Державний торговельно-економічний університет

Кафедра комп'ютерних наук та інформаційних систем

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
на тему:

«Методи обробки зображень на основі згорткових нейронних мереж»

Студента 2 курсу, 4м групи, Коротких Віталій Олександрович

спеціальності
122 «Комп'ютерні науки» _____
підпис студента

Науковий керівник
кандидат снаук _____ Філімонова Тетяна Олегівна
підпис керівника

Гарант освітньої програми
доктор фізико-математичних наук, професор _____ Пурський Олег Іванович
підпис керівника

Київ 2023

Державний торговельно-економічний університет

Факультет інформаційних технологій
Кафедра комп'ютерних наук та інформаційних систем
Спеціальність 122 «Комп'ютерні науки»
Освітня програма «Комп'ютерні науки»

Затверджую
Зав. кафедри _____Пурський О.І.
«9» грудня 2022р.

Завдання
на випускну кваліфікаційну роботу студентці

Коротких Віталій Олександрович
(прізвище, ім'я, по батькові)

1. Тема випускної кваліфікаційної роботи.
«Методи обробки зображень на основі згорткових нейронних мереж»
Затверджена наказом ректора від «6» грудня 2022 р. № 3284
2. Строк здачі студентом закінченої роботи 24 листопада 2023 року
3. Цільова установка та вихідні дані до роботи
Метою роботи: розробка методів обробки зображень на основі згорткових нейронних мереж.
Об'єкт дослідження: процес розробки методів обробки зображень на основі згорткових нейронних мереж.
Предмет дослідження: засоби розробки методів обробки зображень на основі згорткових нейронних мереж.
4. Перелік графічного матеріалу_____
_____
_____
_____
_____

5. Консультанти по роботі із зазначенням розділів, за якими здійснюється консультування:

Розділ	Консультант (прізвище, ініціали)	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Філімонова Т.О.	22.12.2022 р.	22.12.2022 р.
2	Філімонова Т.О.	22.12.2022 р.	22.12.2022 р.
3	Філімонова Т.О.	22.12.2022 р.	22.12.2022 р.

6. Зміст випускної кваліфікаційної роботи (перелік питань за кожним розділом)

Вступ

РОЗДІЛ 1. ОГЛЯД МЕТОДІВ ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ РОЗПІЗНАВАННЯ ЗОБРАЖЕНЬ

1.1 Огляд актуальних проблем в галузі обробки зображень

1.2 Формулювання цілей та завдань дослідження обробки зображень за допомогою згорткових нейронних мереж

1.3 Аналіз теоретичних підходів до вирішення проблем обробки зображень за допомогою згорткових нейронних мереж.

1.4 Обґрунтування важливості теми дослідження

РОЗДІЛ 2. АЛГОРИТМ РОЗПІЗНАВАННЯ ЗОБРАЖЕНЬ ЗА ДОПОМОГОЮ НЕЙРОННИХ МЕРЕЖ

2.1 Опис методів, використаних для тренування та застосування згорткових нейронних мереж у дослідженні обробки зображень.

2.2 Формування та організація набору даних для тренування

2.3 Оцінка точності та результативності моделі

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Розробка програмного забезпечення та середовище його створення

3.2 Представлення отриманих результатів експериментів та оцінок ефективності роботи згорткових нейронних мереж.

3.3 Аналіз результатів

Висновки

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

7. Календарний план виконання роботи

№ Пор.	Назва етапів випускної кваліфікаційної роботи	Строк виконання етапів роботи	
		За планом	фактично
1	2	3	4
1	Вибір теми випускної кваліфікаційної роботи	01.11.2022	01.11.2022
2	Розробка та затвердження завдання на випускну кваліфікаційну роботу	09.12.2022	09.12.2022
3	Вступ	01.05.2023	01.05.2023
4	РОЗДІЛ 1. Огляд методів та програмного забезпечення для розпізнавання пухлин головного мозку	14.06.2023	14.06.2023
5	Підготовка статті у збірник наукових статей магістрів	20.06.2023	20.06.2023
6	РОЗДІЛ 2. Алгоритм розпізнавання пухлин головного мозку на знімках МРТ за допомогою нейронних мереж	08.09.2023	08.09.2023
7	РОЗДІЛ 3. Реалізація програмного забезпечення	20.10.2023	20.10.2023
8	Висновки	02.11.2023	02.11.2023
9	Здача випускної кваліфікаційної роботи на кафедру науковому керівнику	22.11.2023	22.11.2023
10	Попередній захист випускної кваліфікаційної роботи	29.11.2023	29.11.2023
11	Виправлення зауважень, зовнішнє рецензування випускної кваліфікаційної роботи	04.12.2023	04.12.2023
12	Представлення готової зшитої випускної кваліфікаційної роботи на кафедру	06.12.2023	06.12.2023
13	Публічний захист випускної кваліфікаційної роботи	За розкладом роботи ЕК	

8. Дата видачі завдання «9» грудня 2022 р.

9. Керівник випускної кваліфікаційної роботи Філімонова Т.О_____

(прізвище, ініціали, підпис)

10. Гарант освітньої програми Пурський О.І._____

(прізвище, ініціали, підпис)

11. Завдання прийняв до виконання студент Коротких В.О._____

(прізвище, ініціали, підпис)

12. Відгук керівника випускної кваліфікаційної роботи

В випускній кваліфікаційній роботі досліджено методи обробки зображень на основі згорткових нейронних мереж. В результаті розроблена архітектура моделі з використанням згорткових шарів для розпізнавання зображень. Побудовані графіки функції втрат і точності. Розроблено web-сервіс для розпізнавання зображень. Всі поставленні завдання виконані. Робота може бути допущена до захисту.

Керівник випускної кваліфікаційної роботи

_____ 31.05.2023 р.

(підпис, дата)

13. Висновок про випускну кваліфікаційну роботу

Випускна кваліфікаційна робота студента _____Коротких В.О.

(прізвище, ініціали)

може бути допущена до захисту в екзаменаційній комісії.

Гарант освітньої програми _____Пурський О.І.

(підпис, прізвище, ініціали)

Завідувач кафедри _____Пурський О.І.

(підпис, прізвище, ініціали)

«_____»_____2023 р

Анотація

У випускній кваліфікаційній роботі досліджується застосування згорткових нейронних мереж для класифікації зображень мозку з використанням візуальних ознак. Головною метою було створення та оптимізація моделі, яка може ефективно розпізнавати класи: glioma, meningioma, normal, adenoma. У роботі розглядається підготовка даних, аугментація зображень, побудова та аналіз архітектури згорткової нейронної мережі. Експерименти показали високу точність та ефективність запропонованої моделі на валідаційному наборі даних. Отримані результати можуть мати значення для автоматизації процесу діагностики захворювань мозку на ранніх стадіях, сприяючи покращенню швидкості та точності класифікації медичних зображень.

Ключові слова: згорткові нейронні мережі, класифікація медичних зображень, мозкові захворювання, гліома, менінгіома, аденома, нейромережева архітектура, точність класифікації, діагностика захворювань мозку, візуальні ознаки, оптимізація моделі.

Anotation

In the graduation qualifying work, the application of convolutional neural networks for brain image classification using visual cues is explored. The main objective was to create and optimize a model capable of effectively recognizing classes: glioma, meningioma, normal, adenoma. The work covers data preparation, image augmentation, construction, and analysis of the convolutional neural network architecture. Experiments demonstrated high accuracy and effectiveness of the proposed model on the validation dataset. The obtained results may hold significance for automating the process of diagnosing brain disorders at early stages, contributing to improved speed and accuracy in the classification of medical images.

Keywords: convolutional neural networks, medical image classification, brain disorders, glioma, meningioma, adenoma, neural network architecture, classification accuracy, brain disease diagnosis, visual features, model optimization.

ЗМІСТ

ВСТУП

З моменту появи комп'ютерів люди намагалися перенести на комп'ютер рішення якомога більшої кількості проблем. У медичній діагностиці одним із найважливіших і складних завдань є правильний діагноз. Від цього залежить все подальше лікування. Звичайно, вони також намагаються автоматизувати це завдання, щоб мінімізувати людський фактор. Адже воно може бути як позитивним (наприклад, великий досвід лікаря, набутий роками практики), так і негативним (наприклад, погане самопочуття).

Хоча було створено багато систем прийняття медичних рішень, під час їх розробки виникли деякі методологічні труднощі. Адже важко схематизувати дані, коли кілька фахівців можуть по-різному визначити одне і те ж захворювання, як це часто буває. За допомогою зрозумілих алгоритмів не завжди вдається описати якісь складні клінічні картини. Найбільша проблема полягає в тому, що необхідних знань може навіть не бути, коли система створена. У зв'язку з цим робляться спроби розробити систему, яка знає краще, ніж її розробники.

В ідеалі розроблений метод повинен мати 100% чутливість (ймовірність того, що всі люди з позитивним результатом будуть віднесені до потрібної категорії) і 100% специфічність (ймовірність того, що всі люди з негативною ймовірністю класу будуть правильно ідентифіковані).

Часто висока чутливість означає низьку специфічність. Це може бути тому, що не всі сприймають вихід певного параметра за межі прийнятих стандартів як хворобу. Тут спрацьовують індивідуальні особливості організму.

Підвищити чутливість методу, щоб не порушити специфічність, можуть допомогти нейронні мережі. Це нелінійні системи, які здатні класифікувати дані краще, ніж загальновживані лінійні методи. Нейронні мережі вчаться робити висновки, аналізуючи приховані зв'язки, які вони знаходять у даних. При цьому вони не використовують якийсь певний алгоритм міркування для прийняття рішень, а вчаться на прикладах. Крім того, нейронні мережі можуть

виконувати класифікацію, узагальнюючи попередній досвід і використовуючи його для наступних завдань. [1]

Актуальність роботи.

Завдання розпізнавання пухлин головного мозку є затребуваним у медицинських системах. Ці системи можуть використовуватися різними державними установами, лікарнями тощо.

Мета роботи.

Метою роботи є розробка методів обробки зображень на основі згорткових нейронних мереж.

Постановка задачі.

Досягнення цієї мети передбачає вирішення наступних завдань:

- аналіз та вибір архітектури згорткової нейронної мережі;
- збір та підготовка набору даних для навчання моделі;
- розробка, налаштування та тренування згорткової нейронної мережі;
- оцінка ефективності роботи розробленої мережі
- документування результатів та висновків для подальшого використання.

Об’єкт дослідження

Об’єктом дослідження є процес розробки методів обробки зображень на основі згорткових нейронних мереж.

Предмет дослідження.

Предметом дослідження є засоби розробки методів обробки зображень на основі згорткових нейронних мереж.

Наукова новизна

Випускна кваліфікаційна робота полягає в успішному поєднанні сучасних методів обробки зображень та застосування згорткових нейронних мереж для розпізнавання пухлин головного мозку. Впроваджує нейронні мережі у медичну сферу для покращення діагностики.

Публікації.

Результати дослідження опубліковано у збірнику наукових статей студентів, які здобувають освітній ступінь магістра за спеціальністю «Комп'ютерні науки» ДТЕУ. Методи обробки зображень на основі згорткових нейронних мереж // Прикладні комп'ютерні технології : зб. наук. ст. студ. / відп. ред.— Київ : Держ. торг.-екон. ун-т, 2023. – С. 60-65.

Структура та обсяг випускної кваліфікаційної роботи.

Випускна кваліфікаційна робота складається із вступу, чотирьох розділів, висновків, списку використаних джерел із 17 найменувань, додатків і містить 40 сторінки основного тексту, 29 рисунків, 1 таблицю і 8 формул.

РОЗДІЛ 1 ОГЛЯД МЕТОДІВ ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ РОЗПІЗНАВАННЯ ЗОБРАЖЕНЬ

1.1 Постановка задачі

Комп’ютерний зір досі залишається актуальною науковою галуззю, воно вирішує задачі отримання високого рівня розуміння цифрових зображень [1]. Завдання включають методи збору, обробки, аналізу та розуміння цифрових зображень, їх можна розділити на чотири типи [2]:

Класифікація - класифікація зображення за типом об'єкта, який воно містить;

Семантична сегментація - знаходження пікселів об'єктів одного класу на зображенні. Якщо кілька об’єктів одного класу перекриваються, їх пікселі жодним чином не відокремлюються один від одного;

Виявлення об'єктів - виявлення всіх об'єктів необхідних класів і побудова огороджувальної рамки для кожного з них;

Сегментація екземплярів - знаходження пікселів, які містить об'єкт кожного класу окремо;

Ілюстрація сегментації на прикладі повітряних куль наведена на рис. 1.1.

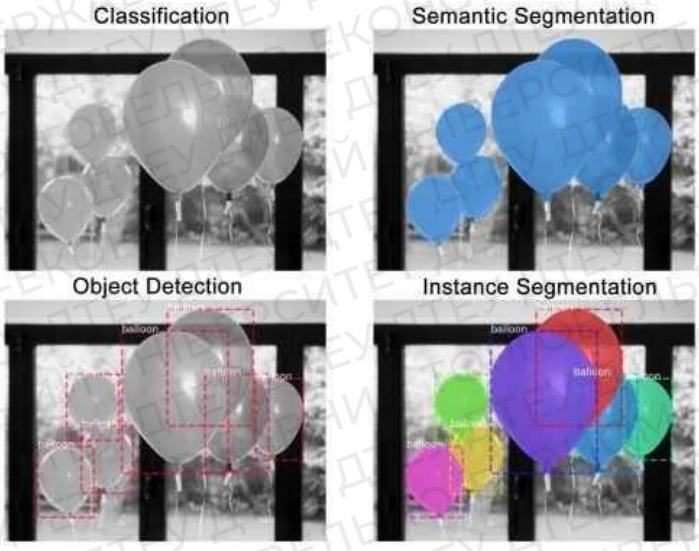

Рис. 1.1 - Ілюстрація типів комп'ютерного зору

Основуючись на вищесказаному метою роботи є розробка програми для аналізу медичних зображень виявлення та сегментація уражень за допомогою методів машинного навчання. На основі роботи розробленої нейронної мережі зображення були класифіковані за наявністю або відсутністю на них пухлини.

Для досягнення поставлених цілей необхідно виконати наступні завдання:

- аналізувати дані зображення МРТ та виконати попередню обробку;
- аналіз існуючих нейронних мереж, придатних для досягнення цілей;
- розробити багатошарову згортову нейронну мережу для ідентифікації пухлин мозку;
- представити результати розробленої нейронної мережі.

1.2 Огляд літератури

Книга [2] Семюела Бернса є вступом до методів глибинного навчання. Автор не робить надто великого акценту на математиці, оскільки цей посібник призначений для розробників, які новачки в галузі глибинного навчання. Книга поділена на розділи, кожен з яких охоплює різні функції бібліотек глибокого навчання (Tensorflow, Keras і PyTorch), доступних у мові програмування Python.

У [3] експериментальні результати описані на основі двох основних параметрів, таких як час і точність. Це дослідження порівнює продуктивність різних алгоритмів, таких як DNN, ANN (штучна нейронна мережа) і KNN. Експериментальні результати та статистичний аналіз показують, що відсотки точності: 93, 90 та 81 відповідно. З результатів стає зрозуміло, що DNN перевершує інші методи KNN і ANN. Зображення МРТ використовувалися як набір даних у цьому дослідженні.

Це дослідження [4] пропонує техніку сегментації, яка допомагає користувачам швидко та ефективно ідентифікувати пухлини за допомогою МРТ головного мозку. На основі аналізу симетрії цей метод застосовується до

багатьох наборів даних про пухлини різного розміру, розташування та інтенсивності для автоматичного виявлення та сегментації різних класів пухлин мозку. Ця техніка дозволяє клініцистам локалізувати пухлини в мозку пацієнта та розрахувати їхню площу, щоб можна було планувати та лікувати ефективне лікування. Обробка зображень за допомогою MATLAB.

Автори [5] застосували метод класифікації SVM до зображень МРТ головного мозку, класифікуючи їх як нормальні та патологічні. Matlab 7.9 використовувався для вилучення функцій.

Отримані результати використовуються як вхідні дані для процесу класифікації, щоб зробити висновки. З точністю 65% нормальні зображення були класифіковані як успішні, а патологічні – ні. Згідно з дослідженням, SVM не може забезпечити надійні результати для великих даних.

В іншому дослідженні [6] зображення МРТ сегментували за допомогою техніки порогової сегментації. Перед процесом сегментації зображення перетворюється на відтінки сірого, а потім фільтрується, щоб видалити шум і зробити зображення яскравішим або різкішим для кращих результатів. Метод класифікації SVM використовується як класифікатор, щоб показати, чи є пухлина злоякісною, доброякісною чи нормальною.

У [7] і [8] для класифікації використовувався підхід нейронної мережі зворотного поширення. Вейвлет-перетворення використовується для виділення ознак, а PCA (аналіз основних компонентів) використовується для вибору функцій. Зменшені дані використовуються для кращих результатів.

1.3 Штучні нейронні мережі

Штучні нейронні мережі (ШНМ) — це складні структури, що складаються з взаємопов’язаних адаптивних елементів, званих штучними нейронами, які можуть виконувати масивні обчислення для представлення знань. Вони володіють усіма основними якостями біологічних нервових систем, включаючи

здатність до навчання, стабільність, нелінійність, високий паралелізм, відмовостійкість, здатність працювати з неточною та неоднозначною інформацією та здатність до узагальнення.

Основні риси:

- нелінійність (краще відповідає даним);
- високий ступінь паралелізму (сприяє швидкій обробці та відмовостійкості обладнання);
- узагальнення (дозволяє застосувати модель до ненавчених даних;
- нечутливість до шуму (навіть невизначені дані та помилки вимірювань добре передбачувані);
- можливість навчання та адаптації (дозволяє моделі оновлювати свою внутрішню архітектуру у відповідь на зміну середовища). [9]

Нейронні мережі ідеально підходять для виконання розпізнавання образів, щоб розпізнавати та класифікувати об’єкти чи сигнали в системах мови, зору та керування. Їх також можна використовувати для прогнозування та моделювання часових рядів.

Нейронні мережі об’єднують багаторівневу обробку за допомогою простих елементів, які працюють паралельно та надихаються біологічними нервовими системами. Він складається з вхідного шару, одного або кількох прихованих шарів і вихідного шару. Кожен шар має кілька вузлів, або нейронів, і кожен шар використовує вихідні дані попереднього шару як вхідні дані, тому нейрони підключаються до різних шарів. Подається вхідний рівень, тобто необроблені дані. Він передає інформацію із зовнішнього світу в мережу. На цьому рівні обчислення не виконуються - вузли просто передають інформацію на прихований рівень. Усі обчислення функцій, що подаються через вхідний рівень, відбуваються у прихованому шарі, який потім передає результати на вихідний рівень. Вихідний рівень відображає результати обчислень, виконаних мережею.

Кожен нейрон у мережі має власну вагу, яка регулюється в процесі навчання, і коли вага зменшується або збільшується, це змінює силу сигналу цього нейрона. Ці ваги автоматично регулюються відповідно до заданих правил навчання, доки ШНМ правильно не виконає бажане завдання.

ШНМ використовують функції активації для виконання складних обчислень у прихованих шарах, а потім передають результати на вихідний рівень. Основна мета цих функцій — ввести нелінійність у нейронну мережу.

Вони перетворюють лінійний вхід вузла в нелінійний вихід, щоб полегшити вивчення поліномів вищого порядку в глибоких мережах кілька разів. Функції активації унікальні тим, що вони диференційовані. Це допомагає їм функціонувати під час процесу зворотного поширення нейронної мережі.

Якщо функція активації не застосована, результатом буде лінійна функція полінома першого ступеня. Хоча лінійні рівняння легко розв’язати, вони мають обмежену складність і, отже, менш здатні вивчати складні функціональні відображення з даних. Отже, без функції активації нейронна мережа була б моделлю кінцевої лінійної регресії.

Найпопулярніші функції активації включають:

– Сигмоїдна

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{1.1}$$

Сигмоїдна функція бере дійсне число та перетворює його на число від 0 до 1. Зокрема, великі від’ємні числа перетворюються на 0, а великі додатні числа перетворюються на 1.

– Tanh

$$\tanh x = \frac{2}{1+e^{-2x}} - 1 \tag{1.2}$$

Функція упаковує дійсні числа в діапазон [-1, 1]. Його вихід має нульовий центр. Ви бачите, що це модифікована сигмоїдна функція $\tanh x = 2\sigma(2x) - 1.$

– Soft Max

Ця функція створює вихідні дані, які коливаються між 0 і 1, із сумою ймовірностей, яка дорівнює 1.

– ReLU (Rectified linear unit)

За останні кілька років лінійні випрямлячі стали дуже популярними. Він обчислює функцію $f(x) = \max(0, x)$. Іншими словами, для активації існує лише нульовий поріг.

– ELU (експоненціальні лінійні одиниці)

$$f(x) = \begin{cases} \alpha(e^x - 1), x \le 0 \\ x, x > 0 \end{cases} \quad (1.3)$$

Тут негативні компоненти моделюються за допомогою експоненціального запису. Однак у нас все одно буде параметр, який можна навчити α.

Нейронна мережа, що складається з двох або трьох з'єднаних нейронних шарів, називається неглибокою нейронною мережею. Мережі глибокого навчання можуть мати багато рівнів, навіть сотні шарів. Обидва ці методи є методами машинного навчання, які навчаються безпосередньо з вхідних даних.

Глибоке навчання особливо підходить для складних програм розпізнавання, таких як розпізнавання облич, переклад тексту та розпізнавання мовлення. [10][11]

1.4 Нейронні мережі в медицині

Останнім часом глибоке навчання все частіше використовується для вирішення різних медичних проблем. Це завдяки збільшеній обчислювальній потужності та великій кількості мічених зображень, а також можливості використання хмарного сховища.

Нейронні мережі використовуються в медицині на наступних рівнях:

– швидке та чітке розпізнавання зображень;

– зменшити можливість лікарських помилок;

– пацієнти самостійно контролюють та аналізують свій стан за допомогою датчиків.

Через існуючі обмеження використання нейронних мереж у медицині було досить обмеженим, а розвиток відбувався повільніше, ніж міг бути.

В даний час вони допомагають лікарям встановлювати діагнози, усувати різні шуми в біологічних сигналах і визначати найважливіші сигнали для певних станів з великої кількості доступних даних. І це лише мала частина можливостей, які відкриває використання штучних нейронних мереж.

Глибокі нейронні мережі вирішують такі проблеми, як інтерпретація різних патологій під час медичного сканування та інтерпретація електрокардіограм.

З їх допомогою також можна визначити деякі види раку, кровотечі, шкірні захворювання, різні переломи кісток і багато інших захворювань. [12]

1.5 Проблеми та обмеження

Основними причинами, які стримують розвиток штучного інтелекту в галузі медичної діагностики, є питання захисту даних і конфіденційності. Дані про пацієнта можна отримати з його історії хвороби. Крім того, не можна виключити можливість навмисного злому системи з метою нанесення шкоди іншим.

Також проблеми можуть виникнути через неточності алгоритму. Наприклад, алгоритм Watson for Oncology, який використовується для лікування хворих на рак, базується на невеликій кількості синтетичних випадків і реальних даних. Він давав неправильні і навіть небезпечні поради щодо лікування, наприклад, використання несумісних ліків, хоча це було строго протипоказано.

Наступна проблема – упереджене та непропорційне використання можливостей ШІ. Через відсутність у наборі даних досліджень із різних

соціальних верств, отримані результати можуть бути не зовсім об’єктивними. У той же час передчасна смерть часто пов'язана з низьким соціально-економічним статусом. [12].

Різні архітектури нейронних мереж використовуються для різних завдань і навчальних даних для медичних діагностичних завдань. Для їх написання використовуються мови різної складності — від простих мов типу BASIC до складних ООП мов типу Python.

На даний момент існує чимало відомих архітектурних моделей для додатків штучної нейронної мережі з різною обчислювальною складністю та показниками подібності до нейронів людського мозку. Ці моделі є унікальними і, на відміну від традиційних статистичних методів, не дозволяють застосувати до них існуючу таксономію.

Однією з основних проблем використання штучних нейронних мереж є неможливість заздалегідь визначити архітектуру проектованої мережі та її складність, що необхідно для бажаного рівня точності результатів. Можливо, доведеться додатково ускладнити архітектуру, оскільки вона надто вимоглива. Нейронна мережа лише з одним прихованим шаром справляється лише з найпростішими завданнями. Збільшення кількості прихованих шарів, необхідних для вирішення більш складних завдань, також вимагає збільшення обчислювальної потужності. Вирішенню цієї проблеми шляхом збільшення можливостей комп'ютерних процесорів і вдосконалення кластерних систем присвячено чимало досліджень. [13]

РОЗДІЛ 2 АЛГОРИТМ РОЗПІЗНАВАННЯ ЗОБРАЖЕНЬ ЗА ДОПОМОГОЮ НЕЙРОННИХ МЕРЕЖ

2.1 Застосування згорткових нейронних мереж

Згорткові нейронні мережі (CNN) широко використовуються у завданнях обробки зображень та вирішенні різних завдань розпізнавання об'єктів. Основними завданнями розпізнавання зображень є наступні:

1. Класифікація: У цьому завданні система повинна визначити, до якого класу належить об'єкт на зображенні.

2. Виявлення: В даному завданні система має виділити об'єкт на зображенні, обведенням його прямокутною рамкою.

3. Сегментація: Сегментація передбачає попіксельну класифікацію об'єктів на зображенні, тобто призначення класу кожному пікселю.

Згорткова нейронна мережа складається з наступних основних шарів:

1. Згортковий шар (Convolutional Layer): Цей шар є одним з ключових будівельних блоків нейронної мережі. Він використовує детектори ознак, які також називають ядрами або фільтрами, для аналізу зображення (рис. 2.1). Детектори ознак рухаються по зображенню, скануючи його різні області. Для кожної області обчислюється скалярний добуток між вхідним пікселем та фільтром, і результат подається у вихідний масив. Після обчислення скалярних добутків отримуємо карту ознак або карту активації.

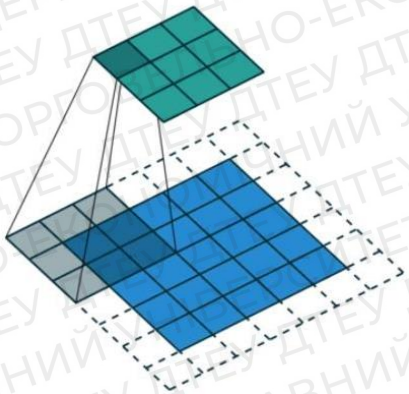

Рис. 2.1 – Візуалізація алгоритму згортки зображення

2. Шар підвибірки (Pooling Layer): Після згорткового шару слідує шар підвибірки (рис. 2.2), який зменшує розмірність та кількість параметрів у вхідних даних. У цьому шарі лише найбільш інформативні регіони зберігаються.

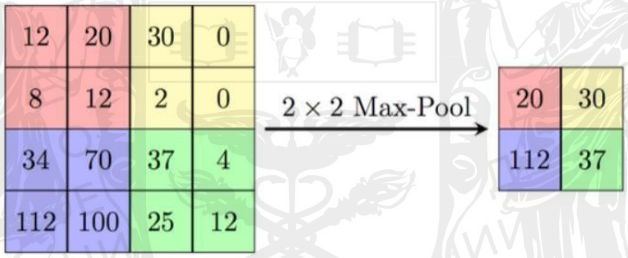

Рис. 2.2 – Приклад прорахунків по алгоритму підвибірки

3. Повністю зв'язаний шар (Fully Connected Layer): Після обробки зображення у попередніх шарах, дані передаються до повністю зв'язаного шару. Цей шар відповідає за регуляризацію та встановлення класу зображення на основі знайдених функцій.

Саме ці основні компоненти згорткової нейронної мережі вирішують завдання розпізнавання об'єктів на зображеннях [14].

2.2 Оцінка точності

Для оцінки точності прогнозованих значень у нашій роботі, ми використовуємо F-міру як науковий показник. Цей інструмент дозволяє нам

здійснити об’єктивну оцінку результатів класифікації. Перш за все, нам необхідно ввести такі важливі поняття, як істинно позитивний (True Positive, TP), істинно негативний (True Negative, TN), хибнопозитивний (False Positive, FP) і хибнонегативний (False Negative, FN). Припустимо, що y – це справжнє значення об’єкта, тоді як $\hat{y}$ – це прогнозоване значення.

Таблиця 2.1

Прогнозування значення

	$y=1$	$y=0$
$\hat{y}=1$	True Positive (TP)	False Positive (FP)
$\hat{y}=0$	False Negative (FN)	True Negative (TN)

Введемо поняття precision та recall:

$$precision = \frac{T \cdot P}{T \cdot P + F \cdot P} \tag{2.1}$$

$$recall = \frac{T \cdot P}{T \cdot P + F \cdot N} \tag{2.2}$$

Precision визначає, яка частина об’єктів, класифікованих класифікатором як позитивні, є дійсно позитивними. З іншого боку, recall визначає, яка частина дійсно позитивних об’єктів класифікується як позитивні класифікатором. У випадку бінарної класифікації, recall позитивного класу (наприклад, зображень з пухлинами) також називається «чутливістю», тоді як recall негативного класу (наприклад, зображень без пухлин) називається «специфічністю». Ці показники дозволяють нам уникнути тенденції класифікатора класифікувати всі об'єкти в одному класі.

Зазвичай для оптимізації параметрів використовується лише одна метрика, і ми слідкуємо за її зміною на тестовій вибірці. Існує різноманітні способи об’єднати precision і recall в один критерій. У нашій роботі ми використовуємо середнє гармонічне, відоме як F-міра.

$$F = \frac{precision * recall}{precision + recall} \quad (2.3)$$

Важливо зазначити, що F-міра має максимальне значення, коли якість класифікації відповідає найвищим стандартам і точність, і відкликання дорівнюють 1. У разі, якщо один із параметрів стає нульовим, F-міра також стає нульовою. Ця метрика допомагає нам об’єктивно оцінити ефективність нашого класифікатора [15].

2.3 Набір даних

У цій роботі ми використовуємо набір даних із 7023 МРТ-зображень головного мозку людини, зібраних із декількох відкритих наборів даних [16]. Дані містять чотири класи зображень: гліома, менінгіома, гіпофіз та зображення без пухлини (рис. 2.3).

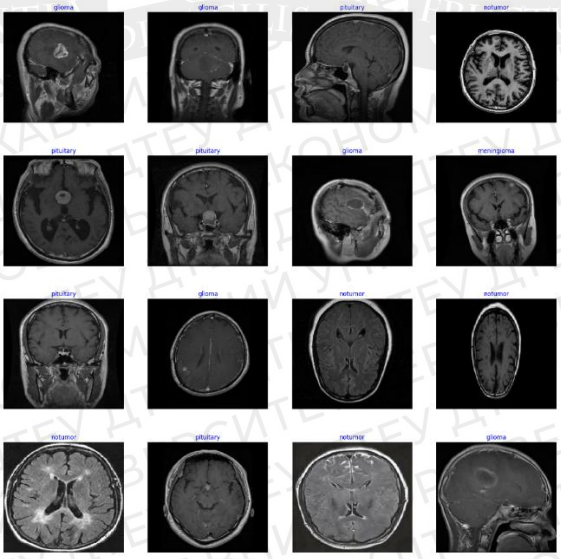

Рис. 2.3 – Навчальні дані

Гліома – це тип пухлини, яка виникає в мозку і спинному мозку. Гліоми розпочинаються в клітинах-клейовинах (глійних клітинах), які оточують нервові клітини і допомагають їм функціонувати. Три типи глійних клітин

можуть утворювати пухлини. Гліома може впливати на функцію вашого мозку і бути загрозливою для життя в залежності від її розташування та швидкості росту. Гліоми є одними з найпоширеніших типів первинних пухлин мозку.

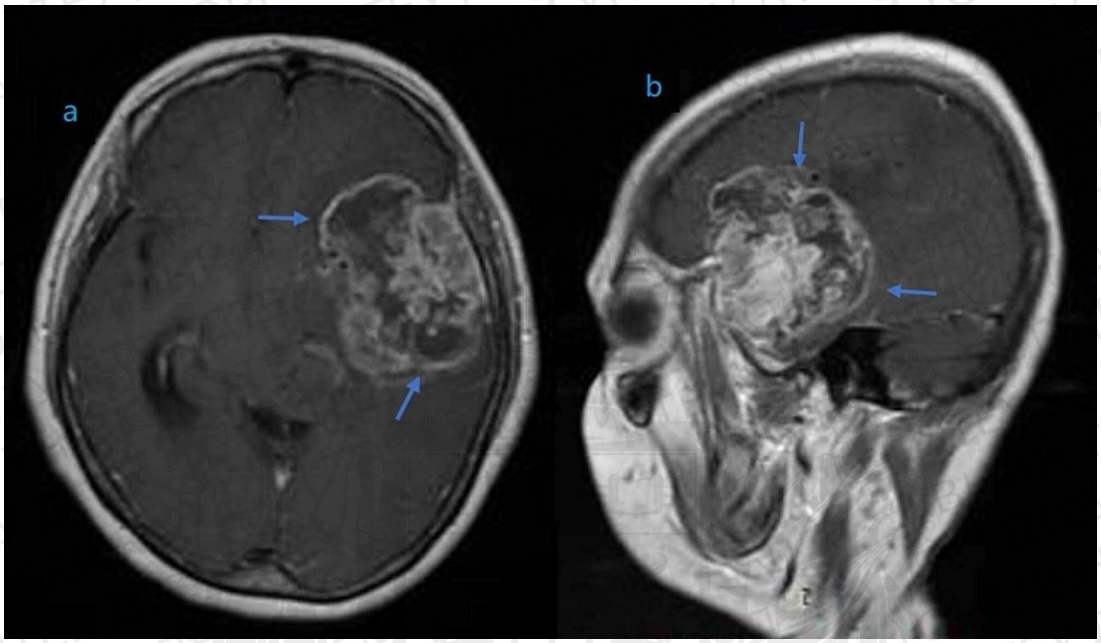

Рис. 2.4 – Приклад гліоми на МРТ знімках

Менінгіома – це пухлина, яка виникає з менінгів - оболонок, що оточують мозок і спинний мозок. Незважаючи на те, що це не технічно пухлина мозку, її включають в цю категорію, оскільки вона може стискати або стискувати суміжний мозок, нерви та судини. Менінгіома є найпоширенішим типом пухлин, які утворюються в голові. Більшість менінгіом ростуть дуже повільно, часто протягом багатьох років, не викликаючи симптомів. Але іноді їх вплив на навколишні тканини мозку, нерви або судини може призвести до серйозної інвалідності.

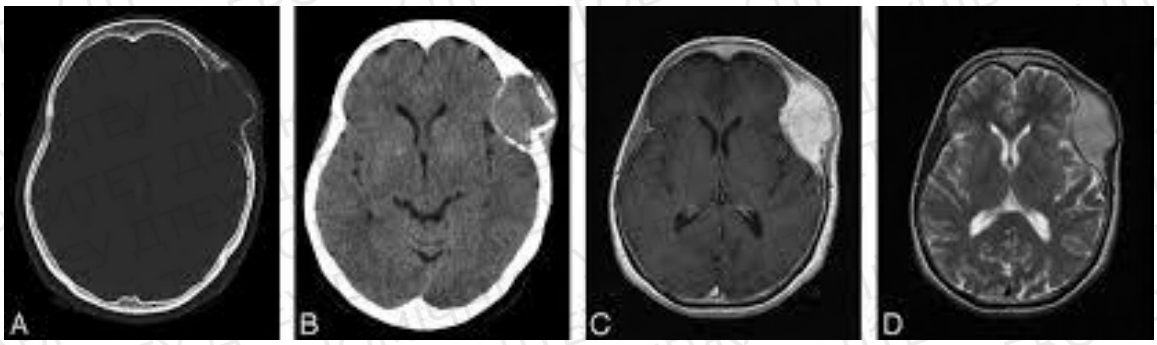

Рис. 2.5 – Приклад менінгіоми на МРТ знімках

Пухлини гіпофізу – це аномальні утворення, які розвиваються у вашій гіпофізній залозі. Деякі пухлини гіпофізу призводять до вироблення занадто великої кількості гормонів, які регулюють важливі функції вашого організму. Деякі пухлини гіпофізу можуть призвести до зниження рівня гормонів, які виробляє гіпофіз. Більшість пухлин гіпофізу є доброякісними (аденомами) утвореннями. Аденоми залишаються в гіпофізній залозі або навколишніх тканинах і не поширюються на інші частини організму.

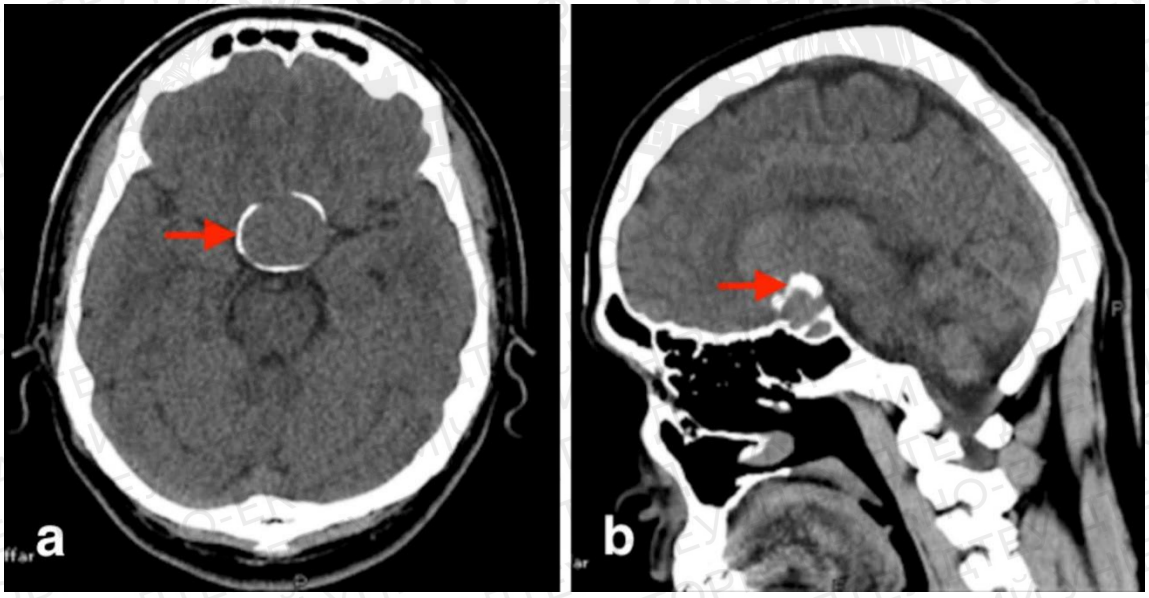

Рис. 2.6 – Приклад пухлини гіпофізу на МРТ знімках

Весь набір даних був розділений на навчальну та тестову частини. Кожна з цих частин містить однакову кількість зображень по кожному з класів захворювання та без нього. Усі 100% даних поділяються наступним чином: 80%

загального набору даних використовується для навчання моделі, а решта 20% використовуються для тестування.

2.4 CNN власної конфігурації

Створення нейронної мережі для обробки зображень, такої як згорткова нейронна мережа (CNN), є важливим етапом у вирішенні завдань комп'ютерного бачення та класифікації об'єктів на зображеннях. Розглянемо кожний з етапів створення такої мережі докладно.

1. Створення послідовної моделі:

На початку створюємо порожню послідовню модель нейронної мережі, використовуючи функцію Sequential(). Послідовна модель дозволяє додавати шари в послідовному порядку.

2. Додавання першого згорткового шару (Conv2D):

Перший згортковий шар – це ключова частина CNN. Ми використовуємо функцію Conv2D, додаючи 32 фільтри (ядра) розміром 3x3 до кожного вхідного зображення. Цей шар допомагає виділити важливі ознаки на зображеннях за допомогою алгоритму згортки. Для активації функції використовується ReLU, яка попереджає проблему зникнення градієнту.

3. Додавання другого згорткового шару:

Потім додаємо другий згортковий шар. Це допомагає нейронній мережі виявляти більш складні ознаки на зображеннях.

4. Додавання першого шару максимального згорткового пулінгу (MaxPool2D):

Після кожного згорткового шару ми додаємо шар максимального згорткового пулінгу. MaxPool2D дозволяє зменшити розмір карти ознак, зберігаючи важливі ознаки. У цьому шарі ми використовуємо ядро розміром 2x2 і вказуємо параметри, такі як стрибки (strides) і доповнення (padding).

5. Додавання інших згорткових шарів:

Додавання подібних згорткових та максимального згорткового пулінгу шарів допомагає нейронній мережі виявляти все більше і більше абстрактних ознак на зображеннях.

6. Додавання шару розгортання (Flatten):

Після відпрацювання всіх згорткових шарів, ми додаємо шар розгортання, який перетворює матрицю ознак у вектор. Це дозволяє передавати вектор ознак наступним повністю з’єднаним шарам.

7. Додавання першого повністю з’єднаного шару (Dense):

Перший повністю з’єднаний шар має 32 нейрони і використовує функцію активації ReLU. У цьому шарі налаштовуються ваги і зсуви для навчання моделі. Крім того, вказується, чи використовувати зсув і тип ініціалізації ваг.

8. Використання Dropout шару:

Вводиться Dropout шар, який випадково вимикає деякі нейрони під час навчання, щоб запобігти перенавчанню. Це допомагає збалансувати модель і підвищити її загальну здатність.

9. Вихідний шар (Dense):

В останньому повністю з’єднаному шарі маємо 4 нейрони, оскільки ця модель призначена для багатокласової класифікації з 4 категоріями. Функція активації залежить від завдання, наприклад, softmax для класифікації.

Завершуючи, створення CNN – це складний процес, де кожен шар виконує певні операції з даними та допомагає моделі вчитися розрізняти важливі ознаки на зображеннях. Налаштування параметрів, таких як кількість фільтрів, розміри ядер, функції активації і регуляризація, грають важливу роль у досягненні бажаного результату в задачах обробки зображень.

2.5 Особливості CNN

Згорткова нейрона мережа, в даному випадку, призначена для класифікації зображень на основі витягнутих ознак, які вона отримує зі згорткових шарів, і видає ймовірності належності до різних класів. Вона добре підходить для задач багатокласової класифікації, таких як класифікація медичних зображень мозку на основі видимих ознак. Розглянемо її особливості більш детально.

1. Згорткові та Пулінг шари:

Модель включає чотири згорткових шари, що дозволяють витягувати важливі ознаки зображення, зазвичай використовуючи ReLU для активації. Два пулінг шари служать для пониження просторового розміру виходів попередніх шарів, зменшуючи кількість параметрів у моделі і покращуючи швидкодію.

2. Повністю з'єднані шари:

Модель включає два повністю з'єднаних шари після згорткових і пулінг шарів для об'єднання і обробки витягнутих ознак. Використовується функція активації ReLU для нейронів цих шарів.

3. Випадкове відключення (Dropout):

Використовується шар випадкового відключення з імовірністю 40%. Це допомагає уникнути перенавчання, зменшуючи зв'язки між нейронами під час тренування.

4. Вихідний шар:

Вихідний шар має 4 нейрони, які відповідають чотирьом класам (glioma, meningioma, normal, adenoma). Функція активації softmax використовується для призначення ймовірностей кожному з класів.

5. Оптимізатор і функція втрати:

Для навчання моделі використовується оптимізатор Adam, який ефективно апдейтує ваги нейронів. Функція втрати встановлена як категоріальна перехресна ентропія, що підходить для задачі багатокласової класифікації.

6. Навчання та валідація:

Модель навчається на тренувальних даних протягом 60 епох та оцінюється на валідаційних даних. Історія навчання зберігається для подальшої аналітики.

7. Збереження моделі:

Після навчання модель зберігається у файл 'weights.h5' для подальшого використання без необхідності перенавчання.

2.6 Функції активації

Функції активації в нейронних мережах використовуються для введення нелінійності в модель і активації нейронів у шарах. Вони допомагають нейронам вивчати та виражати складні відносини між вхідними та вихідними даними. Важливо вибирати правильну функцію активації в залежності від

конкретного завдання та архітектури мережі. Ось декілька типових функцій активації:

1. ReLU (Rectified Linear Unit):

ReLU (Rectified Linear Unit) - це функція активації, яка широко використовується в нейронних мережах для введення нелінійності. Вона задається просто як:

$$f(x) = max(0, x) \tag{2.4}$$

Існує декілька переваг використання ReLU:

1. Нелінійність: ReLU надає нелінійність моделі, дозволяючи їй вивчати складні залежності в даних.
2. Обчислювальна ефективність: Інші функції активації, такі як Sigmoid та Tanh, можуть викликати проблему з вицвітанням градієнту, але ReLU має тенденцію до обчислювальної ефективності, оскільки він призводить до простої операції max.
3. Запобігає вицвітанню градієнту: ReLU допомагає запобігти проблемі вицвітання градієнту, яка може виникнути при навчанні глибоких нейронних мереж.

Згорткові шари з функцією активації ReLU допомагають витягти важливі ознаки зображень і вирізняються високою ефективністю у багатьох завданнях комп'ютерного зору та обробки зображень.

2. Softmax:

Функція активації Softmax використовується для приведення вихідних значень нейронів в останньому повністю з'єднаному шарі до ймовірностей. Основна ідея Softmax полягає в тому, щоб перетворити вихідні значення на шарі вектора у ймовірності, які сумуються до 1. Функція Softmax визначається наступним чином:

$$Softmax(x)_i = \frac{e^{x_i}}{\sum_j e^{x_j}} \quad (2.5)$$

Де x_i - це вихідне значення нейрона, а $\sum_j e^{x_j}$ - сума експонент вихідних значень усіх нейронів.

Основні характеристики Softmax:

1. Ймовірнісний вихід: Softmax перетворює вихідні значення в ймовірності, де кожне значення вказує на ймовірність належності до конкретного класу.

2. Сума ймовірностей рівна 1: Сума всіх ймовірностей для різних класів буде дорівнювати 1, що робить його ідеальним для задач багатокласової класифікації.

3. Використання в останньому шарі мережі: Softmax зазвичай використовується в останньому повністю з'єднаному шарі для отримання ймовірностей класів у задачах класифікації.

РОЗДІЛ 3 РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

В наш час існує величезна розмаїтість мов програмування, а також програмних бібліотек, призначених для використання штучних нейронних мереж і генетичних алгоритмів. Ці бібліотеки доступні на різних мовах програмування, таких як C++, C#, Python і багатьох інших.

3.1 Мова розробки та використані бібліотеки

У цьому розділі курсової роботи розглянемо мову програмування Python, її історію та бібліотеки Tensorflow, Keras і Streamlit, що були використані під час розробки проекту.

Мова програмування Python

Мовою програмування, обраною для вирішення задач та реалізації алгоритмів у даній дипломній роботі, є Python. Python є високорівневою мовою загального призначення, спрямованою на підвищення продуктивності розробника та читабельність коду. Його синтаксис мінімалістичний, але стандартна бібліотека включає значний набір корисних функцій [17].

Мова програмування Python була створена в кінці 1980-х років Гвідо ван Россумом у Нідерландах. Вона була розроблена як високорівнева мова програмування з акцентом на простоті та читабельності коду. Python швидко став популярним серед програмістів завдяки своїй зрозумілості, що сприяло швидкому росту спільноти розробників та розширенню функціональності мови.

У 2000 році було випущено Python 2, який став дуже популярним і знайшов широке застосування. Однак, з розвитком часу, було виявлено деякі недоліки у версії 2, і у 2008 році було запущено Python 3, який вніс багато удосконалень та покращень. На сьогоднішній день Python 3 є актуальною

версією мови і має велику спільноту розробників та багатий екосистему бібліотек.

Приклад програми на мові програмування Python наведено на рисунку 3.1.

```
# Python 3: List comprehensions
>>> fruits = ['Banana', 'Apple', 'Lime']
>>> loud_fruits = [fruit.upper() for fruit in
fruits]
>>> print(loud_fruits)
['BANANA', 'APPLE', 'LIME']

# List and the enumerate function
>>> list(enumerate(fruits))
[(0, 'Banana'), (1, 'Apple'), (2, 'Lime')]
```

Рис. 3.1 – Приклад програми на мові Python

Бібліотеки Tensorflow та Keras

Для розробки моделі машинного навчання для проекту були використані бібліотеки Tensorflow і Keras. Tensorflow, розроблений командою Google Brain, був випущений в 2015 році. Ця бібліотека стала відомою своєю швидкодією та здатністю працювати з глибокими нейронними мережами. Вона стала однією з основних платформ для роботи зі штучним інтелектом та глибоким навчанням.

Keras, з іншого боку, є вищестоячим інтерфейсом для Tensorflow і розроблений таким чином, щоб спростити процес створення та навчання нейронних мереж. Keras надає зручний і інтуїтивний інтерфейс для визначення архітектури мережі та легкої роботи з даними.

Бібліотека Streamlit

Для створення інтерактивного веб-додатку була використана бібліотека Streamlit. Streamlit виникла в 2017 році і за короткий час здобула велику популярність завдяки своїй простоті та швидкості розробки. Вона дозволяє програмістам створювати веб-додатки, просто використовуючи звичайний Python-код. Ця бібліотека стала особливо корисною для розробників, які бажають продемонструвати результати моделей машинного навчання або створити інтерактивні інтерфейси для аналізу даних. Streamlit дозволяє швидко і легко інтегрувати результати роботи моделей Tensorflow і Keras у веб-додаток, що спрощує взаємодію з користувачами через браузер.

3.2 Середовище розробки

PyCharm – це потужна інтегрована середовище розробки (IDE) для мови програмування Python. Ця IDE розроблена компанією JetBrains і визнана однією з найкращих інструментів для розробки Python-програм. Однією з головних переваг PyCharm є його висока продуктивність і зручність використання для розробників.

Однією з ключових функцій PyCharm є автокомплітація, яка допомагає розробникам швидше писати код, заповнюючи ключові слова, методи та змінні автоматично. Також важливою є можливість відлагодження коду в PyCharm, що робить процес виправлення помилок більш ефективним.

PyCharm також має інтеграцію з популярними системами керування версіями, такими як Git, і дозволяє командам розробників легко спільно працювати над проектами. Крім того, вона підтримує велику кількість розширень і плагінів, що дозволяє налаштувати середовище розробки під конкретні потреби проекту.

Загалом, PyCharm – це незамінний інструмент для розробки Python-програм, який сприяє підвищенню продуктивності розробників та полегшує роботу з Python.

3.3 Розробка програми

Тренування моделі CNN

Код містить тренування згорткової нейронної мережі (CNN) для класифікації зображень мозкових опухолей на чотири категорії: «glioma», «meningioma», «normal» і «adenoma». Нижче прокоментуємо кожен етап тренування та використання моделі.

1. Імпорт бібліотек і підготовка даних:

- В цьому кроці імпортуються необхідні бібліотеки, такі як TensorFlow і Keras.

- Зображення для тренувального та валідаційного наборів завантажуються за допомогою ImageDataGenerator, а також застосовує попередню обробку, таку як масштабування та аугментація.

```
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
from keras.preprocessing.image import ImageDataGenerator
from keras.utils import load_img, img_to_array
from keras.models import Sequential
from keras.layers import Conv2D, MaxPool2D, Flatten, Dense, Dropout
from keras.activations import relu, softmax
from keras.optimizers import Adam
from numpy.random import seed

seed(1)
tf.random.set_seed(2)

train_data_gen = ImageDataGenerator(rescale=1. / 255,
                                    shear_range=0.2,
                                    rotation_range=2,
                                    zoom_range=0.2,
                                    horizontal_flip=True,
                                    vertical_flip=True)

training_set = train_data_gen.flow_from_directory(directory="./dataset/Training",
                                                  target_size=(224, 224),
                                                  class_mode='categorical',
                                                  batch_size=32)

validation_data_gen = ImageDataGenerator(rescale=1. / 255)
```

Рис. 3.2 – Код імпортування бібліотек та підготовки даних

2. Визначення структури моделі:

- Модель є послідовною (Sequential), і до неї додаються шари CNN з різними параметрами, такі як кількість фільтрів, розмір ядра та активаційна функція.

- Між деякими свіжими шарами також додані шари пулінгу для зменшення розміру зображення.

- Додаються повністю підключені (fully connected) шари з активаційною функцією ReLU та dropout шар для регуляризації.

```
model = Sequential()
model.add(Conv2D(filters=32, kernel_size=3, activation=relu, input_shape=[224, 224, 3]))
model.add(Conv2D(filters=32, kernel_size=3, activation=relu))
model.add(MaxPool2D(pool_size=2, strides=2, padding='valid'))
model.add(Conv2D(filters=32, kernel_size=3, activation=relu))
model.add(Conv2D(filters=64, kernel_size=3, activation=relu))
model.add(MaxPool2D(pool_size=2, strides=2, padding='valid'))
model.add(Flatten())
model.add(Dense(units=32, activation=relu, use_bias=True))
model.add(Dropout(0.4))
model.add(Dense(units=16, activation=relu, use_bias=True))
model.add(Dense(units=4,activation=softmax))
```

Рис. 3.3 – Код визначення структури моделі

3. Компіляція моделі:

- Вказується оптимізатор (Adam), функція втрати (categorical_crossentropy) та метрика (accuracy) для компіляції моделі.

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

Рис. 3.4 – Кодовий рядок процесу компіляції моделі

4. Тренування моделі:

- Модель тренується за допомогою model.fit, використовуючи тренувальний набір даних training_set та валідаційний набір даних validation_set. Тренування відбувається протягом 60 епох.

```
model_history = model.fit(x=training_set, validation_data=validation_set, epochs=60, verbose=1)

print(model_history.history.keys())
```

Рис. 3.5 – Код процесу тренування моделі

5. Візуалізація результатів:

- Графіки показників тренування, такі як точність та втрата, відображаються за допомогою matplotlib.

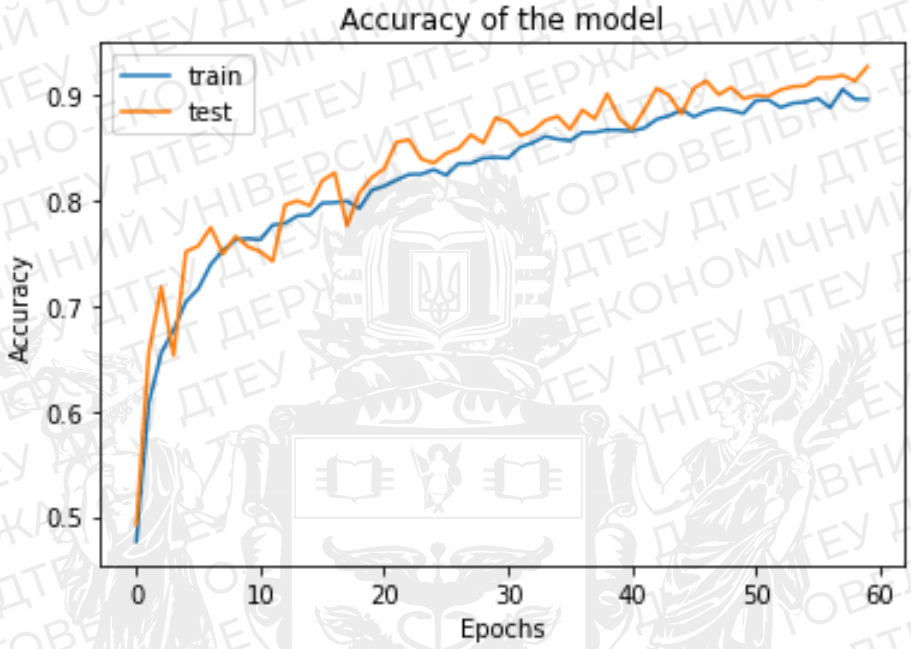

Рис. 3.6 – Візуалізація метрики точності на тестовому та тренувальному датасеті

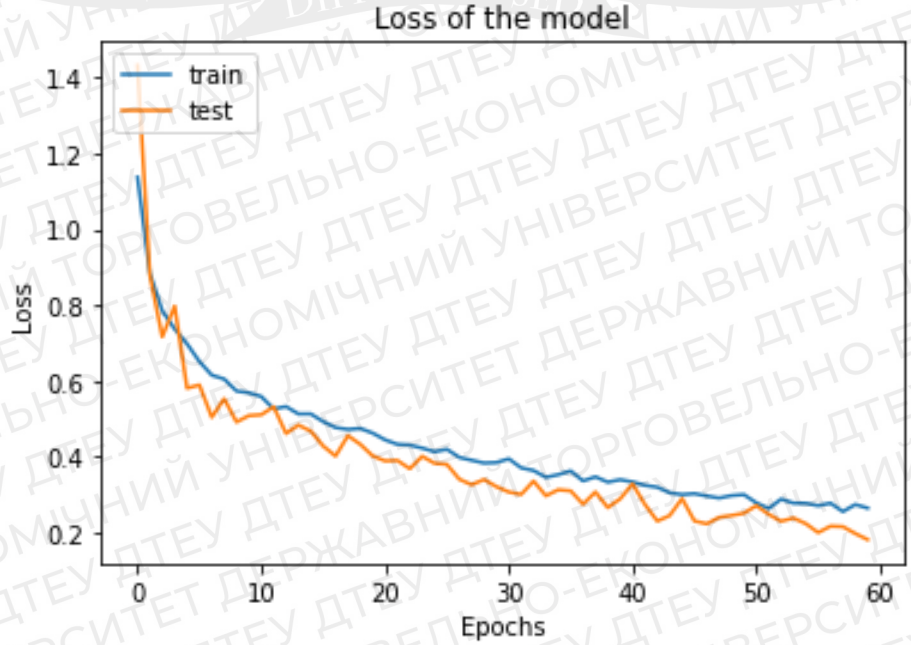

Рис. 3.7 – Візуалізація функції втрат на тестовому та тренувальному датасеті

6. Тестування моделі:

- Модель використовується для класифікації декількох тестових зображень, і результати виводяться на екран разом з назвою класу.

```
index = ['glioma', 'meningioma', 'normal', 'adenoma']

test_image1 = load_img( path: './dataset/Testing/meningioma/Te-me_0018.jpg', target_size=(224, 224))
test_image1 = img_to_array(test_image1)
test_image1 = np.expand_dims(test_image1, axis=0)
result1 = np.argmax(model.predict(test_image1 / 255.0), axis=1)
print(index[result1[0]])

test_image2 = load_img( path: './dataset/Testing/pituitary/Te-pi_0018.jpg', target_size=(224, 224))
test_image2 = img_to_array(test_image2)
test_image2 = np.expand_dims(test_image2, axis=0)
result2 = np.argmax(model.predict(test_image2 / 255.0), axis=1)
print(index[result2[0]])
```

Рис. 3.8 – Код тестування моделі та 2-х тестових зображеннях

7. Збереження моделі:

- Натренована модель зберігається у файлі «weights.h5» для подальшого використання.

```
model.save('weights.h5')
```

Рис. 3.9 – Код збереження моделі

Функціонал коду здійснює всі необхідні кроки для створення, тренування і збереження моделі CNN для класифікації мозкових опухолей.

Використання натренованої моделі CNN

Цей код є частиною додатку для виявлення пухлин на зображеннях головного мозку, використовуючи модель глибокого навчання (CNN). Давайте розглянемо код поетапно:

1. Імпорт бібліотек:

- Код починається з імпорту необхідних бібліотек, таких як streamlit, numpy, tensorflow.keras, matplotlib, і так далі. Ці бібліотеки використовуються для створення веб-додатка, обробки зображень та візуалізації результатів.

```
import streamlit as st
import numpy as np

from tensorflow import keras
from keras.utils import load_img, img_to_array
from matplotlib import cm
import matplotlib.pyplot as plt

from keras.applications.vgg16 import preprocess_input
from tf_keras_vis.utils.model_modifiers import ReplaceToLinear
from tf_keras_vis.utils.scores import CategoricalScore
from tf_keras_vis.saliency import Saliency
```

Рис. 3.10 – Код імпорту бібліотек

2. Налаштування сторінки:

- st.set_page_config() встановлює конфігурацію сторінки додатка Streamlit, включаючи назву сторінки.

```
st.set_page_config(
    page_title="🧠 MRI Brain Tumor Detection",
)

index = {0: 'glioma', 1: 'meningioma', 2: 'normal', 3: 'adenoma'}
```

Рис. 3.11 – Код налаштування назви сторінки

3. Завантаження попередньо навченої моделі:

- Модель для виявлення пухлин на зображеннях головного мозку завантажується з файлу «weights.h5».

```
loaded_model = keras.models.load_model("weights.h5")

st.title("🧠 MRI Brain Tumor Detection")
```

Рис. 3.12 – Код завантаження попередньо навченої моделі

4. Завантаження зображення:

- За допомогою st.file_uploader можна вибрати зображення для обробки.

```
# Upload an image
uploaded_image = st.file_uploader("Upload an MRI Brain Image", type=["jpg", "png", "jpeg"])
```

Рис. 3.13 – Код завантаження зображення

5. Функція smoothgrad:

- Ця функція використовується для створення теплової карти (heatmap) для підкреслення областей зображення, які впливають на прогноз моделі. Вона приймає зображення та модель.

```
def smoothgrad(image, model):
    score = CategoricalScore([1])
    images = np.asarray([np.array(image)])
    x = preprocess_input(images)

    saliency = Saliency(model, model_modifier=ReplaceToLinear(), clone=True)
    cam = saliency(score, x, smooth_samples=20, smooth_noise=0.2)

    return cam, images[0]
```

Рис. 3.14 – Код функції smoothgrad

6. Обробка завантаженого зображення:

- Завантажене зображення конвертується у формат, придатний для передачі до моделі, та змінюється його розмір до 224x224 пікселів.

```python
if uploaded_image is not None:
    st.image(uploaded_image, caption="Image to Process", use_column_width=True)
    origin = load_img(uploaded_image, target_size=(224, 224))

    image_to_pred = img_to_array(origin)
    image_to_pred = np.expand_dims(image_to_pred, axis=0)
```

Рис. 3.15 – Код обробки завантаженого зображення

7. Кнопка «Detect Tumor»:

- По натисканню цієї кнопки виконується передбачення на зображенні, використовуючи завантажену модель. Результат виводиться як інформаційне повідомлення з передбаченою категорією.

```python
if st.button("Detect Tumor"):
    # Make predictions using your model
    result = np.argmax(loaded_model.predict(image_to_pred / 255.0), axis=1)
    st.info(f"Predicted class: **{index[result[0]]}**")

    cam, img = smoothgrad(origin, loaded_model)
    # Overlay the heatmap on the image
    fig, ax = plt.subplots(figsize=(8, 6))
```

Рис. 3.16 – Код кнопки «Detect Tumor»

8. Візуалізація результатів:

- Створюється теплова карта (heatmap), яка накладається на оригінальне зображення для виділення областей, які впливають на прогноз моделі.

```python
# Overlay the heatmap on the image
heatmap = np.uint8(cm.jet(cam)[..., :3] * 255)
ax.imshow(img)
ax.imshow(heatmap[0], cmap='jet', alpha=0.5)
ax.axis('off')

st.pyplot(fig)
```

Рис. 3.17 – Код візуалізації результатів

Функції даного коду дозволяють користувачу завантажувати МРТ зображення головного мозку, передбачати наявність пухлини та візуалізувати, які частини зображення впливають на прийняте рішення моделі.

3.5 Результат роботи програми

Для запуск програми необхідно відкрити файл main.py, та запустити у PyCharm. Перед вами з’явиться наступний веб-інтерфейс (рис. 3.18)

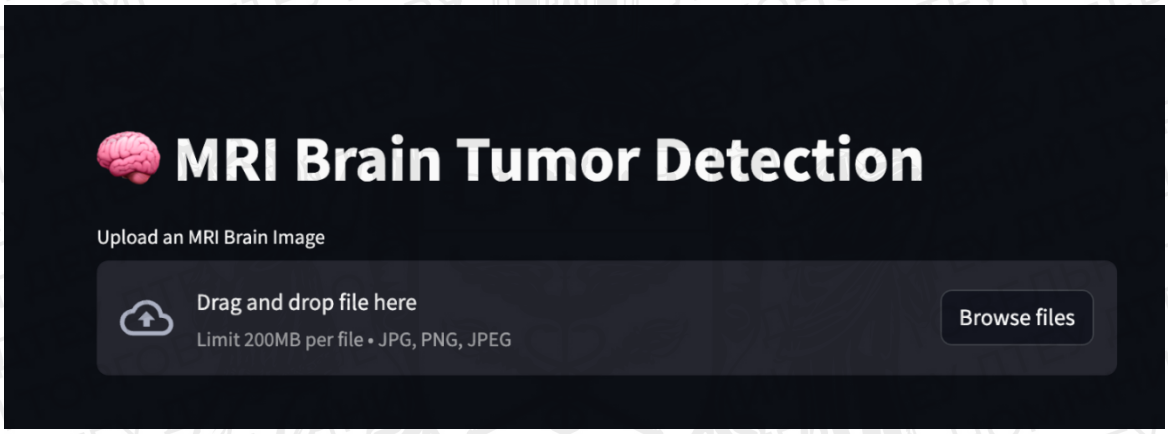

Рис. 3.18 – Початковий веб-інтерфейс програми

Натиснувши на кнопку «Browse files» та вибравши необхідне зображення, пройде завантаження картинки (рис. 3.19).

Рис. 3.19 – Вибір знімку

Далі натискаємо на кнопку «Detect Tumor». Перед вами з’явиться підпис про клас можливої пухлини та області на які необхідно звернути увагу у вигляді heatmap (рис. 3.20)

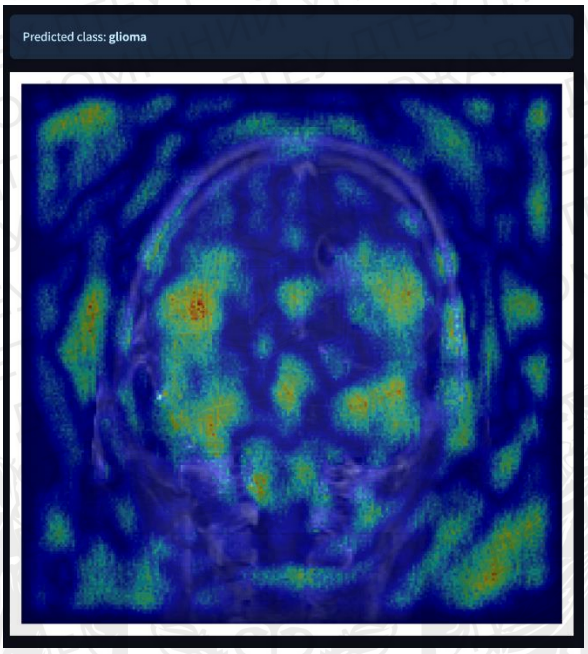

Рис. 3.20 – Області з можливою пухлиною та її клас

Нижче наведено декілька прикладів роботи програми (рис 3.21-3.22)

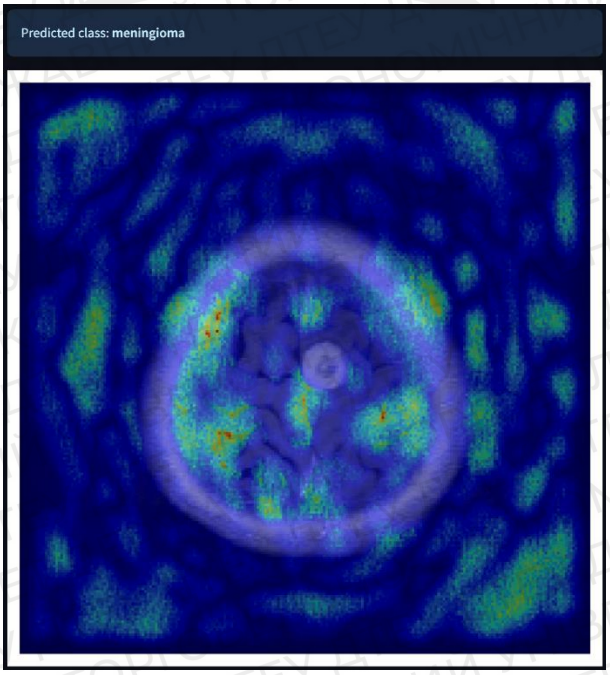

Рис. 3.21 – Тестування програми

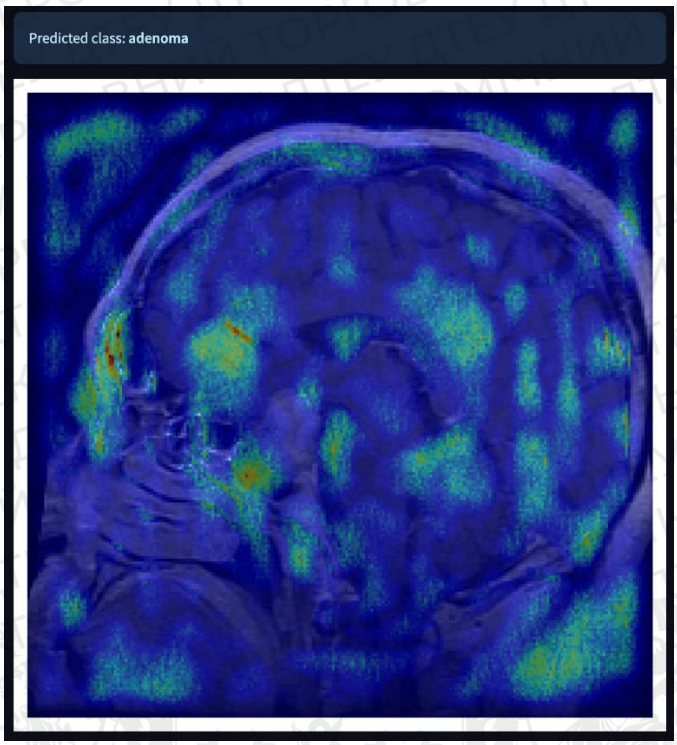

Рис. 3.22 – Тестування програми

3.6 Набір даних

Актуальність набору даних

Цей набір даних є важливим ресурсом для дослідження МРТ-зображень головного мозку людини, оскільки він складається з 7023 зображень, які представляють чотири основні класи: гліома, менінгіома, відсутність пухлини та гіпофіз [16]. Відмінність цього набору даних в тому, що жодні зображення з класу «пухлин» не були включені з набору Br35H, що може вплинути на точність класифікації цього класу.

Проблема з набором даних SARTAJ

Однак важливо відзначити, що набір даних SARTAJ також має свої особливості. Зокрема, відомо, що класифікація зображень гліоми в цьому наборі даних може бути неправильною. Цю проблему було виявлено завдяки результатам робіт інших дослідників та експериментам з різними навчальними моделями. Відповідно, автор набору даних вирішив видалити зображення з цього класу та замінити їх іншими зображеннями, отриманими з сайту figshare. Ця діяльність спрямована на покращення точності класифікації та надання дослідникам надійного набору даних для подальших досліджень у цій області.

Висновок

Цей набір даних є цінним інструментом для наукових досліджень з області медичної обробки зображень та діагностики за допомогою МРТ. Варто враховувати його особливості, зокрема, відсутність зображень пухлин з набору Br35H та проблеми з класифікацією зображень гліоми в наборі SARTAJ. Зусиллями спільноти дослідників ці проблеми можна вирішити та покращити доступ до надійних даних для медичних досліджень і практичного застосування.

3.7 Результати експерименту сегментація знімків МРТ

Налаштування мережі проводилося на 60 епохах і займало 120 хвилин часу, в середньому 2 хвилини на кожну епоху.

Всього CNN має 4 згорткових шари, 2 пулінгових шари та 3 повнозв'язних шари, повна архітектура яких описана вище.

Результати експерименту з сегментації знімків МРТ головного мозку, в якому модель була натренована протягом 60 епох, валідаційна точність склала 92%, точність на тестових даних - 91%, та значення функції втрат (лосс) досягло 0.1820, вказують на високу ефективність та точність моделі в завданні сегментації знімків МРТ головного мозку.

Модель була створена для класифікації чотирьох різних класів: гліома, менінгіома, відсутність пухлини та гіпофіз. Ці класи представляють різні структурні особливості та патології головного мозку, і точність класифікації є важливим показником для правильної діагностики та лікування пацієнтів.

Валідаційна точність на рівні 92% свідчить про те, що модель добре узгоджується з тестовими даними, тобто вона здатна правильно ідентифікувати патологічні області на знім МРТ головного мозку з високою точністю. Це може бути корисним для розпізнавання різних видів пухлин та інших аномалій.

Точність на рівні 91% також є досить високою, враховуючи складність завдання сегментації знімків МРТ. Ця метрика вказує на здатність моделі правильно класифікувати області, які не входять до категорій "гліома" та "менінгіома", такі як "відсутність пухлини" та "гіпофіз".

Значення функції втрат (лосс) 0.1820 свідчить про те, що модель ефективно навчилася під час тренування та має низьку помилку під час класифікації знімків МРТ.

Узагальнюючи, експеримент показав, що навчена модель демонструє високу точність та ефективність у сегментації знімків МРТ головного мозку, що може бути корисним для медичних застосувань, таких як діагностика та відстеження розвитку пухлин та інших патологічних станів у пацієнтів.

3.8 Висновки за результатами експерименту

Висновки за результатами експерименту дозволяють зробити кілька важливих висновків:

1. Ефективність моделі: Модель, навчена для сегментації знімків МРТ головного мозку, продемонструвала високу ефективність, що відображається у високих метриках точності на валідаційних та тестових даних. Валідаційна точність складає 92%, а точність на тестових даних - 91%.

2. Класифікація патологій: Модель була спроектована для класифікації чотирьох різних класів патологій головного мозку: гліома, менінгіома, відсутність пухлини та гіпофіз. Висока точність вказує на здатність моделі вірно ідентифікувати ці патології.

3. Низька помилка: Значення функції втрат (лосс) на рівні 0.1820 свідчить про те, що модель ефективно навчилася та має низьку помилку під час сегментації знімків МРТ головного мозку.

4. Медичний застосунок: Результати експерименту свідчать про великий потенціал моделі у медичних застосуваннях. Вона може бути використана для діагностики та відстеження розвитку різних видів пухлин та інших патологій у пацієнтів.

Узагальнюючи, цей експеримент демонструє успішність навчання моделі для сегментації знімків МРТ головного мозку і вказує на можливості її застосування у сфері медицини для покращення точності діагностики та лікування пацієнтів з патологічними станами головного мозку.

ВИСНОВКИ

Метою цієї роботи є розробка методів обробки зображень на основі згорткових нейронних мереж.

Виконано наступні завдання:

- створено згорткову нейронну мережу для класифікації зображень;
- завантажено датасет, вибірку поділено на тренувану і тестову, дані підготовлено до навчання;
- натреновано модель на підготовлених даних для класифікації зображень;
- побудовано графік точності і функції втрат;
- створено веб-інтерфейс для використання моделі та відображення результатів;

Досліджено методи обробки зображень з використанням згорткових нейронних мереж. В результаті розроблено архітектуру багатошарової нейронної мережі з використанням згорткових шарів. Модель навчено на комбінації датасетів, figshare, SARTAJ, Br35H, що разом становлять 7023 зображення. Валідаційна очність моделі після 60 епох складає склала 92%, точність на тестових данних - 91%, та значення функції втрат досягло 0.1820. В результаті розроблено web-сервіс для розпізнавання пухлин головного мозку.

Підсумовуючи, можна сказати, що всі поставлені завдання виконано. На основі роботи чотирьох розроблених моделей нейронної мережі зображення класифікуються за наявністю пухлини та вибирається найкраще, яке показує значення F-міри, що дорівнює 0,92.

У майбутньому, використання нейронних мереж проявить свою життєздатність у сфері програмного забезпечення, в машинах, що здатні ефективно обробляти великі обсяги даних, і в технологічних системах, які можуть сприймати і виконувати завдання, недоступні для людей. Поточні наукові дослідження, пов'язані з застосуванням нейронних мереж,

демонструють перспективи цього напряму та багато інших неочікуваниих можливостей.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Чурюмова І. Г. Медична система прийняття рішень із використанням нейронної мережі // Вісник Нац. техн. ун-ту "ХПІ": зб. наук. тр. Темат. вип. : Інформатика та моделювання. - Харків: НТУ "ХПІ". - 2004. - № 34. - С. 199-202.

2. Samuel Burns. Python Deep learning // Kindle Edition, 2019, С. 178

3. Ramaswamy Reddy A., Prasad E. V., Reddy L. S. S. Comparative analysis of brain tumor detection using different segmentation techniques //International Journal of Computer Applications. – 2013. – Т. 82. – №. 14. - С. 0975-8887.

4. Das S., Siddiqui N. N., Kriti N., & Tamang, S. P. Detection and area calculation of brain tumour from MRI images using MATLAB //International Journal. – 2017. – Т. 4. – №. 1. - С. 35

5. Othman M. F. B., Abdullah N. B., Kamal N. F. B. MRI brain classification using support vector machine //2011 Fourth International Conference on Modeling, Simulation and Applied Optimization. - IEEE, 2011. - С. 1-4.

6. Кумар DD, Vanhana S., Priya, K. S., & SubashiniS. J.Braintumour image segmentation using MATLAB //vol. - 2015. - Т. 1. - С. 447-451.

7. Zhang Y., Dong Z., Wu L., & Wang S. A hybrid метод для MRI brain image classification //Expert Systems with Applications. – 2011. – Т. 38. – №. 8. - С. 10049-10053.

8. Othman M. F., Basri M. A. M. Probabilistic neural network for brain tumor classification //2011 Second International Conference on Intelligent Systems, Modelling and Simulation. - IEEE, 2011. - С. 136-138.

9. 6 Types of Activation Function in Neural Networks You Need to Know [Електронний ресурс], // URL: https://www.upgrad.com/blog/types-of-activation-function-in-neural-networks/ (дата звернення: 01.10.2023)

10. Neural Network | Machine Learning Tutorial [Електронний ресурс], // URL: https://sci2lab.github.io/ml_tutorial/neural_network/#Common-Activation-Functions (дата звернення: 01.10.2023)

11. What Is a Neural Network? [Електронний ресурс], 1994-2021 // The MathWorks, Inc, URL: https://www.mathworks.com/discovery/neural-network.html (дата звернення: 01.10.2023)

12. Neurohive. Рутинні завдання з мінімальним ризиком [Електронний ресурс], // URL: https://neurohive.io/ru/novosti/nejronnye-seti-v-medicine/ (01.10.2023)

13. Волчек Ю.А., Шишко О.М., Спірідонова О.С., Мохорт Т.В.. Положення моделі штучної нейронної мережі в медичних експертних системах// Juvenis scientia. - 2017. - №. 9.

14. MRI-based brain tumour image detection using CNN based deep learning method [Електронний ресурс], // URL: https://sciencedirect.com/science/article/pii/S277252862200022X (дата звернення: 02.10.2023)

15. Artificial Intelligence and Machine Learning: What You Always Wanted to Know but Were Afraid to Ask [Електронний ресурс], // URL: https://sciencedirect.com/science/article/pii/S277257232100025X (дата звернення: 02.10.2023)

16. Brain Tumor MRI Dataset [Електронний ресурс], // URL: https://kaggle.com/datasets/masoudnickparvar/brain-tumor-mri-dataset (дата звернення: 02.10.2023)

17. Welcome to Python.org [Електронний ресурс], // URL: https://python.org (дата звернення: 02.10.2023)

ДОДАТКИ

Додаток А (main.py)

```
import streamlit as st
import numpy as np

from tensorflow import keras
from keras.utils import load_img, img_to_array
from matplotlib import cm
import matplotlib.pyplot as plt

from keras.applications.vgg16 import preprocess_input
from tf_keras_vis.utils.model_modifiers import ReplaceToLinear
from tf_keras_vis.utils.scores import CategoricalScore
from tf_keras_vis.saliency import Saliency

st.set_page_config(
    page_title="□ MRI Brain Tumor Detection",
)

# Load your pre-trained tumor detection model
index = {0: 'glioma', 1: 'meningioma', 2: 'normal', 3: 'adenoma'}

loaded_model = keras.models.load_model("weights_2.h5")

st.title("□ MRI Brain Tumor Detection")

# Upload an image
uploaded_image = st.file_uploader("Upload an MRI Brain Image",
type=["jpg", "png", "jpeg"])


def smoothgrad(image, model):
    score = CategoricalScore([1])
    images = np.asarray([np.array(image)])
    x = preprocess_input(images)

    saliency = Saliency(model, model_modifier=ReplaceToLinear(),
clone=True)
    cam = saliency(score, x, smooth_samples=20, smooth_noise=0.2)

    return cam, images[0]


if uploaded_image is not None:
    st.image(uploaded_image, caption="Image to Process",
use_column_width=True)
    origin = load_img(uploaded_image, target_size=(224, 224))

    image_to_pred = img_to_array(origin)
```

```
    image_to_pred = np.expand_dims(image_to_pred, axis=0)

    if st.button("Detect Tumor"):
        # Make predictions using your model
        result = np.argmax(loaded_model.predict(image_to_pred /
255.0), axis=1)
        st.info(f"Predicted class: **{index[result[0]]}**")

        cam, img = smoothgrad(origin, loaded_model)
        # Overlay the heatmap on the image
        fig, ax = plt.subplots(figsize=(8, 6))

        # Overlay the heatmap on the image
        heatmap = np.uint8(cm.jet(cam)[..., :3] * 255)
        ax.imshow(img)
        ax.imshow(heatmap[0], cmap='jet', alpha=0.5)
        ax.axis('off')

        st.pyplot(fig)
```

Додаток Б (train.py)

```
import os
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf

from keras.preprocessing.image import ImageDataGenerator
from keras.utils import load_img, img_to_array

# libraries required to build the model
from keras.models import Sequential
from keras.layers import Conv2D, MaxPool2D, Flatten, Dense,
Dropout

from keras.activations import relu, softmax

# optimizer library
from keras.optimizers import Adam

from numpy.random import seed

seed(1)

tf.random.set_seed(2)

train_data_gen = ImageDataGenerator(rescale=1. / 255,
                                    shear_range=0.2,
                                    rotation_range=2,
                                    zoom_range=0.2,
                                    horizontal_flip=True,
                                    vertical_flip=True)

training_set
=
train_data_gen.flow_from_directory(directory="./dataset/Training",

target_size=(224, 224),

class_mode='categorical',
                                                    batch_size=32)

validation_data_gen = ImageDataGenerator(rescale=1. / 255)
# importing our validation set images with the same image size and
batch size
validation_set
=
validation_data_gen.flow_from_directory(directory="./dataset/Testi
ng",

target_size=(224, 224),

class_mode='categorical',
```

```
batch_size=32)
# next function iterates over the training set and separates the
images and their labels
imgs, labels = next(training_set)


# we define a function that prints the first 10 images from the
training set
def plotImages(images_arr):
    fig, axes = plt.subplots(1, 10, figsize=(20, 20))
    axes = axes.flatten()
    for img, ax in zip(images_arr, axes):
        ax.imshow(img)
        ax.axis('off')
    plt.tight_layout()
    plt.show()


# calling our function to print the first 10 images
plotImages(imgs)
# printing all the labels from the first batch of 32 images
print(labels)

# A sequential model is a model which consists of a sequence of
layers.
model = Sequential()
model.add(Conv2D(filters=32,
                 kernel_size=3,
                 activation=relu,
                 input_shape=[224, 224, 3]))
# Adding a secong convolutional layer
model.add(Conv2D(filters=32,
                 kernel_size=3,
                 activation=relu))
model.add(MaxPool2D(pool_size=2,
                    strides=2,
                    padding='valid'))
# adding another Conv2D layer
model.add(Conv2D(filters=32,
                 kernel_size=3,
                 activation=relu))
# adding a fourth Conv2D layer
model.add(Conv2D(filters=64,
                 kernel_size=3,
                 activation=relu))
# adding a second MaxPool2D layer
model.add(MaxPool2D(pool_size=2,
                    strides=2,
                    padding='valid'))
model.add(Flatten())
model.add(Dense(units=32,
```

```
                activation=relu,
                use_bias=True
                ))
model.add(Dropout(0.4))
# adding another fully connected layer
model.add(Dense(units=16,
                activation=relu,
                use_bias=True
                ))
model.add(Dense(units=4,
                activation=softmax))
model.summary()

model.compile(optimizer=Adam(),    loss='categorical_crossentropy',
metrics=['accuracy'])

model_history              =              model.fit(x=training_set,
validation_data=validation_set, epochs=60, verbose=1)

print(model_history.history.keys())

# comparing the training and testing accuracy
plt.plot(model_history.history['accuracy'])
plt.plot(model_history.history['val_accuracy'])
plt.title('Accuracy of the model')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend(['train', 'test'], loc='upper left')
plt.show()

# comparing training and testing loss
plt.plot(model_history.history['loss'])
plt.plot(model_history.history['val_loss'])
plt.title('Loss of the model')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend(['train', 'test'], loc='upper left')
plt.show()

index = ['glioma', 'meningioma', 'normal', 'adenoma']

test_image1      =      load_img('./dataset/Testing/meningioma/Te-
me_0018.jpg', target_size=(224, 224))
test_image1 = img_to_array(test_image1)
test_image1 = np.expand_dims(test_image1, axis=0)
result1 = np.argmax(model.predict(test_image1 / 255.0), axis=1)
print(index[result1[0]])

test_image2      =      load_img('./dataset/Testing/pituitary/Te-
pi_0018.jpg', target_size=(224, 224))
test_image2 = img_to_array(test_image2)
test_image2 = np.expand_dims(test_image2, axis=0)
```

```
result2 = np.argmax(model.predict(test_image2 / 255.0), axis=1)
print(index[result2[0]])

model.save('weights.h5')
```