intro/ (дата звернення: 01.05.2023).

36. Zhang, X. M., Liang, L., Liu, L., & Tang, M. J. (2021). Graph neural networks and their current applications in bioinformatics. *Frontiers in genetics*, *12*, 690049.

37. Ioannidis, V. N., Marques, A. G., & Giannakis, G. B. (2019, December). Graph neural networks for predicting protein functions. In *2019 IEEE 8th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)* (pp. 221-225). IEEE.

38. Zhang, M., & Chen, Y. (2018). Link prediction based on graph neural networks. *Advances in neural information processing systems*, *31*.