НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»
(КПІ ім. Ігоря Сікорського)

ФАКУЛЬТЕТ БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ
кафедра БІОМЕДИЧНОЇ КІБЕРНЕТИКИ

«**До захисту допущено**»
В.о. завідувач кафедри БМК

_____ **Євген НАСТЕНКО**
(підпис) (ім'я, прізвище)

"___"_____2023 р.

Дипломна робота
на здобуття ступеня бакалавра
за освітньо-професійною програмою
«Комп'ютерні технології в біології та медицині»
спеціальності *122 «Комп'ютерні науки»*

на тему: **Система пошуку фрактур кісток за зображеннями комп'ютерної томографії**

Виконав (-ла): студент (-ка) **IV** курсу, групи _БС-93_

БРЮХОВЕЦЬКИЙ МИХАЙЛО ВІКТОРОВИЧ (підпис)
(прізвище, ім'я, по батькові)

Керівник: *ас. каф. БМК Матвійчук Олександр Вадимович* (підпис)
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові)

Рецензент: *доцент каф. біомедичної інженерії, доц., к.ф-мат.н. Соломін Андрій Вячеславовович* (підпис)
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____ _
(підпис)

Київ – 2023 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет біомедичної інженерії
Кафедра біомедичної кібернетики

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 122 «Комп'ютерні науки»

Освітньо-професійна програма «Комп'ютерні технології в біології та медицині»

ЗАТВЕРДЖУЮ
В.о. завідувач кафедри БМК

_____ **Євген НАСТЕНКО**
(підпис) (ім'я ПРІЗВИЩЕ)

«*30» травня* 2023 р.

ЗАВДАННЯ
на дипломну роботу студенту

БРЮХОВЕЦЬКОМУ МИХАЙЛУ ВІКТОРОВИЧУ

(прізвище, ім'я, по батькові)

1. Тема роботи **Система пошуку фрактур кісток за зображеннями комп'ютерної томографії**

Керівник роботи

Матвійчук Олександр Вадимович

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від *«31» травня* 2023 р. **№*2106-с***

2. Термін подання студентом роботи ***11 червня 2023 року***
3. Вихідні дані до роботи: *аналіз існуючих моделей згорткових нейронних мереж для задач класифікації, аналіз даних, представлення додатку для пошуку фрактур кісток за зображеннями комп'ютерної томографії.*
4. Зміст роботи: *анотації (на двох мовах), вступ, огляд літературних джерел, теоретична частина, практична реалізація задачі, загальні висновки, список використаних джерел.*
5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо): *10 слайдів, 31 рисунок, 6 таблиць*

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання ***30 травня 2023 року***

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роб оти	Примітка
1	Отримати завдання за темою ДР на практику	До 15.02.2023р.	*виконано*
2	Переддипломна практика	За графіком	*виконано*
3	Виконання розділів ДР (Вступ, аналітичний огляд літературних джерел, теоретична частина)	До кінця практики	*виконано*
4	Виконання розділів ДР (практична частина, загальні висновки, список джерел)	Не пізніше 1 тижня до засідання каф-ри	*виконано*
5	Перевірка ДР науковим керівником	Не пізніше 1 тижня до засідання каф-ри	*виконано*
6	Подання в електронному вигляді ДР та анотації до неї на перевірку нормоконтролера та плагіат (UNICHECK) .	---- « ------	*виконано*
7	Надання документів на засідання кафедри	За день до засідання	*виконано*
8	Предзахист ДР та допуск до захисту дисертації	Згідно плану каф.	*виконано*
9	Подання ДР рецензенту. Отримання рецензії.	До подання пакету документів до ЕК	*виконано*
10	Подання пакету документів по ДР та супровідних до неї документів до захисту в ЕК[1]	За 5 днів до дати захисту ДР за графіком	*виконано*
11	Захист ДР в ЕК		

Студент _____ **Михайло БРЮХОВЕЦЬКИЙ**
(підпис) (ім'я, ПРІЗВИЩЕ)

Керівник ДР _____ **Олександр МАТВІЙЧУК**
(підпис) (ім'я, ПРІЗВИЩЕ)

Нормоконтролер _____ **Галина КОРНІЄНКО**
(підпис) (ім'я, ПРІЗВИЩЕ)

[1] не пізніше ніж за 5 днів до затвердженої дати захисту ДР в ЕК

АНОТАЦІЯ

Дипломна робота за темою «Система пошуку фрактур кісток за зображеннями комп’ютерної томографії» виконана студентом кафедри біомедичної кібернетики ФБМІ *БРЮХОВЕЦЬКИМ МИХАЙЛОМ ВІКТОРОВИЧЕМ зі спеціальності 122 «Комп’ютерні науки» за освітньо-професійною програмою «Комп’ютерні технології в біології та медицині» та* складається зі: вступу; 3 розділів (*огляд літературних джерел, теоретична частина, практична реалізація задачі)*, висновків до кожного з цих розділів; загальних висновків; списку використаних джерел, який налічує 21 джерело. Загальний обсяг роботи 68 сторінок.

Актуальність теми. Завдяки постійному розвитку технологій, комп'ютерна томографія стала потужним інструментом для отримання детальних зображень кісток. Проте, визначення фрактур на цих зображеннях може бути складним завданням для лікарів, особливо при великій кількості даних. Розробка системи пошуку фрактур кісток, яка заснована на аналізі зображень КТ за допомогою нейромереж та комп'ютерного зору, може значно покращити точність та ефективність діагностики, допомагаючи лікарям вчасно виявляти та локалізувати фрактури.

Метою є створення системи для покращення точності та швидкості виявлення фрактур кісток, що допоможе лікарям у ранньому діагностуванні та локалізації ушкоджень.

Її досягнення передбачає вирішення наступних ***завдань***:

1. Аналіз даних зображень кісток комп’ютерної томографії.
2. Розробка моделей машинного навчання для класифікації зображень кісток комп’ютерної томографії.
3. Створення кросплатформенного додатку для зручної взаємодії з користувачем.

Використані методи: аналіз літературних джерел; розробка архітектури застосунку; вибір технологій та інструментів; реалізація функціональності.

Отримані результати: інтегрована модель класифікації визначення фрактур кісток за зображеннями комп'ютерної томографії у кросплатформений додаток.

***Ключові слова*:** згорткові нейронні мережі, кросплатформеність, фрактури кісток.

Бібліографічний опис ДР:

Брюховецький М.В. Система пошуку фрактур кісток за зображеннями комп'ютерної томографії: дипломна роб. бакалавра : 122 Комп'ютерні науки / Брюховецький Михайло Вікторович. – Київ, 2023. – 67 ст

ABSTRACT

The diploma thesis on "Bone Fracture Detection System using Computed Tomography Images" is conducted by *MYKHAILO BRIUKHOVETSKYI, a student of the Department of Biomedical Cybernetics at the Faculty of Computer Science and Cybernetics. The thesis is completed under the educational-professional program "Computer Technologies in Biology and Medicine" with specialization 122 "Computer Science."* The thesis consists of an introduction, three chapters (literature review, theoretical part, practical implementation of the task), conclusions for each of these chapters, general conclusions, a list of references comprising 21 sources. The total volume of the work is 68 pages.

Relevance of the topic. With the constant advancement of technology, computed tomography has become a powerful tool for obtaining detailed bone images.

However, identifying fractures in these images can be a challenging task for medical professionals, particularly when dealing with a large volume of data. The development of a bone fracture detection system based on the analysis of CT images using neural networks and computer vision techniques can significantly improve the accuracy and efficiency of diagnosis, assisting doctors in timely identifying and localizing fractures.

The aim of this thesis is to create a system that enhances the accuracy and speed of bone fracture detection, aiding doctors in early diagnosis and localization of injuries.

To achieve this aim, the following **objectives are addressed**:

1. Analysis of computed tomography bone image data.
2. Development of machine learning models for the classification of computed tomography bone images.
3. Creation of a cross-platform application for user-friendly interaction.

Methods used include literature review, application architecture design, technology and tool selection, and functional implementation.

The obtained results include an integrated classification model for bone fracture detection using computed tomography images, implemented in a cross-platform application.

Keywords: convolutional neural networks, cross-platform compatibility, bone fractures.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

КТ – Комп’ютерна томографія

ML - Машинне навчання

ПЗ - Програмне Забезпечення

GUI - Графічний інтерфейс користувача

CNN - Згорткова нейронна мережа

RGB - Червоний-Зелений-Синій

SGD - Стохастичний градієнтний спуск

Adam - Адаптивний метод оцінювання моменту

MSE - Середня квадратична помилка

MLP - Багатошаровий перцептрон

RNN - Рекурентна нейронна мережа

ReLU - Прямокутна лінійна функція активації

SDK - Набір розробки програмного забезпечення

ПК - Персональний комп'ютер

API - Інтерфейс програмування додатків

MVP - Мінімально життєздатний продукт

ВСТУП

Система пошуку фрактур кісток за зображеннями КТ є актуальною та важливою задачею в сучасній медицині. Захворювання та травми кісток є поширеними проблемами, які потребують точної та швидкої діагностики для ефективного лікування. КТ-сканування надає детальне зображення кісткової тканини і є одним з найефективніших методів виявлення переломів кісток. Однак, ручний аналіз КТ-зображень дуже складний і вимагає значних зусиль та професійних знань. Тому необхідно розробити автоматизовану систему, яка може швидко та точно виявляти фрактури на КТ-зображеннях кісток. У цій роботі обґрунтовується необхідність розробки такої системи і пропонується метод, який поєднує стек нейронних мереж для досягнення високої точності та швидкості виявлення фрактур кісток на КТ-зображеннях.

Об'єктом нашого дослідження є набір даних MURA, який містить КТ-зображення різних частин тіла з наявністю або відсутністю патологічних змін кісткової тканини. Цей набір даних широко використовується в академічних дослідженнях і містить достатню кількість зображень для навчання та тестування моделей машинного навчання.

Метою нашої роботи є розробка та впровадження системи для виявлення фрактур кісток на КТ-зображеннях. Основними завданнями є створення нейронних мереж для класифікації частин тіла та виявлення деформацій кісток на КТ-зображеннях, а також розробка додатка для взаємодії з користувачем.

Результати роботи можуть бути широко застосовані в медичній практиці. Ця система допоможе лікарям та медичному персоналу швидко та ефективно виявляти переломи на КТ-зображеннях кісток, сприяючи ранній діагностиці та лікуванню пацієнтів. Крім того, розроблений додаток може використовуватись у навчальних закладах для підвищення навичок студентів та молодих фахівців.

У майбутньому можна розширити та покращити систему шляхом використання більших та різноманітних наборів даних, вдосконалення алгоритмів нейронних мереж та врахування інших факторів, які можуть впливати на діагностику переломів. Використання цієї системи в реальних клінічних умовах може покращити процес діагностики та збільшити можливості швидкого лікування пацієнтів з ушкодженнями кісток.

Однією з особливостей останніх років є швидкий розвиток інформаційних технологій, зокрема штучного інтелекту та машинного навчання. Ці технології відкривають нові можливості для багатьох галузей, включаючи медицину. Використання нейронних мереж та комп'ютерного зору в медичних дослідженнях та практиці є активним напрямком, який може значно сприяти поліпшенню процесу діагностики та лікування.

У контексті виявлення фрактур кісток на КТ-зображеннях, існуючі методи, що базуються на ручному аналізі зображень або традиційних алгоритмах, можуть бути обмежені за ефективністю та точністю. Введення нейронних мереж та класифікаторів у цей процес може принести значні переваги, такі як автоматизація та об'єктивна оцінка КТ-зображень, а також прискорення процесу діагностики.

Окрім того, персоналізована медицина стає все більш важливою, оскільки прийняття рішень щодо лікування базується на унікальних характеристиках кожного пацієнта. Використання нейронних мереж у системі відновлення після перелому може сприяти індивідуалізованому підходу до кожного пацієнта та забезпечити більш точну та персоналізовану діагностику.

Отже, дана робота є дуже важливою, оскільки спрямована на розробку системи, яка поєднує сучасні методи машинного навчання і аналіз медичних КТ-зображення кісток. Результати роботи можуть покращити якість медичних послуг, допомогти лікарям ефективно виявляти переломи кісток та покращувати загальний стан пацієнтів

РОЗДІЛ 1
ОГЛЯД ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1.1 Фрактури кісток на зображеннях КТ

1.1.1 Поняття фрактур кісток

Поняття фрактур кісток охоплює важливий аспект медицини та медичної діагностики, пов'язаний з ушкодженням цілісності кісткової структури. Перелом кісток виникає через перевищення механічної сили її меж терпимості і може бути результатом травми, нещасних випадків або захворювань, таких як остеопороз.

Переломи є серйозними медичними ускладненнями, оскільки вони можуть призводити до функціональних обмежень кінцівок, болю, обмеженої рухомості та негативно впливати на якість життя пацієнта. Вірна діагностика перелому має вирішальне значення для правильного лікування та реабілітації пацієнта.

Одна з основних цілей цього дослідження полягає у правильній класифікації та описі типів переломів. Класифікація дозволяє систематизувати переломи за різними ознаками. Це дозволяє лікарям приймати відповідні рішення щодо методів лікування і надавати інформацію про прогноз відновлення пацієнта.

Для точної діагностики кісткових фрагментів застосовуються різні методи, такі як комп'ютерна томографія, рентгенографія, магнітно-резонансна томографія та ультразвук. Ці методи дозволяють отримувати зображення ушкоджених кісток і оцінювати їх стан для визначення найкращих методів лікування. Розуміння концепції фрактур кісток є важливим для ортопедичних хірургів, травматологів та радіологів, які займаються діагностикою та лікуванням кісткових ушкоджень. Дослідження кісткової текстури сприяє поліпшенню методів діагностики та розробці нових технологій для більш ефективного лікування та реабілітації пацієнтів з такими ускладненнями.

Дослідження фрактур кісток дозволяє визначити фактори, що спричиняють ушкодження, такі як зовнішня сила, напрямок удару, тип кістки та індивідуальні особливості пацієнта. Це допомагає розробляти стратегії запобігання травмам та підвищувати безпеку в різних галузях, таких як спорт, автотранспорт та промисловість.

Поняття перелому також має практичне значення для лікування. В залежності від типу та характеру перелому застосовуються різні методи лікування, такі як консервативне лікування (гіпсова пов'язка, фіксація), хірургічне лікування (остеосинтез, артродез) та фізична терапія. Розуміння особливостей та характеристик перелому допомагає лікарям вибрати найефективніший метод лікування та передбачити його результати. Слід зазначити, що поняття фрактури кісток охоплює не тільки їх діагностику та лікування, але й вивчення процесів зростання та згинання переломів.

Здатність кістки зрощуватися та згинатися залежить від багатьох факторів, включаючи тип перелому, стан кісткової тканини, вік пацієнта та загальний стан здоров'я. Дослідження цих процесів сприяє поліпшенню методів лікування та передбаченню процесу відновлення для пацієнтів з кістковими ушкодженнями.

Загалом, концепція перелому охоплює вивчення механізмів ушкоджень, діагностики, лікування та процесів згинання кісток. Це важлива галузь медичної науки, яка сприяє поліпшенню діагностики, лікування та профілактики кісткових ушкоджень, а також підвищенню безпеки та якості життя пацієнтів [20].

1.1.2 ML методи для виявлення фрактур кісток на зображеннях КТ

Глибинне навчання стало ефективним інструментом для виявлення кісткових фрактур на зображеннях КТ. Ці методи базуються на використанні нейромереж, які автоматично навчаються розпізнавати шаблони і текстурні ознаки на медичних КТ-зображеннях.

Одним із найпопулярніших методів глибинного навчання для виявлення кісткових деформацій на знімках КТ є згорткові нейромережі. Ці мережі можуть ефективно аналізувати просторову інформацію на зображенні та виявляти ознаки, які свідчать про наявність переломів. Вони можуть автоматично навчатися на великій кількості зображень з ураженнями та без них, що дозволяє їм ставати все більш точними і надійними. Даний метод для виявлення кісткових фрактур полягає у використанні спеціально створеної архітектури мережі, такої як Inception V3. Ця архітектура має особливу структуру, яка дозволяє ефективно виявляти деталі та локалізувати конкретні області на зображенні. Це особливо корисно для виявлення кісткових фрагментів та їх деталей.

Важливим кроком у використанні методів глибинного навчання для виявлення кісткових фрактур на знімках КТ є підготовка великого набору даних з розміченими зображеннями. Це дозволяє нейромережі навчитися на прикладах та ефективно визначати фрактури на нових зображеннях.

Використання методів глибинного навчання для виявлення кісткових фрактур на знімках КТ має багато переваг, таких як висока точність, швидкість та автоматизація процесу виявлення, сприяючи поліпшенню діагностики та лікування кісткових переломів [21].

1.2 Основи побудови згорткових нейронних мереж

1.2.1 Пряме поширення

Пряме поширення є одним із основних процесів у глибинному навчанні. Цей процес відіграє важливу роль у передачі вхідних даних через нейронну мережу та отриманні вихідних значень.

Задачею прямого поширення є обчислення значень у кожному нейроні мережі від вхідного шару до вихідного. Цей процес починається з передачі вхідних даних у перший шар мережі. На кожному шарі сума зваженого входу обчислюється за допомогою вагів та зсувів (див. рис.1.1).

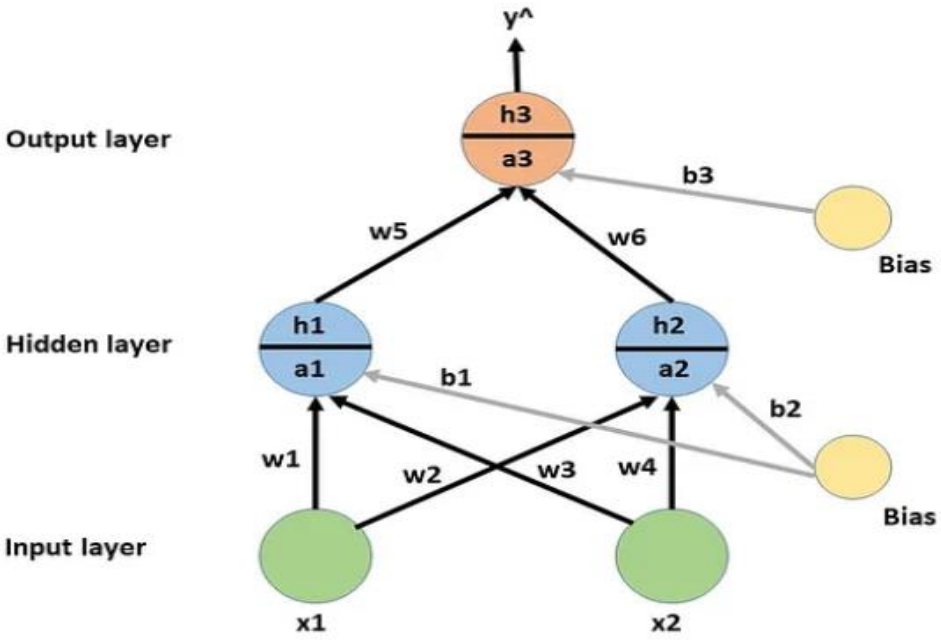


Рисунок 1.1 - Пряме поширення [1]

Кожен нейрон у прихованому шарі або вихідному шарі виконує обчислення на двох етапах: передактивації та активації [1]. Передактивація - це зважена сума вагованих значень вхідних даних. На цьому етапі нейрон вирішує, чи передати цю інформацію наступному шару, залежно від суми зваженого входу та заданої функції активації. Після передактивації, зважена сума обчислених значень вхідних даних проходить через функцію активації. Функція активації - це математична функція, яка надає нелінійність мережі.

Вона дозволяє нейронній мережі виразити складні та нелінійні залежності між вхідними та вихідними значеннями. У глибинному навчанні з використанням багатошарових нейронних мереж, функція активації відіграє важливу роль у наданні виразності моделі.

Процес прямого поширення продовжується від першого до останнього шару мережі. На вихідному шарі отримується вихідне значення, що представляє обчислені результати моделі для конкретного вхідного прикладу.

Після прямого поширення можна використовувати зворотнє поширення для допомоги в налаштуванні параметрів мережі для досягнення кращої точності та універсальності.

1.2.2 Зворотнє поширення

Техніка зворотного поширення є основною технікою навчання нейромережі. Вона включає в себе налаштування ваг нейромережі на основі обчислених помилок з попередніх ітерацій. Правильне налаштування цих ваг може знизити рівень неточностей, підвищити надійність та універсальність моделі.

Зворотне поширення обчислює градієнт функції втрат відносно всіх ваг мережі. По суті, це обчислення виконується шар за шаром, а не безпосередньо для всієї мережі. Алгоритм узагальнює обчислення, що використовуються в правилі дельта (див. рис.1.2).

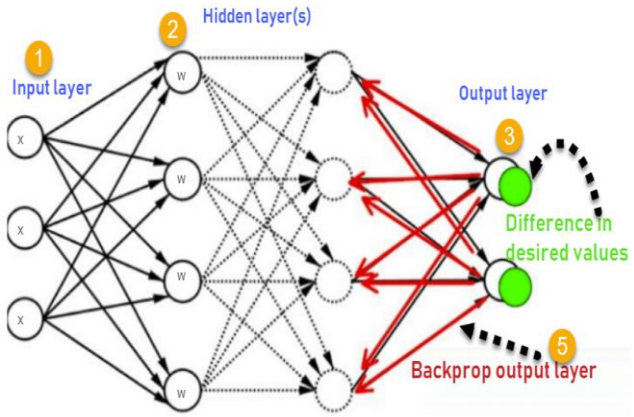


Рисунок 1.2 - Зворотне поширення [2]

Завдяки зворотньому поширенню обчислюється похідна для кожного зваженого коефіцієнта в нейромережі. Похідна вказує на швидкість зміни функції відносно цього зваженого коефіцієнта.

Процес обчислення похідної для зворотного поширення базується на ланцюжковому правилі. Спочатку обчислюється похідна початкової помилки мережі. Потім, враховуючи ваги нейрона, функцію активації та вхідні

значення, ця помилка передається назад через шари мережі. Цей процес продовжується до першого шару, де обчислюються похідні для кожного зваженого коефіцієнта.

Обчислені похідні використовуються для оновлення ваг мережі з метою зменшення помилок та покращення її передбачувальних здібностей. Похідна надає нам інформацію про те, як змінювати ваги з метою оптимізації функції втрат нейронної мережі.

Отже, обчислення похідних зворотного поширення дозволяють нейромережі "навчитися" та адаптуватися до вхідних даних, налаштовуючи ваги для отримання кращої точності.

Переваги зворотного поширення включають швидкість, простоту та легкість використання. Воно потребує мінімальної конфігурації і є гнучким методом, який не вимагає апріорних знань про мережу. Це широко використовувана та зазвичай ефективна методика тренування.

Існують два типи зворотного поширення мереж: статичне зворотне поширення та рекурентне зворотне поширення.

Статичне зворотне поширення є найпоширенішим і фундаментальним типом зворотного поширення. У цьому методі інформація рухається вперед через мережу від вхідного шару до вихідного шару, а потім обчислюється помилка та поширюється у зворотному напрямку для оновлення ваг. Цей підхід навчання дозволяє нейромережі розпізнавати складні залежності між вхідними та вихідними даними.

Рекурентне зворотне поширення використовується у рекурентних нейромережах, які мають зворотні зв'язки між вузлами. Це означає, що вихід передається вперед та повертається назад до попередніх часових кроків. Такий підхід дозволяє моделі зберігати інформацію про попередні стани та використовувати її для прийняття рішень на поточному кроці. Рекурентні нейромережі добре підходять для обробки послідовних даних, таких як текст, мова або часові ряди [2].

Обидва типи зворотного поширення мають свої переваги та обмеження і застосовуються залежно від конкретних завдань та вимог. Статичне зворотне поширення широко використовується для багатьох лінійних ML задач, тоді як рекурентне зворотне поширення є ефективним для моделювання послідовних процесів та залежностей [2].

1.2.3 Згорткові нейронні мережі

Згорткові нейронні мережі є потужним інструментом для обробки зображень. Вони використовуються для автоматичного виявлення та розпізнавання шаблонів, що робить їх особливо корисними для завдань комп'ютерного зору.

Основна ідея, що стоїть за згортковими нейронними мережами, полягає в тому, щоб використовувати згортку для перетворення вхідного зображення. Згортка складається з невеликого фільтра, який переміщується по зображенню та виконує операцію згортки. Після згортки отримується нове зображення, яке відображає виявлені фільтром особливості. Цей процес може повторюватись кілька разів для виявлення все більш складних особливостей.

Згорткові нейронні мережі є особливо ефективними при обробці зображень з двох основних причин. По-перше, вони можуть автоматично виявляти локальні особливості в зображенні, такі як краї, текстури або форми, за допомогою згорткових шарів. По-друге, згорткові нейронні мережі використовують розподілені параметри і загальні ваги, що дозволяє ефективно використовувати навіть обмежену кількість параметрів для великих зображень. Завдяки цим особливостям згорткові нейронні мережі стали стандартним інструментом для багатьох завдань обробки зображень, включаючи виявлення об'єктів, класифікацію зображень, виявлення облич і тд (див. рис.1.3). Вони дозволяють досягати високої точності і ефективно обробляти великий обсяг даних [3].

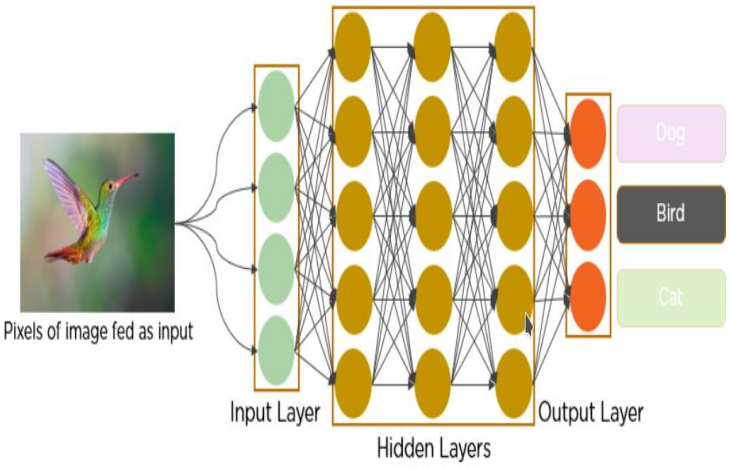


Рисунок 1.3 - Принцип роботи згорткової нейронної мережі для вирішення задачі класифікації [3]

Згорткове ядро (також відоме як фільтр або маска) є одним з основних компонентів згорнутої нейронної мережі. Воно представляє собою невеликий числовий масив, який використовується для виконання операції згортки над вхідним зображенням.

Згорткове ядро переміщується по зображенню (див. рис.1.4), виконуючи операцію згортки на кожному кроці. Водночас ядро охоплює малу область зображення, і його значення множаться на значення відповідних пікселів у цій області. Ця операція називається поелементним множенням.

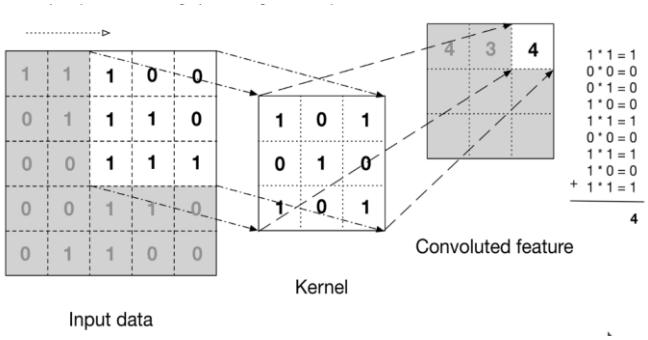


Рисунок 1.4 - Принцип роботи згорткового ядра [3]

Після виконання операції згортки над певною областю, згорткове ядро здійснює переміщення на певний крок (згортковий крок або шаг), щоб

повторити операцію в наступній області. Таким чином, отримується нове зображення, в якому кожний піксель є результатом операції згортки над відповідною областю зображення.

Наприклад, якщо розмір згорткового ядра становить 3x3, то на кожному кроці обчислюється сума елементів, отриманих шляхом поелементного множення ядра на пікселі по одному. Отримана сума стає значенням відповідного пікселя у вихідному зображенні.

Використовуючи різні згорткові ядра, можна виявляти різні особливості в зображенні, такі як особливості обличчя, границі об'єктів або текстурні ознаки. Згортковий крок визначає швидкість переміщення згорткового ядра по зображенню і впливає на роздільну здатність та кількість інформації, яка зберігається у вихідному зображенні.

Обробка кольорових зображень з використанням згорткової нейронної мережі використовує RGB канали для представлення інформації про кольори.

Кожне кольорове зображення можна розглядати як 3D тензор, де кожен піксель має значення червоного, зеленого і синього каналів. Ці значення вимірюють інтенсивність кожного кольору в конкретній точці зображення. Наприклад, для пікселя (x, y) його RGB значення буде (R(x, y), G(x, y), B(x, y)).

У згортковій мережі згорткове ядро також є тривимірним, оскільки воно враховує всі три RGB канали. Кожне згорткове ядро виконує згортку окремо з кожним каналом. Наприклад, якщо у нас є ядро розміром 3x3x3, воно буде виконувати згортку з трьома окремими каналами. Результати згортки кожного каналу об'єднуються, щоб отримати кінцеве вихідне зображення.

Використання RGB каналів дозволяє згортковій мережі моделювати кольорові особливості та залежності у зображенні. Кожен канал має свої власні ваги, які вивчаються під час тренування мережі. Тому мережа може виявляти взаємодію різних особливостей та кольорів для виконання завдань, таких як класифікація або сегментація зображень [4, 5].

1.2.4 Аугментації

Аугментації даних є важливим методом у глибокому навчанні, особливо для згорткових нейронних мереж. Воно використовується для розширення розміру тренувальних даних шляхом створення нових зображень за допомогою застосування перетворень до наявних зображень (див. табл.1.1).

Найпопулярніші методи збільшення даних включають:

- Поворот: обертання зображення на певний кут. Це допомагає моделі розпізнавати об'єкти з різних кутів.
- Зсув: вертикальний і горизонтальний зсув зображення. Це дозволяє моделі бути менш чутливою до положення об'єктів на зображенні.
- Масштабування: зміна масштабу зображення, щоб модель могла розпізнавати об'єкти різних розмірів.
- Відображення: горизонтальне і вертикальне відображення зображення. Це допомагає моделі відрізняти об'єкти з симетричною структурою.
- Шум: додавання певного рівня шуму до зображення. Це допомагає моделі бути менш чутливою до шуму в реальних зображеннях.
- Яскравість, контраст та насиченість: зміна значень яскравості, контрастності та насиченості зображення, щоб допомогти моделі розпізнавати об'єкти в різних умовах освітлення.
- Розмивання: застосування фільтрів розмивання до зображення для зменшення деталей і шуму. Це допомагає моделі фокусуватись на основних ознаках об'єктів і знижує вплив шуму на процес розпізнавання.

Таблиця 1.1

Переваги та недоліки різних методів аугментації даних

Метод аугментації	**Переваги**	**Недоліки**
Поворот	Дозволяє моделі розпізнавати об'єкти з різних кутів	Може призвести до спотворення деяких об'єктів

Продовж. табл. 1.1

Метод аугментації	**Переваги**	**Недоліки**
Зміщення	Зменшує чутливість моделі до положення об'єктів	Може призвести до втрати деякої інформації
Збільшення / зменшення масштабу	Дозволяє моделі розпізнавати об'єкти різного розміру	Може призвести до втрати деталей зображення
Дзеркальне відображення	Допомагає моделі розрізняти симетричні об'єкти	Не підходить для об'єктів зі специфічною асиметрією
Шум	Робить модель менш чутливою до шуму в реальних зображеннях	Може викликати додатковий шум, який заважає розпізнаванню
Яскравість, контрастність, насиченість	Допомагає моделі пристосуватись до різних освітлювальних умов	Може викликати зміну візуальних властивостей зображення

Конкретний вибір методів збільшення даних залежить від характеристик тренувального набору даних та завдання [6].

1.2.5 Оптимізаційні методи

У машинному навчанні існує багато методів оптимізації, які можуть бути використані для ефективного навчання моделей. Оптимізація в машинному навчанні має на меті знайти набір параметрів моделі, що мінімізує функцію втрат або максимізує функцію користі.

Основні методи оптимізації включають:

- Градієнтний спуск
- SGD
- Adam
- Метод Ньютона
- Метод регуляризації

Вибір конкретного методу оптимізації залежить від характеристик поставленої задачі та моделі. Наприклад, SGD часто використовується для навчання великих нейронних мереж, оскільки він має високу обчислювальну ефективність. Проте методи, такі як Adam, можуть забезпечити швидше збіжність за відповідними налаштуваннями. Крім того, якщо модель схильна до перенавчання, може бути використана регуляризація. Важливо спробувати різні методи та параметри, щоб знайти найкраще рішення для вашої конкретної проблеми.

У цій роботі було обрано оптимізатор Adam для навчання нейронної мережі. Adam є одним з найпопулярніших методів оптимізації у глибокому навчанні. Він поєднує підхід градієнтного спуску та RMSprop, додавши адаптивність та динаміку.

Основні переваги методу Adam включають:

- Адаптивність швидкості навчання: Adam автоматично адаптує швидкість навчання для кожного параметра на основі історії градієнту. Він використовує експоненційне згладжування градієнта, що забезпечує більш стабільне та оптимальне навчання.
- Момент: Adam використовує момент для зберігання апріорної інформації про градієнт та його розподіл. Це допомагає знизити шум і прискорити збіжність, особливо на кривих поверхнях функції втрат.
- Висока ефективність на різних завданнях: Adam показує високу ефективність на широкому спектрі завдань машинного навчання. Він підходить для навчання глибоких нейронних мереж з великою кількістю параметрів, а також для обробки різних типів даних та архітектур мереж.
- Легкість використання: Adam дуже простий у використанні та налаштуванні. Більшість бібліотек машинного навчання підтримують Adam, що робить його зручним для використання в практичних проектах.

Незважаючи на ці переваги, у методу Adam також є обмеження та недоліки. Він може бути чутливий до великої кількості шуму та зміщення в даних, що може призводити до некоректного оновлення параметрів. Також потрібно налаштувати деякі гіперпараметри, такі як швидкість навчання та експоненційне згладжування.

У кінцевому рахунку, Adam є потужним та ефективним методом оптимізації, який забезпечує швидше та стабільніше навчання глибоких нейронних мереж. Однак його ефективність може залежати від конкретного завдання та параметрів, тому варто експериментувати та налаштовувати його параметри для досягнення найкращих результатів [7, 8].

1.2.6 Функція втрат

У машинному навчанні функція втрат є математичною функцією, яка вимірює відстань між прогнозованими значеннями моделі та правильними значеннями цільової змінної. Її мета - кількісно оцінити виконання моделі у визначеній задачі.

Функції втрат є ключовим елементом процесу навчання моделі, оскільки вони використовуються для оцінки виконання моделі на навчальних прикладах та визначення напрямку оновлення параметрів моделі під час оптимізації.Важливо підібрати відповідну функцію втрату для конкретної задачі, оскільки це може вплинути на якість та швидкість збіжності моделі під час навчання. Крім того, деякі функції втрат мають властивості, які допомагають вирішувати певні проблеми, наприклад, регуляризацію для уникнення перенавчання моделі (див. табл.1.2).

Таблиця 1.2

Класифікація функцій втрат

Вид функції втрат	**Призначення**	**Приклади задач**	**Приклади архітектур моделей**
MSE	Вимірює середнє	Регресія,	Лінійна регресія,

	квадратичне відхилення між прогнозом та правильним значенням	прогнозування числових значень	MLP
Категоріальна перехресна ентропія	Використовується для задач класифікації з кількома категоріями	Багато Класова класифікація, нейронні мережі	CNN, RNN, Transformer
Бінарна перехресна ентропія	Використовується для бінарної класифікації	Бінарна класифікація, логістична регресія	Logistic Regression, MLP
Перехресна ентропія з вагами	Застосовується до задач з незбалансованими класами, де ваги різних класів різні	Нерівноважна класифікація	CNN, RNN, MLP
Індекс Дайса	Використовується для семантичної сегментації зображень	Сегментація зображень	U-Net, DeepLab, Mask R-CNN

Продовж. табл. 1.2

Вид функції втрат	**Призначення**	**Приклади задач**	**Приклади архітектур моделей**

Huber Loss	Комбінує характеристики середньоквадратичної помилки та середньої абсолютної помилки	Регресія з урахуванням викидів	MLP, Decision Trees

Ці моделі використовують функцію втрат для тренування, яка оцінює ефективність алгоритму при побудові моделі на вхідних даних. Якщо прогноз відхиляється від фактичного результату, значення функції втрат буде великим. За допомогою технік оптимізації і їх впливу на функцію втрат моделі поступово навчаються і зменшують помилки прогнозу.

Не існує універсальної функції втрат, яка підходить для всіх алгоритмів машинного навчання. Вибір функції втрат залежить від кількох факторів, таких як тип вибраного алгоритму, ефективність обчислення похідних та відсоток викидів у наборі даних.

Загалом, функції втрат можна розділити на дві великі категорії в залежності від типу навчального завдання - функції втрат для регресії та функції втрат для класифікації. У класифікації ми намагаємося прогнозувати вихідні значення зі скінченного набору класів. Наприклад, за допомогою великої кількості зображень рукописних цифр, ми класифікуємо їх як одну з цифр від 0 до 9. З іншого боку, регресія використовується для прогнозування неперервних значень, наприклад, ми передбачаємо ціну кімнати на основі даних про площу кімнати, кількість кімнат і т.д [9].

Крім того, існують такі речі, як метрики. Метрики та функції втрат використовуються для оцінки різних аспектів моделі під час навчання машинного навчання.

Функції втрат є цільовою функцією, яку модель намагається мінімізувати під час оптимізації. Вони вимірюють рівень помилки між

прогнозованими значеннями моделі та фактичними даними. Найпоширенішими функціями втрат є MSE для задач регресії та кросс-ентропія для задач класифікації.

З іншого боку, метрики використовуються для оцінки продуктивності моделі після тренування. Вони не є частиною процесу оптимізації і можуть відрізнятись від функцій втрат. Метрики надають додаткову інформацію про продуктивність моделі, таку як точність, повнота, класифікаційна точність тощо. Популярні метрики включають точність, відновлення, F1-показник та ROC-AUC [10].

Незважаючи на те, що функції втрат та метрики можуть використовувати подібні показники помилок, їх підходи та реалізація відрізняються. Функції втрат використовуються для оптимізації моделі та керування тренуванням, тоді як метрики використовуються для оцінки кінцевої продуктивності моделі. Оскільки функції втрат мають на меті мінімізувати помилку, вони можуть бути більш чутливими до великих відхилень, тоді як метрики зазвичай намагаються надати загальну оцінку продуктивності моделі, враховуючи багато аспектів.

Загалом, функції втрат та метрики є важливою складовою оцінки та оптимізації моделей машинного навчання. Комбінування цих двох підходів може призвести до досягнення кращих результатів у машинному навчанні.

1.2.7 Активаційні функції

Активаційні функції є важливими елементами у нейромережах. Вони надають моделі нелінійність та здатність моделювати складні залежності в даних.

Існує багато різних активаційних функцій, але основні з них включають:

- Сигмоїдна функція: ця функція масштабує вхідний сигнал до діапазону від 0 до 1. Вона часто використовується в бінарних класифікаційних задачах або там, де потрібно отримати ймовірність.

- Функція ReLU : ця функція повертає 0 для від'ємних значень вхідного сигналу і залишає позитивні значення без змін. Вона є однією з найпопулярніших активаційних функцій і використовується в багатьох глибоких нейромережах.
- Функція Tanh: ця функція схожа на сигмоїдну функцію, але її вихідний діапазон знаходиться від -1 до 1. Вона також використовується в задачах класифікації і регресії.
- Функція Softmax: ця функція використовується в багатокласовій класифікації. Вона приймає вхідний вектор і перетворює його на вектор ймовірностей для кожного класу.

Активаційні функції повинні мати певні властивості для ефективного використання. Основні вимоги до активаційних функцій включають нелінійність (для моделювання складних залежностей), диференційованість (для обчислення похідних) і обмежений діапазон значень.

При виборі активаційної функції для нейромережі важливо враховувати тип завдання, тип даних та вимоги моделі. Є кілька керівництв, які можуть допомогти вам вибрати відповідну активаційну функцію.

Якщо вам потрібно розділити дані на кілька класів і отримати ймовірність для кожного класу, корисно використовувати функцію Softmax. Якщо вам потрібен лише бінарний вихід, найкраще використовувати сигмоїдну або тангенс гіперболічний активаційний функції.

У завданнях регресії, де потрібно прогнозувати числові значення, рекомендується використовувати функцію ReLU або її модифікації, такі як LeakyReLU або ELU [11].

У глибоких нейромережах найпоширенішою активаційною функцією для прихованих шарів є ReLU. Вона має перевагу в швидкості навчання, але ви можете спробувати інші функції, такі як PReLU або Swish, щоб отримати кращі результати.

Для рекурентних нейромереж часто використовується активаційна функція Tanh.

Якщо ви не впевнені, яка активаційна функція найкраще підходить для вашої моделі, ви можете спробувати різні варіанти і порівняти їх результати на валідаційному наборі даних.

Також слід враховувати властивості активаційної функції, такі як диференційованість, введення нелінійності та здатність запобігати затуханню градієнта [12].

Виключення лінійності у моделі є важливим, оскільки лінійні активаційні функції не можуть моделювати складні залежності. Лінійні комбінації лінійних функцій залишаються лінійними, тому додавання нелінійності за допомогою активаційних функцій допомагає моделі бути більш гнучкою і адаптивною до складних даних.

Кожна активаційна функція має свої переваги та недоліки. Ось деякі загальні переваги та недоліки різних активаційних функцій:

- Сигмоїдна функція:
    - Переваги: Легко інтерпретується як ймовірність, може розрізняти значення.
    - Недоліки: Обмежений діапазон значень.
- ReLU функція:
    - Переваги: Швидка обчислювальна швидкість, уникнення проблеми зниклих градієнтів.
    - Недоліки: Неактивний для від'ємних значень, може призвести до "мертвих" нейронів.
- Тангенс гіперболічний функція:
    - Переваги: Діапазон значень від -1 до 1, диференційований.
    - Недоліки: Можливість виникнення проблеми втрати градієнту, повільна збіжність.
- Softmax функція:

- Переваги: Генерує ймовірність для кожного класу, може розрізняти класи.
- Недоліки: Не підходить для задач регресії, можлива перенавчаність.

1.2.8 Архітектура Inception V3

Архітектура моделі Inception V3 базується на використанні блоків Inception (див. рис.1.5), які дозволяють ефективно виявляти різноманітні функції та особливості в зображеннях. Ця модель є однією з версій серії моделей Inception, розроблених дослідницькою групою Google.

Основна ідея Inception V3 полягає у використанні фільтрів різних розмірів (1x1, 3x3, 5x5) та шарів згортки для аналізу зображення з різних масштабів, що дозволяє виявити як локальні, так і глобальні особливості. Крім того, використання згортки 1x1 допомагає зменшити кількість параметрів моделі та сприяє ефективному навчанню [13].

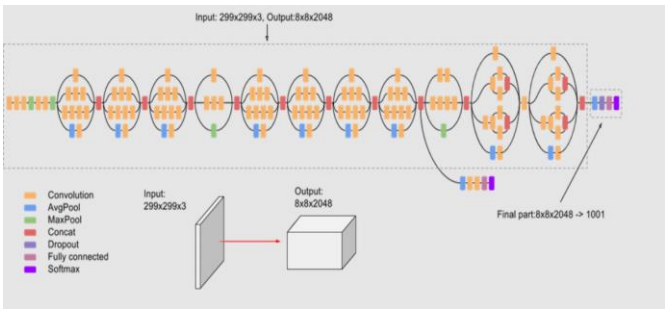


Рисунок 1.5 - Архітектура Inception V3 [13]

Inception V3 також використовує концепцію модулів зниження розмірності, що дозволяють зменшити розмір зображення перед подальшим виявленням особливостей. Це допомагає знизити обчислювальну складність моделі та покращує її продуктивність.

Крім того, Inception V3 використовує інтегровану нормалізацію для стабілізації та прискорення навчання, а також апаратне прискорення з використанням спеціалізованих архітектур, таких як GPU і TPU.

Вся архітектура Inception V3 є глибокою та широкою, що дозволяє моделі ефективно виявляти складні особливості в зображеннях і досягати високої точності класифікації. Це зробило Inception V3 популярним у відомих конкурсах з комп'ютерного зору, таких як ImageNet [13].

1.3 Використання Python для розробки кросплатформенних додатків

1.3.1 Визначення кросплатформенного додатку

Кросплатформовий додаток визначається як програма, яка може працювати на різних операційних системах та апаратних платформах без значних змін у вихідному коді або виконуваному файлі. Це означає, що додаток може бути розроблений один раз і працювати на різних платформах без необхідності повторної компіляції або змін у коді.

Визначення кросплатформового додатку передбачає його сумісність з різними операційними системами, такими як Windows, macOS, Linux, а також різними апаратними платформами, такими як комп'ютери, мобільні пристрої, планшети та інше.

Порівняння кросплатформових та нативних додатків основні відмінності:

Нативний додаток є програмою, яка задовольняє специфічні вимоги операційної системи, використовуючи набір SDK, стек технологій нижчого рівня та апаратні можливості, такі як пам'ять, камера, сенсори та інші додатки, встановлені на пристрої.

Переваги нативних додатків включають:

- Високу продуктивність
- Розширені функціональні можливості
- Плавний користувацький досвід

Кросплатформовий додаток є додатком, який працює на кількох операційних системах, тому він може використовуватися на різних

смартфонах, планшетах, персональних комп'ютерах, годинниках та підключених до Інтернету телевізорах.

Кросплатформові додатки мають багато переваг:

- Використання від 70% до 90% одного й того ж коду
- Легкість обслуговування та оновлення
- Широкий охоплюваний аудиторій
- Швидший вихід на ринок

Існують два типи кросплатформових мобільних додатків: нативно-кросплатформовий додаток, такий як Java або Kotlin для Android та Objective-C або Swift для iOS. Розробники кросплатформових додатків створюють загальний API, який працює на нативному SDK та будують додатки для двох платформ, використовуючи загальний код. Для цієї мети використовуються фреймворки, такі як Xamarin, React Native та Kotlin Multiplatform [14].

Гібридний додаток: хоча мобільні додатки призначені для смартфонів та планшетів, їх бізнес-логіка знаходиться на боці сервера. Оскільки у SDK для iOS та Android є багатий набір веб-компонентів, можна побудувати GUI додатка за допомогою HTML5, CSS та JavaScript. Далі розробники упаковують цей код у WebView - вбудований в мобільний додаток браузер, який може відображати вміст, як звичайний веб-сайт. Деякі гібридні додатки навіть можуть взаємодіяти з апаратними компонентами смартфона, хоча їхні можливості можуть бути обмеженими. На сьогоднішній день найбільш перспективними фреймворками для розробки гібридних додатків є Apache Cordova (раніше відомий як PhoneGap) і Flutter [14].

При розгляді переваг та недоліків розробки кросплатформових додатків, процес вибору найбільш підходящої технології для розробки додатка повинен враховувати основні цілі створюваного додатка.

Наприклад, якщо ви задовольняєте наступні умови, розробка кросплатформового мобільного додатка буде хорошим вибором:

- Ви хочете якомога швидше вивести свій мобільний додаток на ринок.

- Ви хочете перевірити свою концепцію додатка та створити MVP без великого затрати часу та зусиль.
- Основною метою є залучення більшої аудиторії.
- У вас вже є веб-сайт і ви хочете побудувати мобільний додаток на його основі з мінімальними витратами.
- Цей додаток не потребує повної обчислювальної потужності смартфона.
- Додаток не містить складних анімацій та обчислень.

Однак, якщо вам потрібно створити високопродуктивний мобільний додаток зі складним інтерфейсом, багатою анімацією та постійним використанням ресурсів мобільного пристрою, кращим варіантом буде розглянути розробку нативних додатків [15].

1.3.2 Вибір мови програмування для розробки

Кожна мова програмування має власні недоліки та переваги над іншими (див. табл.1.3). Вибір мови програмування Python для розробки кросплатформових додатків має кілька переваг:

- Легкість вивчення: Python має простий і зрозумілий синтаксис, що робить його легким для вивчення навіть для початківців. Це скорочує час, необхідний для вивчення мови програмування та початку розробки кросплатформових додатків.
- Підтримка кросплатформових інструментів: Python підтримує різноманітні інструменти для розробки кросплатформових додатків, такі як PyInstaller, PyOxidizer, Py2exe, що дозволяють упаковувати програми Python у виконувані файли для різних операційних систем.
- Зрілість та популярність: Python є однією з найпопулярніших мов програмування і широко застосовується в сфері розробки програмного забезпечення. Це означає, що ви матимете доступ до великої спільноти розробників, обширної документації та підтримки [16, 17].
- Розширені можливості: має багатий набір стандартних бібліотек і модулів, які дозволяють розробникам швидко втілювати різноманітні

функціональність у своїх додатках. Існує велика кількість сторонніх бібліотек і фреймворків, що дозволяють розширити можливості мови і використовувати її для розробки різноманітних типів додатків, включаючи веб-додатки, мобільні додатки, штучний інтелект тощо.

Таблиця 1.3

Порівняння мов програмування для кросплатформенних додатків

Мова програмування	**Переваги**	**Недоліки**
Python	Простота вивчення, багата екосистема, підтримка	Менша продуктивність порівняно з C++
JavaScript	Широко підтримується веб-браузерами	Обмежена функціональність, швидкість
C#	Можливість використання фреймворку .NET	Обмежена підтримка платформ
Java	Велика кількість фреймворків, платформа Android	Великий обсяг коду, більш важкість вивчення
Dart	Використовується в Flutter для мобільних додатків	Обмежена спільнота розробників, молодий

1.3.3 Огляд кросплатформенних фреймворків та бібліотек для розробки на Python

Flask - Цей легкий фреймворк був розроблений для веб-розробки з використанням Python. Flask підтримує кросплатформову розробку і може використовуватися на різних операційних системах та веб-серверах. Він надає простоту використання і гнучкість у розробці.

Kivy - Цей популярний фреймворк дозволяє розробляти кросплатформові мобільні додатки та інтерфейси для Android, iOS, Windows,

macOS і Linux. Kivy надає високорівневі можливості для створення дружніх користувачу та ефективних інтерфейсів, а також підтримує мультимедіа.

BeeWare - BeeWare надає набір інструментів і фреймворків для розробки кросплатформових додатків з використанням Python. Toga дозволяє будувати нативні інтерфейси користувача, Briefcase допомагає упаковувати додатки в виконувані файли, а Rubicon дозволяє використовувати Python у мові програмування Java.

Pygame - Ця бібліотека дозволяє розробляти ігри та графічні додатки з використанням Python. Pygame підтримує кросплатформову розробку для різних операційних систем і надає багатий набір функцій для графіки, звуку та анімації.

PySide/PyQt - Ці бібліотеки дозволяють розробляти кросплатформові інтерфейси користувача на різних операційних системах, включаючи Windows, macOS, Linux та мобільні платформи. PySide є відкритою альтернативою PyQt і має ліцензію LGPL.

Ці фреймворки та бібліотеки надають зручні та ефективні інструменти для розробки різних типів кросплатформових додатків, включаючи мобільні додатки, веб-додатки, ігри та графічні додатки. Вони дозволяють розробникам фокусуватися на функціональності та концепції додатків, максимально зменшуючи зусилля, необхідні для підтримки різних платформ [18].

У цій роботі програма розроблена з використанням PyQt5. PyQt5 - це фреймворк, що дозволяє розробляти кросплатформові інтерфейси користувача з використанням мови програмування Python. Ось кілька причин, чому PyQt5 може бути хорошим вибором:

- Використання Qt: PyQt5 базується на Qt, популярному фреймворку для розробки інтерфейсів користувача. Qt надає широкий набір готових компонентів і функціональних можливостей, що дозволяє швидко створювати складні інтерфейси з багатофункціональними можливостями.

- Підтримка багатьох платформ: PyQt5 підтримує різні операційні системи, включаючи Windows, macOS, Linux і мобільні платформи. Це дозволяє створювати додатки, що працюють на різних платформах, без необхідності великих змін у коді.
- Обширна документація та активна спільнота: PyQt5 має велику спільноту розробників, яка надає швидку підтримку та доступ до корисної інформації. Існує багато документації, прикладів та ресурсів, що полегшують навчання та розробку з використанням PyQt5.
- Гнучкість та розширюваність: PyQt5 дозволяє розробникам використовувати Python для розробки інтерфейсу користувача, а також підтримує C++ для оптимізації ресурсомістких частин додатків. Крім того, він має вбудовану підтримку для роботи з базами даних, мережами та іншими функціями.
- Гнучкі ліцензійні умови: PyQt5 має дві ліцензії, що дозволяють його використовувати в комерційних та безкоштовних проектах. Це робить його привабливим вибором для розробників з різними потребами та вимогами щодо ліцензування.

1.3.4 Принципи побудови GUI

- Простота: Основна мета полягає в тому, щоб зберігати інтерфейс простим і зрозумілим. Використовуйте прості та зрозумілі елементи керування, уникайте перевантаження інформацією та організовуйте елементи на екрані в логічному порядку.
- Узгодженість: Створіть єдиний стиль та шаблони для всього інтерфейсу. Використовуйте однакові піктограми, кольори, шрифти та елементи керування для забезпечення узгодженості всіх взаємодій користувача.
- Зручність використання: Зробіть користування якомога простішим для користувача. Розмістіть часто використовувані елементи у зручних для користувача місцях, використовуйте зрозумілі терміни та фрази, забезпечуйте легку навігацію та завершення завдань.

- Запобігання помилкам: Надайте захист від помилок та уникайте неправильного введення даних. Використовуйте перевірку даних, надайте чітку інформацію про помилки та можливості виправлення.
- Ефективність: Забезпечте швидкий та ефективний доступ до функціональності та інформації. Мінімізуйте кількість натискань та операцій, необхідних для виконання завдань, і надайте швидкі гарячі клавіші.
- Візуальна привабливість: Зверніть увагу на естетику та дизайн інтерфейсу. Використовуйте привабливі кольори, збалансовану композицію, легкочитані шрифти та відповідні розміри елементів.
- Тестування та зворотній зв'язок: Регулярно тестуйте свій інтерфейс для виявлення та виправлення помилок і дефектів. Збирайте відгуки від користувачів та розглядайте їх пропозиції щодо поліпшення інтерфейсу [19].

Висновки до розділу 1

Проаналізовано літературні джерела, пов'язані з темою розробки системи для виявлення фрактур кісток за зображеннями комп'ютерної томографії. Дослідження охопило такі аспекти, як передавання сигналу у прямому та зворотному напрямках нейромережі, застосування згорткових нейромереж, аугментація даних, методи оптимізації, функції втрати та активації. Також була розглянута архітектура Inception V3, що використано у системі.

Крім того, були розглянуті питання використання крос-платформених додатків та вибір мови програмування Python для розробки системи, що є ефективною та легкодоступною. Було розглянуто кросплатформені фреймворки та бібліотеки для розробки на Python, що допомогло вибрати найкращі інструменти для виконання завдання. Також було приділено увагу

дизайну та плануванню GUI, що забезпечує зручну та інтуїтивну взаємодію користувача з системою для пошуку переломів на зображеннях КТ.

РОЗДІЛ 2
ТЕОРЕТИЧНА ЧАСТИНА

2.1 Опис датасету MURA

MURA - набір даних, який є одним з найпопулярніших та широко використовуваних, що використовуються для класифікації комп'ютерних томографічних зображень, пов'язаних з м'язово-скелетною системою.

Основною метою MURA набору даних є надання даних для виявлення переломів, артриту та інших захворювань кісток і суглобів. Цей набір даних містить понад 40 000 комп'ютерних томографічних знімків, які отримані від пацієнтів з різними симптомами та патологією. Зображення охоплюють різні частини тіла, включаючи плече, руку, лікоть, зап'ясток, стегно, коліно та щиколотку (див. рис.2.1).

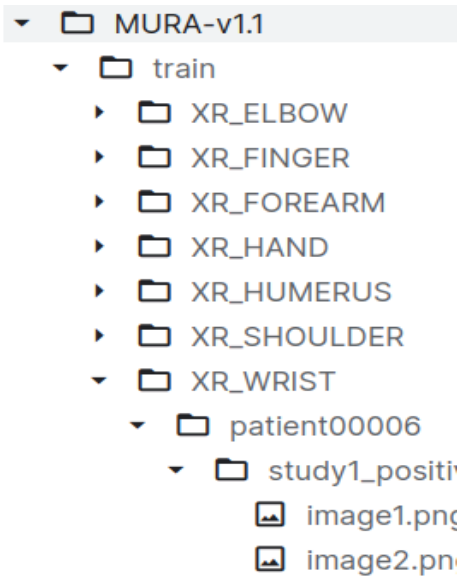


Рисунок 2.1 - Структура датасету MURA

MURA містить експертні надані описи для кожного зображення, які вказують на наявність або відсутність патології на зображенні. Це

представлено у вигляді бінарних міток "позитивний" або "негативний" для кожної частини тіла.

MURA інформацію про класифікацію на різних рівнях, включаючи ідентифікатор пацієнта, частину тіла та номер зображення. Це дозволяє використовувати набір даних для досліджень і аналізу зв'язків між різними мітками та патологіями.

Даний набір даних є викликом для дослідників та розробників, оскільки він містить зображення різної якості, з різним освітленням та перекриттям. Це вимагає відповідного попереднього аналізу та обробки зображень для досягнення оптимальних результатів класифікації.

Використання набору даних MURA у нашій роботі дало змогу навчати нейромережу для розпізнавання патологій на зображеннях комп'ютерної томографії та розробити систему виявлення переломів, яка може бути використана медичними фахівцями для діагностики та лікування м'язово-скелетних захворювань.

Крім основного набору зображень та експертних міток, набір даних містить додаткову інформацію, яка може бути використана для інших досліджень та аналізу. Наприклад, містить дані про вік та стать пацієнтів, що корисно при розгляді статевих та вікових аспектів при пошуку кісткової патології.

Однією з особливостей набору даних MURA є його асиметрія, оскільки кількість зображень для кожної частини тіла неоднакова. Це впливає на баланс класів та досягнення надійних результатів класифікації. Для компенсації цього можна використовувати вагову балансування та стратегії підсилення даних.

При підготовці даних для тренування та тестування моделей потрібно виконати попередню обробку зображень, включаючи зміну розміру, нормалізацію пікселів та можливе видалення шуму. Це допомагає покращити якість даних та сприяє стабільній та ефективній тренуванню моделей.

Крім того, варто зауважити, що набір даних MURA є відкритим та безкоштовним для дослідників та розробників. Це сприяє широкому використанню та порівнянню різних методів та алгоритмів у медичній діагностиці та обробці зображень.

Загалом, набір даних MURA є цінним ресурсом для дослідження та розробки системи пошуку фрактур кісток на основі зображень комп'ютерної томографії. Використання цього набору даних дозволяє розробити та вдосконалити методи та алгоритми класифікації для покращення діагностики та лікування м'язово-скелетних захворювань.

2.2 Алгоритм вирішення задачі

В рамках даної роботи було прийнято рішення створити окрему нейромережу для кожної частини тіла з метою виявлення та класифікації фрактур на зображеннях комп'ютерної томографії. Це обґрунтовано особливостями зображень та їх різнорідністю для кожної частини тіла.

Перш за все, зображення комп'ютерної томографії можуть мати різний формат, розмір та роздільну здатність в залежності від частини тіла, яка була обстежена. Наприклад, зображення плеча можуть мати іншу геометрію та пропорції порівняно зі зображеннями стегна чи руки. Такі різниці у розмірах та форматі зображень можуть впливати на роботу нейромережі та її здатність до адекватного виявлення уражень.

По-друге, внутрішні структури та анатомічні деталі також можуть варіюватись в залежності від частини тіла. Наприклад, структури кісток, суглобів та тканин можуть мати різну форму, розташування та характеристики у різних областях. Використання окремих нейромереж для кожної частини тіла дозволяє більш точно врахувати ці особливості та специфічність структур у кожній області.

Крім того, враховуючи різнорідність зображень та особливості кожної частини тіла, окремі нейромережі забезпечують більшу гнучкість та точність у виявленні фрактур. Кожна нейромережа може бути оптимізована та налаштована спеціально під конкретну частину тіла, що дозволяє досягти більш точних та надійних результатів. Більш точне виявлення фрактур має велике значення в медичній діагностиці та лікуванні, оскільки допомагає забезпечити раннє виявлення та адекватне лікування пацієнтів.

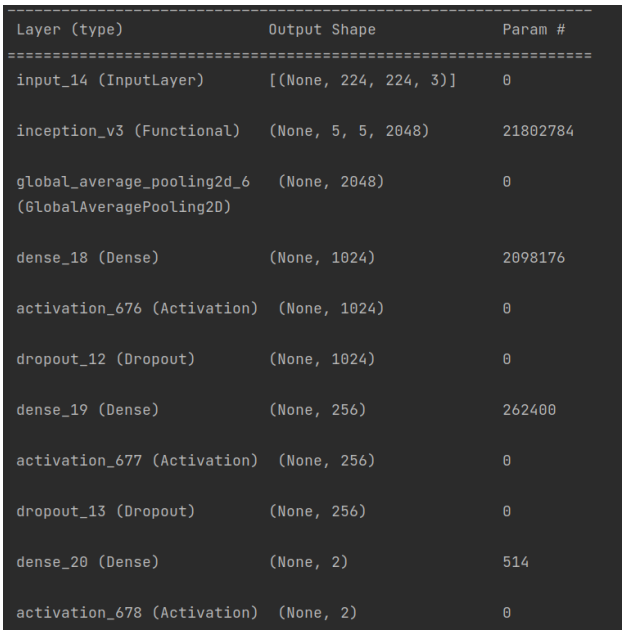
Отже, створення окремих нейромереж для кожної частини тіла виявляється ефективним підходом, оскільки враховує різні розміри, форми та особливості зображень для кожної області. Це дозволяє досягти високої продуктивності та поліпшити систему пошуку фрактур кісток.

2.3 Обрана архітектура нейронної мережі

Ця модель базується на архітектурі Inception V3. Дана модель відома своєю високою точністю та ефективністю в багатьох завданнях сприйняття зображень, включаючи класифікацію зображень.

Inception V3 базується на концепції використання "блоків змінних розмірів", що дозволяє моделі ефективно використовувати фільтри різних розмірів та обробляти інформацію по різних шляхах. Це дозволяє отримувати більш глибокі та широкі представлення зображень і покращувати точність класифікації.

Архітектура моделі, що представлена вище, є власним розширенням базової моделі Inception V3, в яку додано кілька шарів, що значно підвищують точність та повноту моделі. Зокрема, додано GlobalAveragePooling2D, який дозволяє знизити розмір отриманої карти ознак, зберігаючи при цьому її важливі особливості (див. рис.2.2).



Layer (type)	Output Shape	Param #
input_14 (InputLayer)	[(None, 224, 224, 3)]	0
inception_v3 (Functional)	(None, 5, 5, 2048)	21802784
global_average_pooling2d_6 (GlobalAveragePooling2D)	(None, 2048)	0
dense_18 (Dense)	(None, 1024)	2098176
activation_676 (Activation)	(None, 1024)	0
dropout_12 (Dropout)	(None, 1024)	0
dense_19 (Dense)	(None, 256)	262400
activation_677 (Activation)	(None, 256)	0
dropout_13 (Dropout)	(None, 256)	0
dense_20 (Dense)	(None, 2)	514
activation_678 (Activation)	(None, 2)	0

Рисунок 2.2 - Загальний вигляд архітектури нейромережі для виявлення фрактур на частині тіла

Також були додані додаткові шари, включаючи повністю з'єднані Dense шари, активаційний шар (з ReLU активацією) та Dropout шар, які зменшують перенавчання шляхом нормалізації та нелінійної активації для поліпшення продуктивності моделі. Це призводить до наявності двовимірного вихідного шару (Dense(2)), який використовує softmax активацію для розбиття задачі класифікації на дві категорії: пошкоджено та не пошкоджено.

Для поліпшення зручності користувача була створена додаткова нейромережа, яка вміє розпізнавати частину тіла на зображенні комп'ютерної томографії. Це було зроблено з метою забезпечити швидке та автоматичне визначення тілесних областей без необхідності вручну вказувати їх користувачем.

Після отримання зображення КТ, нейромережа класифікатор аналізує його та визначає, до якої конкретної частини тіла воно належить. Це може бути, наприклад, плече, кисть тощо. Після визначення частини тіла, система автоматично вибирає підходящу модель для виявлення уражень саме на цьому конкретному участку тіла. Кожна модель спеціалізується на виявленні

уражень у певній області тіла, що дозволяє досягти більш точних та надійних результатів.

Такий підхід значно спрощує використання системи для користувача, оскільки він лише вибирає зображення КТ, а решта процесу відбувається автоматично. Це забезпечує швидкий та зручний спосіб виявлення фрактур та уражень на різних частинах тіла, що сприяє ефективності та точності діагностики.

2.4 Композиція аугментацій

Використання таких аугментацій дозволяє розширити обсяг та різноманітність тренувального набору даних, що в свою чергу сприяє покращенню універсальності та робастності моделі. Вона стає більш здатною розпізнавати фрактури на зображеннях КТ навіть при змінних умовах освітлення, орієнтації та масштабуванні. Окрім того, це допомагає уникнути перенавчання та покращує узагальнюючу здатність моделі до нових зображень (див. рис.2.3).

```
Compose([
    HorizontalFlip(p=0.5),
    RandomContrast(limit=0.2, p=0.5),
    RandomGamma(gamma_limit=(80, 120), p=0.5),
    RandomBrightness(limit=0.2, p=0.5),
    ShiftScaleRotate(
        shift_limit=0.0625,
        scale_limit=0.1,
        rotate_limit=15,
        border_mode=cv2.BORDER_REFLECT_101, p=0.8
    ),
    ToFloat(max_value=255)
])
```

Рисунок 2.3 - Аугментації, що були накладені на тренувальний набір даних

Ось кілька причин, чому було обрано саме таку композицію аугментації:

- Горизонтальне відображення (Horizontal Flip): Ця аугментація дозволяє створювати варіації зображень шляхом відображення їх горизонтально. Це допомагає моделі навчатися розпізнавати фрактури кісток з різних кутів та позицій.
- Випадковий контраст (Random Contrast): Зміна контрасту зображень сприяє збільшенню різниці між фрактурами та нормальними частинами кісток. Це може поліпшує здатність моделі розрізняти та класифікувати ураження.
- Випадкова гамма-корекція (Random Gamma): Застосування випадкових значень гамми дозволяє регулювати яскравість та контраст зображень. Це покращує видимість фрактур та забезпечує кращу роздільну здатність для моделі.
- Випадкова яскравість (Random Brightness): Зміна яскравості зображень допомагає створювати варіації в освітленні, що допомагає моделі розпізнавати ураження при різних умовах освітлення.
- Зсув, масштабування та поворот (Shift Scale Rotate): Ця аугментація дозволяє випадково зсувати, масштабувати та обертати зображення. Це допомагає моделі навчатися розпізнавати фрактури в різних положеннях та масштабах, що покращує її загальну роботу на реальних зображеннях.

2.5 Опис інтерфейсу користувача

У розробленому додатку, реалізованому з використанням PyQT5, користувачу надається зручний і інтуїтивно зрозумілий інтерфейс для взаємодії з системою пошуку фрактур кісток за зображеннями КТ.

При запуску додатку, користувачу пропонується вибрати зображення КТ для подальшого аналізу. Після вибору зображення, відбувається автоматична

обробка за допомогою нейромережі, яка класифікує, до якої частини тіла належить це зображення.

Якщо нейромережа визначає частину тіла вірно, на екрані з'являється спрогнозована назва цієї частини тіла, а користувачу надається можливість натиснути кнопку "Продовжити". Це дозволяє перейти до наступного етапу, де спеціалізована нейромережа для цієї частини тіла визначає наявність фрактур на зображенні КТ.

У випадку, коли нейромережа неправильно класифікує частину тіла, користувач має можливість самостійно вибрати правильну частину тіла за допомогою інтерактивного вибору. Після вибору правильної частини тіла, процес переходить до наступного етапу, де використовується відповідна нейромережа для аналізу фрактур на цій обраній частині тіла.

Окрім того, зображення КТ візуалізується прямо в додатку, що дозволяє користувачу отримати візуальну інформацію про аналізоване зображення та його результати. Це забезпечує зручність та легкість сприйняття інформації користувачем.

У розробленому додатку також використовуються прогрес-бари для кожного етапу обробки зображень, що дозволяє користувачеві чітко відстежувати прогрес процесу аналізу. Це створює інтуїтивно зрозуміле середовище для користувача та покращує його взаємодію з системою.

Залежно від остаточного діагнозу та виявлення фрактур, колір тла додатка змінюється, що допомагає користувачеві швидко оцінити наявність уражень. У разі, якщо фрактури виявлені, тло підсвічується червоним кольором, а якщо фрактур не знайдено, тло підсвічується зеленим кольором. Цей візуальний ефект допомагає користувачеві швидко сприйняти результати аналізу та зробити приблизну оцінку стану пацієнта.

Дизайн і розташування кнопок і елементів інтерфейсу гармонійно підібрані для забезпечення зручності користувача. Кольори, використані в інтерфейсі, обрані з урахуванням естетичних принципів, що забезпечує

привабливий та привітний вигляд додатку. Розташування кнопок та інших елементів створено з урахуванням логічного потоку взаємодії користувача з системою, що сприяє зрозумілості та легкості використання.

Всі ці дизайн-рішення були впроваджені з метою створення привабливого, зрозумілого та зручного для використання інтерфейсу, який сприяє ефективній взаємодії користувача з системою та поліпшує загальний досвід користувача.

Висновок до розділу 2

В даному розділі було проведено аналіз структури та характеристик датасету MURA, вивчені всі класи, що присутні в датасеті, для розуміння задачі класифікації фрактур на зображеннях КТ. Описано алгоритм, який найкращим чином вирішує поставлену задачу та детально пояснено принципи та кроки алгоритму. Проведено обґрунтування вибору архітектури нейронної мережі, розглянуто переваги та особливості обраної архітектури.

Був складений стек аугментаційних методів, які ефективно використовуються для покращення якості зображень та роблять аналіз більш точним, описано конкретні методи аугментації. Надано детальний опис користувацького інтерфейсу, який забезпечує зручність та легкість взаємодії користувача з системою, обґрунтовано розташування кнопок, вибір кольорів тла та інших елементів інтерфейсу, що сприяють зрозумілості та привабливості додатку.

РОЗДІЛ 3
ПРАКТИЧНА РЕАЛІЗАЦІЯ ЗАДАЧІ

3.1 Обґрунтування програмного середовища

В контексті розробки цього додатку було обрано наступний екосистему: Python, Keras і PyQt5. Вибір цих інструментів здійснено для використання потужних і легких у використанні наборів засобів для розробки програм, які включають машинне навчання, обробку зображень і створення користувацького інтерфейсу. Ці технології дозволяють ефективно, швидко і зручно реалізувати необхідні функціональні можливості.

- Python: є однією з найпопулярніших мов програмування, що використовується для розробки різноманітних програм, включаючи машинне навчання та обробку зображень. Вона має простий і зрозумілий синтаксис, що сприяє швидкій розробці програм і скорочує час написання коду. Python також має велику кількість сторонніх бібліотек і фреймворків, що підтримують машинне навчання і обробку зображень, що робить його ідеальним вибором для цього проекту.
- Keras: це високорівневий фреймворк для побудови нейронних мереж з використанням Python. Він має простий і зрозумілий інтерфейс, що дозволяє легко створювати, навчати та використовувати нейронні мережі. Keras надає багато готових моделей, архітектур і інструментів для обробки зображень, що дозволяє ефективно виконувати завдання розпізнавання та класифікації зображень у цьому проекті.
- PyQt5: це набір бібліотек для розробки графічного інтерфейсу користувача в Python. Він базується на популярній бібліотеці Qt і надає багатий функціонал для створення зручних і привабливих інтерфейсів.
- PyQt5 дозволяє легко і швидко створювати вікна, кнопки, форми та інші елементи GUI, що взаємодіють з функціональністю додатку. Це

- дозволяє розробляти зручний і привабливий користувацький інтерфейс для вашого додатку.

3.2 Навчання та валідація моделей

У роботі було навчено 5 моделей для класифікації наступних частин тіла: плече, лікоть, палець, кисть і долоня. Кожна з цих моделей була навчена для визначення присутності або відсутності ушкоджень на відповідних частинах тіла. Кожна з цих моделей досягла точності 90% і вище, що свідчить про їхню ефективність у виявленні ушкоджень (див. табл.3.1).

Також була навчена модель для класифікації частин тіла. Ця модель використовується для визначення, до якої конкретної частини тіла належить зображення КТ. Вона дозволяє автоматично ідентифікувати частину тіла без потреби вручну вводити цю інформацію. Ця модель також досягла також високих показників.

Варто зазначити, що датасет MURA містить дані про плечову кістку і передпліччя. Однак, в роботі не були представлені результати для цих частин тіла через обмежену кількість даних, наявний шум у даних та недостатню точність моделей при їх використанні. Це свідчить про важкість виявлення ушкоджень на цих конкретних частинах тіла та необхідність подальших досліджень для поліпшення точності і надійності моделей в цьому контексті.

Таблиця 3.1

Досягнута точність моделей на валідаційних даних

Назва моделі	**Досягнута точність**
Класифікатор частини тіла	98%

Продовж. табл. 3.1

Назва моделі	**Досягнута точність на валідаційних даних**
Класифікатор наявності уражень на Кистях	94%
Класифікатор наявності уражень на Пальцях	91%
Класифікатор наявності уражень на Плечах	90%
Класифікатор наявності уражень на Долонях	95%
Класифікатор наявності уражень на Локтях	97%

Для кожної моделі, в рамках розробки, було підібрано оптимальні гіпер параметри та кількість епох з метою досягнення найкращої точності та забезпечення ефективності тренування.

Початковий набір даних був розділений на тренувальну та валідаційну вибірки. Це важливий крок у процесі навчання моделі. Розділення даних на тренувальну та валідаційну вибірки дозволяє оцінити ефективність моделі на незалежних даних та виявити перенавчання (overfitting).

Зазвичай, рекомендована пропорція розділення даних становить близько 80% для тренувальної вибірки і 20% для валідаційної вибірки. Це означає, що 80% даних використовується для навчання моделі, тоді як 20% залишаються для оцінки її ефективності та уникнення перенавчання. Така пропорція дозволяє забезпечити належну кількість даних для тренування, одночасно залишаючи достатньо даних для валідації.

Для підбору гіперпараметрів, таких як learning rate , було проведено експерименти з різними значеннями, щоб знайти оптимальне значення, для кожної моделі яке забезпечує швидку збіжність та стабільність навчання

моделі. Крім того, контроль епох також використовувався для уникнення перенавчання моделі і забезпечення оптимальної точності на валідаційних даних.

В результаті цих процесів підбору гіпер параметрів та контролю навчання, було досягнуто найкращої ефективності моделей, забезпечено їх стабільність та запобіжено перенавчанню.

Модель навчена на КТ-зображеннях Кистей на перші епосі мала значення точності 47%, на восьмій - 94% (див. рис.3.1).

```
Epoch 1/8
304/304 [==============================] - 503s 2s/step - loss: 0.5181 - accuracy: 0.7519 - val_loss: 0.7581 - val_accuracy: 0.4766
Epoch 2/8
304/304 [==============================] - 354s 1s/step - loss: 0.4074 - accuracy: 0.8299 - val_loss: 0.4940 - val_accuracy: 0.7703
Epoch 3/8
304/304 [==============================] - 377s 1s/step - loss: 0.3757 - accuracy: 0.8754 - val_loss: 0.4568 - val_accuracy: 0.7828
Epoch 4/8
304/304 [==============================] - 364s 1s/step - loss: 0.3512 - accuracy: 0.8950 - val_loss: 0.4168 - val_accuracy: 0.8516
Epoch 5/8
304/304 [==============================] - 366s 1s/step - loss: 0.2812 - accuracy: 0.9143 - val_loss: 0.3413 - val_accuracy: 0.8375
Epoch 6/8
304/304 [==============================] - 361s 1s/step - loss: 0.3746 - accuracy: 0.8737 - val_loss: 0.4335 - val_accuracy: 0.8681
Epoch 7/8
304/304 [==============================] - 359s 1s/step - loss: 0.1909 - accuracy: 0.9465 - val_loss: 0.2597 - val_accuracy: 0.9237
Epoch 8/8
304/304 [==============================] - 359s 1s/step - loss: 0.1585 - accuracy: 0.9678 - val_loss: 0.1764 - val_accuracy: 0.9437
```

Рисунок 3.1 - Логи навчання Кистьової моделі

Також процес навчання відображено на графіку (див. рис.3.2).

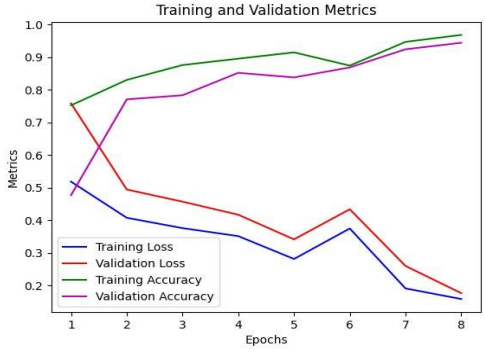


Рисунок 3.2 - Візуалізовані логи навчання Кистьової моделі

Модель навчена на КТ-зображеннях Пальців на перші епосі мала значення точності 64%, на шостій - 91% (див. рис.3.3).

```
Epoch 1/6
159/159 [==============================] - 256s 2s/step - loss: 0.6015 - accuracy: 0.6959 - val_loss: 0.7066 - val_accuracy: 0.6487
Epoch 2/6
159/159 [==============================] - 184s 1s/step - loss: 0.4990 - accuracy: 0.7898 - val_loss: 0.5504 - val_accuracy: 0.7531
Epoch 3/6
159/159 [==============================] - 189s 1s/step - loss: 0.4505 - accuracy: 0.8860 - val_loss: 0.4428 - val_accuracy: 0.8738
Epoch 4/6
159/159 [==============================] - 190s 1s/step - loss: 0.4105 - accuracy: 0.9024 - val_loss: 0.4927 - val_accuracy: 0.8464
Epoch 5/6
159/159 [==============================] - 182s 1s/step - loss: 0.3756 - accuracy: 0.9262 - val_loss: 0.3940 - val_accuracy: 0.9022
Epoch 6/6
159/159 [==============================] - 189s 1s/step - loss: 0.3536 - accuracy: 0.9318 - val_loss: 0.3804 - val_accuracy: 0.9166
```

Рисунок 3.3 - Логи навчання Пальцевої моделі

Також процес навчання відображено на графіку (див. рис.3.4).

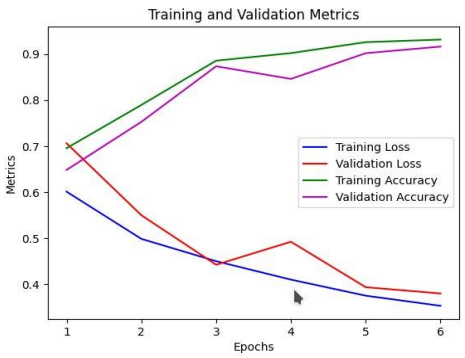


Рисунок 3.4 - Візуалізовані логи навчання Пальцевої моделі

Модель навчена на КТ-зображеннях Плечей на перші епосі мала значення точності 69%, на сьомій - 90% (див. рис.3.5).

```
Epoch 1/7
261/261 [==============================] - 461s 2s/step - loss: 0.8374 - accuracy: 0.6471 - val_loss: 0.6483 - val_accuracy: 0.6990
Epoch 2/7
261/261 [==============================] - 309s 1s/step - loss: 0.5362 - accuracy: 0.7478 - val_loss: 0.6016 - val_accuracy: 0.6513
Epoch 3/7
261/261 [==============================] - 310s 1s/step - loss: 0.4917 - accuracy: 0.8038 - val_loss: 0.5144 - val_accuracy: 0.7935
Epoch 4/7
261/261 [==============================] - 311s 1s/step - loss: 0.4669 - accuracy: 0.8793 - val_loss: 0.4793 - val_accuracy: 0.8521
Epoch 5/7
261/261 [==============================] - 309s 1s/step - loss: 0.4383 - accuracy: 0.8959 - val_loss: 0.4072 - val_accuracy: 0.8892
Epoch 6/7
261/261 [==============================] - 305s 1s/step - loss: 0.4199 - accuracy: 0.8957 - val_loss: 0.5188 - val_accuracy: 0.8708
Epoch 7/7
261/261 [==============================] - 305s 1s/step - loss: 0.4239 - accuracy: 0.9187 - val_loss: 0.4554 - val_accuracy: 0.9024
```

Рисунок 3.5 - Логи навчання Плечової моделі

Також процес навчання відображено на графіку (див. рис.3.6).

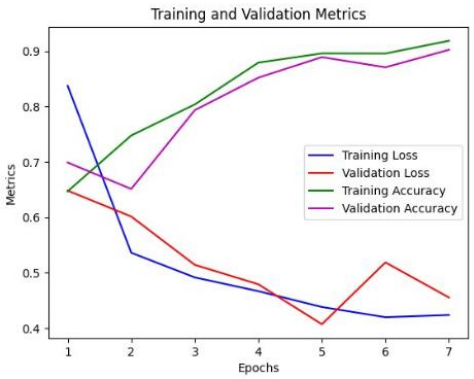


Рисунок 3.6 - Візуалізовані логи навчання Плечової моделі

Модель навчена на КТ-зображеннях Долоней на перші епосі мала значення точності 64%, на восьмій - 95% (див. рис.3.7).

```
Epoch 1/8
173/173 [==============================] - 287s 2s/step - loss: 0.2929 - accuracy: 0.7286 - val_loss: 0.3225 - val_accuracy: 0.6481
Epoch 2/8
173/173 [==============================] - 201s 1s/step - loss: 0.2377 - accuracy: 0.8920 - val_loss: 0.3178 - val_accuracy: 0.8581
Epoch 3/8
173/173 [==============================] - 209s 1s/step - loss: 0.1934 - accuracy: 0.9384 - val_loss: 0.3017 - val_accuracy: 0.9082
Epoch 4/8
173/173 [==============================] - 207s 1s/step - loss: 0.1692 - accuracy: 0.9317 - val_loss: 0.2713 - val_accuracy: 0.9177
Epoch 5/8
173/173 [==============================] - 205s 1s/step - loss: 0.1428 - accuracy: 0.9444 - val_loss: 0.2356 - val_accuracy: 0.9042
Epoch 6/8
173/173 [==============================] - 209s 1s/step - loss: 0.1159 - accuracy: 0.9362 - val_loss: 0.2516 - val_accuracy: 0.9418
Epoch 7/8
173/173 [==============================] - 204s 1s/step - loss: 0.0951 - accuracy: 0.9254 - val_loss: 0.2683 - val_accuracy: 0.9321
Epoch 8/8
173/173 [==============================] - 212s 1s/step - loss: 0.0851 - accuracy: 0.9501 - val_loss: 0.1837 - val_accuracy: 0.9478
```

Рисунок 3.7 - Логи навчання Долоневої моделі

Також процес навчання відображено на графіку (див. рис.3.8).

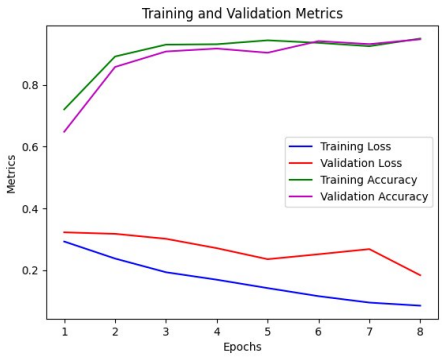


Рисунок 3.8 - Візуалізовані логи навчання Долоневої моделі

Модель навчена на КТ-зображеннях Ліктів на перші епосі мала значення точності 81%, на п’ятій - 97% (див. рис.3.9).

```
Epoch 1/5
154/154 [==============================] - 273s 2s/step - loss: 0.3833 - accuracy: 0.8587 - val_loss: 0.4735 - val_accuracy: 0.8166
Epoch 2/5
154/154 [==============================] - 179s 1s/step - loss: 0.2844 - accuracy: 0.9289 - val_loss: 0.4673 - val_accuracy: 0.8866
Epoch 3/5
154/154 [==============================] - 186s 1s/step - loss: 0.2356 - accuracy: 0.9638 - val_loss: 0.4808 - val_accuracy: 0.9570
Epoch 4/5
154/154 [==============================] - 185s 1s/step - loss: 0.2111 - accuracy: 0.9663 - val_loss: 0.3507 - val_accuracy: 0.9412
Epoch 5/5
154/154 [==============================] - 186s 1s/step - loss: 0.1798 - accuracy: 0.9757 - val_loss: 0.2926 - val_accuracy: 0.9681
```

Рисунок 3.9 - Логи навчання Ліктьової моделі

Також процес навчання відображено на графіку (див. рис.3.10).

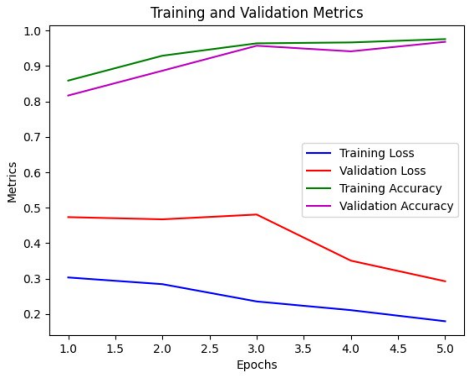


Рисунок 3.10 - Візуалізовані логи навчання Ліктьової моделі

Модель навчена на всіх КТ-зображеннях для загальної класифікації на перші епосі мала значення точності 92%, на четвертій - 98% (див. рис.3.11).

```
Epoch 1/4
1053/1053 [==============================] - 1667s 2s/step - loss: 0.2130 - accuracy: 0.9326 - val_loss: 0.2426 - val_accuracy: 0.9240
Epoch 2/4
1053/1053 [==============================] - 1240s 1s/step - loss: 0.0896 - accuracy: 0.9752 - val_loss: 0.1001 - val_accuracy: 0.9726
Epoch 3/4
1053/1053 [==============================] - 1252s 1s/step - loss: 0.0763 - accuracy: 0.9791 - val_loss: 0.1737 - val_accuracy: 0.9487
Epoch 4/4
1053/1053 [==============================] - 1251s 1s/step - loss: 0.0669 - accuracy: 0.9815 - val_loss: 0.1027 - val_accuracy: 0.9803
```

Рисунок 3.11 - Логи навчання моделі Загального класифікатора

Також процес навчання відображено на графіку (див. рис.3.12).

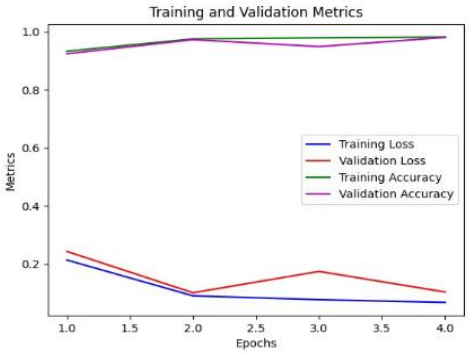


Рисунок 3.12 - Візуалізовані логи навчання моделі Загального класифікатора

Важливим етапом було підібрати learning rate (див. рис.3.13)

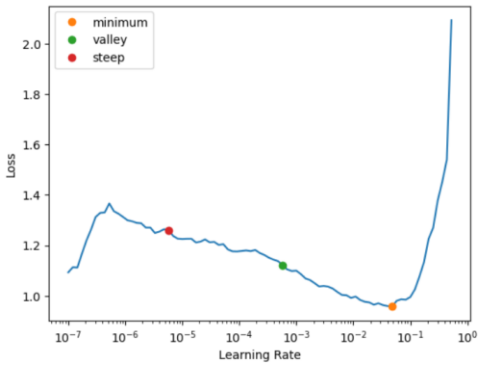


Рисунок 3.13 - процес підбору learning rate

Також було проведено аналіз датасету (див. рис.3.14)

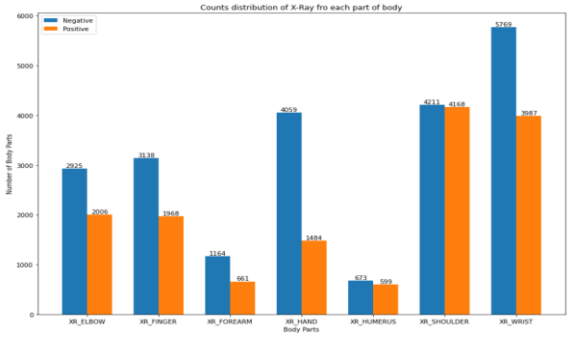


Рисунок 3.14 - Графік розподілу частин тіла та їх міток

3.3 Огляд розробленого інтерфейсу користувача

При запуску додатку з’являється наступний інтерфейс (див. рис.3.15)

Рисунок 3.15 - Початковий вигляд інтерфейсу

При натиску на кнопку “Обрати зображення” користувачу надається можливість відразу валідні для системи КТ знімки (див. рис.3.16)

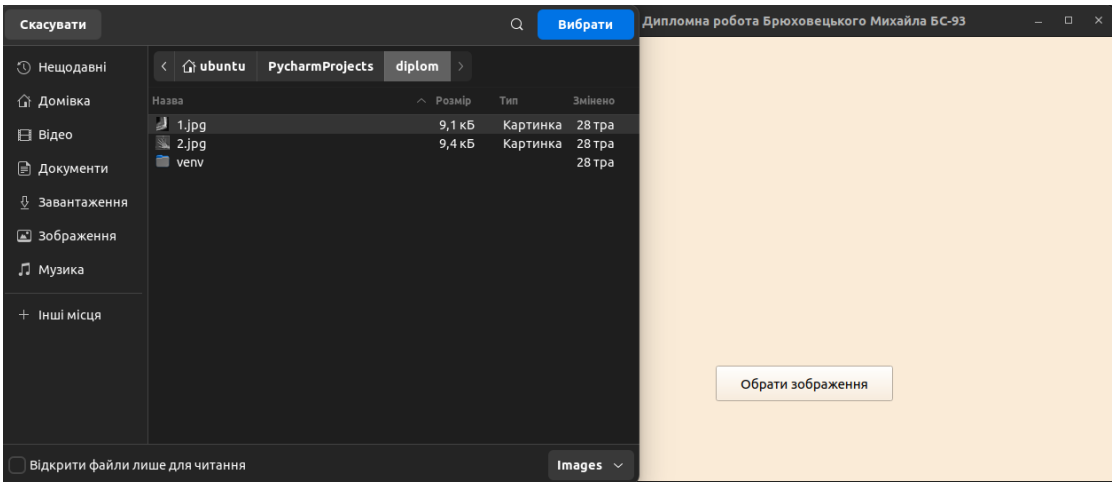


Рисунок 3.16 - Процес обирання КТ-зображення

Далі обране КТ-зображення розміщується в інтерфейсі користувача, а також з’являється можливість визначити тип кістки (див. рис.3.17)

Рисунок 3.17 - Відображення обраного КТ-знімку

Після натискання кнопки “Визначити тип кістки”, починає працювати модель класифікатор, яка займає певний час, і для інтерактиву з користувачем з’являється прогрес бар (див. рис.3.18)

Рисунок 3.18 - Процес визначення типу кістки

Потім відображається текстове поле щодо визначеного типу кістки та з’являється можливість обрати тип кістки самостійно або ж продовжити у випадку вірного розпізнавання (див. рис.3.19).

Рисунок 3.19 - Визначений тип кістки

Після обрання кнопки “Продовжити” відпрацьовує відповідна нейромережа згідно визначеної частини тіла для виявлення пошкодження на ній (див. рис.3.20).

Рисунок 3.20 - Негативний результат виявлення ушкоджень на КТ-зображенні

Відображення меню у випадку невірної класифікації частини тіла та натискання кнопки “Обрати тип самостійно” (див. рис.3.21).

Рисунок 3.21 - Кнопки для вибору частини тіла

Відображення кінцевого меню з інакшим обраним КТ-зображенням (див. рис.3.22).

Рисунок 3.22 - Позитивний результат виявлення ушкоджень на КТ-зображенні

3.4 Розрахунок економічного ефекту ПЗ

Основна функція F_0 - розробка системи, яка вирішує задачу пошуку фрактур кісток за зображеннями КТ. Основні функції ПЗ: F_1(мова програмування), F_2(ML фреймворк), F_3(фреймворк кросплатформеності).

Кожна з них може мати кілька варіантів реалізації:

- F_1
  - Python;
  - R;
- F_2
  - Keras
  - Pytorch
  - Tensorflow

- F_3
  - PyQt5
  - Kivy
  - Shiny

  Імплементації зображено на морфологічній карті (див. рис.3.23).

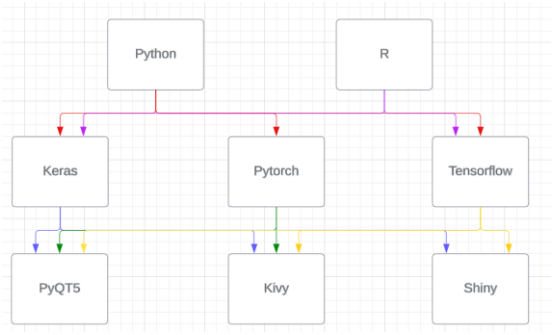


Рисунок 3.23 – Морфологічна карта

Побудова матриці варіантів є систематичним підходом, що дозволяє оцінювати та порівнювати різні варіанти з урахуванням визначених критеріїв (див. табл.4.1). Вона є ефективним інструментом для прийняття рішень, оскільки допомагає структурувати інформацію та зробити об'єктивний вибір.

Таблиця 4.1

Основні функції в позитивно-негативній матриці варіантів

Основні функції	**Варіанти реалізації**	**Переваги**	**Недоліки**
F1	А	Простота, Широке застосування, Легко читається	Повільна інтерпретація
	Б	Швидкість, Спрощена статистика, Велика кількість пакетів	Обмежена машинне навчання, Відсутність універсальності, Відсутність великої спільноти
F2	А	Простота, Широке застосування, Підтримка Tensorflow	Обмежена архітектура, Менша функціональність, Менше ресурсів

Продовж. табл. 4.1

Основні функції	**Варіанти реалізації**	**Переваги**	**Недоліки**
F2	Б	Гнучкість, Швидкість, Зручна розробка	Менша стабільність, Більш вимоглива до пам'яті, Обмежені можливості
	В	Масштабованість, Широке застосування, Велика спільнота	Складність, Вимоглива до ресурсів, Важко налаштовується
F3	А	Широкий набір функціональності, зручність у використанні	Важке вивчення, висока складність розробки
	Б	Вбудована підтримка мультимедіа, швидка розробка інтерфейсу	Обмежені можливості, висока витрата пам'яті
	В	Велика кількість готових компонентів та можливостей	Обмежена кросплатформність, Менша гнучкість

Спираючись на позитивно-негативну матрицю, необхідно відкинути деякі імплементації функцій, оскільки вони не задовольняють поставлені задачі, а саме функції: F1(Б), F2(В), F3(В). Розрахуємо показники якості (див. табл.4.2).

Таблиця 4.2

Розрахунок показників якості

Основні функції	Варіант імплементації	Бальна оцінка	Коефіцієнт рівня якості
F1	А	10	3,44
F2	А	10	2,81
	Б	5	2,19
F3	А	10	1,56
	Б	5	1,09

- $F_1(А) – F_2(А) – F_3(А) = 3,44 + 2,81 + 1,56 = 7,81$
- $F_1(А) – F_2(А) – F_3(Б) = 3,44 + 2,81 + 1,09 = 7,34$

- $F_1(А) – F_2(Б) – F_3(А) = 3,44 + 2,19 + 1,56 = 7,19$
- $F_1(А) – F_2(Б) – F_3(Б) = 3,44 + 2,19 + 1,09 = 6,72$

Найкращою є перша комбінація, а саме - Python, Keras, PyQT5, що і використана у даній роботі, даний варіант має найвищий техніко-економічний коефіцієнт, вартість витрат становить 366 514,84 грн.

Висновок до розділу 3

Розроблений додаток автоматизує діагностику ушкоджень кісток, використовуючи нейромережі та комп'ютерний зір. Він забезпечує швидке та точне виявлення ушкоджень на зображеннях комп'ютерної томографії, полегшуючи роботу лікарів та покращуючи точність діагностики. Інтерфейс додатку, розроблений на PyQt5, є зручним та інтуїтивно зрозумілим. Цей додаток має потенціал для використання в медичній практиці, полегшуючи роботу лікарів та покращуючи результати лікування. Він також має перспективи розвитку та вдосконалення, які можуть включати розширення функціональності та використання додаткових датасетів для донавчання побудованих нейромереж. Виконана робота виправдала очікування та має значний потенціал у медичній сфері.

ЗАГАЛЬНІ ВИСНОВКИ

У даній роботі була розроблена система пошуку фрактур кісток за зображеннями комп’ютерної томографії. У результаті проведених досліджень було досягнуто високої точності виявлення фрактур за допомогою розробленої системи. Нейромережі, побудовані для аналізу зображень КТ, продемонстрували високу ефективність і здатність до автоматичного виявлення фрактур.

Додаток, розроблений на основі системи, дозволяє медичному персоналу швидко та точно аналізувати зображення КТ для виявлення фрактур кісток. Тестування системи на певній кількості користувачів підтвердило її ефективність та практичну цінність.

В ході дослідження були виявлені шляхи подальшого покращення системи, такі як вдосконалення алгоритмів обробки зображень, розширення бази даних та залучення більшої кількості даних для тренування нейромереж. Крім того, система має значні перспективи, оскільки на сьогоднішній день не існує альтернативних рішень, що забезпечують таку високу автоматизацію та точність при виявленні фрактур кісток на зображеннях КТ.

Розроблена система пошуку фрактур кісток на зображеннях КТ є потужним інструментом для медичної діагностики та може забезпечити високу точність та швидкість при виявленні фрактур. Подальше вдосконалення та розширення системи відкриває шляхи до нових можливостей та застосувань у медичній сфері

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Luhaniwal, V. Forward propagation [Електронний ресурс] - Режим доступу до ресурсу: https://towardsdatascience.com/forward-propagation-in-neural-networks-simplified-math-and-code-version-bbcfef6f9250
2. Johnson, D. Backward propagation [Електронний ресурс] - Режим доступу до ресурсу: https://www.guru99.com/backpropogation-neural-network.html
3. Mandal, M. Introduction to CNN [Електронний ресурс] - Режим доступу до ресурсу: https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/
4. Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.
5. Li, F., Karpathy, A., & Johnson, J. Convolutional Neural Networks for Visual Recognition
6. Albumentations [Електронний ресурс] - Режим доступу до ресурсу: https://albumentations.ai/docs/
7. Sagar, R. ML optimization methods [Електронний ресурс] - Режим доступу до ресурсу: https://analyticsindiamag.com/optimisation-machine-learning-methods-gradient-descent/
8. Shaikh, S. Optimization techniques for ML models [Електронний ресурс] - Режим доступу до ресурсу: https://medium.com/geekculture/optimization-techniques-for-ml-models-bd500c8398ce
9. Parmar, R. Loss functions in machine learning [Електронний ресурс] - Режим доступу до ресурсу: https://towardsdatascience.com/common-loss-functions-in-machine-learning-46af0ffc4d23
10. Minaee, S. Machine Learning Metrics [Електронний ресурс] - Режим доступу до ресурсу: https://towardsdatascience.com/20-popular-machine-learning-metrics-part-1-classification-regression-evaluation-metrics-1ca3e282a2ce

11. Oommoo. Вибір шару активації [Електронний ресурс] - Режим доступу до ресурсу: https://habr.com/ru/articles/727506/
12. Neurohive. Функції активації нейромереж [Електронний ресурс] - Режим доступу до ресурсу: https://neurohive.io/ru/osnovy-data-science/activation-functions/
13. Google. Inception V3 [Електронний ресурс] - Режим доступу до ресурсу: https://cloud.google.com/tpu/docs/inception-v3-advanced
14. Klubnikin, A. Cross-platform vs Native Mobile App Development [Електронний ресурс] - Режим доступу до ресурсу: https://andrei-klubnikin.medium.com/cross-platform-vs-native-mobile-app-development-choosing-the-right-dev-tools-for-your-app-project-47d0abafee81
15. Moroz, A. CROSS-PLATFORM MOBILE APP DEVELOPMENT [Електронний ресурс] - Режим доступу до ресурсу: https://www.upsilonit.com/blog/cross-platform-mobile-app-development-all-you-need-to-know
16. Кременчук, В. Топ мов кроссплатформенного програмування [Електронний ресурс] - Режим доступу до ресурсу: https://beetroot.academy/blog/courses/top-5-programming-languages
17. Лазар, А. Особливість багатоплатформного ПЗ [Електронний ресурс] - Режим доступу до ресурсу: https://poradumo.com.ua/221276-iaky-osoblivist-maye-bagat/
18. Jain, N. Python for Mobile App Development [Електронний ресурс] - Режим доступу до ресурсу: https://www.nikhilajain.com/post/mobile-apps-using-python
19. StudLancer. ПРИНЦИПИ ПРОЕКТУВАННЯ GUI [Електронний ресурс] - Режим доступу до ресурсу: https://stud.com.ua/174175/informatika/printsipi_proektuvannya

20. KidsHealth. Fracturas en los huesos [Електронний ресурс] - Режим доступу до ресурсу: https://kidshealth.org/MedStarHealth/es/teens/broken-bones.html
21. Zhou, S.K., Greenspan, H., & Shen, D. (2017). Deep Learning for Medical Image Analysis. Academic Pres.