НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Радіотехнічний факультет

Кафедра прикладної радіоелектроніки

«На правах рукопису»
УДК 004.55

До захисту допущено:
В.о. зав. кафедри

Андрій МОВЧАНЮК
«___»_____20__ р.

Магістерська дисертація

на здобуття ступеня магістра
за освітньо-професійною програмою «Інтелектуальні технології радіоелектронної техніки»
за спеціальністю 172 «Телекомунікації та радіотехніка»
на тему: «Виявлення патологій на зображеннях методами цифрового оброблення сигналів»

Виконав (-ла):
студент (-ка) 2 курсу, групи РЕ-21мп
Жирова Анжела Ігорівна

Керівник:
доцент, к.т.н.

Лащевська Наталія Олександрівна _____

Рецензент:
Ст. викладач, phD, кафедри РТС
Мирончук Олександр Юрійович _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка)

Київ – 2024 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Радіотехнічний факультет
Кафедра прикладної радіоелектроніки

Рівень вищої освіти – другий (магістерський)

Спеціальність – 172 «Телекомунікації та радіотехніка»

Освітньо-професійна програма «Інтелектуальні технології радіоелектронної техніки»

ЗАТВЕРДЖУЮ
В.о.зав. кафедри
Андрій МОВЧАНЮК
« » _____2023 р.

ЗАВДАННЯ
на магістерську дисертацію студента
Жирової Анжели Ігорівни

1.Тема дисертації «Виявлення патологій на зображеннях методами цифрового оброблення сигналів»
науковий керівник дисертації Лащевська Наталія Олександрівна, доцент, к.т.н.
затверджені наказом по університету від «09» листопада 2023 р. № 5206-с

2. Термін подання студентом дисертації 11 січня 2024 року

3. Об’єкт дослідження: процес виявлення патологій на зображеннях за допомогою нейронних мереж

4. Вихідні дані: набір даних зі знімками магнітно-резонансної томографії з цирозом та без цирозу

5. Перелік завдань, які потрібно розробити: огляд літературних джерел, які мали інформацію про цироз печінки, вибір оптимальної архітектури і програмного забезпечення та реалізація нейронної мережі, навчання і тестування нейронної мережі

6. Орієнтовний перелік графічного (ілюстративного) матеріалу: презентація

7. Орієнтовний перелік публікацій: 1 публікація

9. Дата видачі завдання 01 вересня 2023 року

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Отримання теми магістерської дисертації	05.09.2023р.	
2	Розробка плану магістерської дисертації	10.09. 2023р.	
3	Початок збору інформації для дослідження	24.09.2023р.	
4	Вибір архітектури нейронної мережі	10.10.2023р.	
5	Вибір програмного забезпечення	24.10.2023р.	
6	Реалізація та навчання нейронної мережі	05.11.2023р.	
7	Тестування нейронної мережі	15.12.2024р.	
8	Оформлення магістерської дисертації	09.01.2024р.	

Студент  Анжела Жирова

Науковий керівник  Наталія Лащевська

АНОТАЦІЯ

Магістерська робота складається з 52 сторінок пояснювальної записки, яка містить 19 ілюстрацій, 10 таблиць, 1 додаток і 9 джерел посилань.

Актуальність теми: підвищення точності діагностування цирозу печінки шляхом застосування нейромережевого підходу.

Метою дослідження є розробка архітектури нейронної мережі та її навчання для виявлення патологій на зображеннях методами цифрового оброблення сигналів.

У результаті роботи було отримано нейронну мережу, яка з високою точністю у визначає цироз печінки та знімках магнітно-резонансної томографії.

Перелік ключових слів: цироз печінки, згорткові нейронні мережі, машинне навчання, класифікація зображень, трансферне навчання.

ANNOTATION

The master's thesis consists of 52 pages of explanatory note, which contains 19 illustrations, 10 tables, 1 appendix and 9 bibliographic references.

Relevance of the topic: improving the accuracy of liver cirrhosis diagnosis by using a neural network approach.

The aim of the research is to develop the architecture of a neural network and train it to detect pathologies in images using digital signal processing methods.

As a result of the work, a neural network was obtained that detects liver cirrhosis in magnetic resonance imaging images with high accuracy.

Key words: liver cirrhosis, convolutional neural networks, machine learning, image classification, transfer learning.

Пояснювальна записка
до дипломної роботи
на тему: «Виявлення патологій на зображеннях методами цифрового оброблення сигналів»

Київ — 2024 року

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

МРТ — Магнітно-резонансна томографія;

ВСТУП

Актуальність дослідження. Пошук шляхів впровадження нових і вдосконалення існуючих методів діагностування завжди залишається актуальним для медичної галузі. Адже саме від точності та своєчасності їхніх результатів залежить чи буде подальше лікування вдалим і чи виникнуть в його процесі ускладнення.

Використання нейронних мереж – це один з доволі перспективних методів діагностування, який наразі все ще малодосліджений, а отже, потребує подальшого аналізу. Цей метод дозволяє підвищити ефективність та швидкість обробки значних обсягів даних, уникнути використання не зовсім безпечних інвазійних методів діагностування, а також дозволяє забезпечити кращу точність та, як наслідок, правильність встановлення діагнозу, автоматизувати діагностику, збільшити її доступність тощо.

Патології, зокрема цироз печінки, можуть мати слабко виражену симптоматику. При цьому саме своєчасність їх діагностики часто відіграє вирішальну роль в питанні досягнення пацієнтом одужання.

Тому в даній роботі було поставлено за мету зосередитись на пошуку кращої методики в області цифрового оброблення зображення для діагностування цирозу печінки на ранніх стадіях.

При відсутності своєчасного лікування цього захворювання можуть виникнути серйозні ускладнення. Цілком можливий летальний кінцевий результат.

В групі ризику знаходяться люди, що хворі на аутоімунні захворювання, вірусні гепатити, мають хронічні захворювання печінки. А також ті, хто зловживають алкоголем чи неправильним харчуванням. Інколи навіть вживання деяких лікарських засобів може призвести до ушкодження печінки. Тобто цироз — це захворювання, яке вражає різні групи населення.

Отже, дослідження є актуальним тому, що впровадження нейронних мереж, як методу діагностування цирозу печінки, має дозволити збільшити шанси на збереження життя і здоров’я пацієнтів. Функціонал розробленої моделі можна надалі розширити для виявлення й інших патологій, що дозволило б прискорити процес

навчання та зменшити обсяги даних, які потрібні для досягнення необхідної точності.

Мета дослідження. Метою магістерської дисертації є розробка архітектури нейронної мережі та її навчання для виявлення патологій на зображеннях методами цифрового оброблення сигналів; розробка ефективного діагностичного інструменту, який міг би підвищити точність і швидкість виявлення патологій.

Об'єкт дослідження — процес виявлення патологій на зображеннях за допомогою нейронних мереж.

Предмет дослідження — методи виявлення патологій на зображеннях.

Результати роботи можна використати у подальших дослідженнях чи у медичних закладах в якості допоміжного діагностичного інструменту.

1 АНАЛІЗ ЗАДАЧІ

Цироз виникає внаслідок довгострокового пошкодження печінки. Він є незворотнім захворюванням, лікування якого зводиться до роботи над послабленням симптомів та сповільненням подальших деструктивних процесів.

Розглянемо особливості та методи його виявлення більш детально.

1.1 Визначення цирозу печінки

Цироз – стан, що характеризується змінами у структурі печінки. При цьому відбувається заміщення печінкових часток сполучною тканиною з утворенням циротичних вузликів [1]. Як наслідок виникає дисфункція даної залози.

Причинами цієї патології є хронічні запальні процеси. Розглянемо їх нижче.

На рис. 1.1 продемонстровано зображення здорової печінки та печінки з цирозом.

Рисунок 1.1 — Здорова та хвора печінки

Першочерговими ознаками наявності цирозу можуть бути такі характеристики як, наприклад, зміна розмірів печінки (зменшення чи збільшення), деформація її форми, утворення ущільнень. Через некроз тканин втрачаються функціонуючі клітини печінки.

Також погіршується або й зовсім втрачається здатність печінки виконувати свої функції, серед яких обробка токсинів, синтез і зберігання глюкози, білків, метаболізм жовчних кислот і холестерину, гормональний метаболізм, обмін ліпідів і секреція білка плазми.

Ворітна вена та печінкова артерія — два джерела кровопостачання печінки.

Печінкова артерія переносить насичену киснем кров у печінку. Тоді як кров, яка надходить із травної системи, потрапляє в печінку через ворітну вену. При цьому вона переносить матеріали, які призначені для фільтрації печінкою: білки, поживні речовини, ліки або токсини.

Рубцева тканина, яка утворюється при цирозі, блокує кров, яка має проходити через ворітні вену. Це спричиняє підвищення тиску. Оскільки кров з травної системи не може потрапити у печінку, фільтрація поживних речовин та токсинів не відбувається належним чином.

Тоді як підвищення тиску, яке при цьому відбувається, може стати причиною варикозного розширення вен, гепаторенального синдрому тощо.

У свідомості людей цироз асоціюється із хворобами, які спричинені спадковістю або алкоголем, але є багато факторів, які можуть стати його причиною. З-поміж них можна виділити наступні:

– вірусні гепатити;

– аутоімунні захворювання;

– неалкогольні жирові хвороби печінки;

– інфекції;

– лікарська інтоксикація.

Отже, хоча до групи ризику входять люди зі спадковими хворобами та схильностями до надмірного вживання алкоголю, але є значний перелік причин виникнення та прогресування цирозу печінки.

Варто зазначити, що усі хронічні захворювання печінки здатні прогресувати до цирозу. Тоді як його ускладнення, у свою чергу, можуть призвести до гострого ураження нирок, варикозних кровотеч, асциту. Пацієнти з цирозом печінки зазвичай мають підвищений ризик розвитку різноманітних інфекцій (пневмонії, бактеріємії тощо).

Хоча фіброз оборотний, але у якийсь момент пошкодження печінки стануть надто серйозними і тоді печінка вже не зможе відновитися. Хворі на цироз мають значний ризик розвитку раку печінки.

У сучасних реаліях жодна медична методика не може забезпечити повне одужання від цирозу.

Деякі рубці зменшуються у наслідок якісно та своєчасного лікування основних причин, хвороби. Це також може допомогти уникнути виникнення печінкової недостатності для лікування якої була б необхідна трансплантація печінки.

Проте на даний момент не існує жодних чітких стратегій, які б зупинили або повернули назад патологічне прогресування хронічного захворювання печінки.

При цьому, як уже зазначалося раніше, на самому початку цироз має слабко виражену симптоматику, яку можна сплутати із ознаками інших хвороб або зі звичайною втомою.

Так до симптомів цирозу печінки можна віднести:

– швидку втомлюваність;

– втрату апетиту;

– порушення сну та концентрації;

– відчуття гіркоти у роті;

– втрату ваги;

– зміну кольору шкіри;

– здуття;

– висипи на шкірі;

– набряки ніг чи рук;

– слабкість;

– дискомфорт чи навіть біль в області правого підребер’я, що може свідчити про розширення печінки;

– збільшення селезінки;

– печінкова енцефалопатія.

Пацієнти з цирозом печінки також можуть відчувати такі симптом, як:

– зміни настрою;

– апатію;

– сплутаність свідомості;

– депресію.

Звідси слідує, що симптоми цирозу доволі неспецифічні на ранніх стадіях, що ускладнює його діагностування.

1.2 Поширеність захворювання

Цироз печінки є серйозною глобальною проблемою. Щороку близько одного мільйону смертей відбувається через ускладнення цирозу печінки, що робить цю хворобу одинадцятою серед найпоширеніших причин смертності в усьому світі [2].

За дослідженнями, опублікованим Фронтом охорони здоров’я у 2022 році, у всьому світі смертність від цирозу зросла на 47,15% з 1990 по 2017 рік [3].

Саме від своєчасності виявлення цирозу буде залежати успішність його лікування. Проте симптоматика цього захворювання може залишатися слабко вираженою впродовж значного часу. Це значною мірою ускладнює діагностування хвороби і, як наслідок, призводить до такої високої статистики смертності.

1.3 Аналіз існуючих методів обстеження

Для визначення стану тканин печінки використовують різні методи діагностики. Серед них комп’ютерна томографія, магнітно-резонансна томографія, ультразвукове обстеження печінки.

При комп’ютерній томографії для отримання зображень печінки використовуються рентгенівські промені. Приклад такого зображення подано на рис. 1.2.

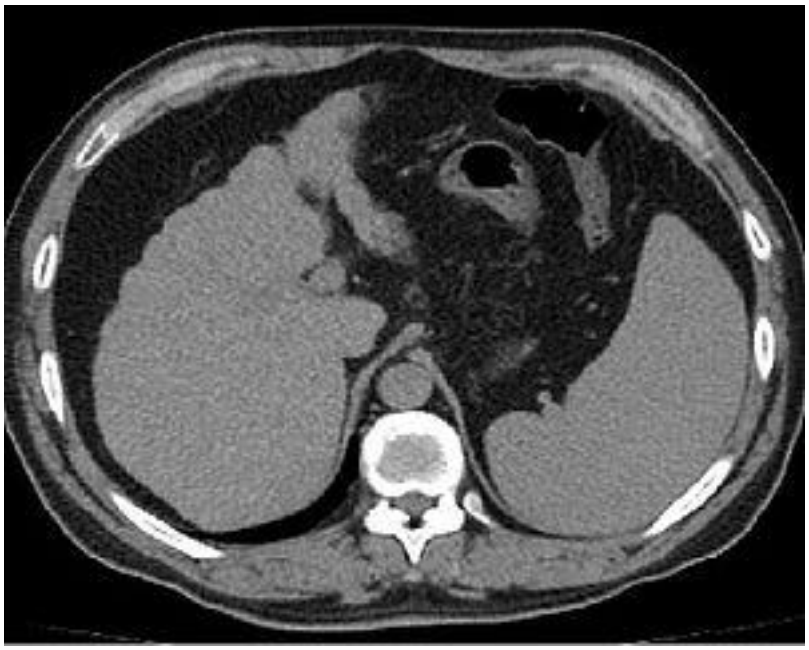

Рисунок 1.2 — Ознаки цирозу печінки на знімку комп’ютерної томографії

При магнітно-резонансній томографії (МРТ) використовуються магнітні поля і радіочастотні хвилі. Приклад зображення отриманого цим методом подано на рис. 1.3.

Рисунок 1.3 — Знімок магнітно-резонансної томографії

При ультразвуковому обстеженні використовуються високочастотні звукові хвилі. Цей метод найдешевший з наведених. Проте знімки отримані за допомогою комп’ютерної та магнітно-резонансної томографії забезпечують більш детальну інформацію про структуру та ураження печінки ніж знімки отримані з ультразвукових досліджень. Вони краще підходять для виявлення цирозу на ранніх стадіях.

Варто зазначити, що якість знімків отриманих при магнітно-резонансній томографії печінки трохи вища, ніж при комп’ютерній. Також цей метод обстеження менш шкідливий для здоров’я пацієнта, адже не вимагає використання рентгенівське опромінення.

Також для точної діагностики може бути необхідна біопсію печінки. Це процедура за якої видаляється і досліджується невеликий фрагмент тканини, що і демонструє рис. 1.4.

Рисунок 1.4 — Біопсія печінки

Цей метод інвазійний та супроводжується ризиком ускладнень, але його результати більш точні ніж ті, які отримуються ультразвуковим дослідженням або комп'ютерною томографією [4].

Порівнюючи перелічені методи обстеження, можна зробити висновок про те, що саме використання знімків МРТ може бути найкращим варіантом для якісного діагностування цирозу печінки за медичними знімками.

1.4 Ознаки цирозу на знімках МРТ

Візуально на МРТ знімках цироз печінки можна помітити якщо звертати увагу на наявність вузликів та рубців, відомих як фіброз.

Рис. 1.5 показує дифузні вузлики з виразними більшими вузликами, які гіперінтенсивні до фонової паренхіми (виділено стрілками) [5].

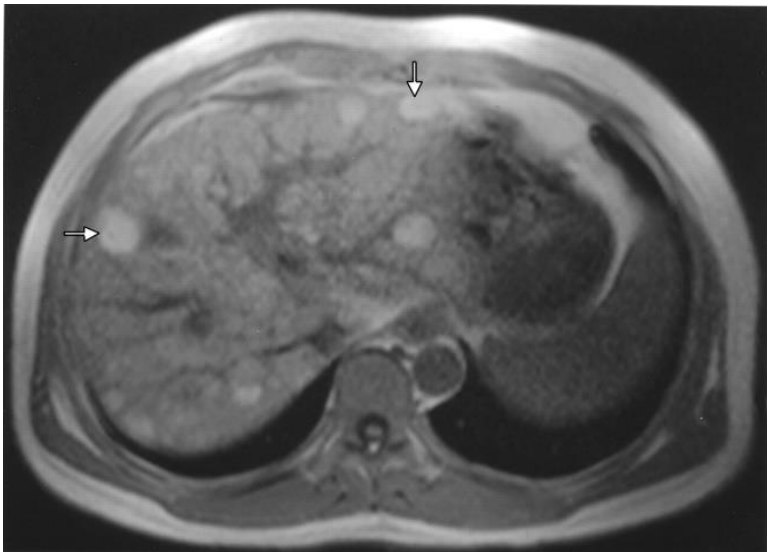

Рисунок 1.5 — МРТ знімок печінки з цирозом

Уражена цим захворюванням печінка також може мати нерівну, неправильну форму та інші відхилення від норми.

1.5 Постановка задачі

Медичні знімки, які отримані з використанням комп’ютерної та магнітно-резонансної томографії, можуть дозволити виявити цироз печінки ще на ранніх стадіях.

Проте діагностування без використання нейронних мереж, як допоміжних інструментів, має певні недоліки:

– нижча швидкість обробки даних;

– суб’єктивність оцінювання через яку різні лікарі можуть давати різні інтерпретації одних і тих самих знімків, адже кожен з них при оцінюванні керується саме власним досвідом та власними знаннями;

– помилки через неуважність, яка може бути спричинена втомою чи надмірною завантаженістю медичних працівників;

– обмеженість часу для ретельного аналізу кожного знімку.

Деякі досвідчені спеціалісти можуть бути недоступними через виїзд закордон або наслідки небезпек, що збільшує навантаження на працівників медичної сфери, сповільняє процес діагностування, призводить до залучення менш досвідчених спеціалістів. У місцях, що наближені до місць бойових дій або у воєнних шпиталях безпосередня консультація з необхідним спеціалістами може виявитися тимчасово неможливою [4].

Тому для забезпечення високої ефективності та точності виявлення цирозу доцільно скористатися допоміжними інструментами.

З цією метою в роботі була поставлена задача – запропонувати таку архітектуру нейронної мережі, яка здатна до швидкого навчання та самонавчання для виявлення цирозу на ранніх стадіях.

У цьому розділі було розглянуто особливості цирозу печінки та методи обстеження, які дозволяють його визначити за знімками. Їхній аналіз дозволяє обрати МРТ знімки печінки у якості набору даних на якому проводитиметься навчання нейронної мережі.

2 ОГЛЯД АРХІТЕКТУР НЕЙРОННИХ МЕРЕЖ

Нейронні мережі успішно виконують розпізнавання облич та певних об’єктів на знімках, а також сегментацію, класифікацію та інші завдання, що пов’язані із обробкою зображень.

Проте для того, щоб результати роботи мережі були точними у контексті конкретного завдання, варто обрати такі архітектуру і метод реалізації аналізу знімків, які будуть оптимальними саме для цього завдання.

Так якщо за задумом в процесі роботи нейронна мережа має стискати та реконструювати аудіодані, то зазвичай у такому випадку обирається такий тип нейронної мережі, як автокодер. Цей тип нейронних мереж зазвичай вдало використовується для завдань, що пов’язані з узагальненням та аналізом звукових функцій. Але його використання не є оптимальним у контексті обробки медичних знімків.

Згорткові архітектури найчастіше застосовуються для виконання таких завдань, як розпізнавання об’єктів та певних особливостей на зображеннях.

Серед найпоширеніших архітектур нейронних мереж можна виділити:

– рекурентні мережі;

– автокодувальники;

– генеративні адверсаріальні мережі;

– мережі трансформери;

– згорткові мережі;

– нейронні мережі прямого поширення.

Розглянемо кожну з перелічених архітектур детальніше.

2.1 Рекурентні мережі

Рекурентні нейромережі розроблені для роботи з послідовними даними, такими як текст або часові ряди. Вони мають зворотні зв’язки, які зберігають інформацію з попередніх кроків. Це дозволяє їм краще працювати з даними, де контекст важливий [6].

Рекурентні мережі мають свої переваги та недоліки.

Серед плюсів рекурентних мереж виділяємо такі пункти:

– здатність до роботи з послідовними даними;

– наявність внутрішньої пам’яті;

– здатність до розпізнавання залежності між елементами послідовності, контексту, врахування структури даних;

– підходять для генерації послідовностей;

– здатність здійснювати роботу з послідовностями будь-якої довжини.

Недоліками рекурентних мереж є:

– проблема зникання градієнту;

– повільне навчання;

– високі обчислювальні витрати;

– обмежена пам’ять, що може призвести до втрати інформації про попередні стани, залежності.

Рекурентні нейронні мережі здебільшого використовуються для виконання завдань машинного перекладу, аналізу тексту тощо. Вони розроблені спеціально для обробки і аналізу послідовних даних, генерації нових послідовностей. Такий тип мереж має перевагу у вирішенні задач в яких інформація залежить від контексту та має послідовну природу.

2.2 Автокодувальники

Автокодувальники використовують для зменшення розмірності та виділення ознак у вхідних даних.

Серед їхніх плюсів можна виділити:

– дозволяють здійснювати стиснення даних;

– можуть використовуватися для боротьби з шумом, а також для відновлення даних, виділення важливих ознак, генерації нових, подібних до вхідних, даних;

– можуть використовуватися для рекомендаційних систем, що аналізували б вибірки даних та за ними складали певні рекомендації, які відповідали б інтересам користувачів.

Недоліками автокодувальників є:

– вразливість до перенавчання;

– високі обчислювальні витрати;

– обмеженість у декодуванні нових даних;

– для ефективного навчання їм необхідні великі обсяги даних, адже інакше може виникнути проблема перенавчання;

Автокодувальники можуть використовуватися для розв’язання задач пошуку ліків, стиснення зображень, виявлення аномалій тощо.

2.3 Генеративні адверсаріальні мережі

Генеративні адверсаріальні мережі складаються з двох мереж, генератора і дискримінатора. Вони використовуються для генерації нових зображень або даних.

Такі мережі мають наступні плюси:

– генерування реалістичного контенту (наприклад, звуку, зображень, тексту);

– новий контент є результатом складних операцій, а не простим повторенням вхідних даних;

– деякі варіанти цих мереж мають семантичне розуміння;

– здатність до навчання на різних рівнях складності;

– вони можуть використовуватися для удосконалення існуючих даних.

Недоліки генеративних адверсаріальних мереж:

– ризик перенавчання;

– для ефективного навчання вони зазвичай потребують великі обсяги даних;

– можливість виникнення в згенерованому контенті розмитості, нерівномірності генерації тощо;

– високі обчислювальні витрати;

– можуть викликати етичні питання та питання про авторське право.

Генеративні адверсаріальні мережі можуть використовуватися для синтезу текстів, покращення якості медичних зображень, створення ринкових прогнозів, відновлення зображень з космосу тощо.

2.4 Мережі трансформери

Трансформери використовуються для обробки послідовностей даних, що робить їх подібними до вищезгаданих рекурентних мереж. Проте трансформери можуть обробляти дані і не послідовно.

Вони мають наступні плюси:

– дана архітектура дозволяє паралельну обробку великих обсягів даних;

– здатність до моделюваннях довгих залежностей;

– здатність до семантичного розуміння тексту;

– можуть використовуватися для широкого спектру завдань.

Недоліки таких нейронних мереж:

– працюють на рівні векторних проекцій без чіткого розуміння змісту даних;

– для ефективного навчання їм зазвичай необхідні великі обсяги даних;

– високі обчислювальні витрати;

– обмежена інтерпретованість.

Трансформатор — це нейронна мережа, яка вивчає контекст і, отже, значення, відстежуючи такі зв’язки в послідовних даних, як-от слова в цьому реченні [7].

2.5 Згорткові мережі

Згорткові нейронні мережі використовуються для обробки зображень і відео. Вони мають спеціальні шари, які допомагають виявляти патерни і особливості в зображеннях.

Серед їхніх плюсів можна визначити наступні:

– здатність до виявляння локальних особливостей, об’єктів;

– здатність до передбачення положення об’єктів на зображеннях;

– ефективність в умовах обмежених ресурсів;

– розпізнання згортковими шарами об’єктів незалежно від їхнього положення на зображеннях;

– можливість забезпечити роботу в реальному часі.

Недоліки таких мереж:

– мало підходять для роботи зі структурованими даними;

– чутливі до зміни чи видалення зображень;

– для ефективного навчання вони зазвичай потребують великі обсяги даних;

– високі обчислювальні витрати.

Використання згорткових мереж більш орієнтоване на роботу з візуальними даними. Тому вони підходять для обробки медичних знімків, але гірше справляються з текстами чи звуками.

2.6 Нейронні мережі прямого поширення

Модель прямого зв’язку є найпростішою формою нейронної мережі, оскільки в ній інформація обробляється лише в одному напрямку. Хоча дані можуть проходити через кілька прихованих вузлів, вони завжди рухаються в одному напрямку і ніколи у зворотному напрямку [8].

Плюси мереж прямого поширення:

– відносна простота і легкість реалізації;

– висока швидкість інференсу;

– здатність до паралельних обчислень;

– здатність до роботи з різними типами даних.

Нейронні мережі з прямим типом поширення цілком ефективні для вирішення простих завдань. Проте такі мережі мають деякі недоліки:

– певна обмеженість у роботі з послідовними даними;

– ризик перенавчання;

– неефективність при роботі з послідовностями різної довжини;

– недостатня здатність до знаходження складних залежностей;

– можливість втрати інформації чи контексту.

2.7 Вибір архітектури

Враховуючі усі наведені факти і перелічені недоліки та переваги можна зробити виважений вибір мережі для роботи. Для неї було обрано згорткові нейронні мережі, які добре підходять для виконання завдання. Їхню базову архітектуру наведено на рис. 2.1.

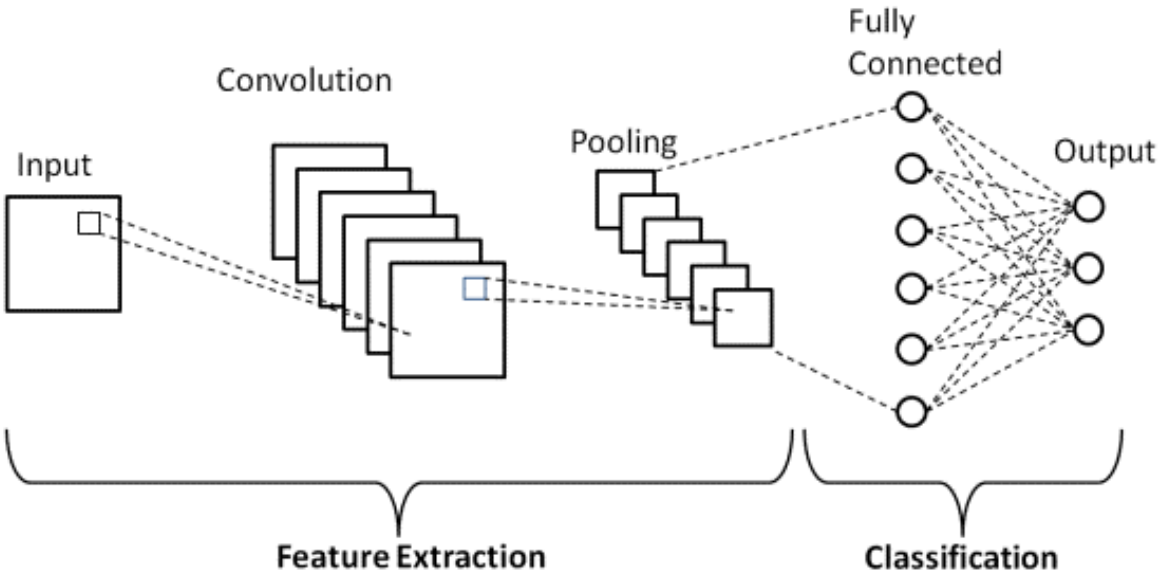

Рисунок 2.1 — Базова структура архітектури згорткової нейронної мережі

Вхідний шар (Input) є початковим шаром, на який подаються вхідні дані.

Згорткові шари (Convolution) використовують фільтри для виявлення різних ознак у вхідних даних. Зазвичай вони виконують операцію згортки.

Шари пулінгу (Pooling) застосовуються для зменшення розмірності вихідних даних, які надходять зі згорткових шарів.

Повністю зв’язані шари (Fully Connected) використовуються для класифікації чи регресії.

Вихідний шар (Output) представляє власне результат роботи нейронної мережі.

Вилучення ознак (Feature Extraction) відбувається у згорткових шарах та шарах пулінгу.

Класифікація (Classification) відбувається у повністю зв’язаних та вихідних шарах, де така нейронна мережа класифікує або ж робить прогнози на основі вилучених ознак.

Згорткові нейронні мережі добре справляються з виділенням особливостей на зображеннях й при правильній розробці та навчанні мають ефективно виконувати задачу розпізнавання патологій на медичних зображеннях.

Зосередившись на поставленому завданні роботи та проаналізувавши потенційні вимоги для його виконання можна дійти до висновку щодо доцільності використання нейронних мереж згорткового типу.

Згорткові нейронні мережі добре справляються з виділенням особливостей на зображеннях. Проте для вдалого розв’язання обраної у магістерській роботі задачі важливо підібрати відповідну архітектуру.

Серед популярних архітектур згорткових нейронних мереж, які можуть підійти для розв’язання поставленої у роботі задачі, можна виділити такі:

– ResNet;

– VGG;

– Inception;

– SqueezeNet;

– DenseNet.

Всі вони у певній мірі підходять для здійснення розпізнавання патологій на медичних зображеннях. Кожна з цих архітектур була розглянута детальніше.

Мета навчання нейронної мережі полягає у мінімізації функції втрат — функції, яка оцінює різницю між фактичним і прогнозованим вихідними значеннями при роботі з навчальними даними. Для оновлення ваг нейронної мережі за допомогою алгоритмів оптимізації використовуються градієнти. Вони вказують напрямок

та швидкість, з якою функція змінюється вздовж кожного параметра моделі. Тобто як саме необхідно змінювати ваги мережі для мінімізування функції.

У випадку, коли градієнти стають надто малими, ваги майже не оновлюватимуться, що, як наслідок, призведе до значного погіршення або й припинення навчання. Проте вибір архітектури ResNet вирішує цю проблему.

ResNet відома своєю інноваційною структурою. Вона працює шляхом додавання залишкових з’єднань, що допомагає навчати нейронну мережу на значній кількості шарів, уникаючи зникання градієнтів. Вона здатна диференціювати об’єкти різних розмірів та форм. Ця архітектура добре підходить для роботи з медичним знімками через її здатність до виявлення навіть невеликих деталей на зображеннях.

VGG має глибоку структуру та використовує невеликий фільтр розмірами 3x3 і велику кількість фільтрів для обробки зображень. Ці нейронні мережі добре підходить для класифікації зображень і детекції об’єктів. Проте VGG також мають високий рівень абстракції через наявність великої кількості шарів згортки та повнозв'язних шарів. Це означає, що вони вирізняють загальні ознаки та шаблони на зображеннях, але можуть втратити деякі деталі та контекстну інформацію, яка завжди важлива для медичних зображень.

Inception використовує модуль, який включає згорткові фільтри різних розмірів і об’єднує їх в одному шарі, що дозволяє ефективно розв’язувати завдання виділення об’єктів різних форм та масштабів.

Проте у порівнянні з ResNet ця архітектура вимагає більше пам’яті, все ще стикається із зникненням градієнту та має складнішу структуру.

SqueezeNet відома своєю компактною архітектурою. Вона може бути ефективною при умовах обмеженості ресурсів. Проте ця архітектура зазвичай поступається ResNet, коли справа стосується необхідності у отриманні глибокого контексту із зображень та роботи над великими обсягами даних.

Всі шари мережі DenseNet пов’язані між собою. Ця архітектура може бути корисною для роботи з медичними зображеннями. Проте ResNet має перспективи досягти вищої точності класифікації, ніж як DenseNet.

З урахуванням вищезазначених фактів, можна зробити висновок, що архітектура ResNet має бути вдалим вибором для подальшої роботи. Вона ефективна в глибокому навчанні та може виділяти навіть невеликі деталі на зображеннях. ResNet використовує конструкцію, яка дозволяє запобігати виникнення згасання градієнтів. Це особливо корисно для навчання на малих наборах даних при якому градієнти можуть бути слабкими.

Для вирішення проблеми втрати градієнту в архітектурі ResNet використовується конструкція skip connections. Вона дозволяє обчислювати вихідні дані, використовуючи як вхідні дані, так і вихідні дані попереднього блоку. Це використання інформації з попередніх блоків для обчислення поточних вихідних даних і допомагає уникати втрати градієнтів.

Так рис. 2.2 демонструє принцип роботи блоків, які є основою ResNet архітектури.

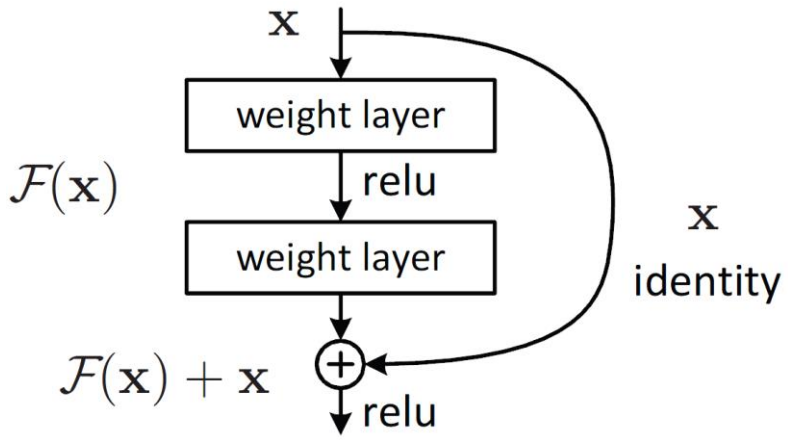

Рисунок 2.2 — Залишковий блок

На цьому рисунку x — це вхідні дані базового блоку.

F(x) — вихідні дані першого згорткового шару.

F(x)+x — вихідні дані шару перехідного зв'язку.

Шар weight layer — ваговий шар.

Функція relu — це функція активації, яка обчислює вихідне значення як максимальне значення між вхідним значенням та нулем.

Функція identity — це функція активації, яка повертає вхідне значення.

Рис. 2.2 демонструє, як два згорткові шари, розділені шаром перехідного зв'язку, можуть бути об'єднані в один блок за допомогою конструкції skip connections.

У цьому випадку шар перехідного зв'язку виконує операцію добутку. Це означає, що вихідні дані першого згорткового шару множаться на параметри шару перехідного зв'язку.

Сума вихідних даних першого згорткового шару і вихідних даних шару перехідного зв'язку використовується як вихідні дані базового блоку.

Розглянемо конкретний приклад реалізації такої структури. На рис. 2.3 подана архітектура однієї з найпопулярніших моделей ResNet архітектури — ResNet-50.

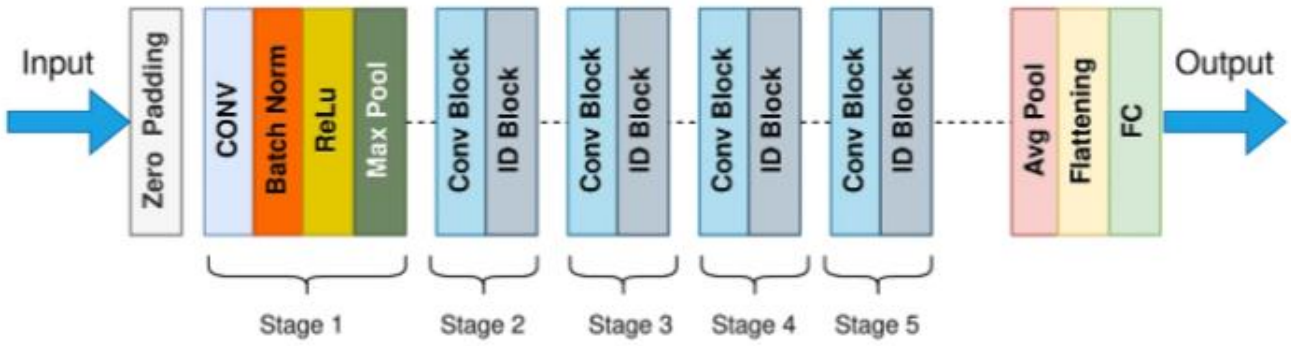

Рисунок 2.3 — Архітектура моделі ResNet-50

ResNet-50 складається з 50 блоків ResNet. Тоді як кожен з них має два згорткові шари за якими слідує residual connection — лінійний шар, який об’єднує вхідні дані з виходами попереднього шару.

На рисунку також показано, що архітектура ResNet-50 містить два додаткових шари: шар глобального середнього значення (avg pool) і шар класифікатора (FC).

Шар глобального середнього значення отримує вихід з останнього блоку ResNet і обчислює глобальне середнє значення, яке далі передається на шар класифікатора.

З виходу шару класифікатора отримується оцінка того, до якого класу належить вхідне зображення.

У даному розділі було проведено аналіз найпопулярніших архітектур. Виділено їхні плюси та недоліки, а також типові задачі для яких вони зазвичай використовуються. Для подальшої реалізації була обрана згорткова ResNet архітектура.

У розділі також було розглянуто особливості структури та принцип роботи блоків такої архітектури.

3 ОГЛЯД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Наступним етапом реалізації нейронної мережі є вибір програмного забезпечення. Саме з його використанням розроблятиметься задана згорткова нейронна мережа.

Існує багато засобів реалізації нейронних мереж. Серед прикладів найбільш відомими є C++, Matlab, Python, JavaScript, Java, R, Go та інші.

Після короткого ознайомлення з особливостями найпопулярніших варіантів, вибір був звужений до необхідності обрати між C++, Matlab, Pyhon та Java. Обидва варіанти безкоштовні, відносно зручні для використання і мають певні переваги та недоліки. Для остаточного вибору варто розглянути їх детальніше.

3.1 Matlab

Matlab має низку переваг:

– власні інструменти для реалізації нейронних мереж (наприклад, Deep Learning Toolbox);

– підтримка графічних процесорів;

– інтуїтивний, зручний і легкий у використанні інтерфейс;

– наявність вбудованої підтримки даних, що дозволяє легко імпортувати та обробляти дані;

– доступ до попередньо навчених моделей;

– велика кількість вбудованих інструментів для обробки сигналів та зображень.

Серед мінусів можна виділити:

– обмеженішу підтримку для глибокого навчання;

– функціонал безкоштовної версії значно обмежений.

Matlab є програмним забезпеченням, яке добре підходить для реалізації і навчання згорткових нейронних мереж. Особливо для тих, які мають обробляти зображення.

Одним із інструментів для такої реалізації в Matlab є Deep Learning Toolbox, який спрощує реалізацію нейронної мережі, що забезпечує його більш вдале використання навіть тими користувачами, які мають обмежений досвід в програмуванні.

Deep Learning Toolbox дозволяє створювати та навчати нейронні мережі, а також, що особливо зручно, здійснювати певну візуалізацію. Так з допомогою Deep Learning Toolbox можна візуалізувати архітектуру мережі, процес навчання, фільтри, результати тощо.

Проте безкоштовна версія Matlab, як вже було зазначено, має обмежений функціонал і не є найкращим вибором для реалізації архітектури ResNet.

3.2 Python

На даний момент Python серед найпопулярніших мов програмування для реалізації нейронних мереж. Серед плюсів Python:

– велика кількість бібліотек (TensorFlow, PyTorch, Keras та інші), які спрощують створення і навчання нейронної мережі;

– відкритий код;

– зручний синтаксис;

– широкий спектр застосування;

– підтримка графічних процесорів;

– доступність навчальних матеріалів та прикладів, які можна вільно знайти у мережі Інтернет;

– зручний інтерфейс;

– наявність бібліотек, що допомагають у обробці зображень;

– зручний для використання трансферного підходу ;

– код, що написано на Python, може успішно використовуватися на будь-якій платформі з тих, які підтримують цю мову;

– інтеграція з іншими мовами та бібліотеками.

При цьому Python має такі недоліки:

– вимагає великих ресурсів;

– обробка зображень вимагає використання додаткових бібліотек.

Python належить широкий спектр інструментів та ресурсів, які доступні для цієї мови.

3.3 Java

Java є достатньо гнучкою мовою програмування, яка має наступні плюси:

– наявність великої кількості бібліотек;

– широкий вибір інструментів для розробки;

– підтримка паралельних обчислень та багатопоточності;

– код, що написано на Java, може успішно використовуватися на будь-якій платформі з тих, які підтримують цю мову;

Серед мінусів використання Java можна виділити:

– складність мови;

– якщо проводити порівняння з Python, то вона має меншу кількість бібліотек;

– її відносна повільність для виконання деяких типів нейронних мереж.

Використання Java вимагає певного рівня попередніх знань з програмування. В порівняння з мовою Python вона доволі складна для вивчення початківцями.

3.4 C++

C++ це високопродуктивна мова, яка є доволі популярним вибором при розробці нейронних мереж. Плюси використання C++ такі:

– гнучкість;

– це потужна мова, яка здатна підтримувати складні алгоритми;

– код написаний на C++, може виконуватися на будь-яких платформах, які підтримують цю мову програмування;

– велика кількість бібліотек.

Тоді як серед недоліків використання C++ можна виділити її складність та те, що вона поступається Python, коли справа стосується кількості інструментів та деяких інших особливостей.

3.5 Вибір програмного забезпечення

Як засіб подальшої реалізації нейронної мережі було обрано Python. Це зумовили такі особливості, як зручність використання, безкоштовність та наявність великої кількості бібліотек.

Розробка моделі нейронної мережі проводитиметься у Google Colab, що дозволить працювати з Python безпосередньо у браузері. Робота у цьому середовищі може здійснюватися з будь-якого пристрою, який підключений до мережі Інтернет.

GoogleColab надає безкоштовний доступ до потужних обчислювальних ресурсів. Він має простий інтерфейс та забезпечує великий вибір між попередньо встановленими бібліотеками та інструментами машинного навчання.

Запис коду в окремих комірках дозволяє не запускати його кожен раз повністю, а працювати з необхідними блоками окремо. Це призводить до економії часу при виконанні кодування.

Проте GoogleColab також має і недоліки. Обмеженість його оперативної пам’яті має бути врахована при виборі шляхів реалізації нейронної мережі та при підготовці набору даних, який на неї подаватиметься.

У ході роботи було обрано наступні бібліотеки:

– TensorFlow, яка використовується для створення, навчання і подальшого використання різних моделей;

– NumPy, яка спрямована на операції з багатовимірними масивами і володіє значним набором функцій та інструментів, які можна використовувати для ефективної обробки числових даних;

– Matplotlib, яка використовується для здійснення візуалізації даних, що включає підтримку різних типів графічних зображень (графіків, діаграм і т.д.), а також надає можливість проводити налаштування їхніх параметрів;

– SciPy та Scikit-image — бібліотеки для обробки зображень;

Бібліотека TensorFlow має вбудований Keras — високорівневий інтерфейс, що використовується для розробки та навчання нейронних мереж. Він надає широкий вибір шарів і оптимізаторів та дозволяє здійснювати швидке навчання моделі нейронної мережі.

У цьому розділі було розглянуто види програмного забезпечення, яка може використовуватися для реалізації нейронних мереж. Вибір було зупинено на мові програмування Python та програмному середовищі Google Colab, які вигідно виділяються зручністю використання, доступністю та наявністю великої кількості необхідних бібліотек, інструментів та особливостей, які стануть у нагоді при подальшій роботі.

4 РЕАЛІЗАЦІЯ НЕЙРОННОЇ МЕРЕЖІ

Реалізація моделі нейронної мережі передбачає наступні етапи:

– Підготовка даних, яка включає завантаження та обробку;

– Реалізація необхідної архітектури;

– Навчання нейронної мережі;

– Тестування нейронної мережі.

4.1 Підготовка даних

Набір даних для дослідження сформовано з набору даних Duke Liver Dataset [9]. Для цього з нього було відібрано МРТ знімки з цирозом та без. Кількість знімків обмежена деякою наповненням оригінального Duke набору і особливостями використання безкоштовної версії Google Colab.

Оскільки нейронна мережа має визначати цироз, інформація про наявність чи відсутність цього захворювання буде використовуватися у якості міток. Тому набір даних для цієї нейронної мережі складається з двох класів: цироз (Cirrhosis) та нормальний (Normal). Відповідні каталоги cirrhosis та normal разом складають навчальний набір. Він збалансований, оскільки обидва каталоги містять однакову кількість медичних знімків.

Приклад зображення із каталогу normal подано на рис. 4.1.

Рисунок 4.1 — Здорова печінка

Для остаточного тестування створено окремий каталог test, який було заповнено тестовим набором даних. Фінальна структура набору даних наведена на рис. 4.2.

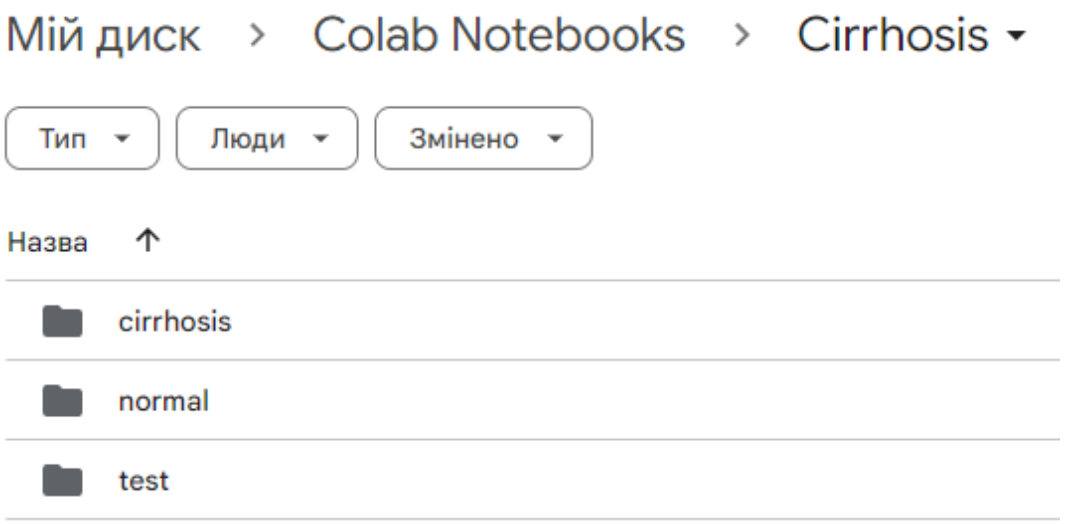

Рисунок 4.2 — Структура набору даних

Даний набір було розміщено на My Driver звідки він був успішно завантажений у середовище виконання Google Colab.

Процес обробки зображень включав зміну розмірності, перетворення у NumPy масиви, нормалізацію значень у діапазоні від 0 до 1. Дані операції були задані безпосередньо у функції завантаження зображень і тому автоматично прикладалися до кожного знімку з каталогів cirrhosis, normal, test. Цю функцію можна побачити на рис. 4.3.

```
# Функція для завантаження зображень
def load_images(filename, target_size=(224, 224)):
    img = load_img(filename, target_size=target_size, interpolation='lanczos')  # Завантаження зображення за заданим filename, зміна розміру на target_size
    img = img_to_array(img)   # Перетворення у NumPy масив
    img = img / 255.0   # Нормалізація значень у діапазоні [0, 1]
    return img
```

Рисунок 4.3 — Функція для завантаження зображень

Для збереження якості знімків при зміні розмірів використовується метод інтерполяції Lanczos. Необхідність у цій зміні виникла через те, що обрані для дослідження ResNet архітектури передбачали, що зображення, які подаються на вхід, мають характеристики 224x224 та кольорові канали RGB.

Перетворення у NumPy масиви дозволяє подати знімки у вигляді тривимірних матриць, виконати нормалізацію та інші операції із знімками. Забезпечує легку інтеграцію даних з бібліотекою TensonFlow.

Вже згадана раніше нормалізація використовується для забезпечення кращої точності, швидкості та стабільності навчання.

Наступним етапом обробки стала аугментація зображень з каталогів cirrhosis та normal. Це дозволило збільшити обсяг навчального набору даних. Аугментація проводилася шляхом застосування зміни кута повороту, яскравості, контрасту та масштабування знімків.

Кожному зображенню із навчального набору була присвоєна мітка, яка відповідає його класу. Так знімки здорової печінки отримали 0, а ті, що з цирозом — 1. Для забезпечення вдалого розрізняння моделлю її класів, було застосовано one-hot кодування міток, яке представляє їх у вигляді векторів, довжина яких дорівнює кількості міток.

Навчальний набір даних було розділено на тренувальний та валідаційний набори. Як слідує з їхньої назви, вони використовуються для тренування та для перевірки (оцінки та налаштування). Код для цієї операції продемонстровано на рис. 4.4.

```
train_n, val_n, y_train_n, y_val_n = train_test_split(t_n, y_train_n, test_size=0.2, random_state=42)
train_c, val_c, y_train_c, y_val_c = train_test_split(t_c, y_train_c, test_size=0.2, random_state=42)
```

Рисунок 4.4 — Поділ на навчальний та валідаційний набори

У наслідок поділу ми отримали 1120 тренувальних та 280 валідаційних знімків.

Для перевірки готовності до використання, здійснено вивід характеристик зображень та міток. На рис. 4.5 подано результат перевірки саме характеристик вхідних даних.

```
Data shape: (224, 224, 3)
Data type: float32
Data range: [0.0, 1.0]
```

Рисунок 4.5 — Фрагмент демонстрації характеристик вхідних даних

З отриманих результатів можна зробити наступні висновки:

– розмірність зображень і відповідних міток збігається;

– дані мають необхідні формат, тип та форму.

Отже, ці дані готові до подальшого використання у моделі нейронної мережі.

4.2 Реалізована модель

Архітектура, що реалізується є модифікацією стандартної ResNet архітектури. Модель має блоки ResNet, згорткові шари, шари пулінгу, пакетну нормаліза-

цію (BatchNormalization), функції активації ReLU, Dropout та завершальний повністю пов'язаний шар. Кожен блок складається з двох згорткових шарів з фільтрами розміром 5x5, за якими слідують пакетна нормалізація та функція активації. Для того, щоб запобігати перенавчання моделі використовується Dropout. Завершальний шар має 2 нейрони та функцію активації sigmoid, яка застосовується для виведення ймовірностей для двох класів.

Головною метою навчання мережі є реалізація оптимальної моделі, яка добре адаптується до даних, які отримує і робить якомога точніші результати. Для оцінки якості навчання використовується функція витрат.

У даному дослідженні застосовувалася бінарна крос-ентропія, яка добре підходить саме для задач бінарної класифікації.

Для мінімізації помилок використовується оптимізатор. Зі списку оптимізаторів, які доступні для бібліотеки TensonFlow, було обрано оптимізатор Adam. Він вирізняється простотою реалізації, загальною ефективність та невибагливістю до пам’яті. Adam самостійно підбирає оптимальну швидкість навчання до кожного параметра окремо, що дещо полегшує процес налаштування моделі.

Для уникання перенавчання використовуються зворотні виклики. ModelCheckpoint зберігає модель, яка має найменші втрати на валідаційному наборі, як найкращу. Далі саме ця модель використовуватиметься для тестування. Тоді як EarlyStopping зупиняє навчання, якщо впродовж визначеної параметром patience кількості епох його результати не покращуються.

Оскільки у коді використовуються і EarlyStopping, і ModelCheckpoint, то найкращою моделлю вважатиметься та, у якої ваги будуть відновлені методом restore_best_weights. Отже, якщо метод EarlyStopping зупинить тренування після початку перенавчання, то модель, яка буде збережена методом ModelCheckpoint, буде тією, у якої навчання не спостерігається.

Варто зазначити, що від розмірів партії (значення параметру batch size) суттєво залежать точність та втрати на валідаційному наборі. Наприклад, при значенні рівному 64 точність моделі протягом багатьох епох трималася рівною 0,5, а її зна-

чення втрат значно перевищувало число 1. На основі цього можна прийти до висновку, що занадто великі значення розмірів партії викликають помилки при валідації.

У ході роботи проведено налаштування параметрів оптимізатора, що дозволило зменшити різкість оптимізації.

Оскільки значення параметра batc_size було обрано рівним 6, що є оптимальним визначеним експериментально варіантом, для уникнення перенавчання береться невелика кількість епох. Вона дорівнюватиме 20. Проте EarlyStopping може здійснити зупинку на більш ранніх стадіях.

Навчання моделі проходить успішно, що продемонстровано на рис. 4.6

```
Epoch 4: val_loss improved from 0.66426 to 0.41660, saving model to best_model
187/187 [==============================] - 568s 3s/step - loss: 0.1023 - accuracy: 0.9643 - val_loss: 0.4166 - val_accuracy: 0.8143
Epoch 5/15
187/187 [==============================] - ETA: 0s - loss: 0.0694 - accuracy: 0.9839
Epoch 5: val_loss improved from 0.41660 to 0.12776, saving model to best_model
187/187 [==============================] - 568s 3s/step - loss: 0.0694 - accuracy: 0.9839 - val_loss: 0.1278 - val_accuracy: 0.9536
```

Рисунок 4.6 — Навчання моделі

У процесі навчання показники втрат для обох наборів зменшуються, а точність зростає.

Результати навчання моделі виведено на рис. 4.7.

```
Final Training Accuracy: 0.9973214268684387
Final Validation Accuracy: 0.9607142806053162
```

Рисунок 4.7 — Результати навчання моделі

Тоді як повний процес навчання на ведено у вигляді графіків на рис. 4.8 та 4.9. На них синім позначено результати для тренувального, а жовтим для валідаційного наборів.

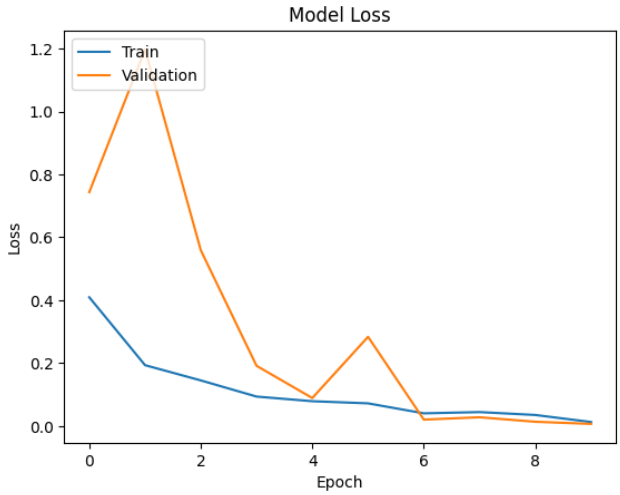

Рисунок 4.8 — Графік втрат при навчанні моделі

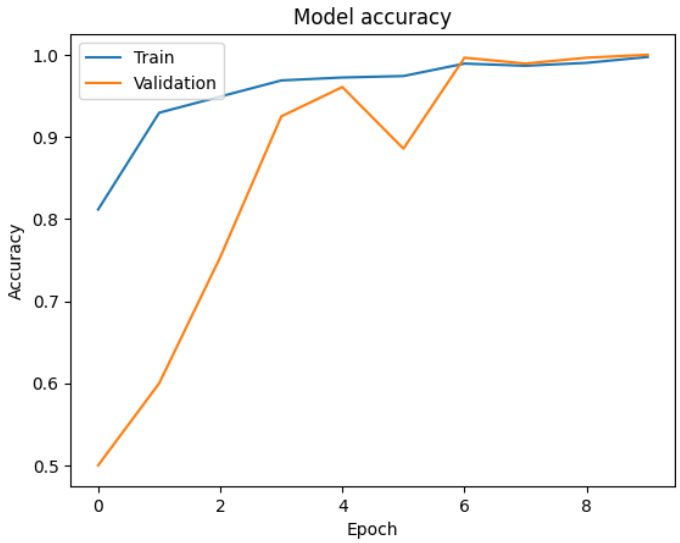

Рисунок 4.9 — Графік точності навчання моделі

У ході роботи було помічено, що навчання моделі особливо чутливе до розмірів набору даних та до його різноманітності. Так при використанні однотипних знімків швидко виникало перенавчання. Надто малі набори даних не дозволяли ефективно на них навчатися і, як наслідок, модель.

4.3 Трансферне навчання

Метод трансферного навчання полягає у використанні вже навчених моделей, що пришвидшує час необхідного навчання мережі та може сприяти підвищенню точності її роботи. Для дослідження цього методу можна скористатися сховищем TensorFlow Hub, яке містить деякі вже навчені на базі даних ImageNet готові моделі. Таким чином отримаємо доступ до попередньо навченої ResNet-50.

У ході роботи був проведений імпорт моделі. Тоді її ваги були заморожені, що дозволило зберегти знання, які модель отримала при навчанні на наборі ImageNet.

Було здійснено вилучення останнього шару моделі і заміна його на класифікатор, який відповідає поставленій у роботі задачі.

Результати навчання моделі виведено у вигляді графіків — рис. 4.10 та 4.11.

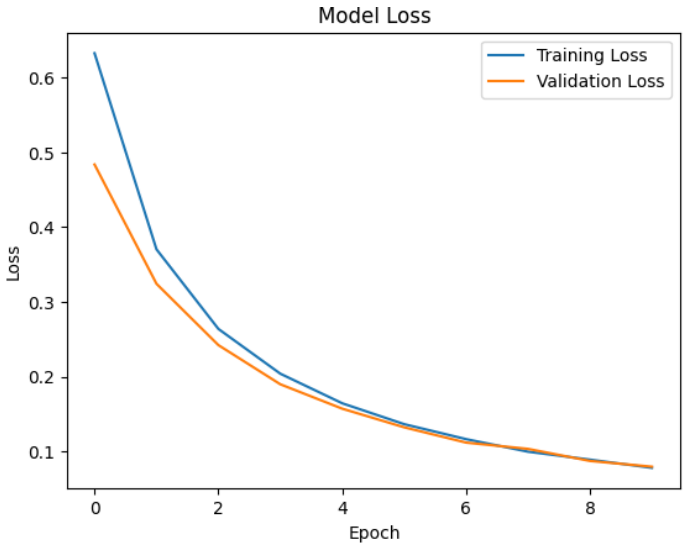

Рисунок 4.10 — Графік втрат при навчанні моделі

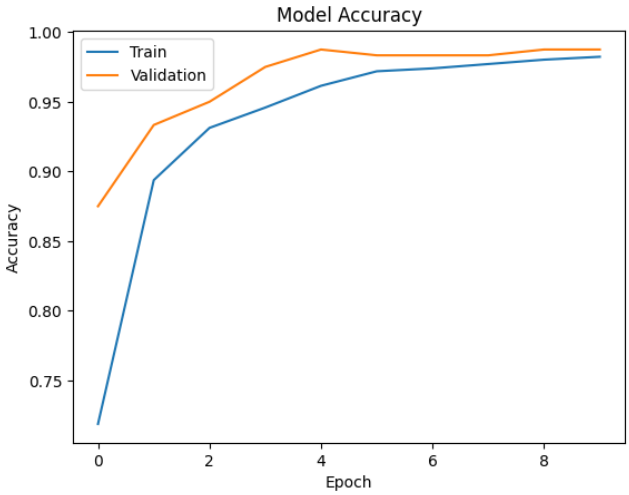

Рисунок 4.11 — Графік точності навчання моделі

З графіків видно, що обидві моделі досягають значної точності під час навчання.

4.4 Аналіз результатів

У ході роботи було використано методи запобігання перенавчанню мережі — використання ранньої зупинки моделі та генерація нових знімків за допомогою аугментації даних. Була запропонована модель, яка вирішує задачу бінарної класифікації. Після запуску вона видала такі результати: точність 99% на тренувальних даних і 96% на валідаційних.

За допомогою трансферного навчання було досліджено попередньо навчену модель ResNet-50. Вона продемонструвала наступні результати: 98% точності на тренувальних і 98% на валідаційних даних.

Для проводження детального аналізу якості виявлення різних особливостей моделями, було проведено тестування на наборі зображень, які не містять міток і раніше не використовувалися при навчанні. Для того, щоб забезпечити більш детальний аналіз якості виявлення різних особливостей на знімках, тестовий набір був розділений на кілька нерівномірних, які були сформовані відповідно до наявності або відсутності певних характеристик на зображеннях. Результати показали, що

моделі мають найвищу точність при перевірці на наборах, які сформовані для порівняння здорової печінки зі знімками, які містять лише чітко виражені зміни форми або значні рубці. Тоді як найнижчу чутливість вони демонструють на наборах, сформованих для порівняння здорової печінки зі знімками печінки, яка містить зачатки вузликів.

Точність реалізованої моделі на кілька відсотків випереджає ResNet. Приблизно 8 із 10 знімків моделями було ідентифіковано вірно, що дозволяє зробити висновок про доцільність використання ResNet архітектур для підвищення точності діагностування цирозу.

У даному розділі була запропонована модель нейронної мережі та проведено її порівняння із попередньо навченою моделлю ResNet.

5 РОЗРОБКА СТАРТАП-ПРОЄКТУ

Розроблення стартап-проекту передбачає здійснення низки кроків, які включають формування ідеї, визначення перспектив її реалізації на ринку та підготовку маркетингової стратегії. У межах розділу розглянемо їх більш детально.

5.1 Опис ідеї проєкту

Результатом магістерської дисертації є нейронна мережа, що розроблена для визначення цирозу печінки на знімках МРТ. Даний результат має практичну цінність для медичної сфери та в подальшому може бути інтегрованим у томографічні системи.

Опис стартап ідеї подано у табл. 5.1.

Таблиця 5.1 — Опис ідеї стартап-проєкту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Використання нейронної мережі у якості допоміжного інструменту при діагностуванні цирозу печінки на знімках магнітно-резонансної томографії. Даний підхід дозволяє підвищити ймовірність вчасного діагностування цього захворювання і, як наслідок, призводить до збільшення шансів пацієнта на одужання	1. Медична діагностика	Збільшення ймовірності виявлення захворювання на ранніх стадіях. Збільшення доступності діагностики та покращення її точності
	2. Дослідження	Розробка нових підходів для визначення цирозу печінки

Цей проєкт направлений на використання у різних медичних закладах. Дана ідея може зацікавити медичних представників – діагностичні центри та приватні

клініки. Потенційних клієнтів потрібно шукати серед тих, хто відомі своїм позитивним відношенням до впровадження новітніх технологій. Серед зацікавлених також можуть виявитися представники, які знаходяться у пошуку шляхів для того, щоб заявити про себе на ринку або раніше не впроваджували значних змін у процесі діагностування, адже використання нової ідеї може вигідно виділити їх серед конкурентів.

5.2 Технологічний аудит

У табл. 5.2 подано технологічний аудит ідеї проєкту. При виборі технологій розробки особливу увагу варто звернути на їхню наявність та доступність.

Таблиця 5.2 — Технологічна здійсненність ідеї проекту

№ п/п	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1	Реалізація моделі нейронної мережі для визначення цирозу на знімках	Python	Наявна	Доступна
		Matlab	Наявна	Доступна
		Google Colab	Наявна	Доступна
Обрана технологія реалізації ідеї проекту: Python, Googl Colab				

Обрані технології доступні для безкоштовного використання. Але важливо зазначити, що комерційне використання набору даних Duke Liver Dataset на якому навчається реалізована нейронна мережа можливе буде правомірним лише при укладанні ліцензійної угоди CC-BY-NC-ND-4.0.

5.3 Аналіз ринкових загроз та можливостей

Визначення ринкових загроз та можливостей дозволить спланувати напрями розвитку проекту.

У табл. 5.3 описана характеристика потенційних клієнтів.

Таблиця 5.3 — Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1	Потреба у визначені наявності цирозу	Приватні та державні клініки, діагностичні центри	Потребують продукт, який буде ефективним та зручним у використанні	Точність
2	Відстеження впливу лікування на стан печінки	Вчені, лікарі	Потребують простий продукт, що буде швидко та точно обробляти дані різних обсягів	Точність, швидкість обробки
3	Дослідження цирозу	Науково-дослідницькі установи	Потребують продукт, який здатен швидко та якісно обробляти високі обсяги даних	Точність, швидкість обробки

Стартап орієнтований на декілька груп. Аналіз показав, що всі вони мають схожі вимоги серед яких найважливішою є точність.

Після визначення потенційної групи клієнтів було проаналізовано ринкове середовище. Для цього у табл. 5.4 і 5.5 розглянуто фактори загроз і можливостей.

Таблиця 5.4 — Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1	Товар недостатньо цікавий користувачам	Користувачі можуть потребувати більш функціональний продукт	Акцентувати увагу на можливості подальшого розширення функціоналу. Здійснити його
2	Невдала рекламна компанія	Рекламна компанія може не принести достатньої кількості потенційних користувачів	Змінити рекламну стратегію. Застосувати систему знижок. Знайти точки додаткової цінності

Таблиця 5.5 — Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	Відсутність конкуренції	Можлива відсутність аналогічного продукту на українському ринку	Максимально поширити інформацію про продукт та його особливість
2	Нові дослідження у медичній сфері	Поява нових досліджень може призвести до покращення алгоритму діагностування цирозу	Оптимізувати продукт через внесення необхідних змін

У таблиці 5.4 подано фактори, що сприяють ринковому впровадженню проекту, а у таблиці 5.5 фактори, що йому перешкоджають. Дані надані в порядку зменшення значущості.

5.4 Стратегія для ринку

Оскільки цироз це глобальне захворюванням від якого страждають мільйони людей в усьому світі, цей стартап має можливості для входу на ринок. Для роботи на ньому було обрано базову стратегію розвитку цього стартап-проєкту. Її визначення наведено у табл. 5.6.

Таблиця 5.6 — Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
	Працюємо з усім ринком, пропонуючи стандартизовану програму	Стратегія масового маркетингу	Простота використання. Задоволення потреб клієнтів і можливість подальшого розширення функціоналу. Нові функції й можливості, яких ще немає на ринку	Стратегія диференціації

Була обрана стратегія диференціації, яка зосереджується на тому, щоб зробити товар привабливішим для споживачів ніж товар конкурентів.

Оскільки стартап націлений на багато груп, йому підійде стратегія масового маркетингу.

Далі здійснимо вибір стратегії конкурентної поведінки. Її визначення подано у табл. 5.7.

Таблиця 5.7 — Визначення базової стратегії конкурентної поведінки

№ п/п	Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
	Так	Шукатиме нових	Ні	Стратегія заняття конкурентної ніші

Далі визначимо стратегію позиціонування до якої входить формування комплексу асоціацій за якими товар має легко ідентифікуватися споживачами. Відповідна інформація подана у табл. 5.8.

Таблиця 5.8 — Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
	Якість, високий рівень точності, зрозумілість при використанні	Стратегія диференціації	Задоволення потреб клієнта	Точність, швидкість обробки, економія ресурсів

Була визначена система рішень щодо поведінки на ринку. Ця система визначатиме напрями поводження на ринку.

5.5 Програма маркетингу

Для досягнення успіху на ринку необхідно розробити ефективну маркетингову стратегію, яка буде спрямована на цільові групи.

Для початку необхідно провести підсумування результатів конкурентоспроможності товару. Дана інформація наведена у табл. 5.9.

Таблиця 5.9 — Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити
1	Точність результатів	Користувач отримує точні результати	Користувач отримує прогнози, які задовольняють його потреби
2	Швидкість обробки медичних знімків	Для отримання результатів не потрібно перебувати у довгому очікуванні	Великі обсяги даних можуть оброблятися одночасно. Навіть в умовах надмірної завантаженості користувача немає необхідності до звернення до сторонніх фахівців для простого попереднього огляду знімків на виражені ознаки цирозу через те, що продукт ефективно впорається сам. Що дозволяє економити ресурси та підтримувати конфіденційність
3	Простота у використанні	Для успішного використання продукту користувачу достатньо прочитати інструкцію з експлуатації	Від користувача не вимагаються глибокі знаннях з програмування

Сформуємо іншу складову маркетингової програми, розробивши концепції маркетингових комунікацій.

Таблиця 5.10 — Концепція маркетингових комунікацій

№ п/п	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
	Потенційні клієнти дізнаються про існуючі товари в мережі інтернет та від уже існуючих клієнтів	Мережа інтернет. Соціальні мережі. Рекомендації	Інтернет маркетинг, SMM маркетинг	Повідомлення про існування товару на ринку. Презентація завдання та характеристик товару. Заохочення до покупки й встановлення товару. Залучення інвесторів	Продемонструвати точність результатів й новизну запропонованого товару

Отже, для того, щоб заохотити споживачів користуватися продуктом буде задіяна реклама. Завдання рекламного повідомлення – презентувати товар та описати ключові характеристики, що виділяють його поміж інших. Як основний канал для поширення реклами доцільно використовувати інтернет-маркетинг.

У даному розділі було проведено розробку стартап-проєкту. На основі проведеної роботи можна зробити висновок, що він має потенціал для успіху. Ідея має значний попит на ринку. Продукт, що пропонується, має потенціал для вирішення декількох проблем у медичній сфері. Подальше впровадження продукту можна вважати доцільним.

При цьому було б корисним залучити одну із цільових груп — медичні заклади — до співпраці. Адже вони можуть надати доступ до більш повного та різноманітного набору даних для навчання.

ВИСНОВКИ

У ході магістерської роботи було досліджено використання цифрового оброблення сигналів, а саме нейромережевого підходу та його методів, для вирішення завдання визначення патологій на медичних зображеннях. Було поставлено за мету зосередитись на пошуку кращої методики для діагностування цирозу печінки.

Дослідження проводилося на основі знімків магнітно-резонансної томографії, які відрізняються точністю та мають високу придатність для спостереження за наявністю патологій.

Під час огляду найпопулярніших архітектур нейронних мереж були визначені їхні переваги та недоліки, а також типові задачі для яких вони зазвичай використовуються. Обрана в дипломній роботі архітектура — згорткова. На вибір вплинули наступні фактори:

– здатність до виявляння локальних особливостей, об’єктів;

– ефективність в умовах обмежених ресурсів;

– розпізнання згортковими шарами об’єктів незалежно від їхнього положення на зображеннях.

Серед відомих згорткових архітектур була обрана ResNet, яка вирішує проблему зникнення градієнту та добре підходить для роботи з медичним знімками через її здатність до виявлення навіть невеликих деталей на зображеннях.

Після дослідження варіантів програмного забезпечення у якості засобу подальшої реалізації нейронної мережі було обрано Python. Серед факторів, які обумовили даний вибір можна виділити:

– велику кількість доступних бібліотек, модулів та інструментів;

– зручність у використанні;

– легкий синтаксис;

– можливість легко застосовувати трансферне навчання;

Робота виконувалася у середовищі Google Colab, яке має зручний інтерфейс, достатню потужність і в цілому добре підходить для реалізації нейронних мереж.

Дане середовище дозволило працювати безпосередньо у браузері на будь-яких пристроях, які підключені до мережі Інтернет.

Методика, використана у роботі полягала у використанні згорткових ResNet мереж, які навчаються на наборі даних, який сформовано зі знімків магнітно-резонансної томографії з цирозом та без цирозу. Під час обробки даних використовувався метод аугментації, який дозволив збільшити об'єм навчальних даних. Для уникання перенавчання використовувалися зворотні виклики.

У результаті виконання кваліфікаційної роботи було розроблено нейронну мережу, яка має доволі високий показник точності при визначенні наявності чи відсутності цирозу печінки на зображеннях. Для порівняння методом трансферного навчання було розглянуто попередньо навчену на наборі даних ImageNet архітектуру ResNet-50.

Аналіз отриманих результатів дозволив зробити висновок про доцільність застосування нейромережевого підходу для підвищення точності діагностування цирозу печінки. Згорткові ResNet мережі показали себе як ефективний діагностичний інструмент навіть на обмеженому наборі даних. Отже, цей напрямок є перспективним для подальшого дослідження. Функціонал запропонованої моделі можна розширити для визначення й інших патологій та для більш чіткої класифікації цирозу на основі поділу на різні стадії розвитку захворювання.

Впровадження продукту на ринку є доцільним. Для реалізації стартап проєкту було б корисно залучити медичні заклади до співпраці. Адже вони можуть надати доступ до більш повного та різноманітного набору даних для навчання та допомогти відкалібрувати точність на основі досвіду лікарів-гепатологів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Цироз печінки. URL: https://uk.wikipedia.org/wiki/Цироз_печінки.

2. Sumeet K. Asrani, Harshad Devarbhavi, John Eaton, Patrick S. Kamath. Burden of liver diseases in the world. Journal of Hepatology, Vol. 70, No. 1. 2018. P. 151–171.

3. Fei Ye, Mimi Zhai, Jianhai Long, Yi Gong, Chutong Ren, Dan Zhang, Xiang Lin, Sushun Liu. The burden of liver cirrhosis in mortality: Results from the global burden of disease study. Front Public Health, Vol. 10. 2022. P. 1-13.

4. Лащевська Н. О., Жирова А. І., Застосування нейронних мереж для діагностування цирозу печінки, V Всеукраїнська науково-технічна конференція студентів та аспірантів «Радіоелектроніка в XXI столітті», 10–12 травня 2023 року Київ, Україна, С. 83-84. URL: https://rtf.kpi.ua/wp-content/uploads/2023/05/5_vntk_radioelektronika_v_hhi_stolitti.pdf.

5. Ankur A. Gupta, Danny C. Kim, Glenn A. Krinsky, and Vivian S. Lee. CT and MRI of Cirrhosis and its Mimics. American Journal of Roentgenology, Vol. 183, No. 6. 2004. P. 1-7.

6. Нейромережа — що це таке, як працює та навіщо потрібна. URL: https://termin.in.ua/neyromerezha/.

7. What Is a Transformer Model? URL: https://blogs.nvidia.com/blog/2022/03/25/what-is-a-transformer-model/.

8. Feed Forward Neural Network. URL: https://deepai.org/machine-learning-glossary-and-terms/feed-forward-neural-network/.

9. Macdonald, J. A., Zhu, Z., Konkel, B., Mazurowski, M., Wiggins, W., & Bashir, M. (2020). Duke Liver Dataset (MRI) (1.0.0) [Data set]. Zenodo. https://doi.org/10.5281/zenodo.6328447.

ДОДАТОК А

```
drive.mount('/content/drive')

n_folder = "/content/drive/MyDrive/Colab Notebooks/dataset/normal"
c_folder = "/content/drive/MyDrive/Colab Notebooks/datset/cirrhosis"
test_folder = "/content/drive/MyDrive/Colab Notebooks/dataset/test"

def load_images(filename, target_size=(224, 224)):
    img = load_img(filename, target_size=target_size, interpolation='lanczos')  # Завантаження зображення за заданим filename, зміна розміру на target_size
    img = img_to_array(img)   # Перетворення у NumPy масив
    img = img / 255.0   # Нормалізація значень у діапазоні [0, 1]
    return img

t_n = []
t_c = []
t_test = []

for filename in tf.io.gfile.glob(f'{n_folder}/*.jpg'):
    img = load_images(filename)
    t_n.append(img)

for filename in tf.io.gfile.glob(f'{c_folder}/*.jpg'):
    img = load_images(filename)
    t_c.append(img)

for filename in tf.io.gfile.glob(f'{test_folder}/*.jpg'):
    img = load_images(filename)
    t_test.append(img)
```

```
def augment_image(image, target_size=(224, 224)):

    angle = np.random.uniform(-10, 10)
    image = rotate(image, angle)

    scale = np.random.uniform(0.8, 1.2)
    zoomed_image = zoom(image, (scale, scale, 1), order=1)

    alpha = np.random.uniform(0.8, 1.2)
    beta = np.random.uniform(-0.2, 0.2)
    zoomed_image = np.clip(alpha * zoomed_image + beta, 0, 1)

    zoomed_image_resized = resize(zoomed_image, target_size)

    zoomed_image_resized = np.clip(zoomed_image_resized, 0, 1)

    return zoomed_image_resized

augmented_t_n = [augment_image(image) for image in t_n]
augmented_t_c = [augment_image(image) for image in t_c]

t_n += augmented_t_n
t_c += augmented_t_c

labels_n = [0] * len(t_n)
labels_c = [1] * len(t_c)

y_train_n = to_categorical(labels_n, num_classes=2)
y_train_c = to_categorical(labels_c, num_classes=2)
```

```
train_n, val_n, y_train_n, y_val_n = train_test_split(t_n, y_train_n, test_size=0.2, random_state=42)
train_c, val_c, y_train_c, y_val_c = train_test_split(t_c, y_train_c, test_size=0.2, random_state=42)

def print_data_info(data, data_name):
    print(f"Info for {data_name} data:")
    print(f"Number of samples: {len(data)}")
    print(f"Data shape: {data[0].shape}")
    print(f"Data type: {data[0].dtype}")
    print(f"Data range: [{np.min(data)}, {np.max(data)}]\n")

print_data_info(t_n, "Оригінальні normal")
print_data_info(t_c, "Оригінальні cirrhosis")

print_data_info(t_test, "Test")

print_data_info(y_train_n, "Мітки для Normal")
print_data_info(y_train_c, "Мітки для Cirrhosis")

input_shape = (224, 224, 3)
num_classes = 2

x_train = np.concatenate([np.array(train_n), np.array(train_c)])
y_train = np.concatenate([y_train_n, y_train_c])

x_val = np.concatenate([np.array(val_n), np.array(val_c)])
y_val = np.concatenate([y_val_n, y_val_c])
```

```
print("Розмірність x_train:", x_train.shape)
print("Розмірність x_val:", x_val.shape)

print("Розмірність y_train:", y_train.shape)
print("Розмірність y_val:", y_val.shape)

y_train_n = to_categorical(labels_n, num_classes=2)
y_train_c = to_categorical(labels_c, num_classes=2)

class block(tf.keras.layers.Layer):
    def __init__(self, channels, stride=1, use_dropout=False):
        super(block, self).__init__()

        self.conv1 = Conv2D(channels, kernel_size=5, strides=stride, padding='same', use_bias=False)
        self.bn1 = BatchNormalization()
        self.relu = ReLU()

        self.conv2 = Conv2D(channels, kernel_size=5, strides=1, padding='same', use_bias=False)
        self.bn2 = BatchNormalization()

        self.shortcut = tf.keras.Sequential()
        if stride != 1:
            self.shortcut.add(Conv2D(channels, kernel_size=1, strides=stride, use_bias=False))
            self.shortcut.add(BatchNormalization())

        self.dropout = Dropout(0.5) if use_dropout else None
```

```
    def call(self, x):
        residual = x

        x = self.conv1(x)
        x = self.bn1(x)
        x = self.relu(x)

        x = self.conv2(x)
        x = self.bn2(x)

        x += self.shortcut(residual)
        if self.dropout is not None:
            x = self.dropout(x)
        x = self.relu(x)

        return x

class NN(tf.keras.Model):
    def __init__(self, num_classes=2):
        super(NN, self).__init__()

        self.conv1 = block(32, stride=2, use_dropout=False)

        self.bn1 = BatchNormalization()
        self.relu = ReLU()
        self.maxpool = MaxPooling2D(pool_size=3, strides=2, padding='same')

        self.layer1 = self._make_layer(32, 2, stride=1, use_dropout=True)
        self.layer2 = self._make_layer(64, 3, stride=2)
        self.layer2_additional1 = self._make_layer(64, 2, stride=1)
```

```
        self.layer2_additional2 = self._make_layer(64, 2, stride=1)

        self.layer3 = self._make_layer(128, 4, stride=2)

        self.avgpool = GlobalAveragePooling2D()
        self.fc = Dense(num_classes, activation='sigmoid')

    def _make_layer(self, channels, num_blocks, stride, use_dropout=False):
        layers = [block(channels, stride, use_dropout)]
        for _ in range(1, num_blocks):
            layers.append(block(channels, stride=1, use_dropout=use_dropout))
        return tf.keras.Sequential(layers)

    def call(self, x):
        x = self.conv1(x)
        x = self.bn1(x)
        x = self.relu(x)
        x = self.maxpool(x)

        x = self.layer1(x)
        x = self.layer2(x)
        x = self.layer2_additional1(x)
        x = self.layer2_additional2(x)
        x = self.layer3(x)

        x = self.avgpool(x)
        x = self.fc(x)

        return x
```

```
model = NN(num_classes)

optimizer = tf.keras.optimizers.Adam(
    learning_rate=0.0001,
    beta_1=0.9,
    beta_2=0.999,
    epsilon=1e-7,
)

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

checkpoint = ModelCheckpoint('best_model', save_best_only=True, monitor='val_loss', mode='min', verbose=1)
checkpoint_path = '/content/drive/MyDrive/Colab Notebooks/dataset/best_model'

early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)

epochs = 20
batch_size = 6

history = model.fit(
    x_train, y_train,
    epochs=epochs,
    batch_size=batch_size,
    validation_data=(x_val, y_val),
    callbacks=[checkpoint, early_stopping]
)

plt.plot(history.history['loss'])
```

```
plt.plot(history.history['val_loss'])
plt.title('Model Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()

plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()

train_accuracy = history.history['accuracy'][-1]
val_accuracy = history.history['val_accuracy'][-1]
print(f"Final training Accuracy: {train_accuracy}")
print(f"Final validation Accuracy: {val_accuracy}")

model.load_weights('/content/drive/MyDrive/Colab Notebooks/dataset/best_model')

predictions = model.predict(np.array(t_test))

predicted_classes = np.argmax(predictions, axis=1)

for i, prediction in enumerate(predictions):
    print(f"Sample {i + 1}: {prediction}, Predicted Class: {predicted_classes[i]}")

actual_labels = {
```

```
    'image_1.jpg': 1,
    'image_2.jpg': 1,
    'image_3.jpg': 1,
    'image_4.jpg': 1,
    'image_5.jpg': 1,
    'image_6.jpg': 0,
    'image_7.jpg': 0,
    'image_8.jpg': 1,
    'image_9.jpg': 1,
    'image_10.jpg': 0
}

actual_labels_list = [actual_labels[f'image_{i + 1}.jpg'] for i in range(len(predictions))]

# Обчислення точності
accuracy = accuracy_score(actual_labels_list, predicted_classes)

print(f"Accuracy on test images: {accuracy}")
```