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 другий (магістерський)

ДОСЛІДЖЕННЯ ВИКОРИСТАННЯ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ ДЛЯ ЗАДАЧІ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ
(тема)

Виконав:
студент 2 курсу, групи ІНФм-22-2
Котихін С.Д.
(прізвище, ініціали)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник проф. Машталір С.В.
(посада, прізвище, ініціали)

Допускається до захисту

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

2024 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти другий (магістерський)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Котихіну Станіславу Дмитровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження використання згорткових нейронних мереж для задачі класифікації зображень

затверджена наказом по університету від 3 листопада 2023 року № 1280Ст

2. Термін подання студентом роботи до екзаменаційної комісії _25__ грудня___ 2023 р.

3. Вихідні дані до роботи науково-методична та науково-технічна література, матеріали конференцій, дані інтернет-мережі, бібліотека глибокого навчання з відкритим кодом Keras, мова програмування Python, середовище розробки Google Colab, відкритий набір зображень автомобілів FER2013.

4. Перелік питань, що потрібно опрацювати в роботі
1. Проаналізувати існуючі проблеми для рішення яких застосовуються згорткові нейронні мережі.
2. Проаналізувати існуючі архітектури згорткових нейронних мереж для рішення існуючих задач.
3. Здійснити вибір інструментального засобу для програмної реалізації.
4. Реалізувати три нейронні мережі на різних архітектурах для вибраної задачі.
5. Порівняти та проаналізувати результати різних архітектур для вибраної задачі.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) актуальність проблеми класифікації зображень, постановка задачі, згорткова нейронна мережа, тестові зображення, перспективи подальшої роботи.

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

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	03.11.2023	
2	Аналіз завдання, підбір літератури	04.11.23-11.11.23	
3	Аналіз літератури з досліджуваної проблеми	11.11.23-22.11.23	
4	Аналіз проблеми класифікації зображень	23.11.23-28.11.23	
5	Дослідження принципів побудови згорткових нейронних мереж	29.11.23-07.12.23	
6	Програмна реалізація	08.12.23-12.12.23	
7	Оформлення пояснювальної записки	13.12.23-20.12.23	
8	Перевірка на плагіат	21.12.2023	
9	Рецензування	23.12.2023	
10	Підготовка презентації та доповіді	24.12.2023	
11	Занесення роботи в електронний архів	03.01.2024	
12	Попередній захист кваліфікаційної роботи	04.01.2024	

Дата видачі завдання 3 листопада 2023 р.

Студент _____
(підпис)

Керівник роботи _____ проф. Машталір С.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 61 с., 1 табл., 17 рис., 44 джерела.

КОМП'ЮТЕРНИЙ ЗІР, КЛАСИФІКАЦІЯ ЗОБРАЖЕНЬ, НЕЙРОННІ МЕРЕЖІ, ГЛИБОКЕ НАВЧАННЯ, ЗГОРТКА, ГРАДІЄНТНИЙ СПУСК, РОЗПІЗНАВАННЯ ЕМОЦІЙ, ТОЧНІСТЬ КЛАСИФІКАЦІЇ.

Об'єктом дослідження є набір зображень об'єднаних одною тематикою.

Метою дослідження є дослідження проблеми класифікації зображень за допомогою згорткових нейронних мереж для подальшого виявлення оптимальної реалізації для поставленої задачі.

У даному дослідженні використані методи інтелектуального аналізу даних, розпізнавання образів, градієнтного спуску, налаштування гіперпараметрів, та апарат оцінки продуктивності моделі. На основі дослідження і проведеного експерименту були виявлені сильні та слабкі сторони різних архітектур згорткових нейронних мереж для задачі класифікації на реальних даних.

В ході дослідження здійснена програмна реалізація порівняння різних архітектур нейронних мереж на різних наборах даних.

COMPUTER VISION, IMAGE CLASSIFICATION, NEURAL NETWORKS, DEEP LEARNING, CONVOLUTION, GRADIENT DESCENT, EMOTION RECOGNITION, CLASSIFICATION ACCURACY.

The object of research is a set of images united by a topic.

The purpose of the research is to explore the problem of image classification using convolutional neural networks to determine the most appropriate implementation for the task.

The methods used in this research are data mining, pattern recognition, gradient descent, hyperparameter tuning, and a model performance evaluation apparatus. Based on the study and the experiment, the strengths and weaknesses of different convolutional neural network architectures for the task of classifying real data were identified.

In the process of the research, a software implementation of the comparison of different neural network architectures on different data sets was implemented.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

CNN – Convolutional Neural Network (згорткова нейронна мережа)

FER – Facial Expression Recognition (задача розпізнавання емоцій на обличчі)

VR – Virtual Reality (віртуальна реальність)

МРТ – магнітно-резонансна томографія

ШНМ – штучна нейронна мережа

ALPR – Automated License Plate Recognition (автоматизоване розпізнавання номерних знаків)

GPU – Graphics Processing Unit (графічний процессор)

NAS – Neural Architecture Search (проєктування архітектури нейронних мереж)

MCC – Matthews Correlation Coefficient (кореляційний коефіцієнт Метью)

ВСТУП

За останні кілька десятиліть комп’ютерний зір набув величезної популярності та цінності завдяки своєму трансформаційному потенціалу в автоматизації та вдосконаленні завдань, що покладаються на візуальні дані [1]. Ця еволюція значною мірою зумовлена згортковими нейронними мережами (CNN) [2, 3]. Вони унікально пристосовані для класифікації зображень завдяки своїй здатності автоматично вивчати просторові ієрархії ознак.

Дизайн згорткових нейронних мереж передбачає виділення локальних ознак за допомогою згорткових шарів, які виявляють зразки, починаючи від простих країв і закінчуючи складними структурами в міру поглиблення мережі. Включення шарів об’єднання зменшує розмірність даних, підвищуючи обчислювальну ефективність і надаючи інваріантності незначним змінам зображення.

Крім того, їхня глибина полегшує наскрізне навчання, усуваючи необхідність ручного виділення ознак, що є основним елементом традиційної обробки зображень. Ця глибина в поєднанні з адаптивністю завдяки навчанню з перенесенням та інтеграції з іншими нейронними архітектурами робить CNN найкращим вибором для класифікації зображень у сфері глибокого навчання.

Використовуючи цю унікальну архітектуру CNN, яка імітує людську зорову систему, для класифікації зображень, промисловість і дослідники можуть ефективно обробляти величезні обсяги візуальних даних, уможливлюючи прорив у багатьох життєво-важливих сферах, тим самим стимулюючи інновації та пропонуючи рішення, які колись вважалися недосяжними [4].

Оскільки ця технологія продовжує розвиватися, адаптуватися і диверсифікуватися, важливість систематичного оцінювання їх можливостей неможливо переоцінити. Як для дослідників, так і для практиків вкрай важливо проводити ретельну і сувору оцінку CNN, щоб визначити їхню

ефективність, сильні сторони і потенційні можливості для вдосконалення в широкому діапазоні завдань і сценаріїв класифікації зображень.

Актуальність дослідження полягає у всебічному аналізі існуючих архітектур згорткових нейронних мереж на задачах класифікації зображення, виявлення сильних сторін кожної та підкреслення вад, які можна вдосконалити в майбутньому.

1 ОГЛЯД ОСНОВНИХ МЕТОДІВ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ ЗА ДОПОМОГОЮ CNN

1.1 Задача класифікації зображень

Класифікація зображень у контексті комп'ютерних наук означає завдання автоматичного присвоєння мітки або категорії зображенню на основі його візуального змісту. Це фундаментальна задача в галузі комп'ютерного зору. Мета полягає в тому, щоб навчити модель, використовуючи набір даних з маркованих зображень, так, щоб при пред'явленні нового, раніше не баченого зображення, модель могла точно передбачити категорію або клас, до якого це зображення належить.

До появи глибинного навчання класифікація зображень в основному покладалася на ручне вилучення ознак із зображень. Ці методи, як правило, включали двоетапний процес: вилучення ознак і класифікацію. Такі методи, як масштабно-інваріантне перетворення ознак (SIFT) і гістограма орієнтованих градієнтів (HOG), були популярними для виявлення основних шаблонів і структур на зображеннях. Після того, як ці ознаки будуть вилучені, класифікатори машинного навчання, такі як машини опорних векторів (SVM) або випадкові ліси, будуть використані для класифікації зображень. Ці методи, хоча й ефективні для деяких завдань, мали обмеження в роботі з великою варіативністю та складнощами, притаманними більш складним наборам даних зображень [5].

Ландшафт класифікації зображень зазнав трансформаційних змін з появою глибокого навчання, зокрема, згорткових нейронних мереж (CNN). Вони, натхненні зоровою системою людини, використовують ієрархічні шари згорткових фільтрів для автоматичного навчання зображень з даних, усуваючи потребу в ручному створенні ознак. Її рішення стимулюють прогрес у таких сферах, як розпізнавання обличь, що має значення для безпеки і персональної ідентифікації, медична візуалізація, де раннє і точне виявлення захворювань

може врятувати життя [6], і автономні транспортні засоби, де класифікація об’єктів у реальному часі має вирішальне значення для безпечної навігації. Крім того, розвиток соціальних мереж і платформ електронної комерції вимагає ефективних систем категоризації та рекомендацій, які значною мірою покладаються на класифікацію зображень [7]. Вирішуючи цю проблему, люди не лише розширюють межі того, що можуть сприймати і розуміти машини, але й каталізують інновації у багатьох галузях, що робить її ключовим викликом з далекосяжними наслідками [8-10].

Сфера класифікації зображень пов’язана з проблемами, що випливають з мінливості та складності візуальних даних. Найважливішою з цих проблем є мінливість зовнішнього вигляду об’єктів через такі фактори, як освітлення, поза, масштаб і оклюзія, які можуть суттєво впливати на продуктивність алгоритмів класифікації. Крім того, фонові перешкоди та інші об’єкти на зображенні можуть заплутати системи класифікації, що призведе до неточної ідентифікації цільового об’єкта. Масштабованість є ще однією проблемою, оскільки зростаючий обсяг даних зображень збільшує труднощі в управлінні та ефективній обробці. Крім того, дисбаланс класів, коли деякі класи мають більше навчальних прикладів, ніж інші, може спотворити процес навчання в бік надмірного представлення певних класів, що призводить до упередженості або неточності моделей [11, 12].

Перенесення моделей на нові або небачені набори даних є критичною проблемою в класифікації зображень. Моделі, які добре працюють на одному наборі даних, можуть виявитися нездатними узагальнити інші набори даних через відмінності в розподілі або якості даних. Іншим аспектом є інтерпретованість і пояснюваність моделей класифікації зображень, особливо моделей глибокого навчання, таких як згорткові нейронні мережі (CNN), які часто розглядаються як моделі «чорного ящика». Забезпечення інтерпретованості цих моделей для експертів предметної області має вирішальне значення для їх прийняття і застосування в реальних сценаріях. Нарешті, вимоги до обчислювальних ресурсів і пам’яті для навчання і

розгортання складних моделей класифікації зображень можуть бути непомірно високими, особливо в середовищах з обмеженими ресурсами або на периферійних пристроях, що створює значні труднощі на шляху до практичного впровадження рішень для класифікації зображень [13, 14].

Вищезгадані проблеми підкреслюють складну природу класифікації зображень, виділяючи її як дуже актуальну і сучасну проблему в галузі комп'ютерного зору. Широкий спектр перешкод, починаючи від варіабельності даних і масштабованості до перенесення моделей і обчислювальних вимог, демонструє благодатний ґрунт для подальших досліджень та інновацій. Більше того, потенціал для покращення ефективності моделей класифікації зображень, особливо в галузевих застосуваннях, підкреслює перспективний шлях для розвитку як теоретичного розуміння, так і практичного застосування класифікації зображень.

1.1.1 Задача розпізнавання емоцій

Задачу розпізнавання емоцій на обличчі (FER) можна вважати, мабуть, одним з найпоширеніших випадків використання класифікації зображень в сучасному світі. Основною метою є виявлення та інтерпретація тонких сигналів у зовнішньому вигляді людини, які вказують на певні емоційні стани, такі як щастя, смуток, гнів, здивування, страх і відраза.

Емоції розпізнаються на основі виразу обличчя, мови тіла. Перший крок передбачає визначення ключових орієнтирів обличчя, таких як очі, брови, ніс, рот і лінія підборіддя. Ці риси слугують орієнтирами для аналізу рухів та змін обличчя.

Після виявлення рис обличчя система аналізує відносне положення і рух цих рис, щоб ідентифікувати конкретні вирази обличчя. Наприклад, підняті брови і широко відкриті очі можуть вказувати на здивування, а опущені губи – на смуток (рис. 1.1).

Рисунок 1.1 – Приклад розпізнавання емоцій

Так чином, розпізнавання емоцій в сфері інтерактивних розваг відкрило інноваційні шляхи для створення захоплюючих і персоналізованих вражень для користувачів. Деякі відеоігри використовують розпізнавання емоцій, щоб оцінити рівень розчарування чи нудьги гравця за виразом його обличчя. Якщо гравець виглядає розчарованим, гра може знизити рівень складності, а якщо йому нудно – додати більш складні елементи.

У VR розпізнавання емоцій можна використовувати для покращення занурення. Наприклад, VR-гра у жанрі жахів може виявити, коли гравець справді наляканий, і посилити моторошні елементи, або послабити їх, якщо гравець надто перевантажений. В іграх із доповненою реальністю, де реальний світ змішується з віртуальним, розпізнавання емоцій можна використовувати для запуску певних ігрових подій. Наприклад, гра з пошуком скарбів у доповненій реальності може виявити приховані скарби, коли гравець демонструє хвилювання.

Розпізнавання емоцій також є потужним інструментом маркетингу [15]. Воно допомагає зрозуміти поведінку та вподобання споживачів. Перед запуском великої рекламної кампанії маркетологи можуть використовувати розпізнавання емоцій, щоб оцінити реакцію фокус-групи. Аналізуючи вираз

обличчя глядачів, компанії можуть визначити, які частини реклами викликають позитивні емоції, а які потребують коригування.

Наприклад в 2015 році міжнародна компанія з виробництва продуктів харчування Kellogg’s скористалась програмним забезпеченням з розпізнавання емоцій Affectiva. Їх ціллю було визначення найбільш привабливої рекламу для своїх пластівців «Crunchy Nut» [16].

Компанія створила кілька версій рекламного ролика для пластівців, у кожному з яких фігурують різні тварини. Одна з версій реклами, в якій була зображена змія, викликала найбільше сміху у глядачів. Однак, коли ті самі глядачі дивилися цю рекламу вдруге, рівень залученості знизився. В іншій версії реклами був зображений інопланетянин. Програмне забезпечення для розпізнавання обличь виявило, що ця версія підтримувала стабільний рівень залученості навіть при повторних переглядах.

Спираючись на ці дані, Kellogg’s вирішила запустити рекламу з інопланетянином замість версії зі змією. Це рішення було продиктоване бажанням мати рекламу, яка б не лише привертала увагу глядачів на початковому етапі, але й підтримувала їхній інтерес протягом тривалого часу. Через рік компанія випустила звіт, що обрана реклама відіграла важливу роль у збільшенні продажів пластівців «Crunchy Nut».

Крім прикладів застосування технології виявлення емоцій для цілей бізнесу, є також приклади її використання для навчання. Деякі платформи електронного навчання інтегрують розпізнавання емоцій, щоб адаптувати контент відповідно до емоційного стану студента. Наприклад, якщо студент виглядає розчарованим або розгубленим, система може спростити контент або надати додаткові ресурси. І навпаки, якщо студенту нудно, система може запропонувати більш складні завдання [17].

У віртуальних класах вчителі можуть використовувати інструменти розпізнавання емоцій, щоб оцінити залученість учнів. Якщо учень виглядає відвернутим або незацікавленим, вчитель може вжити заходів, щоб привернути його увагу або зв’язатися з ним після уроку.

Але треба зауважити, що при розпізнавання емоцій можуть виникати труднощі, в першу чергу через складну природу людських емоцій та їх різноманітні прояви. Такі фактори, як культурні або індивідуальні відмінності в емоційному вираженні та тонкощі міміки обличчя можуть ускладнити точне розпізнавання. Крім того, зовнішні змінні, такі як умови освітлення, перешкоди на обличчі та роздільна здатність зображення, можуть перешкоджати роботі алгоритмів розпізнавання.

1.1.2 Задача розпізнавання захворювань

Одразу після популяризації комп’ютерного зору науковці почали тестувати можливість його застосування в сфері медицини. В умовах глобального зростання кількості хронічних захворювань і постійної загрози нових інфекційних хвороб, своєчасна і точна діагностика має першорядне значення. Тому ця задача розпізнавання стала все дуже цінною у сучасній системі охорони здоров’я завдяки величезному потенціалу.

Програми розпізнавання можуть аналізувати величезні обсяги даних медичних зображень зі швидкістю, недосяжною для людини, що дозволяє виявляти тонкі закономірності і аномалії, які можуть бути пропущені людським оком. Такий швидкий аналіз може призвести до швидшого втручання, зменшення прогресування захворювань і покращення результатів лікування пацієнтів [18].

Крім того, оскільки системи охорони здоров’я в усьому світі борються з такими проблемами, як нестача лікарів, збільшення навантаження на пацієнтів і зростання витрат, автоматизоване розпізнавання хвороб пропонує масштабоване і економічно ефективне рішення. Допомагаючи медичним працівникам у діагностиці, комп’ютерний зір може полегшити частину навантаження на перевантажений роботою медичний персонал, гарантуючи, що пацієнти отримають своєчасну допомогу.

Більшість рішень розпізнавання захворювань базуються на медичних знімках, таких як рентген, комп’ютерна томографія або МРТ. Ці зображення слугують вхідними даними для алгоритмів нейромереж. Вони навчаються на маркованих наборах даних, де кожне зображення асоціюється з певним захворюванням або станом. З часом ці моделі вчяться розпізнавати зразки та ознаки, що вказують на різні аномалії, такі як пухлини, переломи або ознаки захворювань.

На даний момент, незважаючи на велику популярність нейромереж, програми з розпізнавання захворювань не використовуються широко в медичному світі. Однією з головних перешкод є потреба у великій кількості маркованих навчальних даних. Для навчання моделей медичні зображення потребують точних анотацій, часто зроблених експертами-радіологами або медичними працівниками. Збір таких високоякісних анотованих наборів даних може зайняти багато часу, коштувати дорого, а іноді і зовсім неможливий через проблеми з конфіденційністю пацієнтів. Крім того, медичні зображення можуть сильно відрізнятися за якістю, роздільною здатністю і методами візуалізації, що вносить варіабельність, яку повинна враховувати модель. Ця різноманітність може впливати на узгодженість вхідних даних, що робить складним для ШНМ добре узагальнювати різні джерела зображень і умови.

Іншою важливою проблемою є інтерпретованість моделей. У медичних застосуваннях важливо мати не лише модель, яка може робити точні прогнози, але й модель, процес прийняття рішень якої буде зрозумілим для клініцистів. Нейромережі, будучи складними моделями глибокого навчання, часто діють як «чорні скриньки», через що важко зрозуміти, як вони приходять до певного діагнозу. Така непрозорість може призвести до проблем з довірою серед медичних працівників, які покладаються на ці інструменти для прийняття критично важливих рішень. Крім того, ставки в медичній діагностиці неймовірно високі, і навіть незначні помилки можуть мати серйозні наслідки. Забезпечення стійкості, надійності та валідності моделей у різноманітних

реальних клінічних умовах залишається першочерговим завданням у цій галузі.

1.1.3 Задача розпізнавання номерних знаків автомобілей

У сучасному світі, що стрімко урбанізується, можливість автоматичного розпізнавання автомобільних номерних знаків також стала дуже важливою. По-перше, вона відіграє ключову роль у посиленні безпеки та правопорядку. Влада може використовувати системи автоматичного розпізнавання номерних знаків (ALPR) для моніторингу та відстеження транспортних засобів, що становлять інтерес, виявлення викрадених автомобілів або забезпечення дотримання правил дорожнього руху. Наприклад, у випадках наїздів на людей, викрадень чи інших злочинних дій, системи ALPR можуть швидко сканувати величезні обсяги відеозаписів, щоб визначити місцезнаходження конкретного транспортного засобу, допомагаючи в розслідуванні та потенційному розкритті злочинів. Крім того, ці системи можуть бути інтегровані з базами даних, щоб миттєво позначати транспортні засоби, які беруть участь у незаконній діяльності або не відповідають правилам дорожнього руху, забезпечуючи безпеку на вулицях [19].

Крім безпеки, системи автоматичного розпізнавання номерних знаків мають практичне застосування, які спрощують та автоматизують повсякденні процеси, що призводить до підвищення ефективності. Наприклад, при зборі плати за проїзд ручні методи можуть спричинити затори на дорогах і є трудомісткими. За допомогою ж розпізнавання номерних знаків транспортні засоби можуть автоматично стягувати плату при проїзді через ворота, що скорочує час очікування і втручання людини. Аналогічно, в управлінні паркуванням ALPR можна використовувати для перевірки наявності дійсного дозволу на паркування, автоматичного нарахування плати за тривалість паркування або виявлення несанкціонованих транспортних засобів. Крім того,

підприємства можуть використовувати ALPR для логістики та управління автопарком, забезпечуючи своєчасну доставку та ефективне планування маршрутів. По суті, можливість автоматичного розпізнавання автомобільних номерних знаків стала наріжним каменем у сучасному транспорті та управлінні містом, пропонуючи рішення, які є ефективними та масштабованими.

Сам процес розпізнавання номерних знаків зазвичай має декілька послідовних етапів. Спочатку стоїть задача виявлення транспортного засобу. Для цього використовуються такі алгоритми, як YOLO або Faster R-CNN, для визначення місцезнаходження транспортного засобу на зображенні або відеокадрі. Потім номерний знак локалізується за допомогою методів обробки зображень або моделей глибокого навчання, що гарантує точне місцезнаходження номерного знаку.

Після локалізації виконується сегментація символів для виділення кожного символу на номерному знаку, використовуючи алгоритми сегментації для поділу номера на окремі області. Потім виконується розпізнавання символів за допомогою моделей глибокого навчання, таких як згорткові нейронні мережі (CNN), для інтерпретації кожного символу. Розпізнані символи потім обробляються і використовуються для різних додатків, таких як відстеження транспортних засобів або контроль доступу, шляхом порівняння розпізнаного номера з базою даних. Приклад роботи програми продемонстрований на рисунку 1.2.

З іншого боку, завдання розпізнавання автомобільних номерів, незважаючи на технологічний прогрес, стикається з кількома проблемами, які можуть перешкоджати його точності та ефективності. Однією з головних є різноманітність номерних знаків, які можуть бути усіляких форм, розмірів, кольорів та дизайну, залежно від регіону чи країни. Таке розмаїття вимагає надійної моделі, здатної впоратися з безліччю форматів номерних знаків. Крім того, значною перешкодою є оклюзія, оскільки номерні знаки можуть бути

частково закриті брудом, наклейками або іншими об’єктами, що ускладнює точне розпізнавання символів системою.

Рисунок 1.2 – Приклад розпізнавання номерних знаків

Фактори навколишнього середовища також ускладнюють завдання розпізнавання номерних знаків. Різноманітні умови освітлення, такі як яскраве сонячне світло, тіні або нічні сценарії, можуть негативно вплинути на видимість і читабельність номерного знаку. Щоб забезпечити стабільну роботу, система повинна вміти справлятися з цими варіаціями. Крім того, кут огляду, під яким знімається зображення, може спотворювати символи на номерному знаку, що вимагає складних алгоритмів для корекції перспективи і точної ідентифікації символів. Розмиття при русі, особливо на зображеннях автомобілів, що швидко рухаються, також може призвести до розмиття символів, що вимагає застосування передових технологій для покращення зображення та підвищення точності розпізнавання.

Для вирішення цих проблем розробка надійної системи розпізнавання номерних знаків потребує застосування передових методів обробки зображень і глибокого навчання, яким і є згорткові нейронні мережі.

1.2 Огляд існуючих архітектур конволюційних нейронних мереж

У динамічній сфері глибокого навчання згорткові нейронні мережі (CNN) вирішували найрізноманітніші завдання класифікації. Тому поява та розвиток різних архітектур CNN значною мірою зумовлені постійним прагненням підвищити продуктивність та адаптивність для конкретної задачі комп'ютерного зору. Різні додатки, від класифікації зображень і виявлення об'єктів до семантичної сегментації, створюють унікальні перешкоди і вимагають специфічних обчислювальних і репрезентативних можливостей. Наприклад, якщо простіші завдання, такі як розпізнавання цифр, можуть вправно вирішуватися відносно простими архітектурами, то складніші завдання, такі як виявлення кількох об'єктів різного масштабу та орієнтації, потребують більш складних і надійних архітектур [20].

Більше того, технологічний ландшафт, особливо з точки зору обчислювальних ресурсів і апаратного забезпечення, постійно розвивається, що дозволяє розробляти глибші і складніші архітектури CNN. Поява графічних процесорів (GPU) і прогрес у паралельних обчисленнях полегшили навчання більших мереж, сприяючи дослідженню архітектур з більшою кількістю рівнів і параметрів. Як наслідок, дослідники отримали можливість експериментувати і розробляти нові архітектури, які можуть вивчати більш складні функції та ієрархічні представлення даних, тим самим покращуючи прогностичну ефективність моделі.

Крім того, прагнення зменшити перенавчання, покращити узагальнення та підвищити обчислювальну ефективність також відіграло важливу роль у формуванні різних архітектур згорткових нейронних мереж. Різні архітектури впроваджують інноваційні механізми та структури для пом'якшення таких проблем, як зникаючі та вибухові градієнти, які переважають у глибоких мережах.

Таким чином, безліч архітектур CNN відображає об'єднання зусиль, спрямованих на підвищення ефективності прогнозування при одночасному

подоланні викликів і обмежень, пов’язаних з різними завданнями і технологічними обмеженнями.

1.2.1 AlexNet

AlexNet, розроблена Алексом Крижевським, Іллею Суцкевером та Джеффрі Хінтоном, є піонерською згортковою нейронною мережею (CNN) [21], яка суттєво вплинула на сферу глибокого навчання, зокрема, комп’ютерного зору. Представлена в 2012 році, вона була розроблена для класифікації зображень на 1000 різних категорій з великого набору даних ImageNet і стала моделлю-переможцем ImageNet Large Scale Visual Recognition Challenge (ILSVRC) в 2012 році і досягнула чудового показника помилок у 16,4%, що значно нижче, ніж у попередніх моделей. Ця архітектура продемонструвала, що глибоке навчання, зокрема ШНМ, можна використовувати для вилучення ієрархічних ознак з пікселів необроблених зображень, вивчення складних репрезентацій і досягнення чудової продуктивності в задачах візуального розпізнавання. Модель вправно справлялася з проблемами класифікації зображень, такими як обробка різних положень об’єктів, масштабів і спотворень, шляхом навчання надійних і ієрархічних представлень ознак за допомогою своєї глибокої і складної мережевої структури. Таким чином, AlexNet стала каталізатором подальших досліджень і розробок у галузі глибокого навчання та комп’ютерного зору, проклавши шлях до більш складних і надійних моделей у наступні роки.

Сама архітектура моделі безпосередньо складається з восьми шарів, з п’ятьма згортковими шарами і трьома повністю з’єднаними шарами. На вхід мережі подається картинка 224×224 пікселів, яка проходить через перший згортковий шар з 96 ядрами розміром 11×11 та кроком в 4 пікселі (рис. 1.3). Наступні шари включають подальші згортки, операції максимального об’єднання та кроки нормалізації, поступово зменшуючи просторові розміри

та збільшуючи глибину. Варто зазначити, що мережа використовує випрямлені лінійні одиниці (ReLU) як функції активації, що було новим підходом на той час і сприяло її швидшому навчанню. Останні шари мережі є повністю з'єднаними шарами, що призводить до отримання м'якого максимуму на виході для 1000 класів.

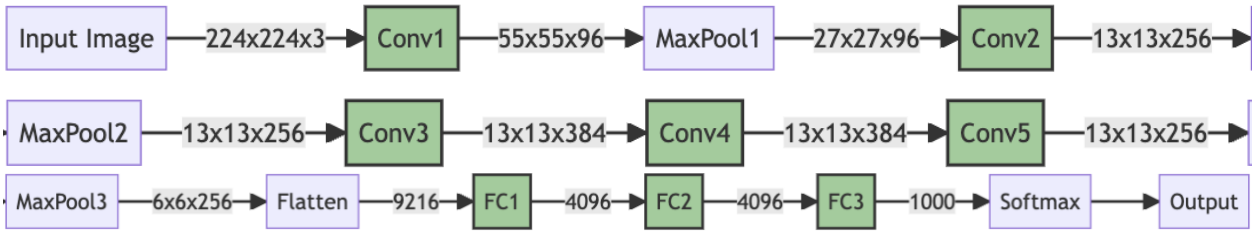

Рисунок 1.3 – Архітектура нейронної мережі AlexNet

Ця архітектура також вирізняється використанням обчислень на графічних процесорах для навчання, використовуючи можливості паралельних обчислень на графічних процесорах для управління процесом навчання, що вимагає значних обчислень. Навчання мережі проводилося з використанням двох графічних процесорів Nvidia GeForce GTX 580, причому модель була розділена між двома графічними процесорами, щоб врахувати велику кількість параметрів. Таке інноваційне використання апаратного забезпечення прискорило навчання глибоких мереж, проклавши шлях для розробки ще більших і складніших моделей у наступні роки. Успіх і методологія AlexNet з тих пір надихнули численні досягнення і архітектури в області глибокого навчання.

1.2.2 VGG-Net

Архітектура VGG, розроблена групою Visual Geometry Group в Оксфордському університеті [22], виділяється своєю глибиною і простотою. Представлена на конкурсі ILSVRC-2014 і досягнув рівня помилок 7,3%, VGG

значно розширила межі комп’ютерного зору, продемонструвавши, що глибина мережі має вирішальне значення для хорошої продуктивності в задачах класифікації зображень. Архітектура поставляється в різних варіантах, найпопулярнішими з яких є VGG16 і VGG19, що позначають моделі з 16 і 19 ваговими шарами відповідно. Мережі VGG характеризуються використанням невеликих згорткових фільтрів 3×3, що дозволяє мережі вивчати складні ієрархічні ознаки зображень, зберігаючи при цьому відносно низьку обчислювальну складність.

Мережі VGG мають просту архітектурну схему: серія згорткових шарів, кожен з яких використовує фільтри 3×3, за якими слідує шар з максимальним об’єднанням, і ця схема повторюється, поступово зменшуючи просторові розміри карт ознак і збільшуючи їхню глибину. Зокрема, шари згортки використовують фільтри 3×3 з кроком 1, а за ними йдуть функції активації ReLU. Об’єднуючі шари зазвичай мають розмір 2×2 з кроком 2. Після кількох згорточних і об’єднуючих шарів мережа вирівнює вихідні дані і пропускає їх через три повністю з’єднані шари, причому на останньому шарі використовується функція активації softmax для виведення класифікації для всіх класів. Глибина і простота VGG роблять його універсальною архітектурою, яку можна застосовувати для різних завдань аналізу зображень, окрім класифікації, таких як виявлення об’єктів і сегментація зображень (рис. 1.4).

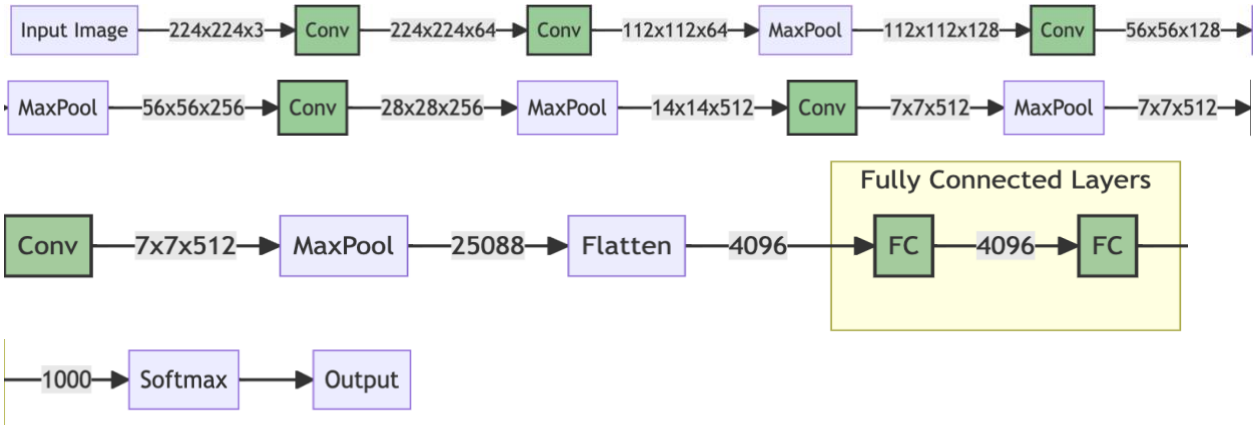

Рисунок 1.4 – Спрощений вигляд архітектури нейронної мережі VGG16

Незважаючи на свою глибину, VGG викликає захоплення своєю простотою і однорідністю. Використання невеликих фільтрів фіксованого розміру по всій мережі спрощує процес налаштування гіперпараметрів і дозволяє дослідникам і практикам зосередитися на інших аспектах проектування та навчання мережі. Однак варто зазначити, що мережі VGG, як правило, є більш пам’ятко ємними порівняно з іншими архітектурами через використання повністю з’єднаних шарів, що може бути обмеженням для певних додатків. Тим не менш, VGG отримала широке розповсюдження у спільноті глибокого навчання і слугує базовою моделлю для різноманітних застосувань та досліджень у галузі комп’ютерного зору.

1.2.3 ResNet

Архітектура ResNet (залишкова мережа), представлена Каймінгом Хе, Сянью Чжаном, Шаоціном Реном та Цзянь Суном [23], зробила значний прорив у галузі глибокого навчання, уможлививши навчання нейронних мереж з надзвичайно великою глибиною. Вона була розроблена для вирішення проблем, пов’язаних з навчанням надзвичайно глибоких нейронних мереж. Коли мережі стають глибшими, вони, як правило, стикаються з проблемою зникаючого/вибухаючого градієнта, коли градієнти або зменшуються майже до нуля, або вибухають до величезних значень під час зворотного поширення, що робить мережу важко оптимізованою. Ця проблема була значною перешкодою у навчанні глибоких мереж і обмежувала їх продуктивність та застосування. ResNet ефективно вирішує цю проблему, дозволяючи навчати мережі з безпрецедентною глибиною і досягати чудової продуктивності для різних завдань і наборів даних.

Архітектура характеризується інноваційним використанням «залишкових блоків» або «пропускних зв’язків», які дозволяють мережі оминати один або кілька шарів під час прямого та зворотного проходження

навчання. Це досягається шляхом додавання входу блоку шарів до його виходу, що математично виражається як:

$$H(x) = F(x) + x, \tag{1.1}$$

де $F(x)$ – це результат шару блоку;

$H(x)$ – фінальний результат;

x – вхід блоку.

Цей механізм пропускання шарів гарантує, що мережа може навчатися функціям ідентичності, що, в свою чергу, допомагає пом'якшити проблему зникаючого та вибухаючого градієнта, яка є типовою для глибоких мереж.

Архітектури ResNet бувають різної глибини, включаючи ResNet-18, ResNet-34, ResNet-50, ResNet-101 і ResNet-152, із зазначенням кількості шарів у кожному варіанті. Архітектура починається з шару згортки і максимального об'єднання, за яким слідує серія залишкових блоків, і завершується глобальним середнім об'єднанням і повністю з'єднаним шаром. Кожен залишковий блок містить згорткові шари, функції пакетної нормалізації та активації ReLU, а також з'єднання, що пропускають, які оминають один або декілька шарів (рис. 1.5). Глибина мережі дозволяє ResNet вивчати широкий спектр функцій, від низькорівневих до високорівневих, що робить її здатною виконувати складні завдання класифікації та досягати вражаючих результатів на різних бенчмарках.

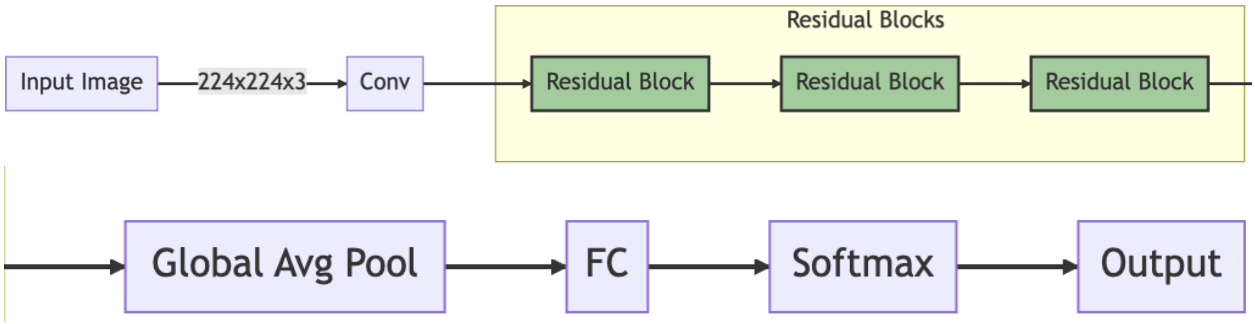

Рисунок 1.5 – Спрощений вигляд архітектури ResNet

Впровадження ResNet мало глибокий вплив на сферу комп’ютерного зору і не тільки, надаючи засоби для навчання більш глибоких мереж і досягнення більш високої точності в різних завданнях, включаючи класифікацію зображень, виявлення об’єктів і семантичну сегментацію. Принцип залишкового навчання, запроваджений ResNet, був широко прийнятий і розширений у різних наступних архітектурах глибокого навчання, ставши основоположною концепцією в розробці нейронних мереж для численних застосувань у різних галузях штучного інтелекту.

1.3 Аналіз літературних джерел щодо існуючих підходів класифікації зображень за допомогою CNN.

Усі розглянуті вище архітектури стали «класичними» в світі глибокого навчання. Їх беруть за основу в більшості найсучасніших архітектур нейронних мереж і модель конфігурується під більш конкретну задачу класифікації. Зараз буде розглянуто деякі з них.

Однією з задач у другій половині десятиріччя було розробки можливості роботи згорткових нейронних мереж на мобільних девайсах. Як наслідок, у 2017 році у статті [24] Говарда та ін. була представлені MobileNets – сімейство невеликих, малопотужних мобільних моделей комп’ютерного зору для TensorFlow з низьким рівнем затримок, розроблених для ефективної максимізації точності, параметри яких налаштовані так, щоб задовольнити ресурсні обмеження для різноманітних випадків використання. Вони можуть бути використані для класифікації, виявлення, вбудовування і сегментації, і призначені для ефективної роботи з різними додатками технічного зору на мобільних і вбудованих пристроях. Архітектура використовує згортки, що розділяються за глибиною, які значно зменшують кількість параметрів і обчислень у мережі, а також вводить два гіперпараметри, які ефективно компромісують між затримкою і точністю.

Модель вводить два нових глобальних гіперпараметри: множник ширини та множник роздільної здатності, які дозволяють легко масштабувати модель з точки зору обчислювальних потреб. Ці гіперпараметри забезпечують систематичний спосіб компромісу між розміром моделі, обчислювальними ресурсами і точністю класифікації, дозволяючи дослідникам і практикам створювати моделі, пристосовані до обмежень їхніх конкретних застосувань. MobileNets демонструють надійну роботу в широкому діапазоні застосувань і використовуються в різних програмах технічного зору, таких як виявлення об’єктів, дрібнозерниста класифікація, атрибути обличь і великомасштабна геолокалізація, забезпечуючи хороший компроміс між затримкою і точністю, які можна налаштувати відповідно до вимог різних застосувань.

Ще одним досить популярним існуючим підходом є інтеграція зовнішніх знань, таких як семантична або ієрархічна інформація про класи, в CNN для підвищення ефективності класифікації, особливо в сценаріях з обмеженою кількістю маркованих даних.

Одними з піонерів цього підходу є Ф.Рен та Т.Ши [25] зі своєю роботою про використання зовнішніх знань для покращення семантики при виявленні емоцій у розмові. Автори пропонують метод, який включає зовнішні знання в попередньо навчену мовну модель, щоб покращити її ефективність у виявленні емоцій у розмовах. Цей метод передбачає використання графа знань для вилучення слів, пов’язаних з емоціями, а потім інтеграцію цієї інформації в модель. Підхід оцінюється за допомогою різних наборів даних, і результати показують, що він перевершує базові методи, демонструючи ефективність використання зовнішніх знань у завданнях розпізнавання емоцій.

З іншого ж боку, є підхід (NAS) до автоматизованого процесу проектування архітектури нейронних мереж, який є критично важливим аспектом глибокого навчання. Традиційні методи проектування мереж часто включають ручне налаштування та експертні знання, що може зайняти багато часу і бути неоптимальним. NAS, з іншого боку, використовує алгоритми пошуку (такі як навчання з підкріпленням, еволюційні алгоритми або

градієнтні методи), щоб дослідити простір можливих мережевих архітектур і автоматично знайти ту, яка найкраще підходить для певного завдання, наприклад, класифікації зображень або мовного моделювання. Цей підхід спрямований на виявлення мережевих архітектур, які забезпечують вищу продуктивність, а також потенційно зменшують обчислювальні ресурси і час, необхідний для розробки моделі. NAS успішно створює найсучасніші моделі в різних галузях, перевершуючи архітектури, розроблені вручну.

Одна з найкращих робіт на цю тему Ченсі Лю, Баррета Зофа та ін. [26] представляє новий підхід до пошуку нейронної архітектури (NAS), який є більш ефективним та результативним. Автори представляють метод під назвою Progressive Neural Architecture Search (PNAS), який використовує послідовну стратегію оптимізації на основі моделей. Ключова ідея полягає в побудові сурогатної моделі, яка передбачає метрику оцінки архітектури нейронної мережі без її навчання. Сурогатна модель навчається на архітектурах, які поступово ускладняються, і використовується для пошуку кращих архітектур.

1.4 Постановка задачі дослідження

Таким чином, дослідження використання згорткових нейронних мереж для задачі класифікації зображень є актуальним завданням глибокого навчання. Тому ставиться завдання виявлення усіх сильних та слабких сторін різних архітектур згорткових нейронних мереж для задачі класифікації на реальних даних.

Об’єктом дослідження є набір зображень об’єднаних одною тематикою.

Метою дослідження є дослідження проблеми класифікації зображень за допомогою згорткових нейронних мереж для подальшого виявлення оптимальної реалізації для поставленої задачі.

Для досягнення мети необхідно вирішити такі завдання:

– провести аналіз існуючих підходів глибокого навчання та архітектур згорткових нейронних мереж;

– розробити порівняльну програму для цих підходів та архітектур з різними наборами даних;

– зробити аналіз результатів порівняння і виявити усі слабкі та сильні сторони кожного підходу або архітектури.

2 МОДЕЛІ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ

2.1 Принципи побудови згорткової нейронної мережі

Розглянемо метод побудови згорткової нейронної мережі. Зазвичай відбувається класифікація кольорових зображень, тому кожне кольорове зображення розбирається на схему 3 різних кольорів – червоного, зеленого, синього. Кожна схема – це матриця значень розміру зображення, кожен піксель буде мати своє значення:

$$\begin{matrix} x_{1,1} & x_{1,2} & \cdots & x_{1,j} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,j} \\ \vdots & \vdots & \vdots & \vdots \\ x_{i,1} & x_{i,2} & \cdots & x_{i,j} \end{matrix}, \tag{2.1}$$

де x – піксель;

$i = 1, 2, \ldots, K$;

$j = 1, 2, \ldots, M$;

K, M – розміри вхідного зображення.

Перед початком побудови мережі по крокам, треба ознайомитися з архітектурою згорткової нейронної мережі на рисунку 2.1, яка буде побудована.

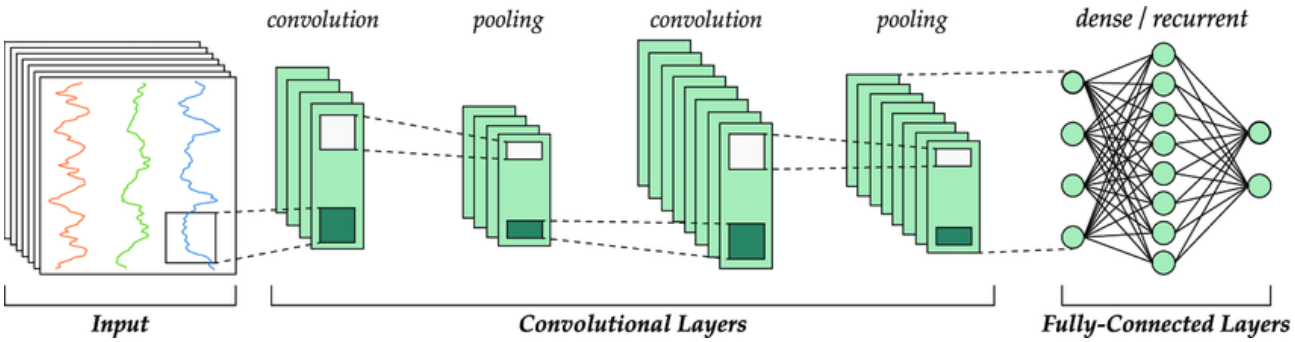

Рисунок 2.1 – Архітектура згорткової нейронної мережі

Першим кроком згорткової нейронної мережі є застосування згортки до зображення [27]. Згортка – це операція виявлення певних ознак на невеликій ділянці зображення. Ідея використання згортки полягає у виявленні локальних особливостей у вхідних даних, таких як краї, текстури або специфічні шаблони. Це робиться за допомогою фільтра або, іншими словами, ядра згортки. Ядро – це зазвичай невелика матриця 3×3, 5×5, 7×7 пікселів, головною метою якої є виявлення одної особливості зображення:

$$\begin{matrix} \omega_{1,1} & \omega_{1,2} & \cdots & \omega_{1,j} \\ \omega_{2,1} & \omega_{2,2} & \cdots & \omega_{2,j} \\ \vdots & \vdots & \vdots & \vdots \\ \omega_{i,1} & \omega_{i,2} & \cdots & \omega_{i,j} \end{matrix}, \tag{2.2}$$

де ω – вага;

$i = 1, 2, \ldots, S$;

$j = 1, 2, \ldots, N$;

S, N – розміри ядра.

Наприклад, така матриця 3×3 буде виявляти тільки вертикальні лінії на ділянці:

$$\begin{matrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{matrix}. \tag{2.3}$$

З математичної точки зору, операція згортки – це скалярний добуток матриць зображення та ядра. Пересуваючи це ядро по матриці зображення на кожній позиції відбувається множення значення в ядрі на вихідні значення пікселів на зображенні, яке воно покриває.

$$\nu_{k,m} = \sum_{i=1}^{S} \sum_{j=1}^{N} x_{i+k,j+m} \cdot \omega_{i,j} + \omega_0 , \tag{2.4}$$

де $\omega_{i,j}$ – значення ваги в конкретному елементі ядра;

$i = 1, 2, \ldots, S$;

$j = 1, 2, \ldots, N$;

S, N – розміри ядра;

$x_{i+k,j+m}$ – значення пікселя в конкретному елементі ядра;

$k = 1, 2, \ldots, K$;

$m = 1, 2, \ldots, M$;

K, M – розміри зображення;

ω_0 – bias.

Ще одним важливим елементом підрахування згортки є bias. Це додатковий параметр нейронної мережі, який використовується для коригування виходу разом зі зваженою сумою входів нейрона. Це схоже на зсув у лінійному рівнянні. Він дозволяє нейрону гнучко формувати вихідні сигнали. Без зміщення вихід нейрона повністю залежав би від його входу. Це означає, що якщо вхід дорівнює нулю, вихід також буде нульовим. Ввівши bias, навіть коли вхідний сигнал дорівнює нулю, нейрон може виробляти ненульовий вихід [27].

Після додавання bias, значення в результуючій матриці проходять через функцію активації ReLU. Це означає, що будь-яке від’ємне значення в матриці встановлюється в нуль, а додатні значення залишаються незмінними (рис. 2.2). Незважаючи на те, що ReLU виглядає як лінійна функція, вона вносить нелінійність у модель. Це дозволяє моделі вчитися на помилках і вносити корективи, що дуже важливо для вивчення складних закономірностей [28, 29]. Це останній крок операції згортки, який на виході буде давати матрицю вихідних характеристик.

$$f(\nu_{ij}) = \max(0, \nu_{ij}), \tag{2.5}$$

де ν_{ij} – це елемент матриці згортки.

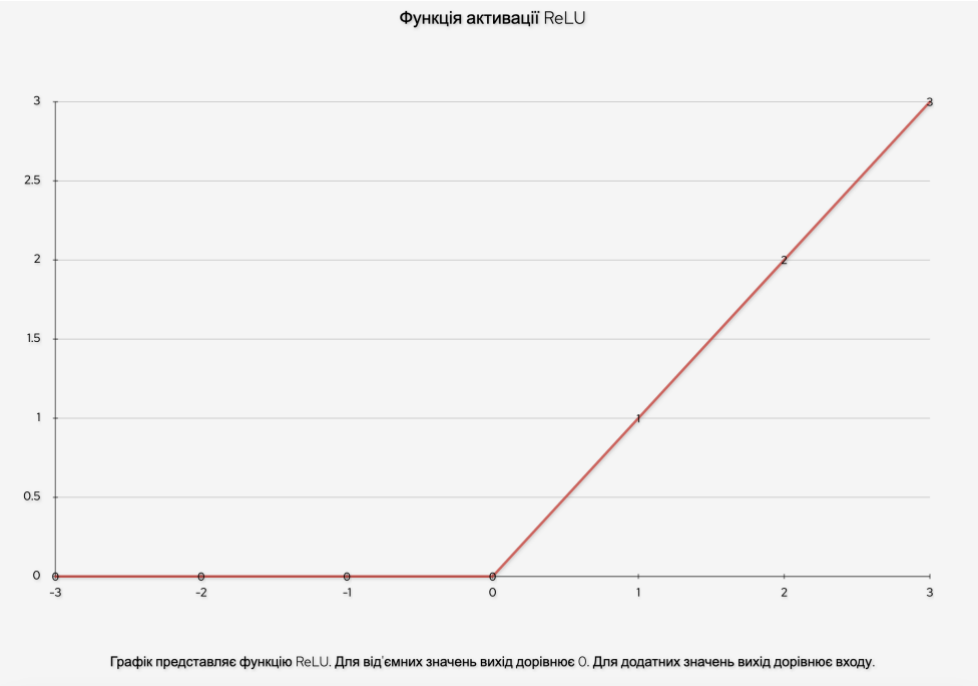

Рисунок 2.2 – графічне зображення функції активації ReLU

Весь процес згортки можна представити в виді блокової діаграми на рисунку 2.3.

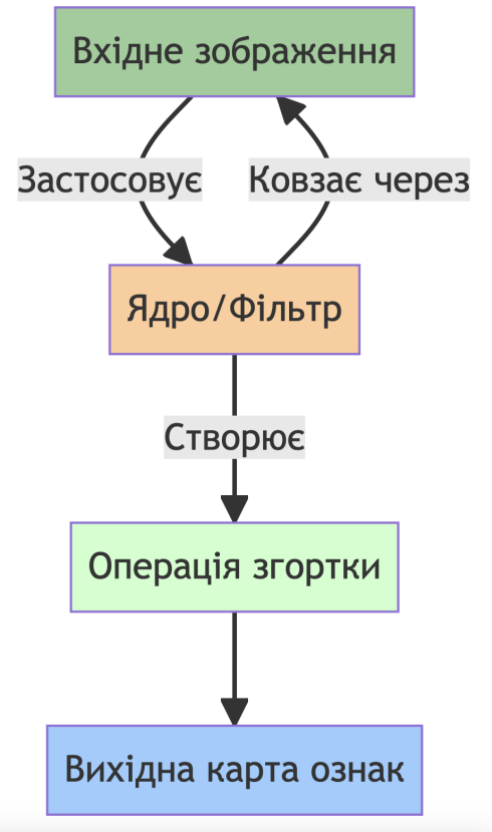

Рисунок 2.3 – Блокова діаграма операції згортки

Після операції згортки, до вихідних значень в згортковій нейронній мережі буде застосовуватися операція MaxPooling:

$$P_{ij} = max_{a=0}^{n-1} max_{b=0}^{n-1} F_{i+a,j+b}\ , \tag{2.6}$$

де i = 1, 2, …, K;

j = 1, 2, …, M;

K, M – розміри матриці характеристик;

a = 1, 2, …, n;

b = 1, 2, …, n;

n – розмір просторової області операції MaxPooling;

$F_{i+a,j+b}$ – значення в $i+a, j+b$ позиції матриці характеристики.

Вона передбачає взяття просторової області (наприклад, області 2×2 або 3×3) з матриці характеристик і взяття максимального значення цієї області для отримання єдиного вихідного значення [28, 29]. Цю операцію зазвичай застосовують у режимі ковзного вікна, подібно до операції згортки (рис. 2.4).

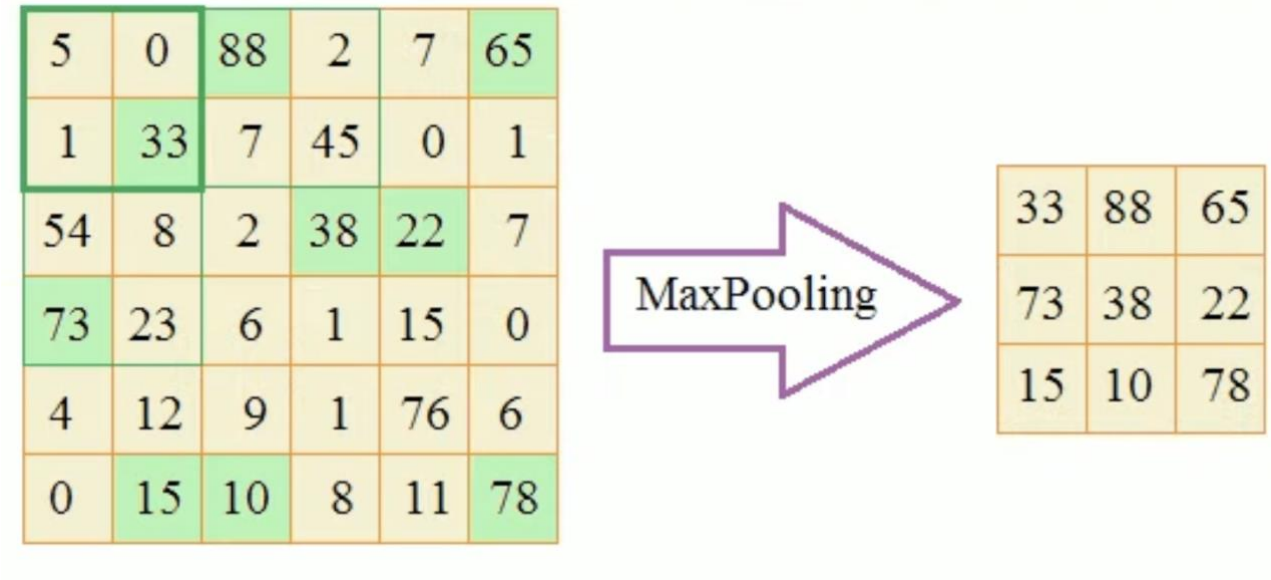

Рисунок 2.4 – Візуалізація операції MaxPooling

Ці кроки можуть повторюватися декілька разів для більш дрібного розбиття вхідного зображення на характеристики [30]. В даній моделі 2 повторення. Після цього, всю цю інформацію треба подати в повнозв'язну нейронну мережу. Але перед тим, привести в зрозумілий для неї вигляд.

На виході з операцій згортки та MaxPooling буде велика кількість багатовимірних матриць, з яких треба зробити одновимірний вектор, адже повнозв'язна нейронна мережа має саме такий вхідний тип даних. Ця операція називається Flatten (рис. 2.5).

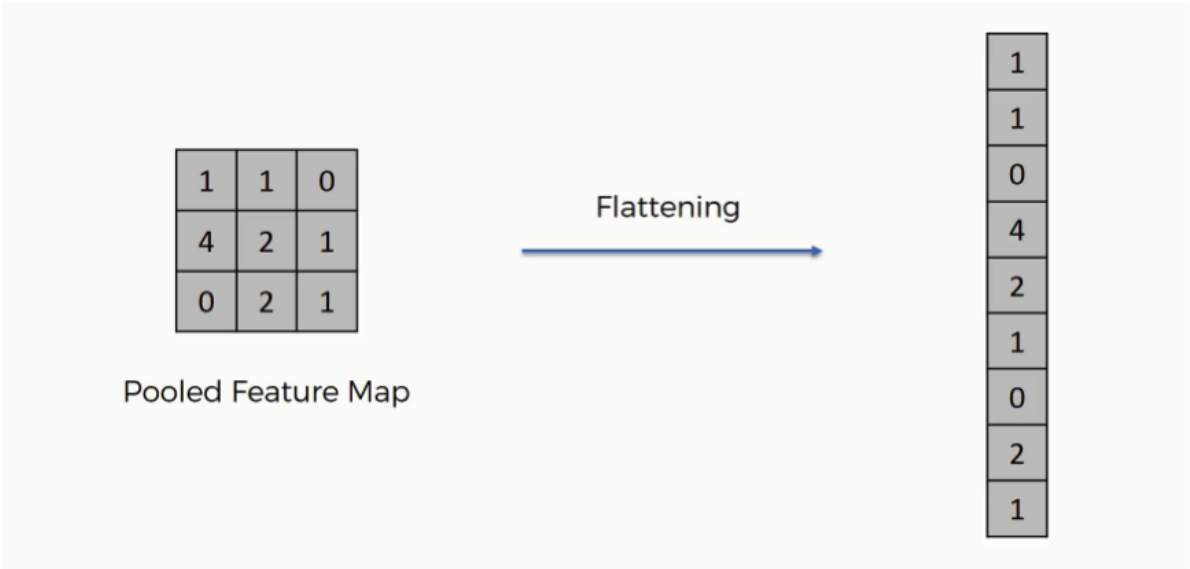

Рисунок 2.5 – Операція Flatten на матриці 3×3

Кожен нейрон у повнозв'язній мережі з'єднаний з кожним нейроном у попередньому шарі. Ось чому він називається «повнозв'язним». Кожен нейрон у повністю повноз'вязному шарі обчислює зважену суму всіх своїх входів. Якщо ми розглянемо один нейрон:

$$v_k = \sum_{i=1}^{S} x_i \cdot \omega_i + \omega_0 , \qquad (2.7)$$

де ω_i – значення ваги в конкретному нейроні;

x_i – значення нейрона;

$i = 1, 2, \ldots, S$;

S – кількість нейронів попереднього слою;

$k = 1, 2, \ldots, K$;

K – кількість нейронів поточного слою;

ω_0 – bias.

Для мережі з декількома нейронами та входами це можна представити у вигляді матриці:

$$\mathrm{H} = \mathrm{X} \cdot W + b\ , \tag{2.8}$$

де X – матриця значень нейронів;

W – матриця ваг всіх попередніх нейронів;

b – вектор значень bias.

Після обчислення зваженої суми, вона пропускається через активаційну функцію для введення нелінійності. Це функція ReLU (2.5), така ж сама, що була застосована після операції згортки. В даному випадку вона буде застосована для матриці H. Такий алгоритм примінюється до кожного слою окрім останнього.

Підрахування останнього слою повнозв'язної нейронної мережі повторює крок з підрахуванням зваженої суми, але застосовується інша функція активації – Softmax. Ця функція слугує для перетворення сирих значень балів у ймовірності для кожного класу [28]. Функція гарантує, що сума ймовірностей для всіх класів дорівнює 1.

$$Softmax(H_j) = \frac{e^{H_j}}{\sum_{k=1}^{K} e^{H_k}}\ , \tag{2.9}$$

де H_j – сирі значення (перед активацією) балів j-го класу;

$j, k = 1, 2, \ldots K$;

K – кількість класів класифікації;

e – основа натурального логарифму.

На виході ми отримаємо ймовірності приналежності зображення до кожного класу. Це буде кінцевим результатом роботи нейронної мережі.

Це модель згорткової нейронної мережі, яка буде застосована у практичній частині наукового дослідження.

2.2 Принципи навчання згорткової нейронної мережі

Згорткові нейронні мережі (CNN) призначені для автоматичного та адаптивного навчання просторових ієрархій ознак на основі вхідних зображень. Навчання, або тренування, ШНМ передбачає коригування його внутрішніх параметрів на основі даних, до яких він отримує доступ, що дозволяє йому робити точні прогнози або класифікації. Без навчання ШНМ не могли би розрізняти різні візуальні ознаки або розпізнавати об’єкти на зображеннях, що робить його неефективним для таких завдань, як класифікація зображень, виявлення об’єктів або будь-яких інших завдань, що базуються на зоровому аналізі.

Навчання CNN передбачає доступ до великого набору даних з міченими зображеннями, що дозволяє мережі ітеративно коригувати свої ваги та зсуви, щоб мінімізувати різницю між її прогнозами та фактичними результатами. Цей процес, покращує розуміння мережею особливостей даних. Тренуючись, CNN краще виокремлює та розпізнає ієрархічні ознаки, від простих ребер до складних структур, що дозволяє їй виконувати поставлене завдання з високою точністю.

Процес навчання буде відбуватися за алгоритмом зворотного поширення (backpropagation) [31]. Систематично обчислюючи градієнт функції втрат для кожної ваги за допомогою ланцюгового правила, зворотне поширення забезпечує механізм коригування ваг і зміщень мережі в напрямку, що зменшує похибку. Цей ітеративний процес налаштування дозволяє мережі

навчатися та адаптуватися до основних закономірностей у даних, тим самим підвищуючи її точність прогнозування та здатність до узагальнення.

Підготовчим кроком буде генерування випадкових ваг для нейронів та їх відхиленя (bias). Ці дані згодом будуть мінятися і коригуватися в ході навчання. Після цього можна приступати до навчання.

Перший крок називається «прямий прохід» або «forward pass» англійською. Прямий прохід у нейронній мережі передбачає обробку вхідних даних через кожен шар мережі послідовно для отримання вихідних даних. Починаючи з вхідного шару, дані проходять через взаємопов'язані нейрони, де вони проходять серію лінійних перетворень (зважені суми вхідних даних плюс зсуви) з подальшою нелінійною активацією. Коли дані проходять кожен шар, ці операції виділяють і трансформують ознаки, завершуючись на останньому шарі, де мережа виробляє своє передбачення [32].

Фактично «прямий прохід» – це просто передбачення нейронної мережі з поточними параметрами, алгоритм якого був описаний разом з формулами в попередньому розділі. Результат передбачення потім порівнюється з фактичними цільовими значеннями для обчислення втрат, які вимірюють точність прогнозування мережі для цього конкретного входу.

Обчислення втрат – це наступний крок навчання. Для цього буде застосована функція втрат крос-ентропія. Існує декілька варіацій цих функцій, але тут буде використана варіація для мультикласифікації – для трьох чи більше класів. Формула функції категоріальної крос-ентропії може бути представлена так:

$$\mathrm{L}(y, \hat{y}) = -\sum_{i=1}^{N} \sum_{j=1}^{K} y_{ij} \log \hat{y}_{ij} , \tag{2.10}$$

де y_{ij} – бінарний індикатор чи клас j вірний для зразка i;

$\hat{y}_{ij}$ – передбачена ймовірність нейронної мережі, що зразок i належить до класу j;

$i = 1, 2 \ldots N$;

N – кількість зразків;

$j = 1, 2 \ldots K$;

K – кількість класів класифікації.

Ця функція втрат є мірою розбіжності між прогнозами мережі та фактичними результатами даних.

Тепер, знаючи розбіжності між результатами, можна зробити налаштування параметрів. Для цього буде застосований алгоритм «зворотного поширення». Він передбачає обчислення градієнта функції втрат щодо кожної ваги та зміщення в мережі, який показує, наскільки зміняться втрати, якщо кожен параметр зміниться на невелику величину [33].

Для різних шарів мережі формули трохи відрізняються, тому треба почати з вихідного шара, він буде перший при проході в зворотньому напрямку. Градієнт для вихідного шару може бути підрахований так:

$$\frac{\partial L}{\partial H_j} = \hat{y}_j - y_j \,, \tag{2.11}$$

де H_j – сирі значення (перед активацією) балів j-го класу (2.9);

y_j – бінарний індикатор чи клас j вірний для поточного зразка;

$\hat{y}_{ij}$ – передбачена ймовірність нейронної мережі, що поточний зразок належить до класу j;

$j = 1, 2 \ldots K$;

K – кількість класів класифікації;

L – функція втрат.

Після вихідного шара йдуть приховані шари. Підрахунок градієнта для них виконується правилом диференціювання складної функції або ланцюговим правилом і ділиться, в данному випадку, на три етапи підрахунків – для значень після активації функцією ReLU, для сирих значень до застосування функції активації та для ваг і відхилень (bias). Формула для

підрахунку першого етапу, значень після активації може бути представлена таким чином:

$$\frac{\partial L}{\partial a_i} = \sum_j \frac{\partial L}{\partial H_j} \times \frac{\partial H_j}{\partial a_i}, \tag{2.12}$$

де a_i – результат після активації нейрона i;

i – індекс поточного нейрону;

j = 0, 1, … S;

S – всі нейрони в шарі після i-го нейрону.

Формула градієнту втрат по відношенню до зваженого входу (до активації) нейрона виглядає таким чином:

$$\frac{\partial L}{\partial H_i} = \frac{\partial L}{\partial a_i} \times f'(H_i), \tag{2.13}$$

де $f'(H_i)$ – похідна функції активації за її входом.

Останнім етапом підрахунку градієнта є його підрахунок для ваг та відхилень, як вже зазначалось вище. Така формула для ваг може бути записана як:

$$\frac{\partial L}{\partial \omega_{ij}} = \frac{\partial L}{\partial H_j} \times a_j, \tag{2.14}$$

де ω_{ij} – вага з'єднувального нейрона j у попередньому шарі з нейроном i в поточному.

А для відхилення (bias):

$$\frac{\partial L}{\partial b_i} = \frac{\partial L}{\partial H_i}, \tag{2.15}$$

де b_i – відхилення (bias) для нейрону i.

Як можна помітити, кожна формула зав’язана на результаті попередньої, саме тому це називається ланцюговим правилом.

В (2.11) – (2.15) була описана одна ітерація «зворотного поширення», тобто для одного шару нейронної мережі. Починаючи з вихідного шару і рухаючись ітеративно назад до вхідного шару, можна обчислити градієнти.

Ці підраховані градієнти тепер треба якось оптимізовано застосувати. Для цього буде застосован алгоритм стохастичного градієнтного спуску [34]. Це модифікація стандартного алгоритму градієнтного спуску. Він, як і базовий алгоритм, буде ітеративно підлаштовувати параметри моделі в напрямку найкрутішого зниження функції втрат, але використовувати одну випадково обрану точку даних на кожній ітерації, що призводить до швидшого оновлення що часто веде до більш нестабільної траєкторії збіжності, але швидших ітерацій, що особливо вигідно для великих наборів даних.

Описаний вище процес «зворотного поширення» – це один крок алгоритму стохастичного градієнтного спуску. Як вже було зазначено в (2.14), (2.15), останній крок підраховує ваги та відхилення. Як відомо, градієнт показує зміну втрат при невеликій зміні параметра, тому цей крок можна узагальнити як:

$$\Delta_\theta L = \frac{\partial L}{\partial \theta}, \tag{2.16}$$

де θ – параметр до коригування, в даному випадку ω або b.

Після цього алгоритм оновлює цей параметр за такою формулою:

$$\theta_{new} = \theta_{old} - \alpha \Delta_\theta L, \tag{2.17}$$

де θ_{new} – нове значення параметру;

θ_{old} – старе значення параметру;

α – швидкість навчання.

Швидкість навчання – це такий гіперпараметр в алгоритмах оптимізації, який визначає розмір кроку для мінімізації функції втрат. Він контролює, наскільки параметри моделі (в цьому випадку, ваги та зміщення) коригуються у відповідь на обчислений градієнт втрат. Висока швидкість навчання може призвести до більших кроків, що, в свою чергу, може призвести до перевищення оптимального рішення, тоді як низька швидкість навчання призводить до менших, обережніших кроків, які можуть повільно збігатися або застрягати в локальних мінімумах.

Для досягнення балансу між швидкою збіжністю та стабільністю до швидкості навчання буде застосоване експоненціальне спадання. Спочатку, коли модель далека від оптимального рішення, більша швидкість навчання допомагає швидко покращити результат. Однак, коли модель наближається до мінімуму функції втрат, корисно зменшити швидкість навчання, щоб уникнути промаху і для точного налаштування параметрів. Експоненціальне спадання поступово зменшує швидкість навчання з часом, дозволяючи моделі більш надійно збігатися до оптимального рішення, а також отримуючи вигоду від швидшого початкового прогресу. Його можна обчисляти кожен крок за формулою:

$$\alpha = \alpha_0 e^{-k \times epoch}, \tag{2.18}$$

де α_0 – початкове значення параметру, яке може бути вибрано випадково;

k – швидкість згасання;

$epoch$ – поточна епоха.

В формулі (2.18) з’являється ще один гіперпараметр – швидкість згасання. Він визначає, наскільки стрімко швидкість навчання знижується з часом. Більші значення призводять до швидшого спаду швидкості навчання.

Таким чином, повторюючи всі описані вище кроки протягом декількох епох або до тих пір, поки втрати не зійдуться до мінімального значення, CNN тренується розпізнавати зображення за певними ознаками. Цей метод

навчання згорткової нейронної мережі буде застосовуватися в практичній частині дослідження.

2.3 Опис подальшого процесу моделювання програмного застосунку та оцінювання ефективності класифікації зображень

В даній роботі буде реалізовано програмну класифікацію зображень на основі трьох популярних архітектур CNN – AlexNet, VGG16, ResNet.

Алгоритм класифікації буде проходити за наступними кроками:

Крок 1. Обирання датасету для тестування. Це може бути власноруч розроблений, або вже існуючий в просторі мережі датасет. В даній роботі будуть використовуватися декілька готових тематичних датасетів, які підходять для дослідження.

Крок 2. Підготовка та нормалізація даних для тренування згорткової нейронної мережі.

Крок 3. Навчання CNN за (2.10) – (2.18) використовуючи алгоритм стохастичного градієнтного спуску.

Крок 4. Тестування нейронної мережі на тестовій вибірці та оцінювання її ефективності за допомогою різних метрик, таких як F1-score.

Крок 5. Повторення Кроків 2-4 на інших архітектурах згорткових нейронних мереж.

Крок 6. Порівняння ефективності між різними архітектурами на обраному датасеті.

F1-Score – це показник, який поєднує в собі точність і відтворення, щоб забезпечити єдину міру продуктивності моделі. Це середнє гармонійне значення точності та відтворення, і обидва показники мають однакову важливість [35, 36].

У сценаріях, де і хибно-позитивні, і хибно-негативні результати є критичними, F1-Score допомагає збалансувати їх. Наприклад, у медичній

діагностиці пропуск позитивного результату (хибно-негативний результат) і помилкова ідентифікація здорової людини як хворої (хибно-позитивний результат) мають значні наслідки.

У випадках, коли один клас недостатньо представлений, точність може вводити в оману. Наприклад, у наборі даних з 95% негативних і 5% позитивних результатів, наївний класифікатор, який прогнозує всі негативні результати, матиме точність 95%, але повністю пропустить клас позитивних результатів. F1-Score може бути більш інформативним у таких сценаріях.

Формула підрахунку метрики F1-Score має такий вигляд:

$$F1 = 2 \times \frac{TP}{2 \times TP + FP + FN}, \tag{2.19}$$

де TP – кількість вірно-позитивних виборів моделі;

FP – кількість хибно-позитивних виборів моделі;

FN – кількість хибно-негативних виборів моделі.

Показник F1-Score коливається від 0 до 1. Вищий показник F1-Score вказує на кращу роботу моделі, де 1 – ідеальна точність, а 0 – найгірша.

3 КОМП'ЮТЕРНА МОДЕЛЬ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ

3.1 Вибір інструментальних засобів для реалізації поставленої задачі

У рамках дослідження був розроблений програмний застосунок для наочного порівняння ефективності класифікації зображень різних архітектур згорткових нейронних мереж на приклади задачі розпізнавання емоцій.

Для реалізації поставленої задачі була обрана мова програмування Python з використанням бібліотек tensorflow, sklearn, matplotlib. Jupiter Notebook був обраний як середовище розробки, а апаратне забезпечення представлено хостинговим сервісом Google Colab.

Python виділяється як найкраща мова для глибокого навчання та розробки нейронних мереж насамперед завдяки своїй простоті та зрозумілості. Її інтуїтивно зрозумілий синтаксис сприяє швидкій розробці і створенню прототипів, що є важливою перевагою для такого роду задач, які часто потребують швидких ітерацій на своїх моделях [37, 38].

Крім того, Python може похвалитися всеосяжною екосистемою, повною бібліотек і фреймворків, пристосованих для маніпулювання даними, аналізу та візуалізації. Потужна підтримка спільноти посилює її привабливість, гарантуючи, що рішення, найкращі практики та ресурси є легкодоступними.

Одна з прикладів дуже просунутої екосистеми є бібліотека TensorFlow. Вона була розроблена командою Google Brain і є однією з провідних бібліотек з відкритим вихідним кодом для чисельних обчислень і машинного навчання. Її гнучка архітектура дозволяє користувачам розгортати обчислення на різних платформах, від CPU і GPU до мобільних і вбудованих систем. TensorFlow надає широкий набір інструментів і бібліотек, які задовольняють як початківців, з високорівневими API, такими як Keras, так і експертів, яким може знадобитися більш тонкий контроль над деталями моделі. Автоматичне диференціювання, критичний компонент для навчання нейронних мереж, є вбудованим, що дозволяє розробникам зосередитися на архітектурі моделі, не

заглиблюючись в обчислення. Крім того, TensorFlow пропонує TensorBoard, інструмент візуалізації, який допомагає зрозуміти, налагодити та оптимізувати нейромережеві моделі. Нарешті, потужна спільнота TensorFlow та регулярні оновлення гарантують, що бібліотека залишається на передньому краю досягнень у галузі глибокого навчання.

Треба зазначити, що у цієї мови програмування є суттєвий недолік – це швидкість виконання коду, порівняно з конкурентами. Це обумовлено низкою причин, серед яких є динамічна типізація, інтерпретатор та інші. Але в випадку будування нейронної мережі швидкість виконання коду не є важливим фактором, тому цей недолік ніяк не вплине на остаточну програму.

Jupyter Notebook пропонує інтерактивне обчислювальне середовище, в якому користувачі можуть поєднувати код, візуалізацію та описовий текст. Це робить його особливо придатним для завдань глибокого навчання, оскільки розробники можуть створювати, тестувати та вдосконалювати свої моделі ітеративно в межах одного документа. Негайний зворотній зв’язок від виконаних клітинок допомагає швидко знаходити несправності та коригувати модель. Крім того, можливість візуалізації даних і результатів безпосередньо в блокноті спрощує попередній аналіз даних та інтерпретацію результатів моделювання. Представляючи як основний код, так і його результати, Jupyter Notebooks сприяють всебічному розумінню робочих процесів глибокого навчання, роблячи процес навчання і розробки більш інтуїтивно зрозумілим і прозорим (рис. 3.1).

Google Colab розширює можливості Jupyter Notebooks, надаючи хмарну платформу з безкоштовним доступом до графічних та обчислювальних процесорів, важливих апаратних прискорювачів для завдань глибокого навчання. Користувачі можуть легко масштабувати свої обчислення, не інвестуючи в дорогу апаратну інфраструктуру. Безшовна інтеграція з Google Drive забезпечує легку співпрацю, спільне використання та зберігання ноутбуків, що підходить як для індивідуальних розробників, так і для команд. Підхід з нульовим налаштуванням означає, що користувачі можуть миттєво

розпочати розробку та навчання моделей глибокого навчання без будь-якої попередньої конфігурації. Поєднання потужних обчислювальних ресурсів, функцій для спільної роботи та простого у використанні інтерфейсу робить Google Colab безцінним інструментом для вирішення складних завдань глибокого навчання.

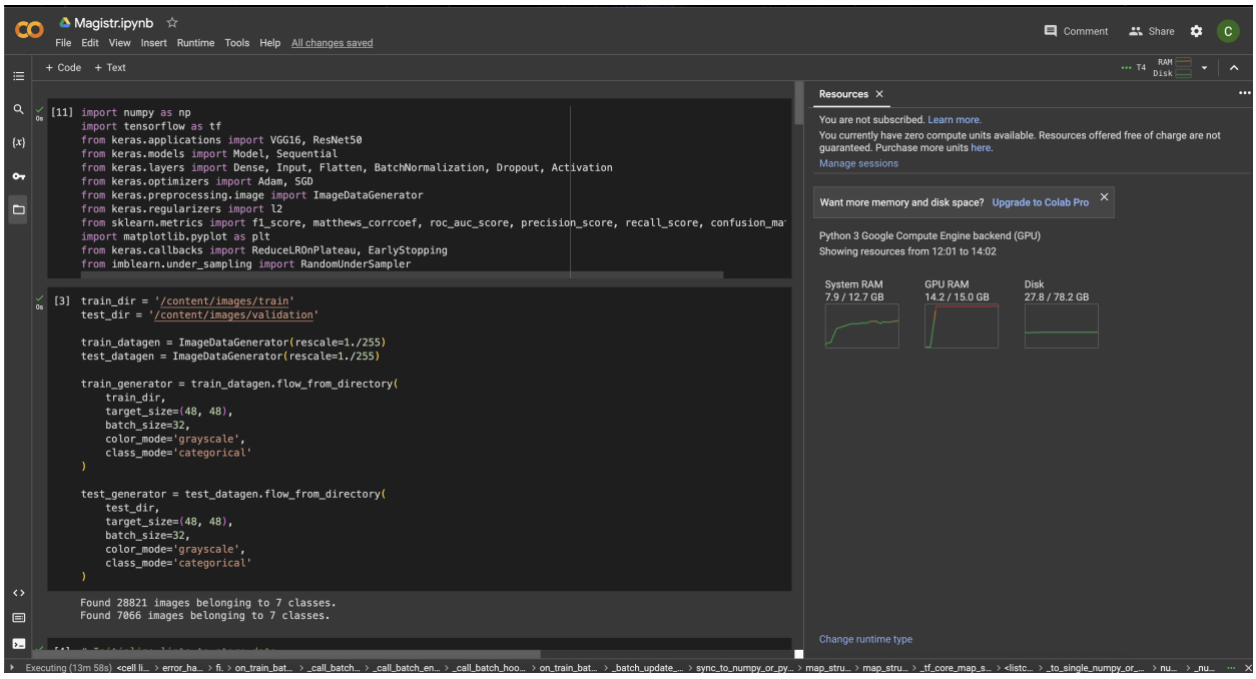

Рисунок 3.1 – Приклад інтерфейсу Jupyter Notebook в Google Colab

Для дослідження був вибраний популярний набір даних FER2013. Він містить значну колекцію з 35887 зображень обличь у відтінках сірого, які класифіковані за сімома різними емоціями: гнів, відраза, страх, щастя, смуток, здивування та нейтральний стан. Різноманітний набір зображень, зібраний від різних людей різної етнічної приналежності, віку та умов освітлення, робить його еталонним набором даних для дослідження. Комплексний характер FER2013 дозволяє розробляти надійні моделі та оцінювати їхню ефективність у реальних умовах [39].

3.2 Програмна реалізація

Спершу потрібна невелика підготовка датасету для використання його в програмі. Для цього всі зображення, які вже в правильному форматі і готові до використання, були розподілені по папкам в вигляді ієрархії. Частина даних, приблизно 80 відсотків всіх зображень знаходяться в папці «train». Вона буде використовуватися для тренування нейромережі. Інша частина буде в папці «validation» для тестування натренованої нейромережі. В кожній з цих папок знаходиться 7 інших папок, які представляють класи між якими буде проходити класифікація. В цих папках вже будуть знаходитись зображення обличь людей в розмірі 48×48 в градації сірого. Ієрархію представлено на рисунку 3.2.

Рисунок 3.2 – Ієрархія даних для програмної реалізації

Сама програмна реалізація розподілена на 3 основних блоки. Перший блок – це загрузка даних датасету в код для подальшого використання. Бібліотека tensorflow має дуже потужні і зручні інструменти для ініціалізації

даних в різних форматах. В даному випадку формат даних – ієрархія папок, тому буде використано клас *ImageDataGenerator*. На вхід він приймає декілька обов'язкових аргументів – шлях до кореневої папки *path*, розмір зображень *target_size*, колір зображень *color_mode* та тип класифікації *class_mode* (лістинг 3.1). При ініціалізації цей клас автоматично приводить зображення в готовий до використання нейронною мережею вигляд.

Лістинг 3.1 Реалізація загрузки даних датасету:

```
train_dir = '/content/images/train'
test_dir = '/content/images/validation'
train_datagen = ImageDataGenerator(rescale=1./255)
test_datagen = ImageDataGenerator(rescale=1./255)
train_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size=(48, 48),
    batch_size=32,
    color_mode='grayscale',
    class_mode='categorical'
)
test_generator = test_datagen.flow_from_directory(
    test_dir,
    target_size=(48, 48),
    batch_size=32,
    color_mode='grayscale',
    class_mode='categorical'
)
```

Наступний етап – це програмне будування архітектур згорткових нейронних мереж. Функції створення нейромереж мають назви *build_alexnet*,

build_vgg16, build_resnet50. Приклад реалізації архітектури VGG-Net представлений на лістингу 3.2.

Лістинг 3.2 Реалізація архітектури VGG-Net:

```
def build_vgg16(input_shape=(48, 48, 1), num_classes=7):
    # Load VGG16 without top classification layers
    vgg_base = VGG16(weights=None, include_top=False, input_tensor=Input(shape=input_shape))

    model=Sequential()
    model.add(vgg_base)
    model.add(Dropout(0.5))
    model.add(Flatten())
    model.add(BatchNormalization())
    model.add(Dense(32,kernel_initializer='he_uniform'))
    model.add(BatchNormalization())
    model.add(Activation('relu'))
    model.add(Dropout(0.5))
    model.add(Dense(32,kernel_initializer='he_uniform'))
    model.add(BatchNormalization())
    model.add(Activation('relu'))
    model.add(Dropout(0.5))
    model.add(Dense(32,kernel_initializer='he_uniform'))
    model.add(BatchNormalization())
    model.add(Activation('relu'))
    model.add(Dense(7,activation='softmax'))
    return model
```

На останок залишається найважливіше – тренування та тестування побудованої згорткової мережі (лістинг 3.3). В ході тестування також

підраховуються всі метрики ефективності і записуються в масиви для подальшої візуалізації та аналізу.

Лістинг 3.3 Реалізація тренування та тестування мережі ResNet:

```
resnet_model = build_resnet50()
resnet_model.compile(optimizer=SGD(learning_rate=0.01, momentum=0.0, nesterov=False), loss='categorical_crossentropy', metrics=['accuracy'])
resnet_model.fit(train_generator,epochs=25, validation_data=test_generator,steps_per_epoch=len(train_generator), validation_steps=len(test_generator), callbacks=[reduce_lr, es])

predictions = resnet_model.predict_generator(test_generator, steps=len(test_generator))
predicted_classes = np.argmax(predictions, axis=1)
# Get true labels
true_classes = test_generator.classes
# Compute metrics
f1 = f1_score(true_classes, predicted_classes, average='weighted')
mcc = matthews_corrcoef(true_classes, predicted_classes)
roc_auc = roc_auc_score(true_classes, predictions, multi_class='ovr')
precision = precision_score(true_classes, predicted_classes, average='weighted')
recall = recall_score(true_classes, predicted_classes, average='weighted')
# Store metrics
f1_scores.append(f1)
mcc_scores.append(mcc)
roc_auc_scores.append(roc_auc)
precision_scores.append(precision)
recall_scores.append(recall)
```

3.3 Аналіз ефективності класифікації зображень

Аналіз ефективності буде проводитися за допомогою кількох відомих метрик для оцінки якості нейронних мереж. Це F1, Matthews correlation coefficient (MCC), ROC AUC [40] та проста точність класифікації на виборці. Сукупність цих метрик дасть можливість правильно оцінити кожну архітектуру для задачі класифікації емоцій і виділити всі недоліки та достоїнства кожної.

Перша метрика, F1, це середнє гармонійне значення точності та повноти. Він дає загальну оцінку точності класифікатора, враховуючи як хибнопозитивні, так і хибнонегативні результати. За результатом дослідження (рис. 3.3), найкращу оцінку отримала архітектура AlexNet, 0,4587. Але і інші архітектури показали не сильно гірший результат, відставши від лідера на 0,01 та 0,015 балів.

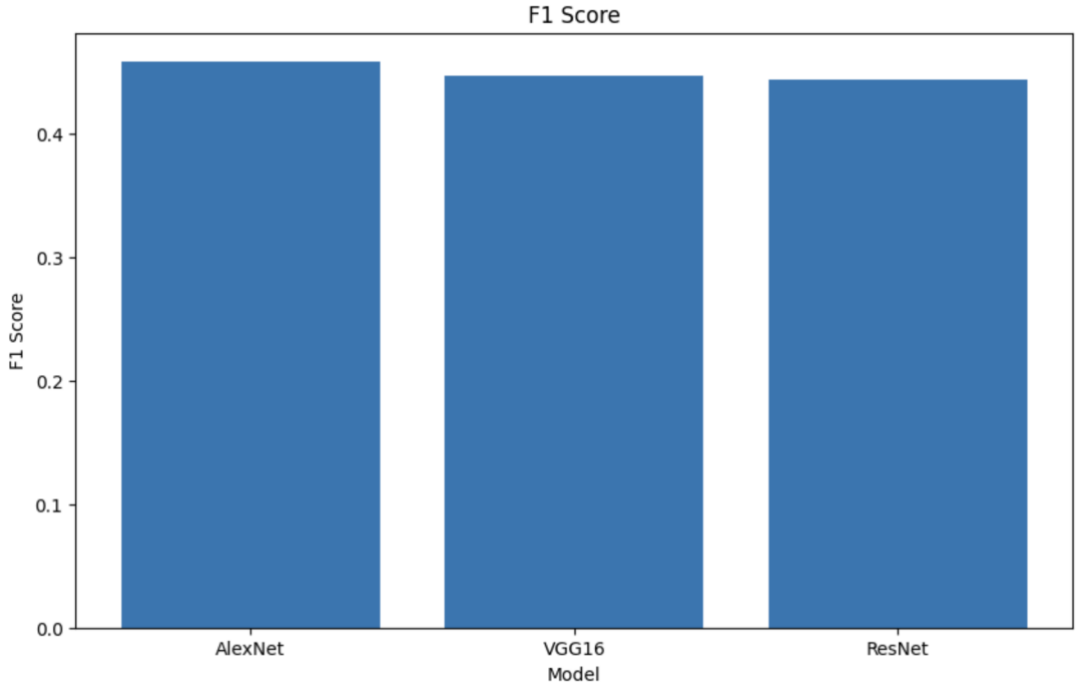

Рисунок 3.3 – Результати метрики F1

Однак ця метрика не враховує істинно-негативні випадки – тобто ті, які модель правильно ідентифікує що вони не належать до позитивного класу.

MCC, з іншого боку, враховує істинно позитивні, хибно позитивні, істинно негативні та хибно негативні результати. Це більш збалансований показник, який може дати високий бал, тільки якщо класифікатор добре працює в усіх чотирьох категоріях [40]. Результати цієї метрики показали (рис. 3.4), що глибокі архітектури мають кращий баланс між класами різного розміру, тобто класи з меншою кількістю тренувальних даних класифікують так же, як і класи з більшою кількістю. Оцінки ResNet та VGG-Net майже однакові, але суттєво вищі за AlexNet.

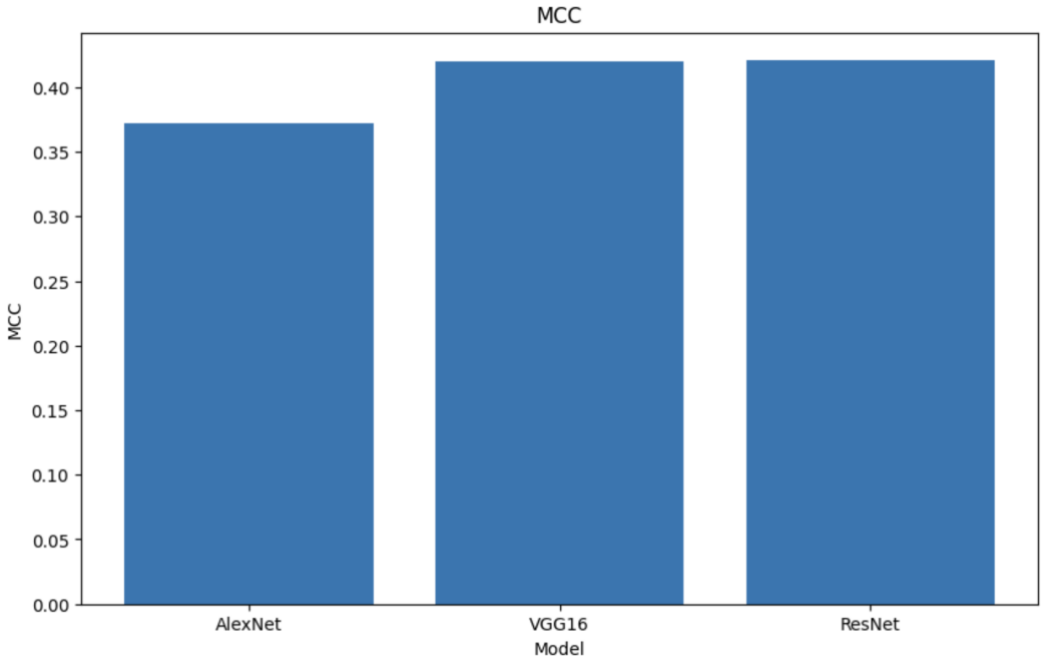

Рисунок 3.4 – Результати метрики MCC

Ще одна метрика – це ROC AUC. Вона є показником ефективності при різних порогових значеннях. ROC – це крива ймовірності, а AUC представляє ступінь або міру відокремлюваності. Проще кажучи, вона показує, наскільки модель здатна розрізняти класи. Результати цієї метрики (рис. 3.5) підтвердили тенденцію глибоких архітектур до кращого розрізняння класів. Архітектура VGG-Net показала найкращий результат – 0,7163, ResNet відстає на 0,015, а AlexNet ще на 0,03.

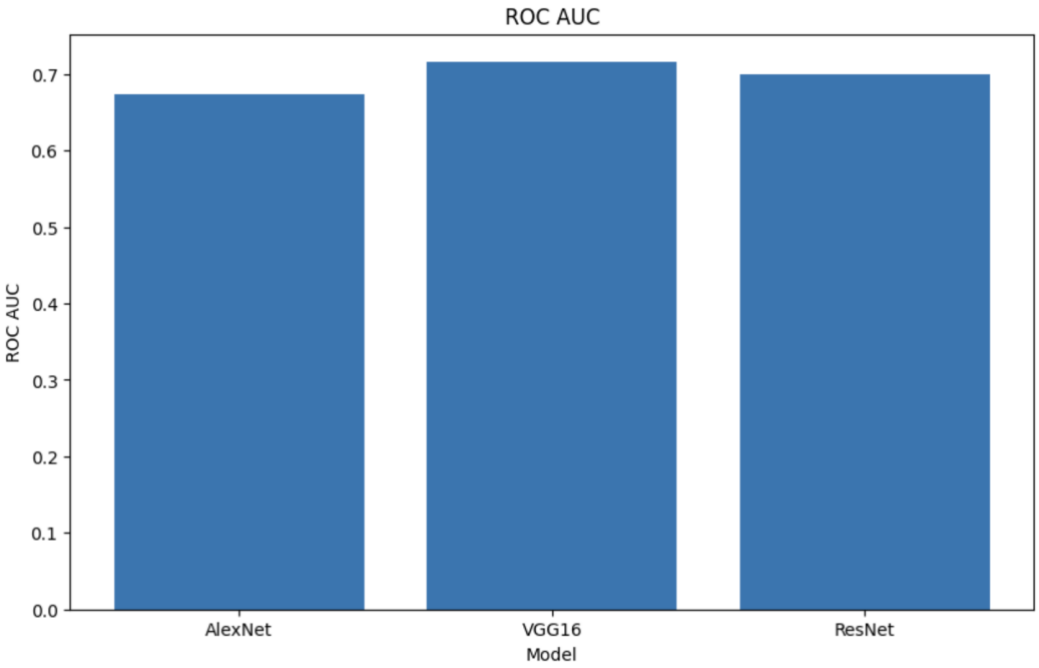

Рисунок 3.5 – Результат метрики ROC AUC

Остання метрика – це процентне співвідношення правильно класифікованих даних до всіх даних набору. Вона показує наскільки добре кожна з архітектур впоралась саме з цією задачею класифікації. Тут найкращий результат має AlexNet, 60,42%. VGG-Net та ResNet в свою чергу, мають результати нижчі на 4% та 6% відповідно (рис. 3.6).

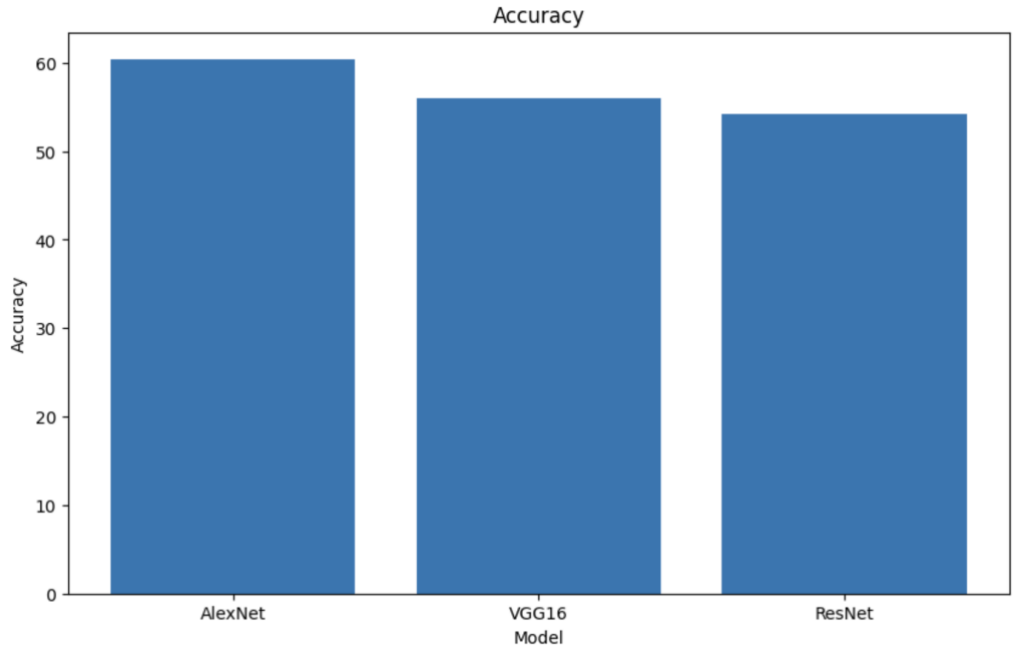

Рисунок 3.6 – Результати метрики точності класифікації

Такі оцінки свідчать про те, що набір, обраний для дослідження має доволі відчутний дисбаланс по кількості даних за класами. До того ж, в більш глибоких архітектурах відчувалась нестача тренувальних даних. Але, не дивлячись на ці невеликі недоліки, набір даних гарно себе показав і дав змогу виділити основні недоліки і переваги кожної з архітектур. Повний список результатів усіх метрик представлено у таблиці 3.1.

Таблиця 3.1 – Результати оцінювання архітектур

Назва архітектури моделі	**F1-Score**	**MCC**	**ROC AUC**	**Precision**	**Recall**	**Accuracy**
AlexNet	0,4587	0,3722	0,6729	0,4519	0,4657	60,42%
VGG16	0,4471	0,4199	0,7163	0,4437	0,4506	55,94%
ResNet50	0,4436	0,4211	0,6995	0,4328	0,4548	54,22%

Виходячи з цих результатів, можна зробити висновки, що для простих задач з невеликими зображеннями, особливо у відтінках сірого, буде достатньо швидкої і простої архітектури AlexNet. Навіть дисбаланс класів не вплинув на її продуктивність. Про це свідчить метрика F1.

З іншого боку, якщо задача буде спрямована не тільки на виявлення правильних класів, а ще і на виявлення неправильних, то тут куди краще працюють VGG-Net та ResNet, виходячи з метрик MCC та ROC AUC. Так, наприклад, це може бути важливим у задачі розпізнавання хвороб, тому що там аналіз завжди передбачає багато факторів і відкидання неправильних варіантів є дуже важливою частиною процесу.

Також результати метрик MCC та ROC AUC відзначають більший баланс між істинно позитивними, хибно позитивними, істинно негативними та хибно негативними випадками в випадку ResNet та VGG. Можна зробити висновок, що з підвищенням складності даних (кольорові зображення,

більший розмір та ін.) точність класифікації буде підвищуватися, особливо в випадку складної архітектури ResNet.

Проте, більші архітектури потребують більших ресурсів. Дане дослідження було проведено повністю в пакетному режимі, тому ресурси, які були витрачені на навчання, не бралися до уваги. Якщо задача буде вирішуватися в online-режимі, де швидкість має значення, то вибір архітектури може бути зроблений на користь менш точної, але більш швидкою. В такому випадку, найкращий баланс швидкості та повноти може продемонструвати архітектура VGG-Net. Вона навчається в 2,5 рази швидше за ResNet (рис. 3.7), а більшість оцінок мають невеликий відрив, або навіть кращі.

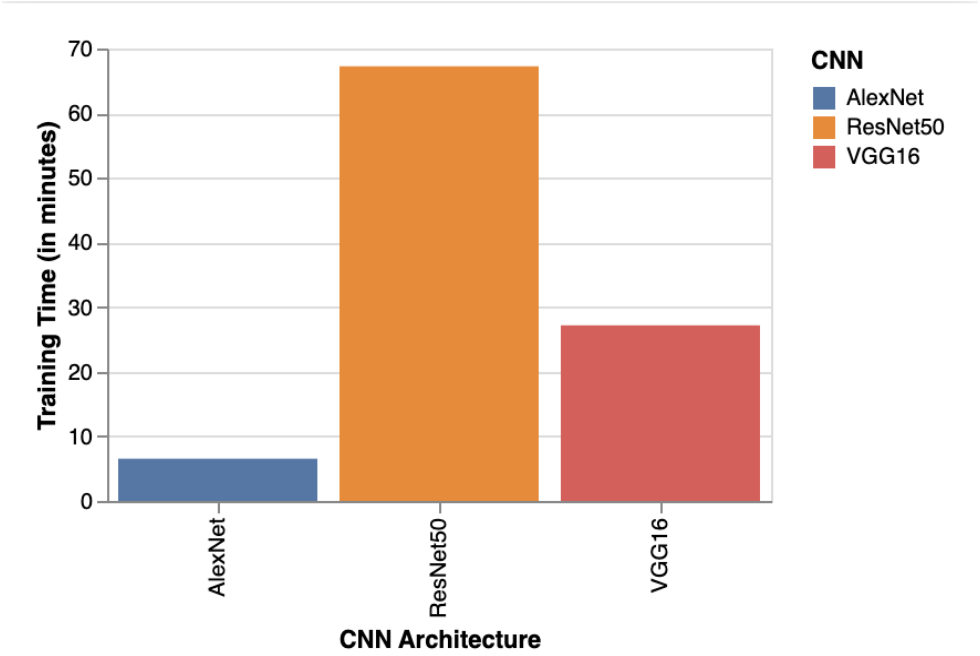

Рисунок 3.7 – Швидкість навчання різних архітектур на одному наборі даних

ВИСНОВКИ

У рамках кваліфікаційної роботи були досліджені три різних архітектури згорткових нейронних мереж для задач класифікації зображень. Був розроблений програмний застосунок з ціллю проведення тестування, на підставі якого були зроблені висновки щодо переваг і вад кожної з них. Було визначено всі необхідні інструментальні засоби для створення застосунку.

Для дослідження були взяті такі популярні архітектури згорткових нейронних мереж, як AlexNet, VGG-Net, ResNet. Були розглянуті задачі виявлення хвороби по медичним знімкам, розпізнавання номерних знаків та розпізнавання емоцій на обличчях людей. В якості метрик аналізу ефективності були взяті F1-score, MCC, ROC AUC та абсолютна точність класифікації вибірки.

Був проведений порівняльний аналіз трьох обраних архітектур на задачі розпізнавання емоцій за допомогою розробленого програмного застосунку. За результатами цього аналізу були виявлені особливості роботи кожної нейронної мережі і запропоновані найбільш відповідні випадки для їх використання. Це надало розуміння можливостей і перспектив подальшого застосування і доопрацювання моделей [41-43].

Практична значущість дослідження полягає у:

– всебічному аналізі різних існуючих архітектур згорткових нейронних мереж, що може посприяти правильному вибору архітектури під задачу класифікації;

– розробленні програмних моделей для тестування різних згорткових нейронних мереж під потрібну задачу класифікації.

Результати дослідження апробовано у вигляді тез доповідей під час Міжнародної наукової конференції «СУЧАСНІ ПРОБЛЕМИ ТА НОВІТНІ ТЕОРІЇ РОЗВИТКУ» [44].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Гороховатський, В. О., & Творошенко, І. С. (2021). Методи інтелектуального аналізу та оброблення даних: навч. посібник.

2. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. nature, 521(7553), 436-444.

3. Nielsen, M. A. (2015). Neural networks and deep learning (Vol. 25, pp. 15-24). San Francisco, CA, USA: Determination press.

4. Bishop, C. M. (1995). Neural networks for pattern recognition. Oxford university press.

5. Bilonoh, B., Bodyanskiy, Y., Kolchygin, B., & Mashtalir, S. (2021, May). Tunable Activation Functions for Deep Neural Networks. In International Scientific Conference “Intellectual Systems of Decision Making and Problem of Computational Intelligence” (pp. 624-633). Cham: Springer International Publishing.

6. Tripathi, S., & Singh, S. K. (2020). Ensembling handcrafted features with deep features: an analytical study for classification of routine colon cancer histopathological nuclei images. Multimedia Tools and Applications, 79, 34931-34954.

7. Lin, W., Hasenstab, K., Moura Cunha, G., & Schwartzman, A. (2020). Comparison of handcrafted features and convolutional neural networks for liver MR image adequacy assessment. Scientific Reports, 10(1), 20336.

8. Bilonoh, B., & Mashtalir, S. (2020, August). Parallel multi-head dot product attention for video summarization. In 2020 IEEE Third International Conference on Data Stream Mining & Processing (DSMP) (pp. 158-162). IEEE.

9. Mashtalir, S., Mikhnova, O., & Stolbovyi, M. (2018, August). Sequence matching for content-based video retrieval. In 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP) (pp. 549-553). IEEE.

10. Mashtalir, S. V., & Stolbovyi, M. I. (2018). ADAPTIVE MATRIX MODELS IN THE VIDEO STREAMS CONTROL PROBLEM. RADIO ELECTRONICS COMPUTER SCIENCE CONTROL, (4), 188-194.

11. Elhagry, A., & Saeed, M. (2022). Investigating the Challenges of Class Imbalance and Scale Variation in Object Detection in Aerial Images. arXiv preprint arXiv:2202.02489.

12. Matsuoka, D. (2021). Classification of imbalanced cloud image data using deep neural networks: performance improvement through a data science competition. Progress in Earth and Planetary Science, 8(1), 1-11.

13. Norrenbrock, T., Rudolph, M., & Rosenhahn, B. (2023). Take 5: Interpretable Image Classification with a Handful of Features. arXiv preprint arXiv:2303.13166.

14. Tvoroshenko, I., & Kukharchuk, V. (2021). Current state of development of applications for recognition of faces in the image and frames of video captures.

15. . Pomazan, V., Tvoroshenko, I., & Gorokhovatskyi, V. (2023). Development of an application for recognizing emotions using convolutional neural networks.

16. Affectiva raises $14 million to bring apps, robots emotional intelligence. URL: https://techcrunch.com/2016/05/25/affectiva-raises-14-million-to-bring-apps-robots-emotional-intelligence/ (дата звернення 05.10.2023).

17. Krithika, L. B., & GG, L. P. (2016). Student emotion recognition system (SERS) for e-learning improvement based on learner concentration metric. Procedia Computer Science, 85, 767-776.

18. Upreti, M., Pandey, C., Bist, A. S., Rawat, B., & Hardini, M. (2021). Convolutional neural networks in medical image understanding. Aptisi Transactions on Technopreneurship (ATT), 3(2), 120-126.

19. Shivakumara, P., Tang, D., Asadzadehkaljahi, M., Lu, T., Pal, U., & Hossein Anisi, M. (2018). CNN-RNN based method for license plate recognition. CAAI Transactions on Intelligence Technology, 3(3), 169-175.

20. Daradkeh Y.I., Gorokhovatskyi V., Tvoroshenko I., and Al-Dhaifallah M. (2022) Classification of Images Based on a System of Hierarchical Features, *Computers, Materials & Continua*, 72(1), pp. 1785–1797.

21. Krizhevsky, A., Sutskever, I. & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks, Adv. Neural Inf. Process. Syst., pp. 1097-1105.

22. Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

23. 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).

24. Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861.

25. Zoph, B., & Le, Q. V. (2016). Neural architecture search with reinforcement learning. arXiv preprint arXiv:1611.01578.

26. Liu, C., Zoph, B., Neumann, M., Shlens, J., Hua, W., Li, L. J., ... & Murphy, K. (2018). Progressive neural architecture search. In Proceedings of the European conference on computer vision (ECCV) (pp. 19-34).

27. Koushik, J. (2016). Understanding convolutional neural networks. arXiv preprint arXiv:1605.09081.

28. What is a Convolutional Neural Network. URL: https://blog.roboflow.com/what-is-a-convolutional-neural-network/#:~:text=The%20convolutional%20layers%20within%20a%20CNN%2C%20through%20the,key%20features%20such%20as%20edges%2C%20textures%2C%20and%20shapes. (дата звернення 15.10.2023).

29. Convolutional Neural Network (CNN) Architecture Explained in Plain English Using Simple Diagrams. URL: https://towardsdatascience.com/convolutional-neural-network-cnn-architecture-explained-in-plain-english-using-simple-diagrams-

e5de17eacc8f#:~:text=The%20first%20convolutional%20layer%20takes,occurs%20in%20each%20convolutional%20layer (дата звернення 16.10.2023).

30. Buduma, N., Buduma, N., & Papa, J. (2022). Fundamentals of deep learning. " O'Reilly Media, Inc.".

31. Amari, S. I. (1993). Backpropagation and stochastic gradient descent method. Neurocomputing, 5(4-5), 185-196.

32. Understanding Backpropagation Algorithm. URL: https://towardsdatascience.com/understanding-backpropagation-algorithm-7bb3aa2f95fd#:~:text=Backpropagation%20algorithm%20is%20probably%20the,propagating%20errors%E2%80%9D (дата звернення 18.10.2023).

33. A Comprehensive Guide to the Backpropagation Algorithm in Neural Networks. URL: https://neptune.ai/blog/backpropagation-algorithm-in-neural-networks-guide#:~:text=This%20article%20is%20a%20comprehensive,works%20in%20the%20backward%20pass (дата звернення 18.10.2023).

34. Stochastic gradient descent. URL: https://optimization.cbe.cornell.edu/index.php?title=Stochastic_gradient_descent#:~:text=Stochastic%20gradient%20descent%20,random%20weight%20vector%20is%20picked (дата звернення 18.10.2023).

35. Yacouby, R., & Axman, D. (2020, November). Probabilistic extension of precision, recall, and f1 score for more thorough evaluation of classification models. In Proceedings of the first workshop on evaluation and comparison of NLP systems (pp. 79-91).

36. F1-Score in Machine Learning: Intro & Calculation. URL: https://www.v7labs.com/blog/f1-score-guide (дата звернення 21.10.2023).

37. Chollet, F. (2021). Deep learning with Python. Simon and Schuster.

38. Millstein, F. (2020). Convolutional neural networks in Python: beginner's guide to convolutional neural networks in Python. Frank Millstein.

39. Khaireddin, Y., & Chen, Z. (2021). Facial emotion recognition: State of the art performance on FER2013. arXiv preprint arXiv:2105.03588.

40. Chicco, D., & Jurman, G. (2023). The Matthews correlation coefficient (MCC) should replace the ROC AUC as the standard metric for assessing binary classification. BioData Mining, 16(1), 1-23.

41. Tvoroshenko I., Gorokhovatskyi V., Kobylin O., and Tvoroshenko A. (2023) Application of deep learning methods for recognizing and classifying culinary dishes in images, International Journal of Academic and Applied Research, 7(9), pp. 57-70.

42. Gorokhovatskyi V., Tvoroshenko I. (2023) Identification of visual objects by the search request. International scientific symposium «INTELLIGENT SOLUTIONS-S». Computational intelligence (results, problems and perspectives). Decision making theory: proceedings of the international symposium, September 28, 2023, Kyiv-Uzhorod, Ukraine, pp. 25-27.

43. Daradkeh Y.I., Gorokhovatskyi V., Tvoroshenko I., Gadetska S., and Al-Dhaifallah M. (2023) Statistical data analysis models for determining the relevance of structural image descriptions, IEEE Access, 11, pp. 126938-126949.

44. Kotykhin, S. (2023). Analysis of existing image classification problems using convolutional neural networks.