МІНІСТЕРСТВО ФІНАНСІВ УКРАЇНИ
ДЕРЖАВНИЙ ПОДАТКОВИЙ УНІВЕРСИТЕТ

ФАКУЛЬТЕТ ФІНАНСІВ ТА ЦИФРОВИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І СИСТЕМ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: МОДЕЛЬ РОЗПІЗНАВАННЯ СИМВОЛІВ ТА ВИЯВЛЕННЯ ОБ'ЄКТІВ

Здобувача вищої освіти другого (магістерського) рівня,
II курсу, групи КМІ-23-1, спеціальності 122 «Комп'ютерні науки»,
освітня програма «Інформаційні управляючі системи і технології в економіці»

(підпис) Шпаковича Володимира Івановича
(прізвище, ім'я, по батькові)

Науковий керівник (підпис) Погорєловська Ірина Дем'янівна, к.е.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Керівник освітньо-професійної програми (підпис) Горбовий Артур Юліанович, д.т.н, професор
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Допущено до захисту за рішенням кафедри протокол від «10» січня 2025 №11

Завідувач кафедри (підпис) Філоненко Михайло Миколайович, к.ф.-м.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Ірпінь – 2025

Реєстрація № 21 27.01.2025 _____
(дата) (підпис відповідальної особи)

Результати перевірки до захисту 27.01.2025 _____
(до захисту / на доопрацювання) (дата) (підпис)

Результат захисту 4 (добре) 04.02.2025 _____
(оцінка) (дата) (підпис)

Голова комісії: _____ В.М. Франчук
(підпис)

Члени комісії:

_____ А.Ю. Горбовий
(підпис)

_____ М.М. Філоненко
(підпис)

_____ І.Д. Погорєловська

БЛАНК ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Об'єкти контролю	Діапазон оцінки в балах	Фактична кількість балів	Примітки	Дата, П.І.Б.: керівника, голови комісії членів комісії,секретаря, (підписи)
ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ КОМІСІЄЮ ПО ЗАХИСТУ				
1. Зміст кваліфікаційної роботи (якість теоретичного матеріалу, його зв'язок з практичним, вміння представляти матеріал в логічній послідовності, наявність висновків та пропозицій)	від 0 до 30	21		04.02.2025 Франчук В.М. Горбовий А.Ю. Філоненко М.М. Погорєловська І.Д. Суховецька Л.П.
2. Дотримання вимог щодо оформлення	від 0 до 10	8		
3. Вчасне виконання роботи на всіх етапах у встановлені кафедрою терміни	від 0 до 5	3		
4. Рецензент	від 0 до 5	4		
РАЗОМ БАЛІВ	від 0 до 50	36		

Оцінювання кваліфікаційної роботи науковим керівником - 35 балів.

ПЕРЕВЕДЕННЯ ОЦІНКИ ЗА 100-БАЛЬНОЮ ШКАЛОЮ ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалоюЄКТС	Оцінка за національною шкалою	Дата	П.І.Б.: голови, членів комісії, секретаря,(підпис)
71	C	4 (добре)	04.02.2025	Франчук В.М. Горбовий А.Ю. Філоненко М.М. Погорєловська І.Д. Суховецька Л.П.

ПРАВИЛА ПЕРЕВЕДЕННЯ ОЦІНОК ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалою ЄКТС	Оцінка за національною шкалою
90-100	A	5 (відмінно)
80-89	B	4 (добре)
70-79	C	
60-69	D	3 (задовільно)
50-59	E	
35-49	FX	2 (незадовільно)

Факультет фінансів та цифрових технологій
Кафедра комп'ютерних та інформаційних технологій і систем
Галузь знань 12 «Інформаційні технології»
Спеціальність 122 «Комп'ютерні науки»
Освітня програма «Інформаційні управляючі системи і технології в економіці»

Затверджую
Завідувач кафедри _____ Філоненко М.М.
«23» грудня 2023

Завдання на кваліфікаційну роботу
здобувачу вищої освіти
Шпаковичу Володимиру Івановичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи Модель розпізнавання символів та виявлення об'єктів
Затверджена наказом ректора від «20» грудня 2023 № 1701
2. Строк здачі закінченої роботи «07» січня 2025
3. Мета роботи дослідження моделей комп'ютерного зору, що використовуються для розпізнавання символів і виявлення об'єктів.
Об'єкт дослідження технології автоматичного аналізу та обробки зображень.
Предмет дослідження методи і моделі розпізнавання символів та виявлення об'єктів.
Орієнтовна база дослідження комплекс сучасних технологій, інструментів, бібліотек та наборів даних, що дозволяють реалізувати, навчати та оцінювати нейронні мережі для обробки зображень.
4. Вихідні дані до роботи моделі комп'ютерного зору для ефективного розпізнавання символів.
5. Перелік графічного матеріалу рисунки та графіки, формули, що пояснюють суть застосованих методів та засобів. Структурні схеми та таблиці, малюнки результатів роботи моделі.
Зміст кваліфікаційної роботи (перелік питань за кожним розділом) РОЗДІЛ 1. Введення в задачу розпізнавання символів і зображення об'єктів 1.1. Аналіз сучасних методів і моделей розпізнавання символів і виявлення об'єктів. 1.2. Типи нейро мереж для вирішення завдань розпізнавання символів. РОЗДІЛ 2. Розробка моделі розпізнавання символів. 2.1. Вибір і підготовка даних. 2.2.Вибір типу нейронної мережі для завдання розпізнавання символів. 2.3. Навчання і тестування моделей РОЗДІЛ 3. Аналіз результатів застосування CNN. 3.1. Аналіз результатів моделі LeNet розпізнавання символів та виявлення об'єктів. 3.2. Аналіз застосування у різних сценарія

6. Календарний план виконання роботи

№ п/п	Назва етапів кваліфікаційноїроботи	Строк виконання етапів роботи	
		За планом	Фактично
1	2	3	4
1	Вибір теми, підбір літературних джерел, формування структури кваліфікаційної роботи	До 15.12.2023	08.12.2023
2	**І розділ** (розкрити сутність і стан дослідження наукової проблематики роботи, після чого навести її загальну характеристику; огляд літератури за темою і вибір конкретних напрямів досліджень)	27.05.2024	27.05.2024
3	**ІІ розділ** (аналіз експериментальної частини про методики досліджень)	02.09.2024	01.09.2024
4	**ІІІ розділ** (проведені теоретичні і (або) експериментальні дослідження та проектні рішення)	18.11.2024	15.11.2024
5	**Висновки**	10.12.2024	10.12.2024
6	Доопрацювання роботи з урахуванням зауважень наукового керівника	23.12.2024	20.12.2024
7	Отримання відгуку керівника	28.01.2025	28.01.2025
8	Отримання зовнішньої рецензії	31.01.2025	31.01.2025

7. Дата видачі завдання «23» грудня 2023

8. Керівник кваліфікаційної роботи _____ (підпис) Погорєловсь (прізвище

9. Керівник освітньо-професійної програми _____ (підпис) Горбовий А (прізвище

10. Завдання до виконання отримав здобувач вищої освіти _____ (підпис) - Шпакович В (прізвище

Анотація

Модель розпізнавання символів і виявлення об’єктів

Мета роботи: дослідження моделей комп'ютерного зору, що використовуються для розпізнавання символів і виявлення об'єктів.

Завдання роботи:

- проаналізувати методи і моделі розпізнавання символів і виявлення об'єктів;
- розглянути типи нейромереж для вирішення завдань розпізнавання символів;
- здійснити вибір моделі для виконання завдань розпізнавання символів;
- зібрати дані та виконати навчання моделі;
- проаналізувати отримані результати та зробити висновки щодо використання моделей розпізнавання символів та виявлення об'єктів.

Об’єкт дослідження: технології автоматичного аналізу та обробки зображень.

Предмет дослідження: методи і моделі розпізнавання символів та виявлення об’єктів.

Інформаційна база дослідження: Методи, алгоритми та програмні засоби розпізнавання символів і виявлення об'єктів.

За результатами дослідження було сформульовано висновки про те, що :модель LeNet, яка була навчена на наборі даних MNIST, є ефективною у простих задачах розпізнавання символів (її точність становить 90,24%), але поступається таким архітектурам, як Faster R-CNN і Vision Transformers.

Одержані результати можуть бути використані при вирішені проблем автоматичного розпізнавання символів, зокрема в OCR-системах, та у сфері відеоспостереження, автономних транспортних засобів.

Кваліфікаційна робота містить: 12 рисунків, 1 таблиця, висновки, перелік використаної літератури, що містить 42 джерела, 1 додаток.

Ключові слова: нейромережі, символи та об’єкти, комп’ютерний зір, машинне навчання, моделі комп'ютерного зору, обробка зображень.

Рік виконання магістерської роботи 2024-2025

Рік захисту роботи 2025

ANNOTATION

Character recognition and object detection model

Purpose of the work: study of computer vision models used for character recognition and object detection.
Tasks of the work:
- analyze methods and models of character recognition and object detection;
- consider types of neural networks for solving character recognition tasks;
- select a model for performing character recognition tasks;
- collect data and train the model;
- analyze the results obtained and draw conclusions regarding the use of character recognition and object detection models.
Object of the research: technologies of automatic image analysis and processing.
Subject of the research: methods and models of character recognition and object detection.
Research information base: Methods, algorithms and software tools for character recognition and object detection.
According to the results of the study, the conclusions were formulated that: the LeNet model, which was trained on the MNIST dataset, is effective in simple character recognition tasks (its accuracy is 90.24%), but is inferior to such architectures as Faster R-CNN and Vision Transformers.
The results obtained can be used to solve problems of automatic character recognition, in particular in OCR systems, and in the field of video surveillance, autonomous vehicles.
The qualification work contains: The qualification work contains: 12 drawings, 1 table, conclusions, a list of used literature containing 42 sources, 1 appendix.
Keywords: neural networks, symbols and objects, computer vision, machine learning, computer vision models, image processing.
Year of completion of the master's thesis 2024-2025
Year of defense of the work 2025

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ І ТЕРМІНІВ 3
ВСТУП 4
РОЗДІЛ 1. Введення в задачу розпізнавання символів і зображення об'єктів 6
1.1. Аналіз сучасних методів і моделей розпізнавання символів і виявлення об'єктів. 6
1.2. Типи нейро мереж для вирішення завдань розпізнавання символів 15
РОЗДІЛ 2. Розробка моделі розпізнавання символів 19
2.1. Структура алгоритмічно-програмних засобів розпізнавання символів 19
2.2.Вибір типу нейронної мережі для завдання розпізнавання символів 24
2.3. Навчання і тестування моделей 37
РОЗДІЛ 3. Аналіз результатів застосування CNN 40
3.1. Аналіз результатів моделі LeNet розпізнавання символів та виявлення об'єктів 40
3.2. Аналіз застосування у різних сценаріях 42
ВИСНОВКИ 45
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 47

ПЕРЕЛІК СКОРОЧЕНЬ І ТЕРМІНІВ

CNN (англ. Convolutional neural network) – Згорткова нейронна мережа.

OCR (англ. Optical Character Recognition) – Оптичне розпізнавання тексту.

RNN (англ. Recurrent Neural Networks) – Рекурентні нейронні мережі.

LeNet – Модель згорткової нейронної мережі, запропонована Яном ЛеКуном зі співавторами у 1989 році.

MNIST (англ. Mixed National Institute of Standards and Technology) - об'ємна база даних зразків рукописного написання цифр.

R-CNN – Конволюційні нейронні мережі на основі регіонів.

YOLO (англ. You Only Look Once) - Одноходова уточнювальна нейронна мережа для виявляння об'єктів.

ViT(англ. *Vision Transformers*) - це трансформер, призначений для задач зорової обробки, таких як розпізнавання зображень.

EMNIST (англ. Extended MNIST) - це розширення MNIST, яке включає не тільки цифри, але й літери латинського алфавіту.

SynthText - це штучно згенерований набір даних з понад 800 000 зображень.

ВСТУП

Актуальність теми. Комп'ютерний зір є одним з найбільш прогресивних і перспективних напрямів штучного інтелекту, що дозволяє здійснювати автоматичний аналіз і обробку зображень і відео з використанням алгоритмів машинного навчання. Зокрема, розпізнавання символів і виявлення об'єктів є фундаментальними завданнями і широко застосовуються в різних галузях, таких як медицина, промисловість, транспорт і безпека.

Розвиток алгоритмів машинного навчання, зокрема глибокого навчання, значно підвищив точність та ефективність рішень для комп'ютерного зору. Нейромережеві архітектури, такі як згорткові нейронні мережі (CNN), революційні нейронні мережі (RNN) та їх вдосконалені варіанти, є основою для створення потужних моделей розпізнавання та виявлення. Іншим важливим аспектом є використання попередньо навчених моделей і методів перехідного навчання, які дозволяють адаптувати існуючі моделі до нових завдань з мінімальними обчислювальними ресурсами.

Мета виконання кваліфікаційної роботи - дослідження моделей комп'ютерного зору, що використовуються для розпізнавання символів і виявлення об'єктів. Для досягнення мети потрібно виконати ряд завдань, зокрема:

- проаналізувати методи і моделі розпізнавання символів і виявлення об'єктів;
- розглянути типи нейромереж для вирішення завдань розпізнавання символів;
- здійснити вибір моделі для виконання завдань розпізнавання символів;
- зібрати дані та виконати навчання моделі;
- проаналізувати отримані результати та зробити висновки щодо використання моделей розпізнавання символів та виявлення об'єктів.

Об’єкт дослідження: технології автоматичного аналізу та обробки зображень.

Предмет дослідження: методи і моделі розпізнавання символів та виявлення об’єктів.

Методи дослідження. Системний аналіз, застосовувався при роботі з інформаційними ресурсами за темою дослідження; методи оцінювання, застосовувалися при аналізі систем та моделей розпізнавання символів; порівняльний аналіз, виконувався при дослідженні результатів вибраних систем та моделей нейронних мереж для вибору системи найбільш відповідну для ефективного розпізнавання символів; аналітичні методи при формуванні таблиць результатів обробки за різними моделями машинного навчання; розробка та тестування системи для розпізнавання номерного знаку автомобіля; за допомогою методу узагальнення формувалися висновки.

Інформаційна база дослідження. Методи, алгоритми та програмні засоби розпізнавання символів і виявлення об'єктів.

Структура дипломної роботи. Кваліфікаційна робота містить вступ, основну частину, яка складається з трьох розділів, до якої входять 12 рисунків, 1 таблиця, висновки, перелік використаної літератури, що містить 42 джерела, 1 додаток.

За результатами дослідження було сформульовано висновки про те, що: модель LeNet, яка була навчена на наборі даних MNIST, є ефективною у простих задачах розпізнавання символів (її точність становить 90,24%), але поступається таким архітектурам, як Faster R-CNN і Vision Transformers.

Науково-практичне значення одержаних результатів: модель LeNet може використовуватися в банківській справі, освіті та архівах для автоматизації введення даних, оцифрування документів та автоматичної класифікації інформації.

Апробація результатів роботи. Результати проведених досліджень були представлені на IV міжн. наук.-практ. Інтернет-конф. „Цифрова трансформація фінансової системи України та країн V-4 в умовах євроінтеграції” (Дубляни, 16 травня 2024 р.), тези доповіді “Комп’ютерний зір” - алгоритми виявлення і розпізнавання об’єктів” опубліковані у збірнику тез конференції [9].

РОЗДІЛ 1. Введення в задачу розпізнавання символів і зображення об'єктів

1.1. Аналіз сучасних методів і моделей розпізнавання символів і виявлення об'єктів.

Оптичне розпізнавання символів (Optical Character Recognition, OCR) використовується для широкого спектру завдань, таких як вилучення тексту зі сканованих документів або розпізнавання символів на дорожніх номерах. Алгоритми розпізнавання зазвичай використовують комбінацію протоколів розпізнавання образів і машинного навчання для точної ідентифікації символів. Програмне забезпечення для розпізнавання можна навчити розпізнавати символи різних мов, шрифтів і стилів.

У разі використання інструменту OCR для вилучення тексту процес зазвичай починається з попередньої обробки зображення.

Замість терміну «розпізнавання» часто використовують інший термін - «класифікація». Ці два терміни часто розглядаються як синоніми, але вони не є повністю взаємозамінними. Кожен з них має своє застосування, а інтерпретація обох термінів часто залежить від конкретної специфікації завдання.

Наведемо деякі типові постановки задач класифікації.

1. Задача ***ідентифікації,*** яка полягає в тому, щоб вирізнити певний конкретний об’єкт серед йому подібних (наприклад, впізнати серед інших людей свою дружину).
2. ***Віднесення об’єкта до того чи іншого класу.*** Це може бути, наприклад, задача розпізнавання літер або прийняття рішення про наявність дефекту у деякій технічній деталі. Віднесення об’єкта до певного класу відображає найтиповішу проблему класифікації, і, коли говорять про розпізнавання образів, найчастіше мають на увазі саме цю проблему. Тому ми розглядатимемо її першочергово.
3. ***Кластерний аналіз,*** який полягає в поділі заданого набору об’єктів на класи – групи об’єктів, схожі між собою за тим чи іншим критерієм. Цю задачу

часто називають ***класифікацією без учителя,*** оскільки, на відміну від задачі 2, класи апріорно не задані.

Проблеми розпізнавання легко вирішуються людиною, зазвичай несвідомо. Спроби побудови штучних систем розпізнавання менш переконливі. Основна проблема полягає в тому, що часто неможливо адекватно ідентифікувати ознаки, які потрібно розпізнати. Для завдань, де такі ознаки можна ідентифікувати, широко застосовуються системи штучного розпізнавання.

Приклади задач розпізнавання образів:

1. Розпізнавання літер.
2. Розпізнавання штрих-кодів.
3. Розпізнавання автомобільних номерів.
4. Розпізнавання осіб та інших біометричних даних.
5. Розпізнавання мови.
6. Розпізнавання зображень.
7. Розпізнавання локальних ділянок земної кори, у яких знаходяться родовища корисних копалин.

Розпізнавання образу (об'єкта, сигналу, ситуації, явища або процесу) - це завдання ідентифікації об'єктів або визначення однієї з їхніх властивостей на основі зображень (оптичне розпізнавання) або звукових записів (акустичне розпізнавання) та інших властивостей.

Одним з основних понять є поняття множини. У комп'ютерах множину представляють набором неповторюваних елементів одного виду. Термін «неповторюваний» означає, що елемент присутній або відсутній у множині. Універсальна множина містить всі можливі елементи для розв'язуваної задачі, в той час як порожня множина не містить жодного елемента [41].

Ключовими компонентами процесу розпізнавання символів і виявлення об'єктів є:

1. Алгоритми для розпізнавання символів.
2. Алгоритми для виявлення об'єктів.

3. Попередня обробка даних.
4. Глибокі архітектури для автоматизації процесу.
5. Інструменти для практичної реалізації.

Поняття “комп'ютерний зір” визначається як галузь досліджень, яка має на меті вирішити проблему надання комп'ютерам здатності бачити. Комп’ютерний зір - це технологія обробки вхідних даних, що представляють собою зображення, для того, щоб робити висновки про навколишній світ [1]. Це міждисциплінарна галузь, яка поєднує в собі штучний інтелект і машинне навчання.

Сьогодні комп'ютерний зір є дуже актуальною науковою областю, яка вирішує задачі для здобуття високого рівня розуміння цифрових зображень. Ці задачі включають методи збору, обробки, аналізу та розуміння цифрових зображень. Існує 4 типи таких задач [2]:

1. Classification - класифікує зображення за типом об'єкта, який воно містить;
2. Semantic segmentation – на зображенні яке містить об'єкт одного класу позначає кожен піксель;
3. Object detection – виявлення об'єктів потрібного класу та створення обмежувального поля навколо кожного з них;
4. Instance segmentation – позначає пікселі на об'єктах кожного класу (рис. 1.1).

Місія комп'ютерного зору полягає у створенні систем, які можуть «бачити», розуміти та інтерпретувати візуальну інформацію так само ефективно, як це робить людина. Основними завданнями комп'ютерного зору є:

1) Автоматизація аналізу візуальних даних:

- Обробка зображень і відео для вилучення інформації, корисної для прийняття рішень.
- Автоматичне розпізнавання об'єктів, тексту, облич і дій на відеозображеннях.

2) Забезпечення взаємодії між людиною і машиною:

- Розробка технологій для поліпшення взаємодії між користувачем і машиною, наприклад, аналіз жестів і емоцій.

3) Оптимізація рутинних процесів:

- Спрощення та автоматизація завдань, що вимагають обробки великих обсягів зображень, наприклад, у медицині (аналіз рентгенівських знімків) та промисловості (контроль якості).

4) Розширення можливостей штучного інтелекту:

- Інтеграція комп'ютерного зору з іншими системами штучного інтелекту для розширення їхніх можливостей.

5) Створення інноваційних технологій:

- Розробка автономних систем (робототехніка, автономні транспортні засоби), які розуміють навколишній світ за допомогою візуальної обробки даних і можуть функціонувати в складних умовах.

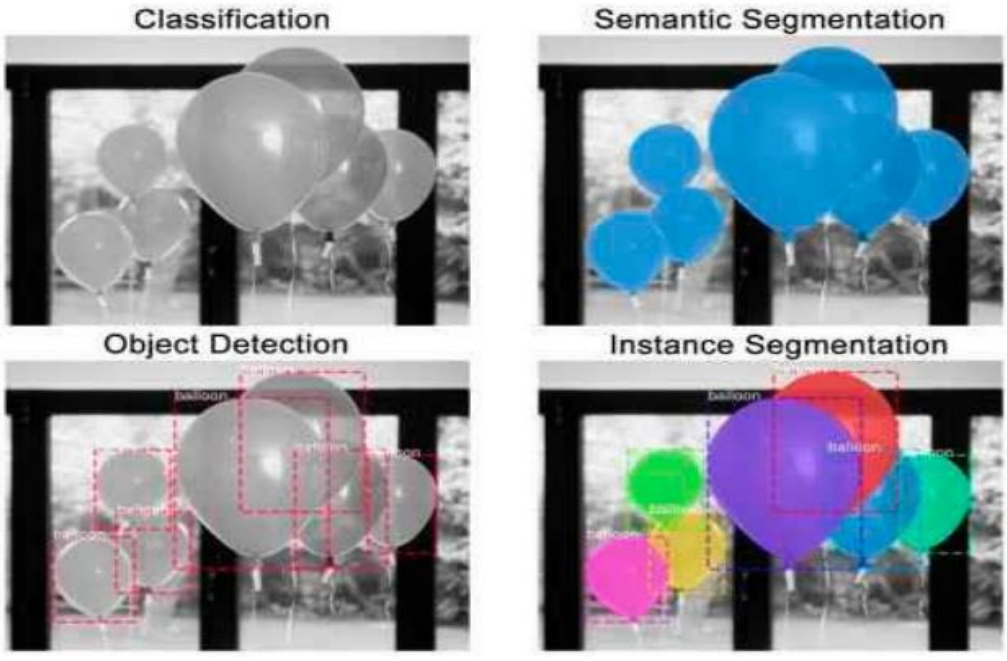


Рис.1.1. Ілюстрація типів комп'ютерного зору [2]

Одним із нових напрямів застосування комп'ютерного зору є автономні транспортні засоби. Системи які базуються на комп'ютерному баченні

допомагають водію у керуванні. Також існують повністю автономні транспортні засоби [3].

Комп'ютерний зір працює через низку послідовних кроків, які перетворюють візуальні дані (зображення або відео) на структуровану інформацію. Основні етапи цього процесу:

1. *Збір та попередня обробка даних.*

- Збір даних: камери та датчики отримують зображення та відео. Наприклад, зображення можна отримати з камер відеоспостереження, смартфонів, дронів тощо.
- Попередня обробка: нормалізація освітлення, зменшення шуму. Масштабування зображення: обрізання до стандартного розміру.
- За необхідності кольорові зображення перетворюються на чорно-білі.

2. *Виділення ознак:*

- Традиційні(ручні) алгоритми, такі як HOG (гістограма орієнтованих градієнтів) і SIFT (масштабно-інваріантне перетворення ознак), використовуються для виділення ключових ознак.
- Автоматичний вибір: у сучасних моделях глибокого навчання (наприклад, CNN) виділення ознак є вбудованим процесом.

3. *Розпізнавання:*

- Класифікація: Моделі визначають, що саме зображено (наприклад, цифру, символ, об'єкт).
- Локалізація: Алгоритми визначають точне місце об'єкта на зображенні.
- Сегментація: Зображення розділяється на області для точнішого визначення форми об'єкта.

4. *Побудова моделей:*

- Глибокі нейронні мережі:
  - CNN (згорткові нейронні мережі): підкреслюють локальні особливості та виконують точну класифікацію.

- Регіональні моделі (швидші R-CNN, YOLO): локалізують і класифікують об'єкти одночасно.
- Трансформатори (ViT, DETR): інтегрують глобальний контекст для обробки складних зображень.

▪ Моделі попереднього навчання: наприклад, ResNet і EfficientNet використовуються для вирішення завдань з меншими обчислювальними витратами.

5. *Постобробка результатів:*

▪ Видалення хибних результатів виявлення.

▪ Об'єднання об'єктів виявлення, що перекриваються.

Далі проводиться нізькорівнева обробка зображень для визначення країв, точок і сегментів зображення, які являють собою прості геометричні фігури.

Популярними методами нізькорівневого аналізу є:

• виділення кордонів або edge detection;

• сегментація;

• класифікація та виявлення об'єктів [4].

Виявлення кордонів - це метод виявлення меж об'єктів на зображенні, який є важливим етапом попередньої обробки в програмах комп'ютерного зору. Межа визначається як область різкої зміни інтенсивності пікселів і зазвичай відповідає контуру об'єкта.

До поширених методів виявлення країв належать:

1) *Оператор Собеля* (рис. 1.2): двовимірний градієнтний фільтр використовується для виявлення горизонтальних і вертикальних змін .

$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} * A, \quad G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * A$$

де:

-A - первинне зображення;

-Gx, Gy - два зображення, що в кожній точці містять наближення горизонтальної та вертикальної похідних, відповідно.

Координату x тут визначено як зростаюча «праворуч», а координату y- як зростаюча «донизу». У кожній точці зображення отримані наближення градієнта можливо об'єднувати, щоб отримувати величину градієнта.

Формула загальної амплітуда градієнта: $G = \sqrt{G_x^2 + G_y^2}$ [10]

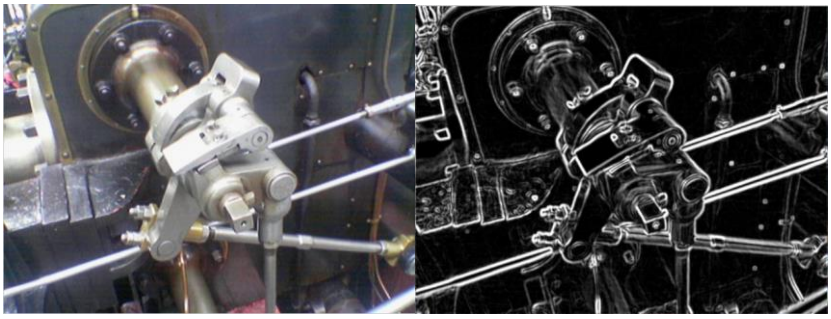


Рис.1.2 . Оператор Собеля був застосований до цього зображення [10].

2) *Метод Кенні* (Canny Edge Detection) (рис. 1.3).

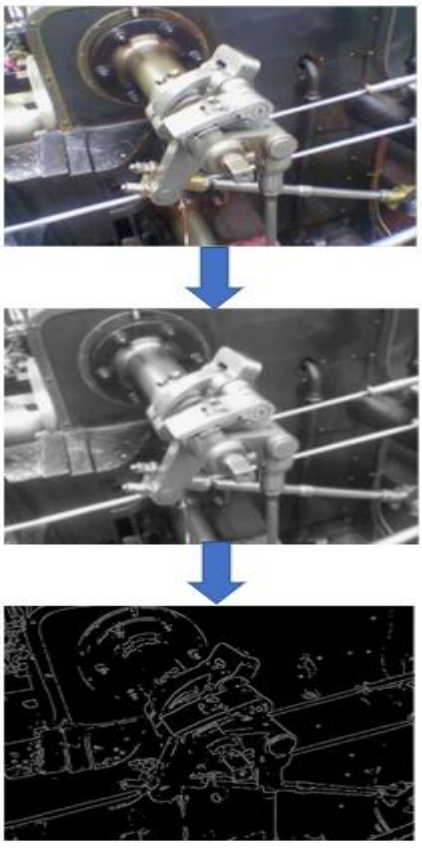


Рис. 1.3. Етапи обробки зображення методом Кенні [11].

Включає кілька етапів, таких як згладжування зображення, обчислення градієнтів, визначення локальних максимумів та застосування подвійного порогового значення для видалення слабких кордонів [11].

3) Сегментація (рис. 1.4), яка спрямована на розділення зображення на кілька частин або сегментів, які представляють окремі об'єкти чи області інтересу/ Даний метод зазвичай використовується для пошуку об'єктів і меж на зображенні. Під час обробки алгоритм присвоює кожному пікселю мітку, щоб їх можна було об'єднати за певними ознаками. Результатом є набір сегментів, що покривають усі частини зображення або контур, виокремлений із зображення [6].



Рис.1.4. Приклад сегментації зображення [6].

Методи сегментації включають:

1. Кластеризація методом k-середніх (k-means): Групує пікселі з подібними характеристиками у кластери. Вибір параметру (кількість кластерів) є важливим аспектом методу.

2. Графічні методи (Graph-based segmentation): Розглядають пікселі як вузли графа, а зв'язки між ними — як ваги ребер. Приклад — алгоритм нормалізованого розрізу (Normalized Cuts).

3. Сегментація на основі порогу (Thresholding): Найпростіший підхід, у якому пікселі розділяються на основі інтенсивності. Методи, як-от метод Оцу, автоматично визначають оптимальний поріг.

4. Глибокі нейронні мережі для сегментації: Такі архітектури, як U-Net і DeepLab, застосовуються для складних завдань сегментації, включаючи медичні зображення та автономне керування [6].

4) Класифікація (рис. 1.5) надає інформацію про зміст зображення. Класифікація зображень є основою для іншого, більш складного алгоритму комп'ютерного зору - виявлення об'єктів. Наприклад, він може відрізнити людину від автомобіля на одному зображенні.

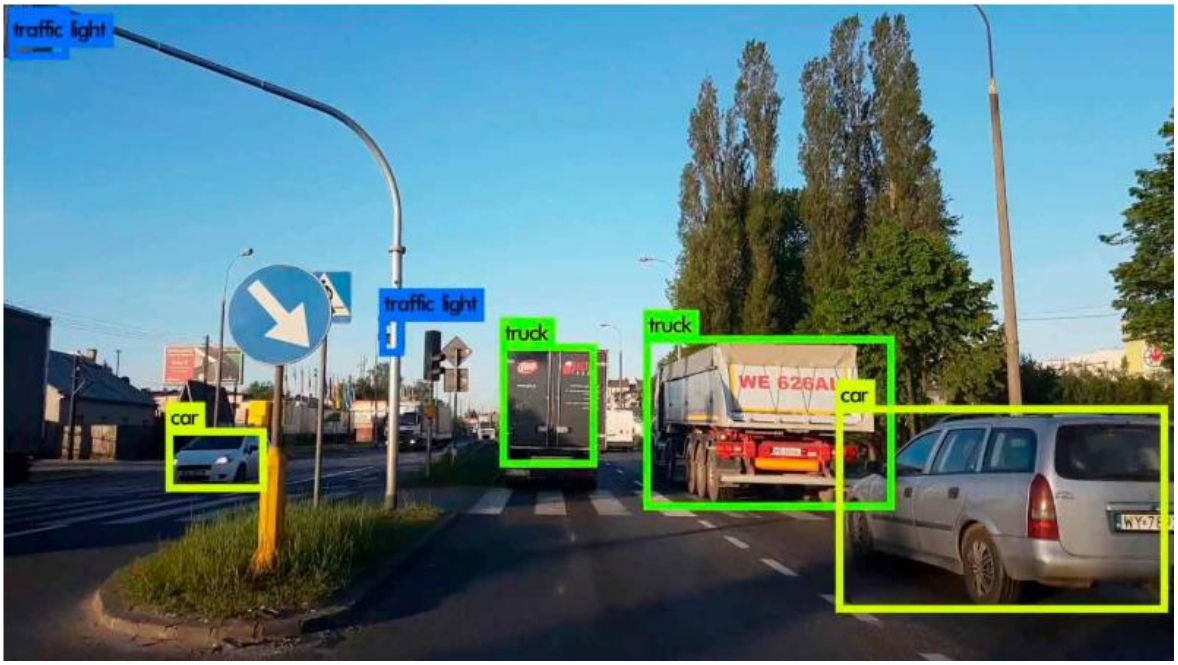


Рис. 1.5. Приклад алгоритму класифікації [4].

Завданням класифікації являє собою визначення, до якого класу належить об'єкт на зображенні, це завдання успішно виконують такі методи, як згорткові нейронні мережі (CNN).

Аналіз та розуміння зображень – це останній крок у комп'ютерному зорі, що дозволяє машинам приймати власні рішення. На цьому етапі використовують високорівневі дані, отримані з попереднього кроку. Прикладом високорівневого аналізу може бути відображення тривимірної сцени, розпізнавання чи відстеження об'єктів [4].

1.2. Типи нейро мереж для вирішення завдань розпізнавання символів

Розпізнавання символів та виявлення об'єктів є важливими завданнями комп'ютерного зору, які активно досліджуються і вдосконалюються. Сучасні методи базуються на поєднанні класичних підходів до обробки зображень і алгоритмів глибокого навчання. Нижче представлені основні методи та моделі з прикладами, формулами, порівнянням їх характеристик та аналізом.

Архітектури нейронних мереж

1) *Convolutional Neural Networks (CNN)*

CNN є базовою архітектурою для розпізнавання символів та об'єктів. Вони працюють на основі згорткових операцій.

Перевагами цих мереж є:

- простота реалізації.
- висока точність для завдань розпізнавання символів.
- стійкість до змін масштабу та деяких видів шуму.

До недоліків відносять:

- велика потреба в обчислювальних ресурсах.
- обмежена здатність до роботи з дуже складними або великими зображеннями без модифікацій архітектури.

Приклад: Модель LeNet використовується для розпізнавання рукописних цифр у базі даних MNIST. На практиці вона досягла точності близько 99% [20].

2) *YOLO (You Only Look Once)*

YOLO - це одноетапний алгоритм, який дозволяє виконувати виявлення об'єктів в реальному часі. Він розбиває зображення на сітку і визначає об'єкти в кожній комірці.

Переваги:

- Висока швидкість обробки (реальний час).
- Простота використання для різних завдань.

Недоліки:

- Менша точність для дрібних об'єктів у порівнянні з іншими методами.
- Може помилково виявляти об'єкти через надмірну генералізацію.

Приклад: YOLO широко використовується в автономних транспортних засобах для виявлення пішоходів та дорожніх знаків [13].

3) Faster R-CNN Faster

R-CNN — це двоетапна архітектура, яка поєднує регіональні пропозиції з CNN для покращення точності виявлення об'єктів. Використовується мережа регіональних пропозицій (RPN) для вибору областей інтересу.

Переваги:

- Висока точність.
- Добре підходить для завдань з великим обсягом даних.
- Сприйнятливість до дрібних деталей об'єктів.

Недоліки:

- Високі вимоги до обчислювальних ресурсів.
- Повільна обробка у порівнянні з одноетапними методами (наприклад, YOLO).

Приклад: Використовується для аналізу медичних зображень (виявлення пухлин на МРТ) [14].

4) Transformers (Vision Transformers - ViT)

ViT — це сучасний підхід до обробки зображень, який використовує механізм уваги, що зарекомендував себе у завданнях обробки природної мови.

Переваги:

- Висока точність для завдань класифікації та виявлення об'єктів.
- Ефективність при роботі з великими наборами даних.
- Гнучкість у навчанні на різних типах даних.

Недоліки:

- Високі вимоги до обчислювальних ресурсів.
- Менша інтерпретованість у порівнянні з CNN [15].

5) Single Shot MultiBox Detector (SSD)

Single Shot Detector (SSD) — це вдосконалений алгоритм, який зробив революцію в цій галузі, дозволивши виявляти об’єкти на зображеннях у реальному часі. У цій статті розповідається про роботу SSD, його архітектуру, ключові переваги та практичне застосування.

Переваги :

• Висока швидкість обробки, що дозволяє працювати в реальному часі.

• Можливість виявлення об'єктів різного розміру завдяки використанню багаторівневих особливостей (feature maps).

• Менша потреба в обчислювальних ресурсах порівняно з Faster R-CNN.

Недоліки :

• Точність нижча, ніж у методів R-CNN, особливо при виявленні малих об'єктів.

• Важче оптимізувати для роботи з дуже дрібними деталями.

Приклад:

Модель SSD300 (з базовою архітектурою VGG16) використовується для виявлення об'єктів у наборі даних COCO та Pascal VOC. У тестах вона досягла продуктивності близько 59 FPS (кадрів в секунду) на GPU Nvidia Titan X при точності mAP ≈ 77,5% на наборі Pascal VOC [42].

У табл. 1.1. наведені основні характеристики систем розпізнавання зображень.

Таблиця 1.1

Порівняння основних характеристик систем розпізнавання зображень

Модель	Точність (%)	Швидкість (FPS)	Основна область застосування	Переваги	Недоліки
CNN (LeNet)	~99 (MNIST)	Висока	Розпізнавання символів	Простота, точність	Обмежена здатність до масштабів
YOLO	90-95	Дуже висока	Виявлення об'єктів у реальному часі	Швидкість, універсальність	Точність на дрібних об'єкта
Faster R-CNN	95-98	Низька	Точне виявлення об'єктів	Точність, робота з дрібними деталями	Повільність
Vision Transformers	97-99	Середня	Класифікація та сегментація об'єкті	Гнучкість, висока точність	Ресурсоємність
SSD	92-96	Середня	Баланс між точністю та швидкіст	Швидкість, зручність	Точність нижче Faster R-CNN

Складено автором за [12-16].

З табл. 1.1 видно, що згорткові нейронні мережі (CNN) демонструють високу точність (~99%) і продуктивність у задачах розпізнавання символів у порівнянні з іншими моделями та можуть бути найкращим вибором для зображень з певними розмірами і чіткими контурами. Водночас інші моделі, такі як YOLO, Faster R-CNN, Vision Transformers і SSD, краще справляються із завданнями виявлення об'єктів, особливо в режимі реального часу і дрібномасштабних додатках. Однак їхні переваги часто супроводжуються обмеженнями, такими як потреба у великих обчислювальних ресурсах (Vision Transformers) або недостатня точність у певних випадках (YOLO).

Підсумовуючи, можна сказати, що CNN залишаються найкращим варіантом для задач розпізнавання символів, тоді як вибір моделі для виявлення об'єктів залежить від специфіки програми та вимог до швидкості і точності.

РОЗДІЛ 2. Розробка моделі розпізнавання символів

2.1. Структура алгоритмічно-програмних засобів розпізнавання символів

Для побудови алгоритмів і програмних засобів для розпізнавання рукописного тексту роботу системи потрібно розділити на чотири етапи.

Спочатку система повинна виконати завдання виявлення тексту на зображенні, результатом якого є конкретні координати тексту або символу на зображенні. Наступний етап - обробка зображення. На цьому етапі система може вирізати текст із зображення для розпізнавання. Останній етап - виділення ключових ознак, які використовуються для розпізнавання символів, і власне розпізнавання символів [32].

На рис. 2 показано основні інструменти системи розпізнавання рукописних символів. Система отримує на вхід певне зображення, а на виході відображає результат - оцифрований документ.

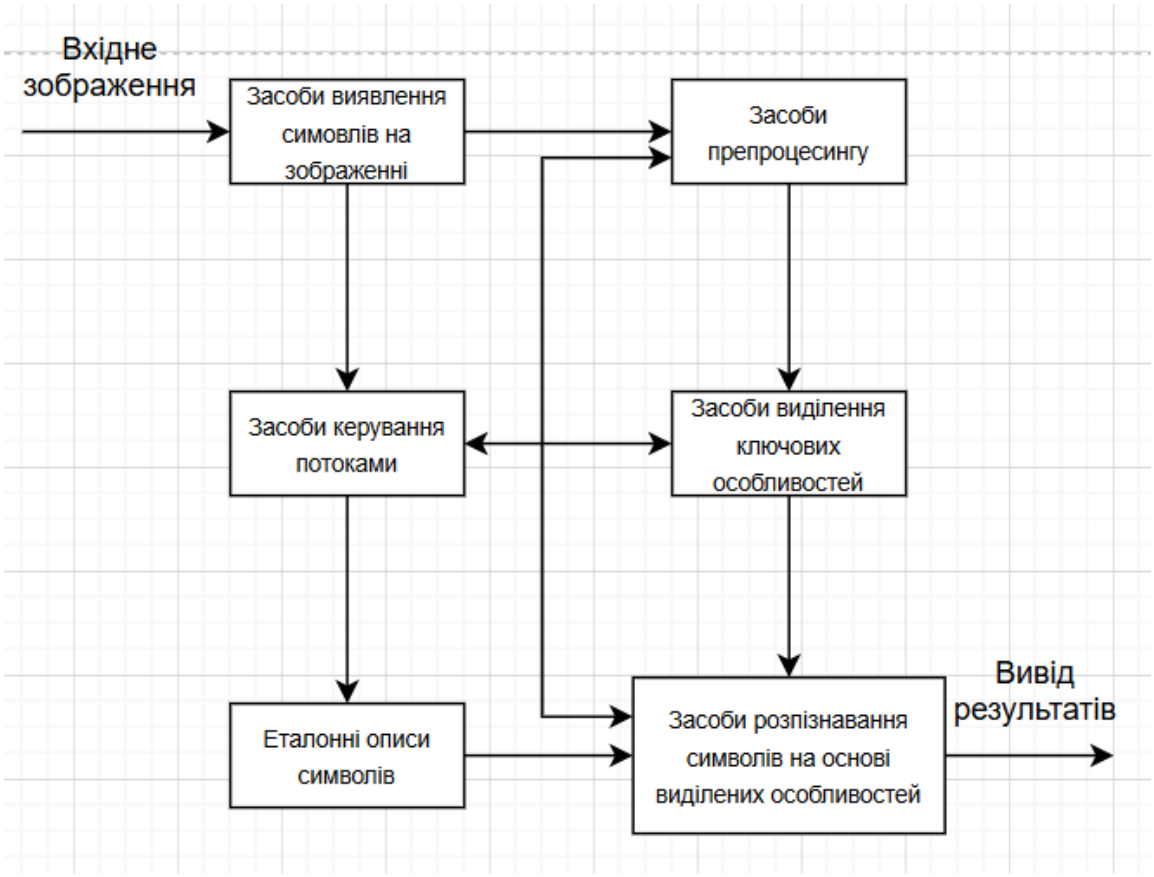


Рис. 2. Структура алгоритмічно-програмних засобів розпізнавання рукописних текстів [29]

Перед початком розробки системи, треба скласти план робіт і побудови системи, це в свою чергу, допоможе зрозуміти характер завдань. Якщо незалежні завдання визначені на етапі розробки, їх можна ідентифікувати з першого погляду і розділити роботу, що прискорює впровадження системи.

Створення моделі розпізнавання символів вимагає вибору відповідного набору даних, який відповідає конкретним вимогам завдання. Найчастіше використовуються такі бази даних, як MNIST, EMNIST і SynthText. У додатку А наведена порівняльна таблиця цих наборів даних.

MNIST (модифікована база даних Національного інституту стандартів і технологій) є класичною для задач розпізнавання рукописних чисел. Ця база даних містить 70 000 чорно-білих зображень розміром 28x28 пікселів. Ці зображення розділені на 60 000 навчальних прикладів і 10 000 тестових прикладів [21]. Завдяки своєму невеликому розміру і простоті, MNIST стала стандартом для тестування базових моделей машинного навчання (рис.2.1).

Тип	Структура	Спотворення	Попередня обробка	Помилка (%)
Лінійний класифікатор	Одношаровий перцептрон	Ні	Ні	12.0
Лінійний класифікатор	Попарний лінійний класифікатор	Ні	Вирівнювання	7.6
Метод найближчих k-сусідів	K-ПН з нелінійною деформацією (P2DHMDM)	Ні	Shiftable edges	0.52
Метод градієнтного підсилювання[en]	Обробка залишків на основі гаароподібних ознак	Ні	Гаароподібна ознака	0.87
Нелінійний класифікатор	40 PCA + квадратичний класифікатор	Ні	Ні	3.3
Метод опорних векторів	Віртуальна система опорних векторів, град-9 поли, 2-пікселя jittered	Ні	Вирівнювання	0.56
Нейронна мережа	2-рівнева мережа 784-800-10	Ні	Ні	1.6
Нейронна мережа	2-рівнева мережа 784-800-10	Пружні деформації	Ні	0.7
Глибока нейронна мережа	6-рівнева мережа 784-2500-2000-1500-1000-500-10	Пружні деформації	Ні	0.35
Згорткова нейронна мережа	6-рівнева мережа 784-40-80-500-1000-2000-10	Ні	Розширення даних для навчання	0.31
Згорткова нейронна мережа	6-рівнева мережа 784-50-100-500-1000-10-10	Ні	Розширення даних для навчання	0.27
Згорткова нейронна мережа	Ансамбль з 35 CNN-мереж, 1-20-C-40-C-150-10	Пружні деформації	З нормалізацією	0.23
Згорткова нейронна мережа	Ансамбль з 5 CNN-мереж, 6-784-50-100-500-1000-10-10	Ні	Розширення даних для навчання	0.21

Рис. 2.1. Результати обробки за різними моделями машинного навчання [21].

Однак її обмежена складність призвела до появи інших, більш складних наборів даних.

Опираючись на результати навчання на цій базі даних, можна сказати, що найкращою мережею є згорткова нейронна мережа, а саме її модель Ансамбль з 5 CNN- мереж, з помилкою лише в 0.21%. В той же час найбільшою помилку має лінійний класифікатор з Однорівневим перцептором в 12.0% помилки. Виходячи з цього можна сказати, що згорткові нейронні мережі мають найкращу структуру, завдяки їхній спеціалізації для обробки зображень.

Extended MNIST (EMNIST) - це розширення MNIST, яке включає не тільки цифри, але й літери латинського алфавіту (рис. 2.2). Цей набір даних містить понад 814 255 зображень і розділений на кілька підмножин: Balanced, ByClass, Letters і т.д. За допомогою EMNIST можна вирішувати більш складні завдання, що вимагають одночасного розпізнавання цифр і букв. EMNIST можна використовувати для вирішення більш складних завдань, які вимагають одночасного розпізнавання цифр і букв. Тому він корисний для створення систем автоматичного розпізнавання тексту у формах та інших документах.

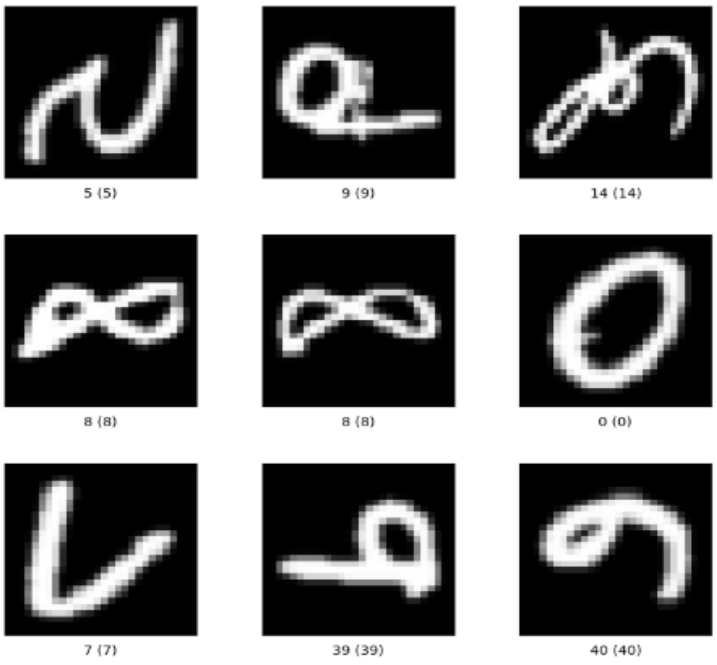


Рис.2.2 .Приклад із набору який містить як цифри, так і букви [17]

SynthText - це штучно згенерований набір даних з понад 800 000 зображень. Текст на цих зображеннях накладено на реальні фотографії (рис. 2.3), що дозволяє моделювати широкий спектр сценаріїв, таких як дорожні знаки, рекламні банери, текст продукту і т.д. SynthText охоплює широкий спектр шрифтів, скольорів, позицій тексту і умов освітлення. Цей набір даних широко використовується в додатках OCR (оптичного розпізнавання символів), особливо в складних реальних умовах.



Рис. 2.3. Synthetic Scene-Text приклади зображень [18].

Цей розділ описав нам, які зазвичай використовуються набори для навчання моделей розпізнавання символів, такі як MNIST, EMNIST і SynthText. Кожен з цих наборів даних має свої переваги та сфери застосування.

З огляду на вибір згорткових нейронних мереж (CNN) для реалізації моделей, набір даних MNIST є найкращим вибором для використання; набір даних MNIST містить збалансовану вибірку рукописних чисел, що дозволяє ефективно навчати та тестувати нейронні мережі. нейронних мереж. Крім того,

MNIST є стандартним еталоном для оцінки якості моделей розпізнавання символів і може порівнюватися з результатами інших досліджень.

Таким чином, вибір MNIST як основного набору даних забезпечує надійність навчання та високу точність розпізнавання символів за допомогою ШНМ.

Підготовка даних є критичним етапом для забезпечення ефективного навчання моделей. Основними кроками навчання моделі є:

1. Нормалізація: масштабування значень пікселів до діапазону [0, 1] або [-1, 1]. Це допомагає уникнути проблем із градієнтами та прискорює процес навчання.
2. Аугментація даних: Додавання випадкових змін до зображень, таких як обертання, масштабування, зсув або інші трансформації. Аугментація підвищує здатність моделі до узагальнення, зменшуючи перенавчання.

Наприклад [14]:

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
data_gen = ImageDataGenerator(
    rescale=1.0/255,
    rotation_range=15,
    width_shift_range=0.1,
    height_shift_range=0.1,
    zoom_range=0.2
)
train_data = data_gen.flow_from_directory('train_dataset_path',
target_size=(28, 28), batch_size=32)
```

3. Розділення даних: Розподіл даних на тренувальний, валідаційний та тестовий набори у співвідношенні 80/10/10. Тренувальний набір використовується для навчання моделі, валідаційний — для налаштування параметрів, а тестовий — для оцінки продуктивності.

4. Збалансованість класів: Якщо класи представлені нерівномірно, модель може упереджувати частіші класи. Тому важливо переконатися у збалансованості або застосувати методи, такі як повторна вибірка чи зважування класів.
5. Очистка даних: Видалення шуму, неправильних міток або зображень низької якості для забезпечення достовірності даних. Це особливо важливо для реальних наборів, таких як EMNIST [17].

Ефективна підготовка даних є ключовим елементом для успішного навчання моделі. Основні етапи попередньої обробки даних, розглянуті в цьому розділі, включають нормалізацію, доповнення, розбиття даних, балансування класів і очищення набору даних. Зокрема, нормалізація допомагає стабілізувати градієнти, а доповнення покращує загальність моделі. Поділ даних на навчальну, перевірочну та тестову множини дозволяє ефективно контролювати якість навчання. Дотримання цих етапів забезпечує стабільне і точне навчання нейромережевих моделей, знижуючи ризик перенавчання і підвищуючи продуктивність.

2.2.Вибір типу нейронної мережі для завдання розпізнавання символів

Нейромережа це деякий набір нейронів поєднаних між собою. Принцип роботи нейромережі базується на перетворенні вхідного вектору даних в вихідний [7].

Для того щоб навчити нейромережу в нейрони записують певний еталон, потім подається вхідний сигнал і порівнюють його з еталоном. Відмінність між еталоном і вхідним сигналом додається до еталону. Чим більше вхідних впливів тим краще навчиться мережа [8].

Завдання розпізнавання символів зазвичай вирішується за допомогою **згорткових нейронних мереж** (CNN). Ці моделі підходять для обробки зображень, оскільки вони можуть виокремлювати локальні ознаки [19].

CNN складаються з кількох ключових компонентів (рис. 2.4):

- Згорткові шари (Convolutional Layers): Виявляють локальні патерни, такі як контури, кути та текстури.
- Шари підвибірки (Pooling Layers): Зменшують розмірність вхідних даних, зберігаючи основні ознаки.
- Шар згладжування (Flatten Layer): Перетворює багатовимірні дані в одномірний вектор, який можна подати на повнозв'язний шар.
- Повнозв'язні шари (Fully Connected Layers): Об'єднують ознаки та виконують класифікацію.
- Активаційна функція (Activation Function): Нелінійна функція, яка додає мережі здатність моделювати складні залежності.

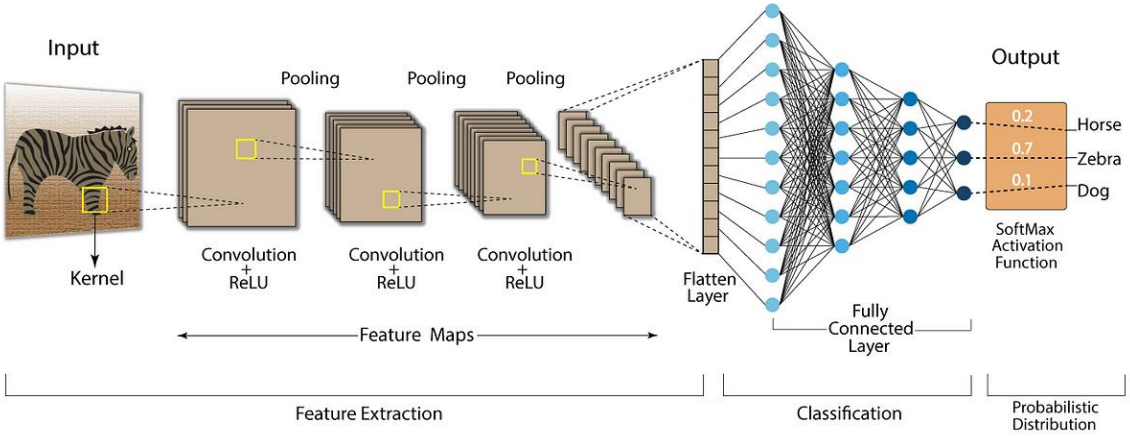


Рис. 2.4. Компоненти CNN [19].

Приклад коду:

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense
model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)),
    MaxPooling2D((2, 2)),
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
```

```
    Flatten(),
    Dense(128, activation='relu'),
    Dense(10, activation='softmax')
])
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

Спочатку імпортуються бібліотеки:

- Sequential: використовується для створення послідовної моделі нейронної мережі, де шари додаються один за одним.
- Conv2D: шар згортки (конволюції), який використовується для виділення просторових особливостей зображення.
- MaxPooling2D: шар максимального підсемплінгу, який зменшує розмірність вхідних даних, зберігаючи найважливіші особливості.
- Flatten: шар, який перетворює багатовимірний вихід (наприклад, 2D-матрицю) в одномірний вектор.
- Dense: звичайний повнозв'язний шар, який використовується для класифікації або регресії.

Після чого починаємо будувати модель:

- **Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1))**:
  - Створює 32 фільтри розміром 3x3.
  - Використовується функція активації ReLU (Rectified Linear Unit) для введення нелінійності.
  - input_shape=(28, 28, 1) визначає розмір вхідних даних (зображення 28x28 пікселів з 1 каналом, тобто чорно-біле зображення).
- **MaxPooling2D((2, 2))**:
  - Застосовує пулінг із вікном 2x2, що зменшує розмірність зображення вдвічі (наприклад, 28x28 -> 14x14).
- **Conv2D(64, (3, 3), activation='relu')**:
  - Другий шар згортки з 64 фільтрами розміром 3x3.

- Використовується ReLU як функція активації.

- **MaxPooling2D((2, 2))**:

- Другий шар пулінгу, зменшує розмірність ще вдвічі (наприклад, 14x14 -> 7x7).

- **Flatten()**:

- Перетворює вихідну багатовимірну матрицю (наприклад, 7x7x64) в одномірний вектор (3136 елементів), щоб його можна було передати до повнозв'язного шару.

- **Dense(128, activation='relu')**:

- Повнозв'язний шар із 128 нейронами, використовує ReLU для активації.

- Використовується для навчання на узагальнених особливостях.

- **Dense(10, activation='softmax')**:

- Вихідний шар із 10 нейронами, де кожен нейрон відповідає одному класу (наприклад, цифри від 0 до 9 для MNIST).

- Функція активації softmax перетворює вихідні значення у ймовірності для кожного класу.

Після побудови моделі, потрібно її скласти разом, для цього використовуємо компіляцію моделі:

- **optimizer='adam.** Оптимізатор ADAM (Adaptive Moment Estimation), який поєднує переваги градієнтного спуску зі змінним кроком і моментуму. Відомий своєю ефективністю в навчанні.

- **loss='categorical_crossentropy'** це функція втрат для багатокласової класифікації. Використовується, коли вихідний шар має softmax.

- **metrics=['accuracy']**. Модель буде оцінювати свою точність (accuracy) під час навчання та тестування.

Нейронна мережа YOLO (You Only Look Once) - один з найпопулярніших підходів до виявлення об'єктів, що передбачає клас і граничну область об'єктів на зображенні за один прохід мережі. Основною

перевагою цієї мережі є швидкість обробки, яка дозволяє одночасно класифікувати та локалізувати об'єкти. Для виявлення об’єктів мережа розбиває зображення на сітки S×S, і кожна комірка мережі прогнозує межі об'єктів у цій комірці.

Ця модель навчається на наборі даних ImageNet-1000. Модель тренується протягом тижня, і вона досягає точності топ-5 у 88% під час перевірки ImageNet 2012, що можна порівняти з GoogLeNet (переможцем ILSVRC 2014), найсучаснішою моделлю на той час. Fast YOLO використовує менше шарів (9 замість 24) і менше фільтрів. Окрім цього, швидкий YOLO має всі параметри, подібні до YOLO. YOLO використовує функцію втрати суми квадратів помилок, яку легко оптимізувати. Однак ця функція надає однакову вагу завданням класифікації та локалізації [28].

Приклад реалізації

```
import cv2
import numpy as np
# Завантаження конфігурації та ваг мережі YOLO
net = cv2.dnn.readNet('yolov3.weights', 'yolov3.cfg')

# Завантаження класів
with open('coco.names', 'r') as f:
    classes = [line.strip() for line in f.readlines()]

# Завантаження зображення
image = cv2.imread('image.jpg')
height, width, channels = image.shape
# Підготовка зображення для мережі
blob = cv2.dnn.blobFromImage(image, 0.00392, (416, 416), (0, 0, 0), True, crop=False)
```

```
        net.setInput(blob)
        # Отримання виходів з мережі
        layer_names = net.getLayerNames()
        output_layers = [layer_names[i[0] - 1] for i in
net.getUnconnectedOutLayers()]
        outs = net.forward(output_layers)
        # Обробка результатів
        class_ids = []
        confidences = []
        boxes = []
        for out in outs:
            for detection in out:
                scores = detection[5:]
                class_id = np.argmax(scores)
                confidence = scores[class_id]
                if confidence > 0.5:
                    center_x = int(detection[0] * width)
                    center_y = int(detection[1] * height)
                    w = int(detection[2] * width)
                    h = int(detection[3] * height)
                    x = int(center_x - w / 2)
                    y = int(center_y - h / 2)
                    boxes.append([x, y, w, h])
                    confidences.append(float(confidence))
                    class_ids.append(class_id)
        # Нанесення обмежувальних рамок на зображення
        indices = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.4)
        for i in indices:
            i = i[0]
```

```
        box = boxes[i]
        x, y, w, h = box
        label = str(classes[class_ids[i]])
        confidence = str(round(confidences[i], 2))
        cv2.rectangle(image, (x, y), (x + w, y + h), (0, 255, 0), 2)
        cv2.putText(image, label + ' ' + confidence, (x, y - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 2)

    # Виведення результату
    cv2.imshow('Image', image)
    cv2.waitKey(0)
    cv2.destroyAllWindows()
```

[27]

Нейронна мережа зворотного розповсюдження (Backpropagation Neural Network)

Цей тип мережі базується на алгоритмі зворотного поширення для корекції ваг зв'язків між нейронами. Процес навчання має три основні етапи: пряме поширення, обчислення помилки та корекція ваг [23] .

Для навчання нейромережі я знаходжу певну функціональну залежність Y=F(X) де X – вхідний, а Y – вихідний вектори. Під час навчання ставиться завдання мінімізації цільової функції помилки нейромережі , яка знаходиться за методом найменших квадратів:

$$E(w) = \frac{1}{2}\sum_{j-1}^{p}(y_i - d_i),$$

де:

y_i– значення j-го виходу нейромережі,

d_i – цільове значення j-го виходу,

p - Число нейронів у вихідному шарі.

Навчання нейромережі виконується за методом градієнтного спуску

$$\Delta w_{ij} = \frac{-E}{w_{ij}},$$

Де: η - параметр, який визначає швидкість навчання.

$$\frac{E}{w_{ij}} = \frac{E}{y_i}\frac{dy_i}{dS_j}\frac{S_j}{w_{ij}},$$

Де:

y_i- значення виходу j-го нейрона,

S_i – зважена сума вхідних сигналів.

При цьому:

$$\frac{S_j}{w_{ij}} = x_i,$$

де:

x_i - значення і-го входу нейрона.

Далі розглянемо визначення першого множника формули.

$$\frac{E}{y_j} = \sum_k \frac{E}{y_k}\frac{dy_k}{dS_k}\frac{S_k}{y_j} = \sum \frac{E}{y_k}\frac{dy_k}{dS_k} \quad {}^{(n=1)}_{jk},$$

де:

k – число нейронів у шарі n+1.

Введемо допоміжну змінну, для обчислення похибки або локального градієнта для нейрона в процесі зворотного поширення помилки.

$${}^{(n)}_{j} = \frac{E}{y_j}\frac{dy_j}{dS_j},$$

Тоді ми зможемо визначити рекурсивну формулу для визначення n-ного шару, якщо відомо наступного (n+1)-го шару.

$${}^{(n)}_{j} = \left[\sum_k {}^{(n+1)}_{j} \; {}^{(n+1)}_{jk}\right]\frac{dy_j}{dS_j},$$

Останній шар визначається за формулою:

$$_{j}^{(N)} = \left(y_i^{(N)} - d_i\right)\frac{dy_j}{ds_j},$$

Запишемо формулу у розкритому вигляді:

$$\Delta w_{ij}^{(n)} = -\ _{j}^{(n)}x_n^i, \text{ [23]}$$

Приклад реалізації:

```
import numpy as np
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

# Створення простої мережі зворотного розповсюдження
model = Sequential([
    Dense(128, input_dim=784, activation='relu'),
    Dense(64, activation='relu'),
    Dense(10, activation='softmax')
])

model.compile(optimizer='adam', loss='categorical_crossentropy',
metrics=['accuracy']) [23]
```

Рекурентна нейронна мережа (RNN)

RNN підходять для обробки послідовних даних. У розпізнаванні символів їх можна використовувати для аналізу тексту як послідовності символів. Слід використовувати дану мережу, для обчислень помилок між фактичним та цільовим результатом.

Формула розрахунку поточного стану[24]:

$$h_t = f(h_{t-1}, x_t),$$

де:

h_t - поточний стан, ф

$h_{t-1} - 1$ -попередній стан,

x_t - вхідний стан.

Формула для застосування функції активації:

$$h_t = tanh(W_{hh}h_{t-1} - 1 + W_{xh}x_t),$$

де:

W_{hh} - вага рекурентного нейрона,

W_{xh}- вага на вхідному нейроні.

Формула розрахунку виходу:

$$y_t = W_{hy}h_t,$$

де:

y_t - вихід

W_{hy} - вага на вихідному шарі

Навчання здійснювалося за наступним алгоритмом:

1. Мережа вимагає одноразового входу в систему.
2. Потім обчислюється поточний стан, використовуючи поточні вхідні дані та набір попередніх станів;
3. На наступному часовому кроці поточний ht дорівнює ht-1;
4. Може бути зроблено будь-яку кількість часових кроків залежно від задачі та інформації з усіх попередніх станів з усіх попередніх станів може бути об'єднана.
5. Після завершення всіх часових кроків, кінцевий поточний стан використовується для обчислення виходу.
6. Вихід порівнюється з фактичним виходом, тобто цільовим виходом, і генерується помилка.
7. Помилка подається в мережу для оновлення ваг, і мережа (RNN) навчається.

Переваги RNN:

- Здатність моделювати часові залежності.
- Підходять для аналізу послідовних даних [24].

Приклад реалізації:

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import SimpleRNN, Dense

# Створення RNN для розпізнавання послідовностей
model = Sequential([
    SimpleRNN(128, input_shape=(28, 28)),
    Dense(10, activation='softmax')
])
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

[24]

Визначення структури і параметрів моделі

Структура моделі включає вибір кількості шарів, розмірів ядер згортки та кількості нейронів у фінальних шарах. Для задачі розпізнавання символів добре підходить згорткові нейронні мережі, саме тому будемо далі використовувати, як приклад мережу LeNet. Оптимально використовувати такі параметри:

1. **Згортковий шар (Conv2D):**

- Кількість фільтрів: **6**.
- Розмір ядра (kernel): **5x5**.
- Функція активації: **ReLU**.
- Вхідна форма даних: **(28, 28, 1)** (зображення розміром 28x28 у відтінках сірого).

2. **Шар середнього згортання (AveragePooling2D):**

- Розмір ядра (pool_size): **2x2**.

3. **Другий згортковий шар (Conv2D):**

- Кількість фільтрів: **16**.

- Розмір ядра (kernel): **5x5**.
- Функція активації: **ReLU**.

4. **Другий шар середнього згортання (AveragePooling2D):**

- Розмір ядра (pool_size): **2x2**.

5. **Шар вирівнювання (Flatten):**

- Перетворює багатовимірні дані в одномірний вектор, щоб передати їх до повнозв'язних шарів.

6. **Перший повнозв'язний шар (Dense):**

- Кількість нейронів: **120**.
- Функція активації: **ReLU**.

7. **Другий повнозв'язний шар (Dense):**

- Кількість нейронів: **84**.
- Функція активації: **ReLU**.

8. **Вихідний шар (Dense):**

- Кількість нейронів: **10** (кількість класів для класифікації).
- Функція активації: **Softmax** (підходить для задачі багатокласової класифікації).

9. **Компіляція моделі:**

- Оптимізатор: **SGD** (градієнтний спуск).
- Функція втрат: **categorical_crossentropy** (використовується для задач багатокласової класифікації).

Напишемо код реалізації під нашу структуру:

```
model = K.Sequential()

model.add(K.layers.Conv2D(filters = 6, kernel_size=(5, 5), activation="relu", input_shape=(28, 28 ,1)))

model.add(K.layers.AveragePooling2D(pool_size=(2, 2)))

model.add(K.layers.Conv2D(filters = 16, kernel_size=(5, 5), activation="relu"))

model.add(K.layers.AveragePooling2D(pool_size=(2, 2)))
```

```
model.add(K.layers.Flatten())
model.add(K.layers.Dense(units = 120, activation="relu"))
model.add(K.layers.Dense(units = 84, activation="relu"))
model.add(K.layers.Dense(units = 10, activation="softmax"))
model.compile(optimizer ="SGD", loss = K.losses.categorical_crossentropy)
```

Переваги та недоліки.

- Переваги: висока точність, здатність виявляти локальні ознаки, масштабованість.

Недоліки: мережа працює тільки з простими об'єктами — невеликими зображеннями з низькою роздільною здатністю [25].

Підводячи підсумки, для створення системи для розпізнавання номерного знаку автомобіля, з використанням технологій комп’ютерного зору, найкраще всього підходять згортокові нейронні мережі (CNN).

2.3. Навчання і тестування моделей

Для навчання кожної моделі використовується підготовлений набір даних, описаний в розділі 2.1. Кожен набір даних був обраний з урахуванням специфічних властивостей моделі:

- У випадку з CNN: MNIST є найкращим вибором завдяки своїй простоті та структурі. Завдяки компактному розміру зображення (28x28 пікселів) і чіткій розмітці класів, CNN можуть ефективно виділяти локальні особливості, такі як контури і кути, які є ключовими для розпізнавання рукописних цифр.
- Для RNN: EMNIST є розширенням MNIST, з символами, організованими в послідовності і з включеними цифрами. Це робить EMNIST ідеальним для рекурентних мереж, які враховують залежності між елементами послідовності, що необхідно для аналізу тексту і рукописних символів.
- Для моделей на основі білінійної інтерполяції: SynthText використовується через його здатність моделювати реальні сценарії з текстом на зображеннях. Штучно створені зображення з різними умовами освітлення, шрифтами та кольорами дозволяють моделям адаптуватися до складних реальних ситуацій. Кожна модель вимагає специфічного підходу до навчання залежно від її структури.

Згорткові нейронні мережі (CNN) навчаються на наборі даних MNIST, який ідеально підходить для початкового навчання завдяки своїй структурі та розміру; MNIST містить компактні зображення рукописних цифр розміром 28x28 пікселів, що дозволяє моделі навчитися ефективно. Цей набір даних, містить компактне зображення рукописних цифр, розміром 28x28 пікселів. Така простота даних зменшує ризик надмірної підгонки та пришвидшує збіжність. Модель використовує оптимізатор Adam та категоричну функцію втрати (categorical crossentropy) перехресної ентропії для стабільного та точного оновлення ваг під час навчання. Навчання виконується на 60 000 зображень з навчального набору, а 10 000 зображень з тестового набору використовуються для валідації.

Приклад коду:

```
history = model.fit(
    train_images, train_labels,
    epochs=10,
    batch_size=32,
    validation_data=(test_images, test_labels)
)
```

Побудова графіку втрат:

```
import matplotlib.pyplot as plt
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.legend()
plt.show() [26]
```

Нейронна мережа зворотного розповсюдження (Backpropagation Neural Network)

Для навчання цієї моделі використовуються зображення, попередньо перетворені у 784 вектори (28x28 пікселів). Навчання виконується на тому ж наборі даних MNIST.

Приклад коду:

```
history = model.fit(
    x_train, y_train,
    epochs=10,
    batch_size=64,
    validation_data=(x_test, y_test)
)[23]
```

Рекурентна нейронна мережа (RNN) навчається на наборі даних EMNIST, і кожне зображення розглядається як послідовність рядків. Такий підхід дозволяє моделі аналізувати контекст символів.

Приклад коду:

```
history = model.fit(
   emnist_train, emnist_train_labels,
   epochs=15,
   batch_size=128,
   validation_data=(emnist_test, emnist_test_labels) [24]
```

Тестування моделей на окремому тестовому наборі даних

Так, як найбільш підходящим варіантом стали згорткові нейронні мережі, тестування буде проводитись на моделі LeNet на окремих наборах даних, щоб оцінити їх продуктивність.

Приклад тестування:

```
result = np.argmax(model.predict(np.expand_dims(X_test, -1)), axis = 1)
print (classification_report(result, y_test, digits = 4))
```

У цьому розділі різні моделі були навчені та протестовані на відповідних наборах даних. Згорткові нейронні мережі (CNN) найкраще підходять для побудови моделей розпізнавання символів і показали високу продуктивність на наборі даних MNIST. Завдяки компактному розміру зображень та структурованій розмітці, ШНМ здатні ефективно виокремлювати локальні ознаки символів. Процес навчання та тестування підтверджує, що за допомогою MNIST можна досягти високої точності розпізнавання рукописних цифр, гарантуючи стабільність та швидку збіжність моделі.

РОЗДІЛ 3. Аналіз результатів застосування CNN

3.1. Аналіз результатів моделі LeNet розпізнавання символів та виявлення об'єктів

За результатами навчання (рис. 3.1), а саме в 100 епох можна сказати, що фінальне значення втрат(loss): 0.3382. Цей результат вказує на те, що модель має низьке значення функції втрат, отже модель добре адаптувалася до навчальних даних і зменшила помилки класифікації від 2.3500 до 0.3382.

```
Epoch 100/100
2/2 ━━━━━━━━━━━━━━━━━━━━ 19s 6s/step - loss: 0.3382
<keras.src.callbacks.history.History at 0x7a484dbd8820>
```

Рис. 3.1. Результат навчання моделі LeNet.

Точність під час тестування (рис. 3.2), на тестовому наборі(10 000 зображень) стало значення 90.24%(загальна точність). Це свідчить про те, що модель класифікує трохи більше 90% тестових прикладів.

```
313/313 ━━━━━━━━━━━━━━━━━━━━ 2s 7ms/step
              precision    recall  f1-score   support

           0     0.9449    0.9325    0.9387       993
           1     0.9771    0.9677    0.9724      1146
           2     0.8798    0.8790    0.8794      1033
           3     0.8347    0.9451    0.8864       892
           4     0.8809    0.8827    0.8818       980
           5     0.8576    0.9307    0.8926       822
           6     0.9332    0.9351    0.9342       956
           7     0.9144    0.9409    0.9275       999
           8     0.9281    0.7847    0.8504      1152
           9     0.8622    0.8471    0.8546      1027

    accuracy                         0.9024     10000
   macro avg     0.9013    0.9046    0.9018     10000
weighted avg     0.9040    0.9024    0.9020     10000
```

Рис. 3.2. Результати тестування мережі LeNet.

Також слід відмітити самі метрики продуктивності. Модель класифікує 10 класів (наприклад, числа від 0 до 9). Для кожного класу розраховуються такі метрики:

1) Presicion(точність). Відсоток правильно класифікованих прикладів серед усіх передбачених прикладів для даного класу.
2) Recall(Відтворюваність). Відсоток правильно класифікованих прикладів серед усіх прикладів у класі
3) F1-Score. Середнє гармонійне значення точності та повноти.
4) Support (Підтримка): кількість прикладів у тестовому наборі для кожного класу.

Варто зазначити, що:

- Найвища точність: Клас 1: 97,71% - модель дуже добре класифікує приклади з цифрою 1.
- Найкраща відтворюваність: Клас 3: 94,51% - майже всі приклади з цифрою 3 розпізнаються правильно.
- Найкращий показник F1: Клас 1: 97,24% - модель добре балансує між точністю і неточністю.
- Найгірші показники: Клас 8: найнижчий показник F1 (85,04%) і повноти (78,47%). Це значить, що модель має труднощі з правильною ідентифікацією цифри 8.

Останньою метрикою є середні значення (AVG):

1) Macro avg (середнє по класах):
   - Precision: 90.13%
   - Recall: 90.46%
   - F1-score: 90.18%
2) Weighted avg (зважене середнє з урахуванням кількості прикладів у кожному класі):
   - Precision: 90.40%
   - Recall: 90.24%
   - F1-score: 90.20%

Ці результати показують, що модель збалансована в більшості класів, хоча в деяких класах (наприклад, 8) їй може бути складно розпізнати.

Отже, модель LeNet добре показала себе на тестових даних з точністю понад 90%. Найкращі результати спостерігаються для класів з більш простими рисами (наприклад, 1, 3), а найгірші - для більш складних форм (наприклад, 8). Це вказує на необхідність подальшого вдосконалення моделі або розширення навчальної вибірки для складних класів.

3.2. Аналіз застосування у різних сценаріях

Розгляд можливих областей застосування, переваг та обмежень моделей.

Можливі застосування моделей LeNet.

Завдяки своїй архітектурі та продуктивності, модель LeNet широко використовуються в таких додатках комп'ютерного зору, як: розпізнавання рукописного тексту: ідеально підходить для таких завдань, як розпізнавання номерів банківських чеків, поштових індексів і рукописного тексту, особливо корисні для завдань класифікації з низьким рівнем шуму завдяки своїй точності для цифр 1, 3 і 0.

Класифікація зображень у контрольованих середовищах.

Модель використовуватися в додатках автоматизації, де класи обмежені, а дані чітко структуровані (наприклад, маркування продукції на конвеєрних стрічках).

Переваги моделей LeNet.

- Простота: відносно проста архітектура дозволяє легко впроваджувати і швидко навчатися.
- Ефективність на невеликих наборах даних: висока точність на відносно невеликих навчальних вибірках, що робить її придатною для завдань з обмеженими ресурсами.

- Висока продуктивність для простих класів: ідеально підходить для класів з чіткою структурою даних і низьким рівнем шуму (наприклад, числа 1 і 3).
- Швидке тестування: невеликий розмір і проста архітектура дозволяють моделі працювати швидко навіть на слабких апаратних ресурсах.

Обмеження моделі LeNet.

Низька точність для складних класів: наприклад, точність для класу 8 значно низька (індекс F1 становить 85,04%). Це може обмежити застосування моделі для задач, де важливим є розпізнавання складних об'єктів або схожих класів.

- Вразливість до шуму та складних зображень: модель погано працює із зображеннями, що містять багато шуму та деформацій.
- Обмежена масштабованість: через свою просту архітектуру LeNet не може ефективно обробляти завдання з великою кількістю класів і даних високої роздільної здатності.
- Неуніверсальність: модель підходить для вузьких і спеціалізованих завдань, таких як розпізнавання символів, але може бути неефективною для більш складних завдань розпізнавання об'єктів.

Аналіз моделі LeNet показав її високу продуктивність у задачах розпізнавання символів, особливо в умовах добре структурованих даних і низького рівня шуму. Завдяки своїй простій архітектурі та обчислювальній швидкості модель підходить для використання в системах автоматичної класифікації, розпізнавання рукописного тексту та маркування товарів.

Водночас модель має обмежену масштабованість і низьку чутливість до складних зображень з високим рівнем шуму. Точність класифікації значно знижується для складних і схожих класів, що обмежує її використання в більш загальних задачах комп'ютерного зору. Тому, хоча LeNet залишається ефективним рішенням для спеціалізованих завдань, у більш складних

сценаріях краще використовувати більш просунуті моделі, такі як глибші ШНМ або архітектури перетворення.

У цьому розділі проаналізовано результати використання згорткових нейронних мереж (CNN) для розпізнавання символів. Якість класифікації оцінюється на основі метрик точності, повноти та F1. Також розглянуто вплив попередньої обробки даних на результати навчання, проаналізовано збіжність моделі та її поведінку на тестовому наборі.

ВИСНОВКИ

Під час дослідження моделей комп'ютерного зору для розпізнавання символів та виявлення об'єктів, автор проаналізував сучасні нейромережеві архітектури, їхню ефективність та сфери застосування. Були зроблені наступні висновки:

Ефективність та характеристики архітектури: результати експериментів підтверджують, що модель LeNet показує високу точність в задачах розпізнавання рукописного тексту, особливо цифр (середня точність 90,24%, метрика F1 для певних класів досягала 97,24%). Однак її простота обмежує її використання в більш складних завданнях, таких, як виявлення об'єктів у реальному часі та сегментація зображень.

Порівняння моделей: модель LeNet добре працює в контрольованих середовищах даних, таких як набір даних MNIST. Однак для задач, які вимагають обробки великих обсягів складних даних, краще використовувати сучасні архітектури з набагато кращою масштабованістю, точністю і універсальністю, такі як YOLO, Faster R-CNN і Vision Transformers.

Переваги мережі CNN, а саме моделі LeNet: проста архітектура, швидке навчання і тестування, роблять LeNet придатною для завдань, де висока продуктивність не є пріоритетом. Модель може бути ефективно використана в сценаріях, де кількість класів обмежена, а роздільна здатність вхідних даних низька.

Рекомендоване використання моделі LeNet найкраще підходять для освітніх цілей і завдань, що вимагають високої продуктивності і низьких ресурсів, таких як автоматизація простих операцій або аналіз рукописних документів.

Перспективи розвитку а отже і подальше вдосконалення архітектури нейронних мереж спрямоване на підвищення їх продуктивності та універсальності. Використання трансформаторів, гібридних підходів (наприклад, поєднання CNN і RNN) та інтеграція з методами навчання з

підкріпленням відкривають нові можливості для застосування комп'ютерного зору в різних галузях.

Підводячи підсумки треба сказати, що проведене дослідження показало, що вибір моделі комп’ютерного бачення залежить від специфіки задачі, доступних ресурсів і вимог до продуктивності. Модель LeNet, попри свої обмеження, є важливою базовою архітектурою, яка стала основою для розвитку сучасних, більш досконалих нейронних мереж.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Shashirangana J, Padmasiri H, Meedeniya D, Perera C (2020) Automated license plate recognition: a survey on methods and techniques. IEEE Access 9:11203–11225.
2. Roy S (2020) Automatics number plate recognition using convolutional neural network. Azerbaijan Int J High Perform Comput 234–244.
3. Real-Time Automatic Vehicle Management System Using Vehicle Tracking And Car Plate Number Iden- tification / Lee H., Kim D., Kim D., Bang S.Y. // ICME. – 2003. – № 7. – P. 353-356.
4. Acosta B.D. (2004) Experiments in Image Segmentation for Automatic US License Plate Recognition / B.D. Acosta // Thesis submitted to the Faculty of the Virginia Polytechnic Institute and State University in partial fulfill- ment of the requirements for the degree of MASTER OF SCIENCE in Computer Science. – 104 p.
5. Arshi Parvaiz, Muhammad Anwaar Khalid, Rukhsana Zafar, Huma Ameer, Muhammad Ali, Muhammad Moazam Fraz. (2023). Vision Transformers in medical computer vision – A contemplative retrospection. Engineering Applications of Artificial Intelligence. Retrieved from https://doi.org/10.1016/j.engappai.2023.106126 [in English]
6. Dhanya, V. G., Subeesh, A., Kushwaha, N. L., Dinesh Kumar Vishwakarma, Nagesh Kumar, T., Ritika, G., Singh. A. N. (2022). Deep learning based computer vision approaches for smart agricultural applications. Retrieved from https://doi.org/10.1016/j.aiia.2022.09.007 [in English].
7. 10.Zhong, Zhuoyao, Lei Sun, and Q. Huo. (2019). “Improved Localization Accuracy by LocNet for Faster R-CNN Based Text Detection in Natural Scene Images.” Pattern Recognition 96.
8. 11.A.Conci, J. E. R. de Carvalho, T. W. Rauber. (2009) A Complete System for Vehicle Plate Localization, Segmentation and Recognition in Real Life Scene, Ieee Latin America Transactions, vol. 7, no. 5, September.

9. Шпакович В. І. “Комп’ютерний зір” - алгоритми виявлення і розпізнавання об’єктів /Шпакович В. І., Погорєловська І.Д. //Цифрова трансформація фінансової системи України та країн V-4 в умовах євроінтеграції. Digital transformation of the financial system of Ukraine and v4 countries in the context of european integration. Збірник тез IV міжн. наук.-практ. Інтернет-конф. (Дубляни, 16 травня 2024 р.). Частина І. Дубляни: ЛНУП, 2024. 337 с. 361-365.
10. Оператор Собеля. URL: https://uk.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%BE%D1%80_%D0%A1%D0%BE%D0%B1%D0%B5%D0%BB%D1%8F
11. Оператор Кені. URL: https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%BE%D1%80_%D0%9A%D1%8D%D0%BD%D0%BD%D0%B8
12. Yann LeCun et al. "Gradient-based learning applied to document recognition." *Proceedings of the IEEE*, 1998: с. 2278–2284.
13. Joseph Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection," *CVPR*, 2016: с. 779–788.
14. Shaoqing Ren et al., "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *NeurIPS*, 2015: с. 91-99.
15. Alexey Dosovitskiy et al., "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," *ICLR*, 2021: с.5-7.
16. Wei Liu et al., "SSD: Single Shot MultiBox Detector," *ECCV*, 2016: с.21-37.
17. Emnist - Dataset. URL: https://www.tensorflow.org/datasets/catalog/emnist?hl=ua
18. GitHub: Code for generating syntetic text. URL: https://github.com/ankush-me/SynthText
19. Використовуємо CNN для обробки зображень. Частина перша. URL: https://dou.ua/forums/topic/48368/

20. Чичкарьов Є. А., Зінченко О. В., Балалаєва О. Ю. Розпізнавання рукописних українських літер та цифр з використанням синтетичного набору даних та згорткових нейронних мереж //ГРААЛЬ НАУКИ, №23 (2022). С. 241-253.
21. MNIST(базаданих). URL: https://uk.wikipedia.org/wiki/MNIST_(%D0%B1%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85)
22. Keras 3API Documentation. URL: https://keras.io/api/data_loading/image/?utm_source=chatgpt.com
23. Backpropagation in Neural Network. URL: https://www.geeksforgeeks.org/backpropagation-in-neural-network/
24. Introducion to Recurrent Neural Networks. URL: https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/?ref=header_outind
25. Еволюція архітектур нейромереж у комп'ютерному зорі. URL: https://habr.com/ru/companies/slsoft/articles/855602/
26. Introduction to Pooling Layer. URL: https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/?ref=header_outind
27. YOLOv4. URL: https://github.com/AlexeyAB/darknet
28. YOLO:YOU LOOK ONLY ONCE. URL: https://www.geeksforgeeks.org/yolo-you-only-look-once-real-time-object-detection/?ref=header_outind
29. Paramud Y., Yarkun V. Algorithmic and software means of handwritten symbol recognition // Bulletin of the National University “Lviv Polytechnic” “Computer Systems and Networks” No 881, Lviv, Ukraine. 2017. – P. 98–106.
30. Yarkun V., Paramud Y. Algorithmic and software synchronization of information exchange // Bulletin of the National University “Lviv Polytechnic” “Computer Systems and Networks” No 857, Lviv, Ukraine. 2016. – P.111–118.
31. Datong Chen, Jean-Marc Odobez, Herve Bourlard. Text detection and recognition in images and video frames // Pattern Recognition journal, 2003. – P. 595–608.
32. Convolutional Neural Network [Electronic resource]/ Stanford. – Access mode: http://ufldl.stanford.edu/tutorial /supervised/ConvolutionalNeuralNetwork/.
33. Prashanth, D., Mehta, R. V. K., Kadiyala, R. & Bhaskar, V. (2022). Handwritten devanagari character recognition using modified lenet and alexnet Convolution Neural Networks. Wireless Personal Communications, (122(2)).
34. Vovchuk, O. & Kyrychenko, M. (2019). Recognition of Handwritten Cyrillic Letters using PCA. URL: https://www.researchgate.net/publication/336987544_Recognition_of_Handwritten_Cyrillic_Letters_using_PCA
35. Cyrillic-oriented MNIST. (2022). URL: https://github.com/GregVial/CoMNIST
36. Khavalko, V., Mykhailyshyn, V., Zhelizniak, R., Kovtyk, I. & Mazur, A. (2020) Economic efficiency of innovative projects of CNN modified

architecture application. CEUR Workshop Proceedings. 1st International conference on cyber hygiene and conflict management in global information networks (CyberConf 2019) (pp. 182–193). November 30, 2019, Kyiv, Ukraine.URL: https://ceur-ws.org/Vol-2654/paper14.pdf

37. Rajavelu, A., Musavi, M. T. & Shirvaikar, M. V. (1989). A neural network approach to character recognition. Neural Networks, (2(5)), 387–393. URL: https://doi.org/10.1016/0893-6080(89)90023-3
38. Krizhevsky A., Sutskever I., Hinton G. ImageNet Classification with Deep Convolutional Neural Networks. Advances in Neural Information Processing Systems. 2012. Vol. 25, No 2. P. 1097.
39. Girshick R. Fast R-CNN. IEEE International Conference on Computer Vision : conference materials, Santiago, 7-13 Dec. 2015. Santiago, 2015. P. 1440.
40. Копча-Горячкіна Г. Е. ТЕОРІЯ РОЗПІЗНАВАННЯ ОБРАЗІВ : Посібник. Ужгород : ДВНЗ «Ужгор. нац. ун-ту», 2016. URL: https://org2.knuba.edu.ua/pluginfile.php/55687/mod_resource/content/1/Посібник-Теор-розпізн-образ-ч-I.pdf.
41. GeeksforGeeks. How single-shot detector (SSD) works? - GeeksforGeeks. *GeeksforGeeks*. URL: https://www.geeksforgeeks.org/how-single-shot-detector-ssd-works/?ref=header_outind

ДОДАТОК А

Набір даних	Кількість зображень	Розмір зображень	Класи	Основні сценарії використання
MNIST	70,000	28x28	10	Розпізнавання рукописних цифр
EMNIST	814,255	28x28	62	Розпізнавання цифр і букв
SynthText	800,000+	Різні	—	OCR для тексту на реальних зображеннях

Таблицю складено на основі джерел:[17, 18, 21]