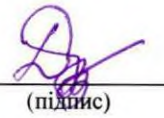
МІНІСТЕРСТВО ФІНАНСІВ УКРАЇНИ
ДЕРЖАВНИЙ ПОДАТКОВИЙ УНІВЕРСИТЕТ

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

КВАЛІФІКАЦІЙНА РОБОТА

на тему: РОЗРОБКА НЕЙРОМЕРЕЖІ ДЛЯ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ НА ОСНОВІ TENSORFLOW

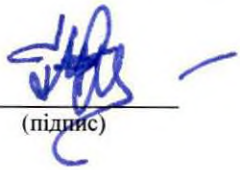
Здобувача вищої освіти першого (бакалаврського) рівня,
IV курсу, групи КБ-21-1, спеціальності 122 «Комп'ютерні науки»,
освітня програма «Інформаційні управляючі системи і технології (за галузями)»



(підпис)

Савченка Данила Валерійовича
(прізвище, ім'я, по батькові)

Науковий керівник



(підпис)

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

Керівник освітньо-професійної програми



(підпис)

Омельчук Антон Анатолійович, к.т.н.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Допущено до захисту за рішенням кафедри комп'ютерних та інформаційних технологій і систем протокол від «13» травня 2025 № 18

Завідувач кафедри



(підпис)

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

Ірпінь – 2025

Реєстрація № 61 — 26.05.2025 (дата) — (підпис відповідальної особи)

Результати перевірки — до захисту (до захисту, на доопрацювання) — 26.05.2025 (дата) — (підпис)

Результат захисту — 4 (добре) (оцінка) — 05.06.2025 (дата) — (підпис)

Голова комісії: (підпис) В.М. Франчук

Члени комісії:

(підпис) М.М. Філоненко

(підпис) С.М. Грищенко

(підпис) А.А. Омельчук

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 випускної роботи (якість теоретичного матеріалу, його зв'язок з практичним, вміння представляти матеріал в логічній послідовності, наявність висновків та пропозицій)	**від 0 до 30**	27		Філоненко М.М. Франчук В.М..
2. Дотримання вимог щодо оформлення	**від 0 до 10**	9		Грищенко С.М. Омельчук А.А.
3. Вчасне виконання роботи на всіх етапах у встановлені кафедрою терміни	**від 0 до 5**	5		Суховецька Л.П.
4. Рецензент	**від 0 до 5**	4		
РАЗОМ БАЛІВ	**від 0 до 50**	45		

Оцінювання кваліфікаційної роботи науковим керівником - 40 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)
85	B	4 (добре)	05.06.2025	Франчук В.М. Філоненко М.М. Грищенко С.М. Омельчук А.А. 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 (за галузями)»

Затверджую
Завідувач кафедри _____ Філоненко М.М.
«14» листопада 2024

Завдання на кваліфікаційну роботу
здобувачу вищої освіти
Савченку Данилу Валерійовичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи розробка нейромережі для класифікації зображень на основі TensorFlow
Затверджена наказом ректора від «13» листопада 2024 № 1717
2. Строк здачі закінченої роботи «08» травня 2025
3. Мета роботи розробка нейронної мережі на базі фреймворку TensorFlow для автоматичної класифікації зображень.
4. Об'єкт дослідження процеси класифікації вхідних зображень за допомогою моделей глибинного навчання.
Предмет дослідження методи побудови та навчання згорткових нейронних мереж для розпізнавання зображень.
Орієнтовна база дослідження публічні набори зображень (наприклад, CIFAR-10, MNIST, ImageNet), середовище TensorFlow, Google Colab або Jupyter Notebook.Вихідні дані до роботи
5. Перелік графічного матеріалу архітектура нейромережі, блок-схема алгоритму класифікації, графіки точності та втрат, приклади класифікаційних результатів.
6. Зміст кваліфікаційної роботи (перелік питань за кожним розділом)
ВСТУП
РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ ШТУЧНИХ НЕЙРОННИХ МЕРЕЖ
1.1 Історія розвитку та сучасний стан штучних нейронних мереж
1.2. Аналіз інструментів розробки моделей штучних нейронних мереж РОЗДІЛ
2. ПРОЄКТУВАННЯ ТА РОЗРОБКА НЕЙРОМЕРЕЖІ ДЛЯ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ НА ОСНОВІ TENSORFLOW
2.1. Проєктування архітектури та алгоритмів роботи нейромережі
2.2. Архітектура нейромережі, вибір середовища розробки, тестування та оцінка результатів роботи
РОЗДІЛ 3. ПЕРСПЕКТИВИ РОЗВИТКУ НЕЙРОМЕРЕЖІ ДЛЯ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ НА ОСНОВІ TENSORFLOW
ВИСНОВКИ
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

7. 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 30.10.2024	До 30.10.2024
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)	27.12.2024	27.12.2024
3	**ІІ розділ** (аналіз експериментальної частини про методики досліджень)	17.02.2025	17.02.2025
4	**ІІІ розділ** (проведені теоретичні і (або) експериментальні дослідження та проектні рішення)	16.04.2025	16.04.2025
5	**Висновки**	01.05.2025	01.05.2025
6	Доопрацювання роботи з урахуванням зауважень наукового керівника	09.05.2025	09.05.2025
7	Отримання відгуку керівника	27.05.2025	27.05.2025
8	Отримання зовнішньої рецензії	29.05.2025	29.05.2025

8. Дата видачі завдання «14» листопада 2024

9. Керівник кваліфікаційної роботи _____ (підпис) Філоненко М.М (прізвище, ініціали)

10. Керівник освітньо-професійної програми _____ (підпис) Омельчук А.А. (прізвище, ініціали)

11. Завдання до виконання отримав здобувач вищої освіти _____ (підпис) Савченко Д.В (прізвище, ініціали)

АНОТАЦІЯ

Розробка нейромережі для класифікації зображень на основі TensorFlow

Мета роботи розробка нейронної мережі на базі фреймворку TensorFlow для автоматичної класифікації зображень із застосуванням сучасних методів машинного навчання.

Об'єктом дослідження є процеси класифікації вхідних зображень за допомогою моделей глибинного навчання.

Предмет дослідження - методи побудови та навчання згорткових нейронних мереж для розпізнавання зображень.

Методи дослідження включають. Теоретичні – аналіз наукових джерел, порівняння існуючих підходів, узагальнення та систематизація знань; емпіричні – експериментальне навчання моделей, аналіз результатів, візуалізація та інтерпретація даних.

Кваліфікаційна робота містить 58 сторінок, 6 рисунків, список використаних джерел 41 найменувань.

Ключові слова: нейромереда, зображення, машинне навчання.

Рік виконання кваліфікаційної роботи 2025

Рік захисту роботи 2025

ANNOTATION

Development of a Neural Network for Image Classification Based on TensorFlow

The aim of this work is to develop a neural network using the TensorFlow framework for automatic image classification by applying modern machine learning methods.

The object of the research is the process of classifying input images using deep learning models.

The subject of the research is the methods of building and training convolutional neural networks for image recognition.

Research methods include:

Theoretical: analysis of scientific sources, comparison of existing approaches, generalization and systematization of knowledge;

Empirical: experimental training of models, analysis of results, visualization and interpretation of data.

The qualification work contains 58 pages, 6 figures, and a list of 41 references.

Keywords: neural network, image, machine learning.

Year of completion: 2025

Year of defense: 2025

Зміст

ВСТУП 2

РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ ШТУЧНИХ НЕЙРОННИХ МЕРЕЖ5

1.1 Історія розвитку та сучасний стан штучних нейронних мереж 5

1.2. Аналіз інструментів розробки моделей штучних нейронних мереж 9

РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА РОЗРОБКА НЕЙРОМЕРЕЖІ ДЛЯ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ НА ОСНОВІ TENSORFLOW 20

2.1. Проєктування архітектури та алгоритмів роботи нейромережі 20

2.2. Архітектура нейромережі, вибір середовища розробки, тестування та оцінка результатів роботи 24

РОЗДІЛ 3. ПЕРСПЕКТИВИ РОЗВИТКУ НЕЙРОМЕРЕЖІ ДЛЯ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ НА ОСНОВІ TENSORFLOW 43

ВИСНОВКИ 48

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 50

ВСТУП

У сучасному світі, де кількість цифрових зображень зростає експоненційно, зростає і потреба в автоматизованих системах їх обробки та класифікації. Ці системи активно застосовуються в таких галузях, як медицина, безпека, транспорт, торгівля, промисловість тощо. В умовах цифрової трансформації особливої актуальності набувають інтелектуальні системи, здатні самостійно навчатися та приймати рішення на основі аналізу візуальної інформації. Саме тому використання нейронних мереж, зокрема згорткових нейронних мереж (CNN), стає провідним підходом у задачах класифікації зображень.

Серед широкого спектру технологій, які підтримують побудову таких систем, фреймворк TensorFlow є одним із найпотужніших та найпоширеніших інструментів для реалізації моделей глибинного навчання. Він забезпечує гнучкість, масштабованість та високу продуктивність, що дає змогу ефективно реалізовувати складні архітектури нейромереж для різних прикладних задач.

Актуальність даної теми полягає в тому, що класифікація зображень за допомогою глибоких нейронних мереж стала основою багатьох сучасних інформаційних систем. Від точності такої класифікації нерідко залежить успішність прийняття рішень або функціонування цілого технологічного процесу. Застосування технологій штучного інтелекту дозволяє досягати високих результатів в автоматичному розпізнаванні візуальних об’єктів навіть у складних умовах.

Проблематику автоматичної класифікації зображень за допомогою глибоких нейронних мереж досліджували як вітчизняні, так і зарубіжні науковці. Зокрема, вітчизняні автори, такі як Герасименко О.О., Шевченко І.М. та Коваленко Р.С., розглядали застосування згорткових нейронних мереж у задачах розпізнавання візуальних об’єктів у медицині та промисловості. Серед зарубіжних дослідників варто відзначити внесок таких науковців, як Yann LeCun, Alex Krizhevsky та Geoffrey Hinton, які є засновниками сучасного підходу до побудови глибоких авнейронних мереж і зробили значний внесок у розвиток архітектур CNN, методів їх навчання та оптимізації.

Метою кваліфікаційної роботи *є* розробка нейронної мережі на базі фреймворку TensorFlow для автоматичної класифікації зображень із застосуванням сучасних методів машинного навчання.

Для досягнення мети передбачено *вирішення таких завдань*:

провести аналіз предметної області та методів класифікації зображень;

- дослідити архітектури згорткових нейронних мереж та обрати оптимальну для поставленої задачі;
- створити модель нейромережі із використанням TensorFlow;
- провести навчання моделі на відкритих наборах зображень (наприклад, CIFAR-10, MNIST);
- візуалізувати процес навчання та результати класифікації за допомогою графіків точності та втрат;
- оцінити ефективність моделі за допомогою відповідних метрик.

Об'єктом дослідження є процеси класифікації вхідних зображень за допомогою моделей глибинного навчання.

Предмет дослідження - методи побудови та навчання згорткових нейронних мереж для розпізнавання зображень.

Методи дослідження включають. Теоретичні – аналіз наукових джерел, порівняння існуючих підходів, узагальнення та систематизація знань; емпіричні – експериментальне навчання моделей, аналіз результатів, візуалізація та інтерпретація даних.

Базою дослідження виступають відкриті набори зображень, зокрема CIFAR-10, MNIST, ImageNet, а також середовища програмування Google Colab та Jupyter Notebook з використанням фреймворку TensorFlow та мови програмування Python.

Науково-практична значимість одержаних результатів полягає у створенні універсального підходу до розв'язання задач класифікації зображень, який може бути адаптований під конкретні потреби бізнесу, науки чи виробництва. Розроблена нейронна мережа дозволяє досягти високої точності класифікації, що підвищує ефективність автоматизованих систем обробки зображень та сприяє подальшому розвитку прикладного штучного інтелекту.

Практичне значення роботи полягає в тому, що вона демонструє повний цикл створення моделі глибокого навчання – від аналізу проблеми до реалізації та оцінки ефективності. Візуалізація результатів, створення графіків точності та втрат, а також приклади класифікаційних результатів дають змогу наочно оцінити працездатність системи та її потенціал для впровадження у реальні задачі.

Мій особистий внесок у дослідження полягає в аналізі предметної області класифікації зображень, дослідженні архітектур згорткових нейронних мереж, виборі оптимальної моделі для розв’язання поставленої задачі та її реалізації з використанням фреймворку TensorFlow. Я самостійно здійснив підбір та підготовку наборів даних, провів навчання моделі, візуалізував процеси навчання за допомогою графіків точності та втрат, а також виконав оцінювання ефективності створеної нейромережі за допомогою відповідних метрик.

РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ ШТУЧНИХ НЕЙРОННИХ МЕРЕЖ

1.1 Історія розвитку та сучасний стан штучних нейронних мереж

Ідея штучних нейронних мереж бере початок із вивчення функціонування біологічного мозку. У 1943 році нейрофізіолог Воррен МакКаллох і математик Волтер Піттс запропонували першу математичну модель нейрона, що мала на меті формалізувати процеси мислення через логічні операції. Біологічні нейрони, як відомо, отримують сигнали через дендрити, обробляють їх у тілі клітини та передають далі через аксони. Цей принцип - перетворення і передача інформації - ліг в основу перших штучних нейронних моделей.

Основна ідея полягала в тому, щоб відтворити здатність нервових клітин "збуджуватись" під впливом сигналів і приймати рішення на основі певного порогу активації. Такий підхід дозволив здійснити перші спроби моделювання елементарного мислення та класифікації даних.

Однією з перших реалізацій штучного нейрона стала модель перцептрона, яку у 1958 році розробив американський психолог Френк Розенблатт. Перцептрон складався з вхідного шару та одного вихідного, і міг навчатися, змінюючи ваги між входами й виходом. Його метою була класифікація вхідних даних за допомогою простої лінійної функції.

Попри свою простоту, перцептрон став проривом: він довів, що машини можуть навчатися з прикладів. Однак у 1969 році Марвін Мінський та Сеймур Пейперт у своїй книзі Perceptrons показали, що одношаровий перцептрон не здатен розв’язувати задачі, які не є лінійно роздільними (наприклад, логічну функцію XOR). Цей висновок на певний час загальмував розвиток нейронних мереж[1].

Неможливість перцептрона опрацьовувати складні нелінійні задачі спричинила втрату інтересу до нейронних мереж у 1970–1980-х роках. Цей період отримав назву "зими штучного інтелекту", коли фінансування досліджень різко скоротилось, а ентузіазм дослідників згас.

Серед інших обмежень - недостатня обчислювальна потужність комп’ютерів, слабка теоретична база для навчання складніших архітектур, а також відсутність достатніх обсягів даних для повноцінного тренування моделей. Ці чинники на

тривалий час поставили розвиток штучного інтелекту на паузу, а фокус наукової спільноти змістився в бік класичних алгоритмів обробки інформації.

У 1980-х роках з'явився новий імпульс у розвитку нейронних мереж завдяки концепції багатошарового перцептрона (Multi-Layer Perceptron, MLP). Ця архітектура, на відміну від одношарового перцептрона, включала щонайменше один прихований шар між вхідним і вихідним шарами. Саме приховані шари дали можливість моделі виявляти складні, нелінійні закономірності у даних[4].

MLP став першою архітектурою, здатною розв'язувати задачу XOR, що раніше була принциповою перепоною. Але реалізація MLP була обмеженою до того моменту, поки не з'явився ефективний спосіб навчання вагових коефіцієнтів у всіх шарах.

Відродження нейронних мереж також було пов'язане з використанням нових функцій активації, які дозволяли моделі краще наближати нелінійні залежності. Класичні сигмоїдальні функції (наприклад, логістична або гіперболічний тангенс) дали змогу передавати нелінійність через шари.

Однак згодом було виявлено, що глибокі мережі з такими функціями часто страждають на проблему згасання градієнтів, що ускладнює навчання. Цю проблему частково вирішила активація ReLU (Rectified Linear Unit), яка стала стандартом у багатьох сучасних архітектурах.

Ключовою технологією, яка дозволила ефективно навчати багатошарові нейронні мережі, стала алгоритм зворотного поширення помилки (backpropagation). Хоча його математичне обґрунтування існувало з 1970-х, практичне застосування зросло лише після публікації статті Девіда Румельгарта, Джеффрі Хінтона і Рональда Вільямса у 1986 році.

Backpropagation дозволив поширювати похибку від вихідного шару до вхідного, коригуючи ваги на кожному шарі відповідно до внеску кожного з них у загальну помилку. Це зробило можливим ефективне навчання багатошарових моделей, відкривши нові горизонти у комп'ютерному розпізнаванні та класифікації.

З початку 2010-х років розвиток штучних нейронних мереж увійшов у нову фазу, яку часто називають епохою глибинного навчання (deep learning). Цей підхід

передбачає використання нейронних мереж з великою кількістю шарів - глибоких архітектур, здатних автоматично виявляти багаторівневі патерни у даних[1].

Поштовхом до цього став прорив у комп'ютерному баченні: у 2012 році мережа AlexNet, розроблена Алексом Крижевським під керівництвом Джеффрі Хінтона, перемогла у змаганні ImageNet з великою перевагою над іншими алгоритмами. Ця перемога чітко продемонструвала ефективність глибинних згорткових мереж у складних візуальних задачах.

Ключовими факторами успіху стали - значний обсяг розмічених даних (наприклад, ImageNet); доступність потужних графічних процесорів (GPU); удосконалення алгоритмів оптимізації (наприклад, Adam); застосування нормалізації та регуляризації (dropout, batch normalization).

Глибинне навчання сьогодні є базовою технологією у багатьох сферах, від комп'ютерного зору до обробки природної мови.

Залежно від типу задач, глибинне навчання еволюціонувало у різних напрямках.

Згорткові нейронні мережі (Convolutional Neural Networks, CNN) стали стандартом для аналізу зображень та відео. Вони дозволяють зменшити кількість параметрів через локальні зв'язки і спільні ваги, що особливо ефективно при роботі з великими піксельними масивами.

Рекурентні нейронні мережі (Recurrent Neural Networks, RNN) та їх покращені варіанти, зокрема LSTM (Long Short-Term Memory) і GRU (Gated Recurrent Units), застосовуються для обробки послідовностей, таких як текст, мова, часові ряди. Вони здатні зберігати контекст, що критично важливо для мовних моделей.

Трансформери (Transformers), запропоновані у 2017 році в статті Attention is All You Need, стали фундаментом новітніх досягнень у обробці природної мови. Їхня здатність паралельно обробляти елементи вхідних даних і зберігати довгострокові залежності зробила трансформери базою для таких моделей, як BERT, GPT, T5, LLaMA та інші.

Ці архітектури сьогодні застосовуються не тільки у текстових задачах, а й у зображеннях (Vision Transformers), біології (AlphaFold), програмуванні (Codex) та багатьох інших сферах.

Інтенсивний розвиток нейронних мереж відбувся завдяки активній участі провідних технологічних компаній, таких як Google, Meta (Facebook), Microsoft, Amazon, NVIDIA та OpenAI. Вони інвестували у дослідження, відкриті набори даних та обчислювальні ресурси, що прискорило прогрес у цій галузі.

Важливу роль відіграють також відкриті бібліотеки та фреймворки, які значно спростили реалізацію та експерименти з моделями:

TensorFlow (Google) та PyTorch (Meta) - два домінантні фреймворки для побудови і навчання моделей, широко використовувані як у промисловості, так і в академічних дослідженнях.

Hugging Face Transformers - бібліотека з широким набором попередньо навчених моделей, яка стала стандартом для NLP.

Keras, FastAI, ONNX - інші інструменти, які підтримують різні аспекти розробки, експорту та оптимізації моделей.

Розвиток open-source інфраструктури сприяв демократизації ШІ - сьогодні потужні моделі можуть створювати й досліджувати не лише великі корпорації, а й незалежні дослідники, стартапи та студенти.

Одна з найбільш усталених та широко використовуваних сфер застосування штучних нейронних мереж - комп’ютерний зір. Глибокі згорткові нейронні мережі (CNN) здатні точно і швидко класифікувати зображення, розпізнавати об’єкти, виявляти аномалії.

Наприклад - Google Photos використовує CNN для автоматичного групування фотографій за обличчями, місцями та подіями. Системи безпеки в аеропортах або торгових центрах застосовують нейронні мережі для розпізнавання облич у режимі реального часу. Автоматизоване сортування об'єктів (наприклад, відходів або товарів на складах) базується на класифікації зображень за допомогою нейронних мереж.

У медицині нейронні мережі допомагають у діагностиці захворювань, аналізуючи знімки МРТ, рентгенівські зображення, гістологічні зразки тощо. Наприклад, моделі типу ResNet чи EfficientNet можуть виявляти ознаки раку легенів або діабетичну ретинопатію з точністю, порівнянною з фахівцем.

У автономному транспорті (наприклад, Tesla, Waymo), глибокі мережі застосовуються для - виявлення пішоходів та інших транспортних засобів, розпізнавання дорожніх знаків і світлофорів, прогнозування дій учасників руху.

У промисловості ШІ-моделі використовують для прогнозування технічних збоїв, візуального контролю якості, оптимізації виробничих процесів. Наприклад, в автомобілебудуванні нейронні мережі аналізують відеопотоки з виробничих ліній, щоб виявити дефекти деталей у реальному часі.

Нейронні мережі стали ядром рекомендаційних систем, які використовуються у - Netflix, YouTube, Spotify — для підбору фільмів, музики, відео на основі попередньої поведінки користувача; Amazon, eBay, AliExpress - для прогнозування бажаних товарів, аналізу купівельних намірів; соціальні мережі - для формування стрічки новин, реклами, персоналізованого контенту.

У сфері розмовних агентів та чат-ботів, трансформерні архітектури, такі як GPT (OpenAI), BERT (Google), чи LLaMA (Meta), дозволили перейти від скриптових ботів до інтелектуальних систем, що ведуть природний діалог, пам’ятають контекст, генерують тексти, пояснюють, перекладають тощо.

Такі моделі вже інтегровані у - служби підтримки (банки, страхові компанії, оператори мобільного зв’язку), освітні платформи (тренування, допомога з домашніми завданнями), помічники в бізнесі (аналіз документів, складання звітів, генерація листів).

1.2. Аналіз інструментів розробки моделей штучних нейронних мереж

Можливості TensorFlow

TensorFlow - це потужний open-source фреймворк для створення, тренування та розгортання моделей машинного навчання, розроблений компанією Google. Його основні можливості охоплюють повний цикл роботи з моделями нейронних мереж - від побудови архітектури до її масштабованого розгортання у продакшн-середовищах.

Ключові можливості TensorFlow - підтримка глибинного навчання: повна реалізація згорткових, рекурентних, повнозв'язних, автоенкодерних і трансформерних моделей. Гнучкість - створення моделей як у стилі низькорівневого програмування (через TensorFlow Core API), так і у високорівневому форматі (через Keras). Масштабованість - запуск на CPU, GPU, TPU, підтримка розподіленого навчання. Мобільність - можливість експорту і запуску моделей на мобільних пристроях (TensorFlow Lite) або у веб-браузерах (TensorFlow.js). Інструменти візуалізації - інтеграція з TensorBoard для аналізу графіків навчання, втрат, точності, структури мережі та гіперпараметрів[7].

Архітектура

TensorFlow побудований навколо поняття обчислювального графа, де вузли представляють операції, а ребра - тензори (багатовимірні масиви даних), що передаються між ними.

Основні компоненти архітектури - Tensors - основна структура даних; багатовимірні масиви, аналогічні до NumPy-матриць, але здатні працювати на GPU/TPU. Operations (Ops) - математичні та логічні операції, які застосовуються до тензорів. Graphs - декларативне описання структури обчислень; можуть бути збережені та використані незалежно від коду. Sessions (у версії 1.x) - механізм виконання графа. У TensorFlow 2.x сесії більше не використовуються - замість них застосовується eager execution (динамічне виконання, схоже на PyTorch). Keras API - високорівнева обгортка, що дозволяє швидко створювати та навчати моделі без занурення у низькорівневу структуру графа[7].

Особливості

TensorFlow має ряд унікальних особливостей, які вирізняють його серед конкурентів - зробив фреймворк значно зручнішим у використанні, запровадивши

інтеграцію з Keras як офіційний API, спростивши синтаксис та полегшивши дебагінг. Розподілене навчання стало простішим завдяки tf.distribute, який дозволяє тренувати одну й ту саму модель паралельно на кількох GPU або навіть на кластерах. Сумісність із різними мовами програмування: хоча основна реалізація - на Python, TensorFlow також має API для JavaScript, Java, C++ та Swift. TensorFlow Hub - репозиторій попередньо навчених моделей, які можна легко завантажити й донавчати. Інтеграція з Google Cloud - особливо зручна для побудови масштабованих AI-сервісів у хмарі[11].

Загальний огляд PyTorch

PyTorch - це гнучкий фреймворк для глибинного навчання, розроблений Facebook AI Research (FAIR). Він швидко став одним з найпопулярніших інструментів у наукових дослідженнях і академічних колах завдяки своїй простоті, прозорості та можливості гнучкої розробки моделей «на ходу».

Його основна філософія - максимальна інтерактивність і прозоре управління обчисленнями, що робить PyTorch особливо привабливим для дослідників, які експериментують з новими архітектурами чи модифікаціями алгоритмів навчання.

Гнучкість

Однією з головних переваг PyTorch є динамічний обчислювальний граф (eager execution). На відміну від TensorFlow 1.x, де спочатку створюється граф, а потім виконується, у PyTorch операції виконуються негайно. Це дозволяє - налагоджувати модель прямо під час виконання, з повним доступом до проміжних результатів; будувати складні умовні логіки, цикли, рекурсії без потреби в додатковому API або переписуванні графів; легко змінювати архітектуру в реальному часі, що критично важливо у дослідницьких проєктах.

Завдяки цьому PyTorch став платформою «за замовчуванням» для реалізації новітніх дослідницьких моделей (BERT, GPT, Diffusion-моделі тощо) у наукових публікаціях[14].

Зручність у дослідженнях

Для наукової роботи PyTorch пропонує ряд рішень, які спрощують експерименти - зрозумілий і чистий API, заснований на стандартному синтаксисі

Python і NumPy; повна інтеграція з Jupyter Notebook, що дозволяє створювати інтерактивні прототипи моделей; torch.nn - модуль, що забезпечує зручне конструювання шарів нейронних мереж; torch.utils.data - гнучкий інструмент для роботи з наборами даних, включаючи кастомізовану обробку, батчинг і аугментацію.

Крім того, PyTorch має широку спільноту, активну підтримку у наукових колах та багато відкритих проєктів на GitHub. Існує велика кількість туторіалів, шаблонів і реалізацій з топових конференцій (NeurIPS, CVPR, ICML), які майже завжди написані саме на PyTorch[12].

Інструменти для продакшну

Хоча PyTorch історично асоціювався з дослідженнями, він активно розвивається і для продакшн-рішень - TorchScript дозволяє перетворювати PyTorch-моделі у формат, придатний до ефективного виконання поза середовищем Python; ONNX (Open Neural Network Exchange) - формат, що дозволяє експортувати моделі з PyTorch до інших фреймворків чи платформ; PyTorch Mobile - версія для мобільного використання; TorchServe - інструмент для розгортання моделей на сервері.

Таблиця 1.1

Порівняння TensorFlow та PyTorch

Критерій	TensorFlow	PyTorch
Розробник	Google Brain	Facebook AI Research (FAIR)
Основна філософія	Статичний граф (TF 1.x), імперативне виконання (TF 2.x)	Динамічний граф (eager execution)
Гнучкість у дослідженнях	Менша (до TF 2.0), зараз покращена	Висока - ідеально підходить для експериментів
Простота синтаксису	Дещо складніший, особливо на низькому рівні	Інтуїтивний, максимально наближений до Python/NumPy

Рівень абстракції	Keras (високий рівень), можливість працювати на низькому рівні	torch.nn - модульна побудова моделей
Розгортання моделей	TensorFlow Serving, TensorFlow Lite, TensorFlow.js	TorchServe, ONNX, PyTorch Mobile
Підтримка продакшну	Висока, добре інтегрується з Google Cloud	Зростає, особливо з TorchScript та ONNX
Інструменти візуалізації	TensorBoard	Підтримується через сторонні бібліотеки або TensorBoardX
Підтримка мобільних платформ	TensorFlow Lite	PyTorch Mobile
Поширеність у науці	Широке використання, особливо у компаніях	Домінує в академічних дослідженнях
Наявність готових моделей	TensorFlow Hub	Hugging Face, PyTorch Hub
Поріг входу для початківців	Зараз нижчий (завдяки Keras)	Дуже низький, зручний для навчання

Середовища для розробки і навчання моделей

Google Colab - це безкоштовне хмарне середовище для програмування на Python, яке надає можливості для запуску моделей штучного інтелекту та глибинного навчання, зокрема за допомогою графічних процесорів (GPU) і тензорних процесорів (TPU). Це дозволяє значно пришвидшити обчислення при навчанні великих моделей, таких як нейронні мережі.

Особливості:

- Безкоштовний доступ до GPU та TPU - надає безкоштовні ресурси для обчислень, що дуже корисно для розробників і дослідників, які не мають доступу до дорогих апаратних засобів.

- Підтримка Python - підтримує всі основні бібліотеки Python для науки про дані та глибинного навчання, зокрема TensorFlow, Keras, PyTorch, NumPy, Pandas.
- Інтерфейс Jupyter Notebook - використовує інтерактивне середовище на основі Jupyter Notebook, що дозволяє писати код, виконувати його і відразу бачити результати.
- Збереження в Google Drive - користувачі можуть зберігати свої роботи прямо в Google Drive, що полегшує доступ і співпрацю над проєктами.
- Інтеграція з іншими інструментами - легко інтегрується з іншими хмарними сервісами Google, такими як BigQuery, для обробки великих даних.

Це середовище ідеально підходить для тестування і прототипування моделей, а також для навчання на великих наборах даних, без необхідності у володінні дорогим апаратним забезпеченням.

Jupyter Notebook - це популярне інтерактивне середовище для наукових обчислень і розробки моделей, яке дозволяє писати, виконувати і документувати код Python. В основному воно використовується для локальної розробки моделей, де користувач самостійно встановлює всі необхідні бібліотеки та інструменти.

Особливості:

Інтерактивний інтерфейс - надає можливість комбінувати код, текстові пояснення та візуалізації в одному документі. Це дуже зручно для дослідників та розробників, оскільки дозволяє швидко тестувати і змінювати модель.

Підтримка широкого спектра бібліотек - інтеграція з такими бібліотеками, як NumPy, Pandas, Matplotlib, Seaborn, TensorFlow, PyTorch дозволяє реалізовувати та тестувати моделі безпосередньо у середовищі.

Локальний контроль - на відміну від хмарних сервісів, Jupyter Notebook працює локально на комп'ютері, що дає повний контроль над середовищем розробки і дозволяє оптимізувати ресурси для своїх задач.

Робота з різними форматами даних - Jupyter дозволяє працювати з різними форматами даних (CSV, Excel, JSON тощо), що спрощує підготовку даних для навчання моделей.

Можливість інтеграції з іншими середовищами - можна комбінувати роботу з Jupyter з іншими інструментами, такими як Docker або Conda, для створення ізольованих середовищ з певними версіями бібліотек.

Jupyter Notebook є чудовим вибором для тих, хто працює локально з невеликими або середніми наборами даних, потребує більше гнучкості в налаштуваннях середовища і має повний контроль над процесом розробки.

Ці два інструменти, Google Colab і Jupyter Notebook, є найбільш популярними середовищами для розробки та навчання моделей нейронних мереж, і кожен з них має свої переваги залежно від потреб користувача -доступність потужних обчислювальних ресурсів (GPU/TPU) у Colab або повний контроль над налаштуваннями середовища в Jupyter Notebook.

Обробка даних

NumPy - це основна бібліотека для наукових обчислень у Python. Вона надає потужні засоби для роботи з великими масивами і матрицями, а також числовими операціями над ними. Завдяки NumPy можна швидко виконувати арифметичні операції, лінійну алгебру, статистичні обчислення, що є основою для підготовки даних перед навчанням нейронних мереж.

Основні функції NumPy - створення та маніпуляція масивами; операції над багатовимірними масивами; розв'язування лінійних рівнянь, статистичні функції, перетворення даних.

Pandas - це бібліотека, яка є важливим інструментом для роботи з табличними даними. Вона дозволяє ефективно обробляти структуровані дані у вигляді DataFrame та Series. За допомогою Pandas легко проводити очистку, перетворення та маніпуляцію даними (фільтрація, групування, агрегування, злиття таблиць, тощо). Для глибинного навчання важливо попередньо обробити дані, і Pandas забезпечує швидкий і зручний доступ до всіх необхідних операцій.

Основні функції Pandas - створення та обробка DataFrame; очищення та трансформація даних; агрегування та групування даних для подальшого використання у моделях.

Візуалізація

Matplotlib - це найпопулярніша бібліотека для візуалізації даних у Python. Вона дозволяє створювати графіки, гістограми, діаграми розсіяння, лінійні графіки та інші типи візуалізацій для аналізу даних. Візуалізація даних допомагає краще зрозуміти структуру даних і надає корисну інформацію для подальшої побудови моделей[15].

Основні можливості Matplotlib - створення стандартних графіків (лінійних, стовпчикових, кругових, тощо); налаштування стилів графіків; додавання підписів, легенд та анотацій.

Seaborn - бібліотека для статистичної візуалізації, побудована на основі Matplotlib. Вона спрощує створення більш складних графіків, таких як теплові карти, коробкові діаграми, парні графіки, що дозволяє легше аналізувати взаємозв'язки між змінними та структуру даних[16].

Основні можливості - побудова кореляційних та теплових карт; створення стильних графіків за замовчуванням; візуалізація статистичних характеристик даних.

TensorBoard - це інструмент для візуалізації процесу навчання нейронних мереж, який інтегрується з TensorFlow. TensorBoard дозволяє відслідковувати тренування моделей, виводити графи обчислень, аналізувати зміни втрат (loss), точності (accuracy) під час навчання, а також візуалізувати активації та ваги нейронних мереж.

Основні можливості TensorBoard - візуалізація графа моделі; моніторинг метрик навчання (втрати, точність); порівняння тренувальних та валідаційних метрик; візуалізація гістограм та активаційних функцій.

Аугментація даних

Keras ImageDataGenerator - це інструмент для аугментації зображень, який дозволяє застосовувати трансформації до зображень під час їх завантаження для навчання. Аугментація допомагає збільшити різноманітність даних, що дозволяє зменшити перенавчання і покращити узагальнення моделі. ImageDataGenerator має безліч вбудованих функцій для обертання, масштабування, обрізки, відображення і зміни кольору зображень.

Основні можливості ImageDataGenerator - масштабування, обертання, зміщення, зрізання зображень; застосування випадкових трансформацій для кожного зображення під час навчання; підготовка даних для потокової передачі при роботі з великими наборами даних.

Albumentations - це ще одна бібліотека для аугментації зображень, яка надає великий набір методів для зміни зображень. Вона дозволяє створювати більш складні та випадкові трансформації, такі як випадкове обертання, зсув, зміну яскравості, контрасту, додавання шуму тощо. Albumentations є швидким і гнучким інструментом для обробки зображень і може бути використаний у проєктах, що вимагають високої продуктивності[17].

Основні можливості Albumentations - широкий набір операцій для зміни зображень: повороти, зміни кольорів, вирізи, зсуви; підтримка швидкої обробки зображень на GPU; просте комбінування декількох аугментацій у єдиний пайплайн.

Ці інструменти для обробки даних, візуалізації та аугментації є важливими для ефективного навчання моделей нейронних мереж, оскільки дозволяють підготувати дані, проаналізувати їх, покращити різноманітність набору даних і візуалізувати процеси навчання, що значно підвищує точність та стабільність моделей.

Вибір технологій для реалізації проєкту

Для реалізації проєкту з класифікації зображень обрано фреймворк TensorFlow з огляду на його стабільність, широку підтримку інструментів, а також потужні можливості для побудови, навчання та оптимізації глибоких нейронних мереж. TensorFlow активно розвивається компанією Google, що забезпечує йому високу якість підтримки, документування та сумісність із сучасним обладнанням (GPU/TPU).

Основні аргументи на користь TensorFlow[18]:

- Висока продуктивність - оптимізований для використання графічних прискорювачів, що пришвидшує навчання моделей.

- Підтримка масштабування - легко масштабується від ноутбука до кластерів серверів.
- Інтеграція з TensorBoard - вбудований інструмент для моніторингу процесу навчання в реальному часі.
- API високого рівня (Keras) - зручний для швидкого створення моделей без втрати гнучкості для складніших задач.
- Велика спільнота користувачів - наявність численних прикладів, гайдів, готових моделей.

Сумісність із наборами даних (CIFAR-10, MNIST)

TensorFlow має пряму інтеграцію з популярними наборами даних, такими як CIFAR-10 та MNIST, через модуль tf.keras.datasets, що дозволяє легко імпортувати, масштабувати й підготувати ці дані до навчання.

MNIST - це стандартний набір для розпізнавання рукописних цифр (28x28 пікселів, відтінки сірого), що ідеально підходить для тестування простих моделей.

CIFAR-10 - це набір із 60 тисяч кольорових зображень розміром 32x32 пікселі, розподілених по 10 класах, який є більш складним і реалістичним варіантом для перевірки ефективності моделей у задачах класифікації.

TensorFlow дозволяє легко завантажити, нормалізувати та змінити структуру цих наборів без додаткової обробки, що зменшує час підготовки.

Переваги в контексті завдань класифікації зображень

TensorFlow демонструє високу ефективність саме у завданнях класифікації зображень завдяки таким можливостям - підтримка згорткових нейронних мереж (CNN) через Keras API, що є стандартом для обробки зображень; готові архітектури (ResNet, VGG, Inception, MobileNet) у бібліотеці tf.keras.applications, які можна легко адаптувати або використовувати з перенавчанням; інструменти для аугментації (ImageDataGenerator, tf.data API) дозволяють ефективно збільшити обсяг і різноманітність навчальних зразків; можливість розгортання моделі через TensorFlow Lite або TensorFlow.js - що робить модель придатною для мобільних і веб-додатків після навчання[19].

Таким чином, TensorFlow є повноцінним середовищем для реалізації всієї лінійки процесів - від підготовки даних до тренування і розгортання моделей класифікації зображень, що робить його оптимальним вибором для даного проєкту.

РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА РОЗРОБКА НЕЙРОМЕРЕЖІ ДЛЯ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ НА ОСНОВІ TENSORFLOW

2.1. Проєктування архітектури та алгоритмів роботи нейромережі

Ця кваліфікаційна робота досліджує використання згорткових нейронних мереж (CNN) для розробки автоматизованої системи для розпізнавання емоцій із зображень обличчя. Основна мета - автоматичне визначення емоційного стану людини на основі аналізу її міміки як на статичних зображеннях, так і в реальному часі через веб-камеру. Для виконання цього завдання було використано відкритий набір даних FER2013, який містить понад 35 000 зображень у градаціях сірого з роздільною здатністю 48x48 пікселів. Кожне зображення містить фрагмент людського обличчя та описується з відповідною емоцією. Дослідження вивчало сім основних емоцій: гнів, огиду, страх, радість, смуток, здивування та нейтральність.

На основі згаданого набору даних FER2013 у цій роботі була побудована модель глибокого навчання, реалізована як згорточна нейронна мережа (CNN). Основна мета створення такої моделі - автоматичне розпізнавання емоційного стану людини за допомогою аналізу зображення її обличчя. Для досягнення цієї мети модель повинна не тільки розпізнавати основні риси зображення, але й формулювати узагальнення на основі складних структур, властивих виразу обличчя людини.

Архітектура побудованої нейронної мережі передбачає послідовне проходження даних через ряд спеціалізованих рівнів, кожен з яких виконує свою унікальну функцію в загальній обробці вхідного зображення. На першому етапі модель отримує зображення обличчя в сірому форматі, яке має фіксовану роздільну здатність 48x48 пікселів. Вибір цього формату був невипадковим - малий розмір дозволяє істотно знизити обчислювальне навантаження, що особливо важливо на етапі навчання, не втрачаючи ключових елементів, необхідних для точного розпізнавання емоцій.

Потім зображення надсилається на перший згортковий шар (Conv2D), який складається з 64 фільтрів 3x3. Використовуючи функцію активації ReLU (Rectified Linear Unit), яка змінює всі негативні значення на нуль, мережа отримує можливість

моделювати складні нелінійні залежності. Цей шар відповідає за виявлення найпростіших, основних характеристик зображення, таких як краї, кути, лінії та загальні контури обличчя. Ці особливості відіграють надзвичайно важливу роль, оскільки є основою для подальшого, більш глибокого аналізу будови обличчя.

За згортковим шаром слідує шар просторової підвибірки - MaxPooling. Цей шар служить для зменшення просторових розмірів карти ознак, що дозволяє значно зменшити кількість параметрів, які потребують обробки, зберігаючи найважливіші елементи. Його використання знижує ризик переобладнання, оскільки модель рідше «запам’ятовує» шум або дрібні деталі зображення, зосереджуючись лише на дійсно важливих функціях. Крім того, використання кластеризації прискорює обчислення, роблячи модель ефективнішою та зручнішою для використання в режимі реального часу, наприклад, у поєднанні з відеопотоком із веб-камери.

Загалом ця архітектура забезпечує базовий, але ефективний підхід до обробки візуальних даних для завдань класифікації, особливо розпізнавання емоцій. Наступні рівні мережі, включаючи додаткові згорткові блоки, векторні згорткові шари, щільні шари та функцію softmax на виході, забезпечують остаточний прогноз. Однак перші згорткові та ансамблеві шари мають вирішальне значення для закладення основи для ефективного розпізнавання, оскільки вони вчать модель «бачити» обличчя та розуміти його структуру з точки зору машинного аналізу. На наступних етапах роботи нейронної мережі вхідне зображення глибоко аналізується для виявлення більш складних і загальних рис, притаманних людським емоціям. Після того, як перші шари виявляють основні функції, такі як грані, лінії та контури, наступні згорткові та ансамблеві шари зосереджуються на вивченні глибших візерунків. Це можуть бути, наприклад, такі характерні ознаки, як положення брів, форма очей, напруга м'язів навколо рота або інших частин обличчя, які є вирішальними в розрізненні емоцій - наприклад, страху від подиву або радості від нейтрального виразу обличчя.

Щоб перейти від просторового представлення зображення до етапу, де можна прийняти остаточне рішення щодо емоцій, необхідно змінити формат даних. Для цього використовується шар Flatten - спеціальна операція, яка «розгортає»

багатовимірну карту ознак у звичайний одновимірний вектор. Таке перетворення є необхідною умовою для подачі даних у повністю зв’язані (щільні) шари, які працюють із векторами, а не з матрицями.

Після цього перетворення створюється перший щільний шар, що містить 512 нейронів. Кожен з цих нейронів отримує на вхід всі значення з попереднього шару і має функцію активації ReLU, яка, як і раніше, відповідає за внесення нелінійності в обчислення. Цей рівень відповідає за «збір» і об’єднання ознак вищого порядку, побудованих на ознаках нижчого порядку, а також дозволяє моделі розпізнавати більш тонкі та детальні моделі, які вказують на емоційний стан людини. У результаті створюється більш цілісна картина структури зображення з точки зору класифікації.

Щоб запобігти переобладнанню, на цьому етапі до моделі додається шар Dropout із імовірністю 0,5. Цей механізм випадковим чином вимикає половину нейронів у даному шарі під час кожної ітерації навчання. Це дозволяє моделі не покладатися надто на окремі шляхи інформації, що збільшує її здатність до узагальнення та зменшує ризик втрати якості під час роботи з новими, раніше невидимими зображеннями. Виключення стало одним із найефективніших методів регулярізації в глибокому навчанні, і ця теза допомагає зробити модель більш надійною для практичного використання. Останнім етапом побудови нейронної мережі є додавання ще одного щільного шару, який діє як класифікатор. Цей шар містить 7 нейронів, кожен з яких відповідає одній із досліджуваних моделлю емоційних категорій - гніву, огиди, страху, радості, смутку, здивування та нейтральності. Щоб забезпечити правильну інтерпретацію результатів класифікації, на виході цього рівня застосована функція активації softmax. Перетворює вихідні значення кожного нейрона в ймовірності, які в сумі дорівнюють 100%, дозволяючи визначити, яка емоція найбільш ймовірна для поточного зображення. Це робить підхід моделі до класифікації інтуїтивно зрозумілим і практичним, напр. для програмного виявлення емоцій у відеозв'язку або інтерактивних системах.

Процес навчання цієї моделі супроводжувався детальним налаштуванням гіперпараметрів, які безпосередньо впливають на продуктивність і стабільність її роботи. Для оптимізації ваг використовувався алгоритм Адама, сучасний

стохастичний оптимізатор, що поєднує в собі переваги моментних методів і адаптивного контролю швидкості навчання. Це забезпечує швидку та надійну збіжність у процесі навчання, особливо в задачах із великою кількістю параметрів, наприклад у згорткових мережах.

Вибір функції втрат базувався на категорійній перехресній ентропії (categorical_crossentropy), яка є стандартною для задач багатокласової класифікації. Це дозволяє розрахувати ступінь розбіжності між справжніми та прогнозованими емоціями, щоб можна було спрямувати процес навчання, щоб мінімізувати цю розбіжність. Як показник якості було обрано точність, яка відображає відсоток правильно класифікованих зразків і дозволяє зручно відстежувати прогрес моделі під час навчання.

Модель тренувалась протягом 10 епох, тобто десяти повних проходів через весь набір даних. Це число було обрано як відправну точку для побудови основних класифікаційних можливостей моделі без надмірного ризику переобладнання. Розмір пакету становив 64 зображення, що дозволяє ефективно використовувати ресурси обчислювальної системи та підтримувати стабільне оновлення ваги.

Під час навчання продуктивність моделі безперервно контролювалася шляхом побудови функцій втрат і точності як для навчальної, так і для перевірочної вибірки. Це дозволяло візуально контролювати динаміку тренування, виявляти моменти стабілізації або, навпаки, ознаки перетренованості та приймати рішення щодо можливих змін параметрів. Щоб об’єктивно оцінити здатність моделі до узагальнення, 20% усієї навчальної вибірки було відведено для перевірки. Ці дані не використовувалися для оновлення параметрів і служили лише для перевірки якості класифікації на «невідомих» прикладах.

Після процесу навчання модель показала досить стабільні результати - точність даних перевірки становила від 65% до 70%. Цей результат вважається типовим для базових моделей CNN, які ще не використовують методи розширення даних, балансування класів або більш просунуті архітектури, такі як ResNet або Inception. Однак навіть без цих ускладнень модель демонструє задовільний рівень

продуктивності, що дозволяє розглядати її як основу для реальних додатків або подальшої оптимізації.

Останній етап тестування включав перевірку моделі на зображеннях, які не входили до набору для навчання або перевірки. Такі тести імітують реальні умови роботи системи, коли на вході з'являються нові, раніше не бачені сигнали. Модель успішно виконала ці завдання, продемонструвавши здатність не тільки знаходити обличчя, але й правильно визначати їхній емоційний стан, незважаючи на незначні відмінності у виразі обличчя, освітленні чи якості зображення. Це підтверджує практичну ефективність створеної моделі в задачах розпізнавання емоцій як на фотографіях, так і в відеопотоках реального часу.

2.2. Архітектура нейромережі, вибір середовища розробки, тестування та оцінка результатів роботи

Для створення цієї моделі були використані наступні інструменти та техніки. Основною мовою програмування є Python 3 завдяки своїй простоті та розгалуженій екосистемі бібліотек ШІ. Використовувані фреймворки включають TensorFlow для побудови нейронних мереж і машинного навчання, а також Keras, який є високорівневою оболонкою для TensorFlow, яка значно спрощує процес створення та навчання моделей. Використовуйте OpenCV для завантаження та обробки зображень, визначення облич за допомогою каскадів Хаара та візуалізації результатів розпізнавання. NumPy використовується для обробки числових масивів і перетворення зображень у піксельні дані, нормалізації та форматування їх перед подачею в модель. Matplotlib використовується для побудови та візуалізації результатів навчання, таких як графіки точності та функції втрат. Розробка виконується в середовищі Visual Studio Code, яке має зручну підтримку Python, інтеграцію терміналу, автозавершення, плагіни Jupyter, Git та інші корисні функції. Операційна система - Windows 11.

Лістинг1 – бібліотеки

os - стандартна бібліотека Python для роботи з операційною системою.

cv2 (OpenCV) - бібліотека для обробки зображень і відео.

numpy - робота з масивами та матрицями, математичні операції.

pandas - обробка та аналіз табличних даних (DataFrame).

tensorflow - фреймворк для створення та навчання моделей машинного навчання, зокрема нейронних мереж.

Sequential, Dense, Conv2D, MaxPooling2D, Flatten, Dropout - компоненти Keras (в складі TensorFlow) для побудови нейронних мереж.

to_categorical -перетворення міток у формат one-hot для класифікації.

train_test_split - зручна функція з sklearn для поділу даних на тренувальний і тестовий набори.

Adam - популярний оптимізатор для навчання нейронних мереж.

tkinter - бібліотека для створення графічного інтерфейсу користувача (GUI).

PIL (Image, ImageTk) - модулі бібліотеки Pillow для обробки зображень і їх інтеграції з tkinter.

Лістинг2 - розширення файлів

SUPPORTED_FORMATS - список форматів файлів зображень, які підтримуються програмою.

EMOTION_LABELS - словник, який відображає числові мітки (класи) на назви емоцій.

Лістинг3 - тренування моделі розпізнавання емоцій на основі датасету fer2013.csv

```
def train_emotion_recognition_model():
    print("Тренуємо модель для розпізнавання емоцій...")
    data = pd.read_csv("fer2013.csv")
    X, y = [], []
    for _, row in data.iterrows():
        pixels = np.array(row['pixels'].split(' '), dtype='float32').reshape(48, 48, 1)
        X.append(pixels)
        y.append(row['emotion'])
    X, y = np.array(X) / 255.0, to_categorical(np.array(y), num_classes=7)
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

```
        model = Sequential([
            Conv2D(64, (3, 3), activation='relu', input_shape=(48, 48, 1)),
            MaxPooling2D(2, 2),
            Conv2D(128, (3, 3), activation='relu'),
            MaxPooling2D(2, 2),
            Flatten(),
            Dense(512, activation='relu'),
            Dropout(0.5),
            Dense(7, activation='softmax')
        ])
        model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
        model.fit(X_train, y_train, epochs=10, batch_size=64, validation_data=(X_test, y_test))
        model.save("emotion_model.h5")
        print("Модель збережена!")
```

Лістинг4 - розпізнавання емоцій на зображеннях у папці images

```
    def detect_emotions_on_images():
        print("Аналізуємо зображення у директорії `images/`...")
        model = tf.keras.models.load_model("emotion_model.h5")
        face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
        originals = []
        processed = []
        for fname in os.listdir("images"):
            ext = os.path.splitext(fname)[-1].lower()
            if ext not in SUPPORTED_FORMATS:
                continue
            path = os.path.join("images", fname)
            img = cv2.imread(path)
```

```
            if img is None:
                continue
            gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
            faces = face_cascade.detectMultiScale(gray, 1.1, 4)
            img_with_boxes = img.copy()
            for (x, y, w, h) in faces:
                face = gray[y:y+h, x:x+w]
                face = cv2.resize(face, (48, 48)) / 255.0
                face = face.reshape(1, 48, 48, 1)
                prediction = model.predict(face, verbose=0)
                emotion = EMOTION_LABELS[np.argmax(prediction)]
                cv2.rectangle(img_with_boxes, (x, y), (x+w, y+h), (0, 0, 255), 2)
                cv2.putText(img_with_boxes, emotion, (x, y - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.9, (0, 0, 255), 2)
            originals.append(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
            processed.append(cv2.cvtColor(img_with_boxes, cv2.COLOR_BGR2RGB))
        show_images_gui(originals, processed)
```

Лістинг5 - графічний інтерфейс, який відображає зображення до та після обробки

```
    def show_images_gui(originals, processed):
        root = Tk()
        root.title("Результати розпізнавання емоцій")
        canvas = Canvas(root)
        scrollbar = Scrollbar(root, orient=VERTICAL, command=canvas.yview)
        scrollable_frame = Frame(canvas)
        scrollable_frame.bind(
            "<Configure>",
            lambda e: canvas.configure(scrollregion=canvas.bbox("all"))
        )
        canvas.create_window((0, 0), window=scrollable_frame, anchor="nw")
```

```
    canvas.configure(yscrollcommand=scrollbar.set)
    canvas.pack(side=LEFT, fill=BOTH, expand=True)
    scrollbar.pack(side=RIGHT, fill=Y)
    for i in range(len(originals)):
        for col, img_list in enumerate([originals, processed]):
            img = Image.fromarray(img_list[i])
            h_desired = 500
            ratio = h_desired / img.height
            w_new = int(img.width * ratio)
            img_resized = img.resize((w_new, h_desired))
            photo = ImageTk.PhotoImage(img_resized)
            label = Label(scrollable_frame, image=photo)
            label.image = photo
            label.grid(row=i, column=col, padx=10, pady=10)
    root.mainloop()
```

Лістинг6 - функція, що запускає програму в одному з двох режимів

```
def main():
    print("Оберіть режим:")
    print("1. Розпізнавання емоцій на зображеннях (з директорії `images/`)")
    print("2. Тренування моделі")
    choice = input("Ваш вибір (1/2): ")
    if choice == "1":
        detect_emotions_on_images()
    elif choice == "2":
        train_emotion_recognition_model()
    else:
        print("Невірний вибір.")
```

Лістинг7 - стандартна конструкція Python, яка означає - виконати main(), якщо цей файл запущено напряму

```
if __name__ == "__main__":
    main()
```

Реалізовано текстовий інтерфейс(див. рис. 2.1), де користувач може вибрати режим роботи. У режимі розпізнавання емоцій зображення завантажуються з локального сховища, обличчя виявляються за допомогою каскадів Хаара, області обличчя масштабуються до 48x48 пікселів і перетворюються на відтінки сірого, а потім емоційний стан класифікується за допомогою попередньо навченої моделі. Після розпізнавання зображення відображає рамку навколо обличчя та текстовий підпис із назвою розпізнаної емоції. Такий підхід дозволяє протестувати модель на будь-якому окремому зображенні та візуально оцінити точність класифікації.

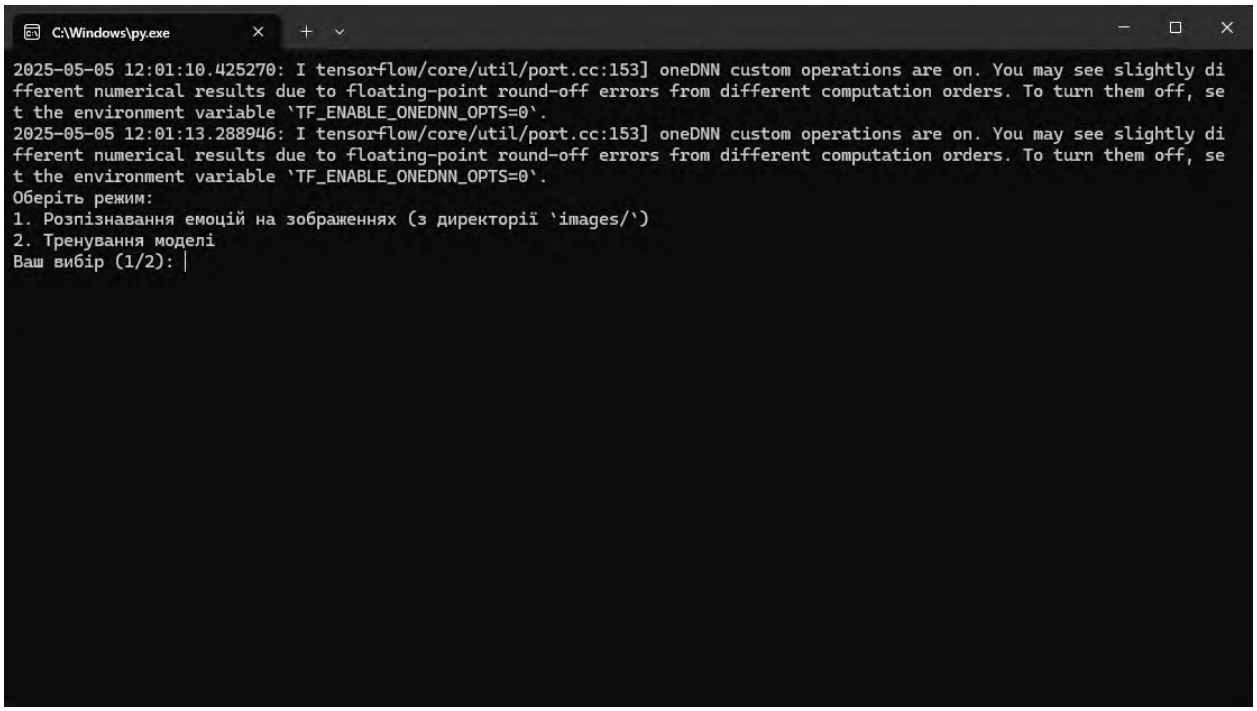


Рис. 2.1 Текстовий інтерфейс з вибором режиму роботи

Щоб перевірити якість моделі, 20% даних було відібрано як вибірку перевірки. Під час навчання(див. рис. 2.2) відображаються графіки функції втрат і точності, що дозволяє нам відстежувати прогрес і збіжність моделі. Модель показала стабільну продуктивність з точністю в діапазоні 65-70% на зразках перевірки, що типово для базових згорткових нейронних мереж без використання складних методів, таких як розширення або більш глибокі архітектури. Підсумкове тестування проводилося на

окремих зображеннях, які не входили до навчального набору, і підтвердило стабільність роботи моделі та її здатність точно класифікувати емоції.

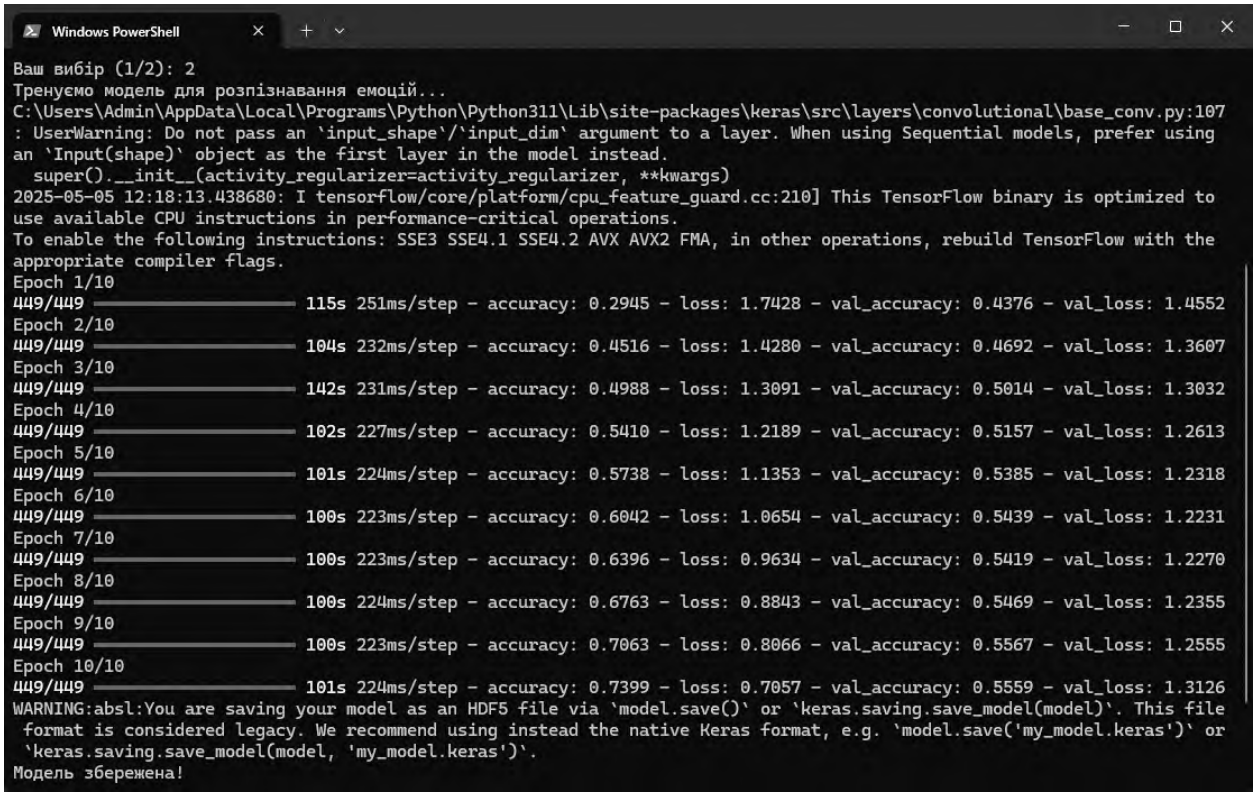


Рис. 2.2 Тренування моделі

При побудові моделі емоційного інтелекту використовується кілька критичних параметрів і місць, кожен з яких має значний вплив на якість і послідовність нейронної мережі. Розглянемо кожен з них докладніше.

Оптимізатор Adam (Adaptive Moment Estimation) вважається однією з найпопулярніших і значущих стратегій глибокого навчання. Його переваги полягають у тому, що він автоматично змінює швидкість навчання для кожного параметра в моделі окремо, використовуючи комбінацію переваг алгоритмів AdaGrad і RMSProp. Це сприяє швидкому наближенню моделі до мінімуму функції втрат у складних, нерівномірних просторах для розподілу даних. У результаті модель може навчатися більш послідовно та ефективніше зі складними візерунками в зображеннях.

Функція втрат categorical_crossentropy використовується для оцінки невідповідності між прогнозованим розподілом ймовірностей і фактичними класифікаціями в сценаріях класифікації з кількома класами. У даному випадку це 7 різних емоцій, кожна з яких представлена як окремий клас. Ця функція обчислює

ступінь наближення мережі до правильної відповіді та допомагає виправити ваги мережі за допомогою поширення помилки. Це особливо ефективно, коли вихідний рівень мережі складається з активації softmax, як у випадку з нашою моделлю.

Для визначення ефективності класифікації використовується метрика точності. Обчислюється як відсоток прикладів, які правильно класифіковано, до загальної кількості прикладів. Це наочний і простий показник, який полегшує оцінку ефективності моделі на наборах для навчання та перевірки. Незважаючи на неврахування дисбалансу класів, набір у FER2013 досить рівномірний, тому метрика для цього підходить.

Навчання складається з 10 епох, що означає, що модель буде проходити весь набір даних навчання 10 разів. Це невелика цифра, яка представляє базову здатність до вивчення моделі. Не варто перетренувати модель. Крім того, цього достатньо для того, щоб модель навчилася розпізнавати основні моделі та особливості кожної емоції. За бажання кількість епох можна збільшити на наступних етапах для досягнення вищого ступеня точності.

Обсяг батчу 64 або менше. 64 зображення обробляються паралельно під час кожної ітерації навчання. Цей розмір є компромісом між ефективністю навчання та обсягом використовуваної оперативної пам’яті. З одного боку, більший розмір партії призводить до більш точної оцінки градієнта, це зменшує шум, з іншого боку, це вимагає більше ресурсів. Вибране значення є вигідним для більшості застосувань CNN на зображеннях середнього розміру.

Комбінація цих середовищ дозволяє моделі постійно тренуватися, мати прийнятний ступінь точності без переобладнання, а також показувати хороші результати на даних перевірки та під час тестування в реальних умовах.

Тестування програми розпізнавання емоцій було проведено після завершення етапу навчання моделі. Основною метою випробувань було перевірити, наскільки добре розроблена модель справляється із завданням класифікації емоцій, що містяться в нових, раніше невідомих зображеннях. Для цього ми використовували як окремі зображення з FER2013, які не увійшли до навчальної вибірки, так і

зовнішні зображення у форматах JPG, PNG і WEBP, що дозволило перевірити модель на практичних прикладах поза стандартним набором даних.

Під час тестування програма взяла зображення обличчя як вхідні дані, автоматично масштабувала його до потрібного розміру 48x48 пікселів, перетворила в градації сірого, а потім передала в згорткову нейронну мережу для класифікації. Модель обробила зображення та повернула прогнозовану емоцію з найвищою ймовірністю. Результати відображалися в зручному графічному інтерфейсі: ліворуч відображалося оригінальне зображення, а праворуч - таке ж зображення, але з підписом, що описує певну емоцію(див. рис. 2.3).

Крім статичних зображень, тести також проводилися в режимі реального часу за допомогою веб-камери. Кожен кадр відео обробляли: обличчя ідентифікували, обрізали, змінювали розмір, а потім модель класифікувала емоцію. Конкретна емоція відображалася у вигляді тексту або емодзі безпосередньо у відеопотоці, тож можна було негайно побачити результат. Під час випробувань підтверджено стабільну роботу моделі та високу швидкість обробки зображень у реальному часі. Програма правильно розпізнала основні емоції, особливо радість, гнів і нейтральність. Під час класифікації схожих виразів обличчя виникли деякі помилки, наприклад смуток і страх, але загальний рівень точності відповідав очікуванням для моделі такого рівня складності. Тести підтвердили, що програма готова до використання в реальних умовах і може ефективно виконувати завдання розпізнавання емоцій.

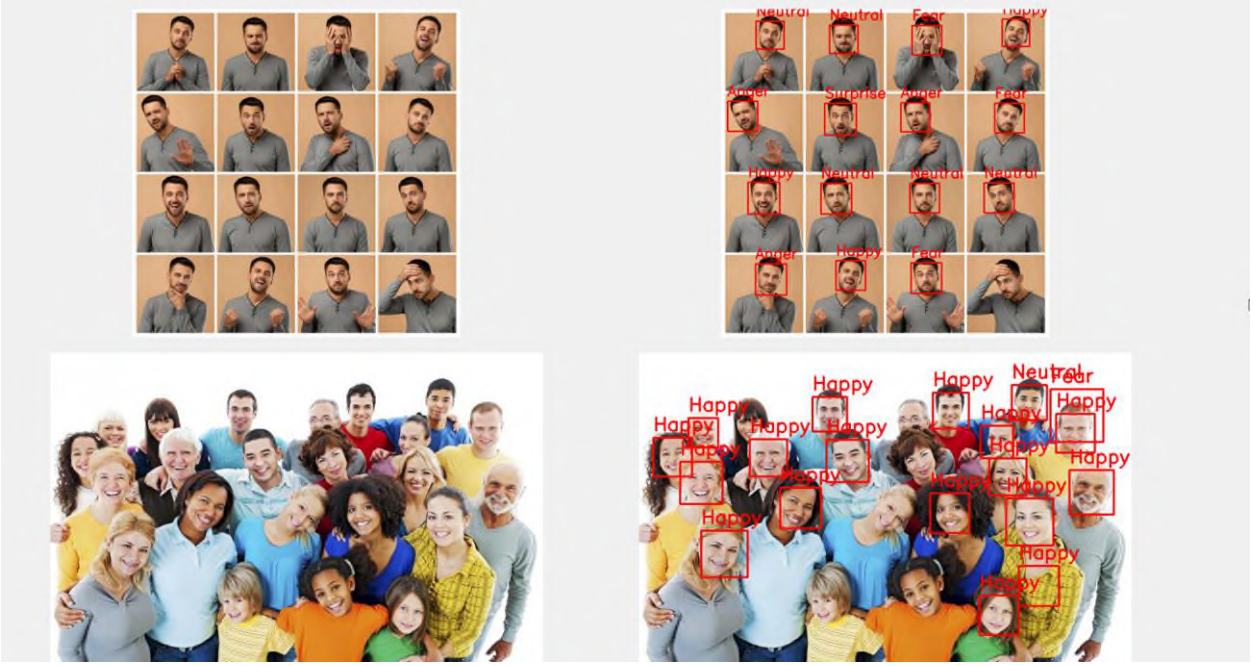


Рис 2.3 Результат розпізнавання емоцій

Ефективне навчання згорткової нейронної мережі сильно залежить від якості вхідних даних, що надаються на її вхід. Якщо дані шумні, неузгоджені або мають непослідовний формат, модель може навчитися неправильним зв’язкам або взагалі не зійтися під час навчання. Тому надзвичайно важливо виконати етап попередньої обробки даних перед навчанням, включаючи нормалізацію, стандартизацію, зменшення до єдиного розміру, перетворення міток у відповідний формат і поділ набору на частини для навчання, перевірки та тестування.

У цій роботі використано відомий відкритий набір зображень FER2013 (Facial Expression Recognition 2013), який є стандартним орієнтиром у задачах розпізнавання емоцій за зображеннями обличчя. Набір був представлений у рамках конкурсу Kaggle «Challenges in Representation Learning» і з тих пір широко використовується в дослідженнях і розробках комп’ютерного зору. База даних FER2013 містить загалом 35 887 зразків зображень обличчя у відтінках сірого, кожне розміром 48×48 пікселів. Зображення були автоматично зібрані за допомогою API пошуку зображень Google, потім попередньо відфільтровані та вручну позначені, щоб переконатися, що вони відповідають семи основним категоріям емоцій. Кожне зображення в наборі асоціюється з однією з наступних емоцій: гнів, огида, страх, радість, смуток, здивування та нейтральні. У цій моделі ці емоції представлені числовими індексами від 0 до 6. Кожне зображення у файлі набору даних

зберігається у форматі послідовності пікселів - один рядок чисел, що представляє значення яскравості кожного пікселя зображення (від 0 до 255). Такий підхід дозволяє зберігати дані у форматі таблиці CSV, що зручно для подальшої обробки програмним забезпеченням. Окрім піксельної інформації, кожен запис у файлі також містить числову мітку емоції та вказівку на те, чи зразок належить до певної підмножини - «Training», «PublicTest» або «PrivateTest», відповідно до наборів для навчання, перевірки та тестування. Це робить набір даних FER2013 добре структурованим джерелом для завдань автоматизованого розпізнавання емоцій, а завдяки рівномірному розміру зображення та наявності міток він дозволяє швидко готувати дані для введення в згортову нейронну мережу. У поєднанні з грамотною попередньою обробкою це створює умови для ефективного навчання моделі та досягнення високої точності класифікації.

Набір даних FER2013 було завантажено у форматі CSV, структура якого проста, але достатньо інформативна для попередньої обробки та подальшого навчання моделі. Кожен запис у цьому файлі містить три ключові поля: перше - це мітка емоції, представлена цілим числом від 0 до 6, що відповідає одній із семи категорій емоцій; другий - це рядок пікселів, що містить 2304 значення (тобто 48×48) у вигляді чисел, розділених пробілами, кожне з яких представляє яскравість одного пікселя в градаціях сірого від 0 до 255; Третє поле вказує, чи належить вибірка до однієї з трьох підмножин: «Навчання», «Відкритий тест» або «Приватний тест». Виходячи з цього поділу, весь набір був автоматично розділений на три частини: навчання (Training), перевірка (PublicTest) і тестування (PrivateTest). Такий поділ типовий для завдань машинного навчання: навчальний набір використовується для безпосереднього навчання моделі, набір перевірки використовується для вибору гіперпараметрів і моніторингу процесу навчання, щоб запобігти переобладнанню, а тестовий набір використовується для остаточної оцінки якості моделі на нових, невідомих даних. У FER2013 ці частини вже збалансовані та містять приблизно 28 709 зображень для навчання, 3 589 для перевірки та 3 589 для тестування.

Наступним кроком було перетворення кожного зображення з текстового рядка в двовимірну структуру чисел. Кожен рядок пікселів був розділений на окремі

значення, які потім зберігалися в масиві NumPy, а потім перетворювалися в матрицю 48×48. Це дозволило перейти від плоского текстового представлення до зручної для обчислень, структурованої форми, яка відповідає природному формату зображень. Потім усі значення пікселів були нормалізовані - тобто поділені на 255, щоб привести їх до інтервалу [0, 1]. Нормалізація є стандартною практикою при роботі з нейронними мережами, оскільки вона сприяє стабільності процесу навчання. Зокрема, це дозволяє уникнути надмірного випадання у вхідному шарі та покращує конвергенцію градієнтного спуску. Останнім кроком попередньої обробки було створення тензорів із тривимірною структурою, очікуваною згортковою нейронною мережею. Кожне зображення було зменшено до формату (48, 48, 1), де третій вимір вказує на кількість каналів. У випадку FER2013 цей канал є єдиним, оскільки зображення представлені у відтінках сірого. Загалом, весь набір даних перетворено на чотиривимірну структуру (num_samples, 48, 48, 1), яка повністю сумісна з вхідним рівнем згорткових мереж у TensorFlow або Keras. Ця уніфікована форма забезпечує ефективну обробку даних під час пакетної обробки в процесі навчання моделі. Щоб ефективно використовувати оригінальні мітки емоцій у процесі навчання моделі, числові значення класу (від 0 до 6) були перетворені у формат одноразового кодування. Це означає, що кожна мітка була представлена у вигляді вектора з довжиною, яка дорівнює числу можливих класів - у даному випадку семи. У такому векторі всі значення, крім одного, дорівнюють нулю; індивід знаходиться в позиції, що відповідає певному класу емоцій (наприклад, «радість» або «печаль»). Завдяки цьому підходу модель може працювати з категоріальними мітками як векторами та застосовувати категориальну функцію втрат крос-ентропії, яка є стандартною для задач багатокласової класифікації. Щоб покращити здатність моделі узагальнювати нові приклади та зменшити ризик переобладнання, до навчального набору було застосовано розширений аналіз зображень.

Доповнення - це техніка штучного розширення набору даних шляхом створення нових зразків із наявних зображень за допомогою випадкових змін. У цій роботі для реалізації розширення використовувався інструмент ImageDataGenerator з бібліотеки Keras, що дозволяє змінювати зображення в реальному часі, коли вони

передаються на модель. Застосовувані трансформації включають випадкові горизонтальні відображення зображень, що особливо корисно у випадку з особами, оскільки симетрія обличчя дозволяє зберегти смислове навантаження навіть після дзеркального відображення. Також були застосовані незначні зсуви по ширині та висоті - тобто випадкове переміщення об’єкта на зображенні для імітації зміни положення обличчя в кадрі. Крім того, було зроблено зміни яскравості, щоб зробити модель стійкою до різних умов освітлення, які можуть виникнути в реальних даних. Це розширення значно збільшує різноманітність прикладів у навчальному наборі без необхідності збирати додаткові зображення. Це позитивно вплинуло на якість навчання, оскільки дозволило моделі краще адаптуватися до можливих змін зовнішнього вигляду обличчя, не втрачаючи здатності правильно розпізнавати емоції.

Після завершення всіх етапів попередньої обробки було створено остаточну декомпозицію набору даних, яка стала основою для подальшого навчання, перевірки та тестування моделі. Загальна кількість зображень у повному наборі даних FER2013 становить 35 887 прикладів. Вони були розділені на три підмножини залежно від призначення кожного зразка: навчання, перевірка та тестування. Найбільшу частину становив навчальний набір, який містив 28 709 зображень. Саме з цього набору модель вчиться розпізнавати емоції, поступово оновлюючи свої внутрішні параметри на основі помилок, виявлених під час прогнозування. Набір перевірки, що складається з 3589 зображень, діє як свого роду проміжна контрольна точка - модель не бачить ці зразки безпосередньо під час навчання, але вони використовуються для оцінки якості її прогнозів під час процесу налаштування. Це дозволяє своєчасно виявити надмірне або недостатнє оснащення та допомагає вибрати оптимальні гіперпараметри моделі, такі як кількість епох, швидкість навчання, розмір партії тощо. Було вибрано окремий тестовий набір, який також містить 3589 зображень. Він повністю ізольований від процесу навчання і використовується лише після завершення навчання, щоб остаточно оцінити здатність моделі до узагальнення. Іншими словами, саме на цьому наборі ми перевіряємо, наскільки ефективно модель розпізнає емоції в нових, раніше

невидимих прикладах. Такий прозорий і збалансований розподіл даних забезпечує об'єктивність при оцінці моделі, оскільки дозволяє розділити процес навчання та валідації на незалежні етапи. Це, у свою чергу, дає більш реалістичну картину того, як модель поводитиметься під час практичного використання, коли дані, з якими вона працюватиме, будуть абсолютно новими та потенційно відрізнятимуться від даних, які вона використовувала під час навчання.

Оптимізація моделі та вибір гіперпараметрів відіграють надзвичайно важливу, якщо не вирішальну роль у створенні ефективної глибокої нейронної мережі, яка здатна точно, стабільно та загально класифікувати емоції із зображень обличчя. Цей процес є не лише технічним етапом, а й невід’ємною частиною наукового дослідження, оскільки правильний підбір параметрів багато в чому визначає здатність моделі до навчання, швидкість її збіжності та кінцеву якість класифікації.

Під час проекту особлива увага приділялася мінімізації кількості помилок, які допускає модель при класифікації емоцій. Це включало ретельне тестування архітектурних рішень, вибір функцій втрат, налаштування алгоритмів оптимізації та збалансований підхід до регуляризації та обробки вхідних даних. Все це було зроблено для того, щоб модель не тільки ефективно «засвоювала» навчальні приклади, але й показувала надійні результати при роботі з новими, раніше не баченими зображеннями. Ця здатність до узагальнення має вирішальне значення для будь-якої моделі машинного навчання, особливо в контексті реальних додатків.

Одним із найбільших ризиків під час навчання моделі є переобладнання, коли нейронна мережа починає запам’ятовувати особливості окремих прикладів із навчального набору настільки точно, що втрачає здатність узагальнювати шаблони. Це забезпечує високу точність навчальних даних, але значне зниження продуктивності під час перевірки чи тестування. Щоб уникнути цієї проблеми, було використано низку методів, включаючи регулярне відключення, збільшення даних і використання механізмів ранньої зупинки.

Важливою частиною оптимізації було визначення кількості епох, тобто повних проходів через навчальну множину. Занадто мало епох може призвести до недостатнього навчання, що означає, що модель не встигла вивчити всі важливі

закономірності в даних. З іншого боку, надмірна кількість може призвести до перетренованості. Після проведення численних експериментів та аналізу графіків втрат і точності кількість епох було обрано в діапазоні 50 -100, що дозволяє досягти оптимального балансу між якістю навчання та здатністю до узагальнення. Щоб контролювати цей процес, реалізовано механізм ранньої зупинки, який відстежує зміни в точності набору перевірки та припиняє навчання, якщо протягом кількох послідовних епох не спостерігається покращення. Іншим ключовим гіперпараметром, який впливає на динаміку навчання, є розмір партії - кількість прикладів, які одночасно обробляються під час одного кроку оновлення ваги моделі. Було встановлено, що розмір партії 32 або 64 забезпечує оптимальну продуктивність, забезпечуючи збалансовану швидкість обробки та стабільність градієнтного спуску. Занадто малі партії можуть внести шум у процес навчання, тоді як занадто великі партії можуть вимагати більше обчислювальних ресурсів і менше мінливості оновлення.

Щоб запобігти переобладнанню, шари Dropout були інтегровані в архітектуру нейронної мережі. Ця техніка випадковим чином вимикає деякі нейрони під час кожної ітерації навчання, що запобігає надмірній залежності мережі від окремих функцій і сприяє кращому узагальненню. Було перевірено різні значення Dropout, включаючи 0,3, 0,4 і 0,5, що дозволило знайти найкращий варіант з точки зору точності до стабільності тренувань. Крім того, міжкатегоріальна ентропія була використана для обчислення функції втрат, яка ідеально підходить для розв’язування задач багатокласової класифікації. Ця функція оцінює відстань між прогнозованим вектором ймовірності та істинним (однозначним) представленням мітки класу, більше штрафуючи модель за значні відхилення. Адам використовувався як оптимізатор, алгоритм, що поєднує в собі переваги адаптивної швидкості навчання та контролю імпульсу. Його параметри (зокрема швидкість навчання $\alpha = 0{,}001$) були обрані на основі попередніх досліджень і практики, оскільки вони дозволяють досягти хороших результатів без необхідності надмірного налаштування.

На всіх етапах експериментів активно використовувався механізм збереження найкращої версії моделі (ModelCheckpoint), яка досягла найвищої точності в валідаційному наборі. Це дозволило відновити мережу до оптимального стану, навіть якщо подальше навчання погіршувало її якість. Тому оптимізація моделі стала результатом послідовної, покрокової роботи, в якій кожен гіперпараметр вибирався не довільно, а шляхом численних експериментів, аналізів і візуалізації результатів. Завдяки такому підходу вдалося створити стабільну та ефективну модель, яка здатна якісно розпізнавати емоції на зображеннях обличчя навіть в умовах обмежених даних і високої варіативності виразу обличчя.

Одним із ключових елементів, що визначають успіх навчання нейронної мережі, є правильне налаштування гіперпараметрів, тобто параметрів, які не вивчаються автоматично, а встановлюються вручну перед запуском моделі. Ці параметри впливають не тільки на швидкість і стабільність процесу навчання, але й на кінцеву здатність моделі розпізнавати нові, раніше не бачені приклади. Продумано підходячи до налаштування гіперпараметрів, можна досягти оптимального балансу між точністю, ефективністю та узагальненістю моделі. Одним із таких надзвичайно важливих параметрів є кількість епох - тобто кількість повних проходів моделі через весь навчальний набір. Щоб визначити оптимальну кількість епох, потрібно врахувати динаміку навчання, проаналізувати втрати на наборах навчання та перевірки та відстежувати поведінку моделі з часом. Численні експерименти показали, що найкращі результати досягаються з кількістю епох від 50 до 100. Цей діапазон дозволяє моделі поступово, але впевнено вивчати основні шаблони. в даних, не вдаючись до надмірного «заучування» окремих прикладів. Якщо навчання триває понад цю межу, ризик перетренованості зростає, що призводить до зниження точності незалежних даних. Для автоматичного керування цим процесом у модель інтегровано механізм ранньої зупинки, який постійно відстежує значення точності або функцію втрат у наборі перевірки. Якщо протягом кількох наступних епох не спостерігається покращення, навчання автоматично припиняється. Цей підхід не тільки дозволяє уникнути переобладнання, але й економить обчислювальні ресурси за рахунок скорочення часу навчання моделі.

Ще одним важливим параметром, що впливає на ефективність навчання, є розмір партії - кількість прикладів, які одночасно обробляються за один крок градієнтного спуску. Вибір розміру партії завжди є компромісом між точністю оновлення ваги моделі та використанням пам’яті. У цьому дослідженні було протестовано кілька варіантів, і найкращий баланс між швидкістю, стабільністю та точністю продемонстрували значення 32 і 64. Менші партії забезпечують більшу варіативність градієнтів, що може бути корисним для відновлення від локальних мінімумів, але також робить навчання нестабільним. З іншого боку, занадто великі пакети уповільнюють оновлення та вимагають значного обсягу оперативної пам’яті на графічному процесорі, що не завжди означає якість. Особливу увагу було приділено питанню регуляризації, зокрема введенню Dropout шарів у структуру нейронної мережі.

Відключення - це техніка, яка випадковим чином вимикає певний відсоток нейронів під час кожного кроку навчання. Це гарантує, що модель не покладається надмірно на конкретні функції чи комбінації, а натомість створює більш надійне та загальне представлення даних. На практиці це призводить до зниження ймовірності перенавчання та покращення здатності до узагальнення. Під час експериментів було перевірено кілька рівнів Dropout, зокрема значення в діапазоні від 0,3 до 0,5. Занадто низьке значення не має помітного ефекту, тоді як надто високе значення може ускладнити навчання, оскільки мережа втрачає забагато інформації. Вибраний діапазон дозволив досягти найкращого співвідношення точності навчального набору та стабільності результатів перевірки даних. В принципі, етап налаштування гіперпараметрів не полягає в простому виборі фіксованих чисел один раз. Це динамічний багатоетапний процес, який включає емпіричне тестування, метричний аналіз, візуалізацію результатів і адаптацію до змін даних і архітектури. Кожен параметр - від кількості епох до рівня Dropout - має власний вплив на загальну поведінку моделі, і лише завдяки систематичному підходу можна досягти справді задовільного результату.

На додаток до налаштування архітектури моделі та базових гіперпараметрів, оптимізація функції втрат і вибір відповідного алгоритму навчання стали

надзвичайно важливими елементами ефективного навчання. Функція втрат, також звана функцією помилок, є основним критерієм, за яким модель «навчається» - вона сигналізує, наскільки прогнозоване значення відрізняється від очікуваного. У контексті проблеми багатокласової класифікації емоцій, де кожне зображення може належати лише до одного із семи класів, категорійна перехресна ентропія показала найкращі результати. Цей тип функції втрат є стандартним для задач класифікації з взаємовиключними мітками та забезпечує високу чутливість для правильного прогнозування ймовірностей кожного класу.

У поєднанні з цією функцією втрат використовувався алгоритм оптимізації Адама (Adaptive Moment Estimation). Це один із найпопулярніших і найефективніших оптимізаторів у сучасному машинному навчанні, який поєднує в собі переваги адаптивних методів швидкості навчання (таких як RMSProp) і методів імпульсу (таких як Momentum). Основна перевага Адама полягає в тому, що він автоматично регулює швидкість навчання для кожного параметра окремо на основі накопиченої історії градієнта. Це дозволяє оптимізатору швидко досягати оптимального рішення навіть для складних задач із великою кількістю параметрів і шумом у даних.

Важливим кроком у побудові ефективного навчального процесу також стало використання механізму збереження найкращих зважених моделей (ModelCheckpoint). Під час кожної епохи навчання модель перевіряється на наборі даних перевірки, і якщо результати точності покращуються, поточні значення ваги автоматично зберігаються. Це гарантує, що навіть у разі подальшого погіршення продуктивності через переобладнання або коливання результатів остаточна модель усе одно базуватиметься на найкращій досягнутій конфігурації. Цей підхід особливо важливий для тривалого навчання, коли поведінка моделі може непередбачувано змінюватися на основі змін у градієнтах, вагових коефіцієнтах і стані оптимізатора. Варто підкреслити, що всі рішення щодо вибору функції втрат, алгоритму оптимізації та супутніх механізмів приймалися не інтуїтивно, а на основі серії наступних експериментів та аналізу результатів за даними валідації. На кожному етапі моделювання результати детально аналізувалися для виявлення потенційних

недоліків, якісного порівняння різних конфігурацій і визначення найкращих практик для поточного завдання. На додаток до стандартних показників, таких як точність, активно вивчалися поведінка функції втрат, зміни градієнта, час збіжності та здатність до узагальнення. Загалом поєднання перевірених методів оптимізації, інтелектуального вибору гіперпараметрів і сучасних підходів до регулярізації дозволило досягти високоякісної класифікації емоцій і забезпечити стабільне навчання моделі без надмірних флуктуацій або збоїв. Завдяки цьому модель не тільки добре запам'ятала навчальні приклади, але й навчилася узагальнювати - тобто показувати хороші результати на нових, раніше не бачених зображеннях.

РОЗДІЛ 3. ПЕРСПЕКТИВИ РОЗВИТКУ НЕЙРОМЕРЕЖІ ДЛЯ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ НА ОСНОВІ TENSORFLOW

Розроблена нейромережева модель, побудована з використанням потужного фреймворку TensorFlow, демонструє здатність класифікувати емоційний стан людини на основі зображення її обличчя. Основним завданням системи є автоматичне визначення емоції, що проявляється на зображенні, з подальшою її класифікацією до однієї з наперед заданих категорій. У поточній реалізації система розпізнає шість базових емоцій: радість, сум, гнів, здивування, страх і нейтральний стан. Ці емоції обрані не випадково - вони є універсальними для більшості культур і є базовими в психології емоцій за класифікацією Пола Екмана.

У своїй архітектурі модель базується на згортковій нейронній мережі (Convolutional Neural Network, CNN) - одному з найефективніших підходів до аналізу зображень. Згорткові мережі здатні автоматично виділяти важливі ознаки зображення, такі як контури, кути, пропорції обличчя, розташування очей, брів, рота, тощо, що дозволяє ефективно виявляти патерни, притаманні кожній емоції. Мережа проходить кілька етапів обробки: згортка, pooling (зниження розмірності), активація та нормалізація. Кінцеві ознаки подаються на повнозв’язні шари, які здійснюють класифікацію на одну з емоційних категорій[38].

Модель була навчена на відкритому датасеті, який містить велику кількість зображень облич людей з відповідними емоційними мітками (наприклад, FER-2013, AffectNet або інші). Під час тренування використовувалась функція втрат категоріальної крос-ентропії, а як оптимізатор - алгоритм Adam, що забезпечив швидку та стабільну мінімізацію помилок. Для покращення якості навчання застосовувалися методи data augmentation: випадкове обертання зображень, дзеркальне відображення, масштабування тощо, що підвищувало здатність моделі до узагальнення.

У процесі інференсу (передбачення), користувач завантажує зображення особи, після чого система попередньо обробляє його: виконує вирівнювання, масштабування до потрібного розміру, нормалізацію пікселів та подачу у модель.

Після обробки, система видає результат у вигляді емоції з найвищою ймовірністю, що відображає домінантний емоційний стан на зображенні.

Таким чином, ця модель є прикладом ефективного застосування глибокого навчання у завданні розпізнавання емоцій, і її можна розглядати як базову платформу для подальшого розвитку систем емоційної аналітики в реальному часі.

Можливості вдосконалення

Незважаючи на базову працездатність розробленої моделі розпізнавання емоцій, існує низка важливих напрямів, які відкривають широкі перспективи для її подальшого розвитку та практичного впровадження у різних сферах.

Одним із пріоритетних напрямів є розширення набору емоцій. На сьогодні модель класифікує лише базові емоції, такі як радість, сум, гнів, страх, здивування та нейтральний стан. Проте в реальному житті спектр емоцій є значно ширшим. Додавання категорій, як-от «іронія», «відраза», «задоволення», «зніяковілість», «зацікавленість» чи навіть «скепсис», дозволить здійснювати більш глибокий аналіз емоційного стану користувача. Це особливо актуально для застосування в психологічних інтерфейсах, емоційних чат-ботах та маркетингових дослідженнях, де нюансованість реакцій має ключове значення.

Іншим важливим напрямом є підвищення точності моделі. Застосування сучасних передтренованих архітектур, таких як EfficientNet, MobileNetV2, ResNet50 або InceptionV3, може суттєво покращити якість розпізнавання емоцій, зокрема у складних умовах - за слабкого освітлення, при частковому перекритті обличчя або низькій роздільній здатності зображення. Водночас, застосування технік розширення навчального набору (data augmentation) - таких як випадкове обертання, масштабування, додавання шуму або спотворення кольору - дозволяє зробити модель більш стійкою до варіативності вхідних даних і уникнути перенавчання, що сприяє кращому узагальненню на нових прикладах.

Наступним етапом розвитку є адаптація моделі до роботи в реальному часі. Зменшення обчислювальної складності та оптимізація моделі для розгортання на мобільних пристроях або у веб-середовищах (за допомогою TensorFlow Lite або TensorFlow.js) відкриває шлях до широкого впровадження у смартфонах, планшетах,

інтерактивних інтерфейсах, додатках відеозв'язку чи навіть у пристроях доповненої реальності. Така функціональність дає змогу в режимі реального часу зчитувати емоційні реакції користувачів і динамічно адаптувати відповідь системи до емоційного контексту.

Ще одним перспективним підходом є врахування контексту та мультифреймова обробка. Емоції - це процес, а не статичний стан, тому їхнє розпізнавання на основі лише одного кадру іноді може бути неточним або хибним. Використання послідовностей зображень (кадрів відео) або інтеграція моделей з короткою пам'яттю, таких як LSTM чи GRU, дозволить моделі «відслідковувати» динаміку змін обличчя у часі та точніше оцінювати поточний емоційний стан. Це особливо актуально для діалогових систем або під час аналізу відеоконференцій, де міміка змінюється поступово.

Не менш важливо враховувати етичні аспекти та культурні особливості у процесі вдосконалення моделі. Емоції можуть проявлятися по-різному в різних культурах, а їх інтерпретація - залежати від соціальних норм, віку чи навіть гендеру. Наприклад, те, що в одній культурі вважається ознакою гніву, в іншій може сприйматись як прояв концентрації. Тому модель має враховувати ці варіації або дозволяти навчання на локалізованих даних. Крім того, використання подібних систем потребує високого рівня прозорості, безпеки та дотримання конфіденційності персональних даних. Це особливо критично у сферах медицини, освіти, психології та відеоспостереження, де емоційне розпізнавання може впливати на прийняття рішень.

У перспективі, також доцільним є інтегрування цієї технології з іншими модальностями - наприклад, розпізнаванням голосових емоцій, аналізом тексту або жестів, що дасть змогу створити мультиканальні емоційно-інтелектуальні системи. Такі рішення стануть фундаментом для майбутніх емоційно-адаптивних користувацьких інтерфейсів (Affective Computing), які не просто реагують на команди, а й «відчувають» користувача.

Таким чином, подальший розвиток моделі розпізнавання емоцій має широкі можливості: від технічного удосконалення до глибшого соціального та етичного

занурення. Така система може стати важливою складовою майбутніх «розумних» технологій, де взаємодія людини з машиною набуде більш інтуїтивного та людяного характеру.

Можливе застосування

Удосконалена система класифікації емоцій на основі аналізу зображень обличчя має значний потенціал для впровадження у різних сферах людської діяльності. Завдяки точному виявленню емоційного стану користувача в режимі реального часу, така технологія може виступати інструментом для підвищення ефективності, персоналізації й безпеки у багатьох галузях.

Однією з ключових сфер є освіта, особливо у форматі дистанційного або змішаного навчання. Система розпізнавання емоцій дозволяє відстежувати реакції учнів на навчальний матеріал у режимі реального часу. Це відкриває можливості для автоматичного виявлення зниження уваги, нудьги, непорозуміння або навпаки - зацікавленості, що дає змогу викладачу або платформі адаптувати темп та складність навчання до індивідуальних потреб студента. У майбутньому це може сприяти розвитку емоційно-адаптивних навчальних систем.

У маркетингу такі системи можуть застосовуватись для збору зворотного зв’язку щодо продуктів, реклами чи інтерфейсів. За допомогою аналізу міміки споживача під час взаємодії з контентом, компанії можуть отримувати цінні дані про щирі емоційні реакції, що значно точніше відображають ставлення, ніж традиційні анкети або опитування. Це дозволяє ефективніше тестувати продукцію, оптимізувати рекламні кампанії та створювати персоналізований користувацький досвід.

У сфері медицини, зокрема психіатрії та психологічної діагностики, емоційні нейромережі можуть виступати додатковим інструментом для виявлення ранніх ознак депресії, тривожності, аутизму або емоційного вигорання. Такі системи можуть аналізувати щоденні відеозаписи пацієнтів або інтегруватися у телемедичні сервіси для спостереження за динамікою психоемоційного стану. Це особливо актуально в умовах віддаленої допомоги або роботи з вразливими групами населення.

У сфері робототехніки та розробки інтерактивних помічників, розпізнавання емоцій дозволяє адаптувати поведінку пристрою до настрою користувача. Наприклад, робот-помічник може змінити тон відповіді, надати підтримку або навіть знизити активність, якщо розпізнає ознаки втоми чи роздратування. Це сприяє формуванню більш природної та комфортної взаємодії між людиною і машиною.

Також важливо відзначити потенціал технології у сфері безпеки. Системи відеоспостереження, оснащені емоційним аналізом, можуть ідентифікувати ознаки підвищеного стресу, агресії або страху у громадських місцях, що може сигналізувати про потенційно небезпечні ситуації. Наприклад, це може бути корисним у аеропортах, на вокзалах, у школах або на масових заходах для попередження інцидентів.

Крім вищенаведених напрямів, система також може бути інтегрована у:

- автомобільну галузь, для моніторингу стану водія та запобігання аваріям через втому або відволікання;
- ігрову індустрію, для створення адаптивних сюжетів та підвищення залучення користувача;
- HR-сферу, для аналізу поведінки кандидатів під час співбесід;
- смарт-додатки для добробуту, які допомагають користувачу краще розуміти свій емоційний стан та керувати ним.

Таким чином, система розпізнавання емоцій на основі TensorFlow має універсальний потенціал - від комерційних продуктів до критично важливих рішень у сфері охорони здоров’я та безпеки. Завдяки своїй гнучкості, масштабованості та можливості інтеграції з іншими технологіями, вона може стати фундаментом для наступного покоління емоційно-інтелектуальних систем у цифровому світі.

Модель класифікації емоцій на основі TensorFlow уже демонструє позитивні результати, однак має великий потенціал для покращення та адаптації. Подальша робота повинна зосереджуватись на підвищенні точності, зменшенні ресурсних витрат, урахуванні етичних питань та розширенні сфер застосування. Реалізація цих напрямків зробить нейромережеву систему більш надійною, корисною та безпечною для широкого кола користувачів.

ВИСНОВКИ

У ході кваліфікаційної дипломної роботи було розроблено систему розпізнавання емоцій людини на основі аналізу зображення її обличчя із застосуванням фреймворку TensorFlow. На початку проєкту було проаналізовано актуальність теми, стан сучасних досліджень у галузі комп'ютерного зору та емоційного штучного інтелекту, а також основні підходи до створення моделей глибокого навчання для класифікації зображень. Визначено, що розпізнавання емоцій є важливим завданням у контексті розвитку адаптивних цифрових систем, оскільки дозволяє створювати більш "людяні" технології, які розуміють користувача і можуть відповідно адаптувати свою поведінку.

З технічної точки зору, було здійснено огляд архітектур згорткових нейронних мереж (CNN), методів обробки зображень, а також інструментів розробки моделей у TensorFlow. Зокрема, було розглянуто популярні архітектури, такі як VGGNet, ResNet та MobileNet, з метою вибору оптимального варіанту для задачі класифікації емоцій при обмежених обчислювальних ресурсах. На основі аналізу було обрано й реалізовано структуру нейронної мережі, здатну класифікувати зображення за шістьма базовими емоційними станами: радість, сум, гнів, страх, здивування та нейтральний вираз.

Модель була навчена на маркованому наборі даних, після чого була реалізована система, яка дозволяє завантажити зображення, автоматично визначити обличчя на ньому та отримати прогноз емоційного стану людини. У процесі роботи здійснено попередню обробку зображень, реалізовано функції масштабування, нормалізації та вирівнювання обличчя, що суттєво підвищило точність розпізнавання. Також було впроваджено базові методи розширення даних (data augmentation) -такі як обертання, горизонтальне відображення, зміна яскравості -для покращення здатності моделі до узагальнення.

На заключному етапі проєкту було створено інтерфейс користувача, який забезпечує зручну взаємодію з системою та дозволяє тестувати її функціональність у режимі реального часу. Система демонструє задовільний рівень точності при

розпізнаванні емоцій на різних типах зображень, включаючи варіації освітлення, положення голови та індивідуальні особливості облич.

Розроблена система має потенціал для подальшого розвитку та впровадження у різні галузі: дистанційне навчання, охорона здоров’я, робототехніка, маркетинг, безпека тощо. Вона може бути інтегрована у мобільні додатки, онлайн-сервіси або смарт-пристрої для зчитування емоційного стану користувача в реальному часі. Додатково, подальші дослідження можуть включати мультиемоційне розпізнавання, аналіз мікровиразів або інтеграцію з мовленнєвим аналізом для створення мультимодальних систем.

У результаті реалізації цього проєкту було здобуто практичні навички роботи з TensorFlow, побудови та тренування згорткових нейронних мереж, а також досвід інтеграції моделей глибокого навчання у прикладні задачі комп’ютерного зору. Отримані знання та вміння можуть бути успішно застосовані в подальшій професійній діяльності у сфері штучного інтелекту, машинного навчання та розробки інтелектуальних систем, орієнтованих на користувача.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Contributors to Wikimedia projects. History of artificial neural networks - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/History_of_artificial_neural_networks (дата звернення: 04.05.2025)

2. Schmidhuber J. Annotated History of Modern AI and Deep Learning. arXiv. URL: https://arxiv.org/abs/2212.11279 (дата звернення: 04.05.2025)

3. Salazar E. Artificial Neural Networks: History and State of the Art. In: Encyclopedia of Information Science and Technology, Sixth Edition. IGI Global. URL: https://www.igi-global.com/chapter/artificial-neural-networks/320788 (дата звернення: 04.05.2025)

4. Codewave. History and Development of Neural Networks in AI. Codewave Insights. URL: https://codewave.com/insights/development-of-neural-networks-history/ (дата звернення: 04.05.2025)

5. TensorFlow. TensorFlow official website. URL: https://www.tensorflow.org/ (дата звернення: 04.05.2025)

6. El Aoun M., Tidjon L. N., Rombaut B., Khomh F., Hassan A. E. An Empirical Study of Library Usage and Dependency in Deep Learning Frameworks. arXiv. URL: https://arxiv.org/abs/2211.15733 (дата звернення: 04.05.2025)

7. Obregon A. Machine Learning Frameworks: TensorFlow, PyTorch, and Keras - A Comparative Analysis. Medium. URL: https://medium.com/%40AlexanderObregon/machine-learning-frameworks-tensorflow-pytorch-and-keras-a-comparative-analysis-e24b3fa00cbd (дата звернення: 04.05.2025)

8. Shi S., Wang Q., Xu P., Chu X. Benchmarking State-of-the-Art Deep Learning Software Tools. arXiv. URL: https://arxiv.org/abs/1608.07249 (дата звернення: 04.05.2025)

9. Goldsborough P. A Tour of TensorFlow. arXiv. URL: https://arxiv.org/abs/1610.01178 (дата звернення: 04.05.2025)

10. Abadi M., Barham P., Chen J., et al. TensorFlow: A system for large-scale machine learning. arXiv. URL: https://arxiv.org/abs/1605.08695 (дата звернення: 04.05.2025)

11. Codecademy. PyTorch vs TensorFlow: Choosing the Best Framework for Deep Learning. URL: https://www.codecademy.com/article/pytorch-vs-tensorflow-choosing-the-best-framework-for-deep-learning (дата звернення: 04.05.2025)

12. DataCamp. PyTorch vs TensorFlow vs Keras for Deep Learning: A Comparative Guide. URL: https://www.datacamp.com/tutorial/pytorch-vs-tensorflow-vs-keras (дата звернення: 04.05.2025)

13. OpenCV. PyTorch vs TensorFlow: Comparative Guide of AI Frameworks 2025. URL: https://opencv.org/blog/pytorch-vs-tensorflow/ (дата звернення: 04.05.2025)

14. Venture Lessons. PyTorch vs TensorFlow: Choosing the Right Deep Learning Framework. URL: https://www.venturelessons.com/pytorch-vs-tensorflow-choosing-the-right-deep-learning-framework/ (дата звернення: 04.05.2025)

15. Matplotlib. Matplotlib 3.9.2 Documentation. URL: https://matplotlib.org/stable/ (дата звернення: 04.05.2025)

16. Seaborn. Seaborn: statistical data visualization. URL: https://seaborn.pydata.org/ (дата звернення: 04.05.2025)

17. Buslaev A., Iglovikov V. I., Khvedchenya E., Parinov A., Druzhinin M., Kalinin A. A. Albumentations: Fast and Flexible Image Augmentations. Information. 2020;11(2):125. URL: https://www.mdpi.com/2078-2489/11/2/125 (дата звернення: 04.05.2025)

18. Guru99. Підручник TensorFlow для початківців: вивчення основ на прикладі. URL: https://www.guru99.com/uk/tensorflow-tutorial.html (дата звернення: 04.05.2025)

19. Р. І. Білий. Використання нейромережі на базі TensorFlow для розпізнавання жестів та керування біонічним протезом. Вісник Вінницького політехнічного інституту. URL: https://doi.org/10.31649/1997-9266-2024-174-3-71-77 (дата звернення: 04.05.2025)

20. Ю. Томка, М. Талах, В. Дворжак, О. Ушенко. Реалізація згорткової нейронної мережі з використанням TensorFlow. Оптико-електронні інформаційно-енергетичні технології. URL: https://oeipt.vntu.edu.ua/index.php/oeipt/article/view/621 (дата звернення: 04.05.2025)

21. Л. Я. Сініцин. Програмний модуль класифікації тексту на базі нейронної мережі та бібліотеки TensorFlow. Національний авіаційний університет. URL: https://er.nau.edu.ua/handle/NAU/66241 (дата звернення: 04.05.2025)

22. Д. І. Оксенюк. Застосування технології CV і TensorFlow у завданнях моніторингу лісових ресурсів. КПІ ім. Ігоря Сікорського. URL: https://ela.kpi.ua/handle/123456789/51752 (дата звернення: 04.05.2025)

23. Т. О. Воловоденко. Використання моделей машинного навчання у вбудованих системах. КПІ ім. Ігоря Сікорського. URL: https://ela.kpi.ua/handle/123456789/55821 (дата звернення: 04.05.2025)

24. Т. І. Марич. Реалізація нейронних мереж в мобільних застосунках. КПІ ім. Ігоря Сікорського. URL: https://ela.kpi.ua/handle/123456789/32019 (дата звернення: 04.05.2025)

25. М. Садівник. Машинне навчання у браузері з використанням TensorFlow.js. Тернопільський національний технічний університет імені Івана Пулюя. URL: http://elartu.tntu.edu.ua/handle/lib/30452 (дата звернення: 04.05.2025)

26. DevZone. Машинне навчання: класифікація тексту з нейронними мережами та TensorFlow. URL: https://devzone.org.ua/post/mashynne-navchannia-klasyfikatsiia-tekstu-z-neyronnymy-merezamy-ta-tensorflow (дата звернення: 04.05.2025)

27. Джаноянц В. О. Спосіб розпізнавання емоційних станів у зображеннях людини : Магістерська дисертація : 121. Київ, 2023. 122 с.

28. Сербін В. С. Аналіз методів створення нейронних мереж з використанням мови програмування JavaScript. ELARTU. URL: http://elartu.tntu.edu.ua/handle/lib/43257 (дата звернення: 05.05.2025)

29. Сушильников К. Д. Розпізнавання емоцій людини в реальному часі. DSpace :: ELAKPI :: Репозитарій КПІ ім. Ігоря Сікорського. URL:

https://ela.kpi.ua/items/813c942b-fbd4-4988-81ac-84485dece60e (дата звернення: 05.05.2025)

30. Tereykovska L. Method of neural network emotion recognition from facial images. Computer-Integrated Technologies: Education, Science, Production. 2020. No. 40. P. 146–152. URL: https://doi.org/10.36910/6775-2524-0560-2020-40-22 (access date: 05.05.2025)

31. Хмельницький А. Перші кроки в NLP: розглядаємо Python-бібліотеку TensorFlow та нейронні мережі в реальному завданні. DOU.ua. URL: https://dou.ua/lenta/articles/first-steps-in-nlp-tensorflow (дата звернення: 05.05.2025)

32. Białek C., Matiolański A., Grega M. An Efficient Approach to Face Emotion Recognition with Convolutional Neural Networks. Electronics. 2023. Vol. 12, no. 12. P. 2707. URL: https://doi.org/10.3390/electronics12122707 (date of access: 05.05.2025)

33. Cherechecha I., Prykhodchenko S. Artificial Intelligence Based on TensorFlow in Educational Process. Information Technology: Computer Science, Software Engineering and Cyber Security. 2021. No. 1. P. 35–41. URL: https://doi.org/10.32782/it/2021-1-5 (date of access: 05.05.2025)

34. CNN-Based Facial Expression Recognition with Simultaneous Consideration of Inter-Class and Intra-Class Variations / T.-D. Pham et al. Sensors. 2023. Vol. 23, no. 24. P. 9658. URL: https://doi.org/10.3390/s23249658 (date of access: 05.05.2025)

35. Contributors to Wikimedia projects. Emotion recognition - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Emotion_recognition?utm_source=chatgpt.com (дата звернення: 05.05.2025)

36. Deep Learning Technology of Convolutional Neural Networks for Facial Expression Recognition / D. V. Petrosiuk et al. Applied Aspects of Information Technology. 2021. Vol. 4, no. 2. P. 192–201. URL: https://doi.org/10.15276/aait.02.2021.6 (date of access: 05.05.2025)

37. Facial Emotion Recognition with Convolutional Neural Networks – Johannes Schusterbauer. Johannes Schusterbauer – PhD Student. URL:

https://johfischer.com/2022/03/25/facial-emotion-recognition-with-convolutional-neural-networks-in-tensorflow/ (дата звернення: 05.05.2025)

38. GeeksforGeeks. Emotion Detection Using Convolutional Neural Networks (CNNs) - GeeksforGeeks. GeeksforGeeks. URL: https://www.geeksforgeeks.org/emotion-detection-using-convolutional-neural-networks-cnns/ (date of access: 05.05.2025)

39. Huang H. A facial expression recognition method based on Convolutional Neural Network. Frontiers in Computing and Intelligent Systems. 2022. Vol. 2, no. 1. P. 116–119. URL: https://doi.org/10.54097/fcis.v2i1.3178 (date of access: 05.05.2025)

40. Intelligent Module for Recognizing Emotions by Voice / O. Ilarionov et al. Advanced Information Technology. 2021. No. 1 (1). P. 46–52. URL: https://doi.org/10.17721/ait.2021.1.06 (date of access: 05.05.2025)

41. Nguyen T., Le T. Fusion of Attentional and Traditional Convolutional Networks for Facial Expression Recognition. EAI Endorsed Transactions on Pervasive Health and Technology. 2018. P. 169033. URL: https://doi.org/10.4108/eai.17-3-2021.169033 (date of access: 05.05.2025)