Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження методів штучного інтелекту для аналізу дерматоскопічних зображень при діагностуванні захворювань шкіри
(тема)

Виконав:
студент 2 курсу, групи СШІм-22-1
Малихін О.С.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)
Освітня програма Системи штучного інтелекту
(повна назва спеціалізації)
Керівник доц. Селіванова К.Г.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ В.О. Філатов
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

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

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Малихіну Олександру Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження методів штучного інтелекту для аналізу дерматоскопічних зображень при діагностуванні захворювань шкіри

затверджена наказом університету від 1 квітня 20 24 р. № 260Ст
2. Термін подання студентом роботи до екзаменаційної комісії 11 червня 20 24 р.
3. Вихідні дані до роботи науково-методична література, дані з інтернет джерелж, нейронні мережі: LeNet, ResNet18, ResNet18, VGG19, AlexNet, GoogLeNet, RegNet_Y_32GF, бібліотеки: pytorch, numpy, pandas, torchvision, sklearn, matplotlib. набір даних HAM10000 з дерматоскопічними зображеннями у форматі jpg, середа розробки PyCharm.

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі та постановка задачі
2) Аналіз сучасних методів для класифікації та аналізу зображень
3) Аналіз наявних методів обробки й аналізу дерматоскопічних зображень
4) Практична реалізація та проведення експериментів.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	01.04.2024	виконано
2	Аналіз предметної галузі	03.04.2024	виконано
3	Огляд інформаційних джерел	06.04.2024	виконано
4	Огляд наявних рішень	10.04.2024	виконано
5	Пошук та аналіз набору даних	15.04.2024	виконано
6	Аналіз наявних методів	17.04.2024	виконано
7	Проведення експериментів	20.04.2024	виконано
8	Обробка та аналіз експериментів	27.04.2024	виконано
9	Написання пояснювальної записки	03.05.2024	виконано
10	Перевірка на академічний плагіат	10.05.2024	виконано
11	Перевірка на нормоконтроль	13.05.2024	виконано
12	Підготовка презентації та доповіді	17.05.2024	виконано
13	Попередній захист	05.06.2024	виконано
14	Рецензування	07.06.2024	виконано
15	Захист перед екзаменаційною комісією	11.06.2024	виконано

Дата видачі завдання 1 квітня 2024 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Селіванова К.Г.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 100 с., 51 рис., 6 табл., 3 дод., 32 джерел.

ГЛИБИННЕ НАВЧАННЯ, ЗАХВОРЮВАННЯ ШКІРИ, МАШИННЕ НАВЧАННЯ, НЕЙРОНІ МЕРЕЖІ, ТЕЛЕМЕДИЦИНА, ШТУЧНИЙ ІНТЕЛЕКТ, CNN.

Об’єкт дослідження – наявні методи штучного інтелекту для аналізу й класифікації дерматоскопічних зображень при діагностуванні захворювань шкіри

Предмет дослідження – ефективність наявні методі штучного інтелекту для аналізу й класифікації дерматоскопічних зображень при діагностуванні захворювань шкіри

Мета роботи – пошук ефективних методів штучного інтелекту для аналізу й класифікації дерматоскопічних зображень при діагностуванні захворювань шкіри

Методи дослідження – аналіз теоретичних джерел, технічної літератури, проектування й імплементація інтелектуальних моделей для аналізу й класифікації дерматоскопічних зображень при діагностуванні захворювань шкіри, порівняння результатів роботи побудованих моделей, експериментальне дослідження ефективності побудованих моделей на різних вибірках даних.

В роботі було досліджено ефективність різних методів побудови штучного інтелекту для аналізу й класифікації та захворювань шкіри.

В результаті проведених теоретичних і експериментальних досліджень рекомендовано низку ефективних методів для аналізу й класифікації та захворювань шкіри

ABSTRACT

Master’s thesis contains: 100 pp., 51 fig., 6 tabl., 3 ann., 32 references.

DEEP LEARNING, SKIN DISEASES, MACHINE LEARNING, NEURAL NETWORKS, TELEMEDICINE, ARTIFICIAL INTELLIGENCE, CNN

The object of the research is existing artificial intelligence methods for analyzing and classifying dermatoscopic images in the diagnosis of skin diseases

Subject of research – the effectiveness of existing artificial intelligence methods for analyzing and classifying dermatoscopic images in the diagnosis of skin diseases

The subject of the research is to find effective methods of artificial intelligence for the analysis and classification of dermatoscopic images in the diagnosis of skin diseases

Research methods – analysis of theoretical sources, technical literature, design and implementation of intelligent models for analyzing and classifying dermatoscopic images in the diagnosis of skin diseases, comparison of the results of the built models, experimental study of the effectiveness of the built models on different data samples.

In this study the effectiveness of different methods of building artificial intelligence for analyzing and classifying skin diseases.

As a result of the theoretical and experimental studies, a number of effective methods for analyzing and classifying skin diseases are recommended.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

Класифікатор – алгоритм машинного навчання, що здатен передбачити клас об'єкту на основі деяких вхідних даних;

ШІ – штучний інтелект – розділ комп'ютерної лінгвістики та інформатики, що опікується формалізацією проблем та завдань, які подібні до дій, що виконує людина;

ШНМ – штучна нейронна мережа – алгоритм або математична модель, яка імітує роботу людського мозку і використовується для розв'язання складних завдань в області штучного інтелекту;

AI – Artificial Intelligence – штучний інтелект;

CNN – Convolutional Neural Network – тип нейронної мережі, яка спеціалізується на обробці даних, які мають просторову структуру, такі як зображення;

DL – Deep Learning – підгалузь машинного навчання, що використовує нейронні мережі з багатьма шарами для навчання на великих наборах даних і розв'язання складних завдань;

ML – Machine Learning – галузь штучного інтелекту, яка вивчає алгоритми, які навчають комп'ютери робити рішення на основі вхідних даних, без явного програмування;

OCR – Optical character recognition – технологія, яка дозволяє автоматично розпізнавати текст на зображеннях, та конвертувати його у відповідний текстовий формат, який може бути подальше обробленою або збереженою;

SVM – Support Vector Machines – набір схожих алгоритмів навчання з викладачами, що використовуються для задач класифікації та регресійного аналізу.

ВСТУП

В останній час штучний інтелект сильно впливає та трансформує медичну сферу, надаючи нові можливості для діагностики та лікування захворювань. Однією з найбільш актуальних напрямків використання цієї технології знаходиться аналіз дерматоскопічних зображень для діагностики захворювань шкіри. В умовах зростаючого обсягу медичної інформації та постійного розвитку методів штучного інтелекту виникає необхідність глибокого дослідження та вдосконалення цих методів у сфері дерматології.

Дерматоскопія, яка є найефективнішою технікою для вивчення пігментованих утворень на шкірі, надає велику кількість інформації, але її інтерпретація залишається викликом для лікарів. Розробка та вдосконалення методів штучного інтелекту, спрямованих на аналіз дерматоскопічних зображень, може значно полегшити роботу фахівців та покращити точність діагностики захворювань шкіри.

Дослідження методів штучного інтелекту в області дерматоскопії має потенціал виявити нові шляхи для розвитку медичної науки та покращення якості надання медичних послуг у галузі дерматології. Це допоможе не лише прискорити процес діагностики, але й сприятиме ранньому виявленню захворювань, підвищуючи ефективність лікування та зменшуючи ризики для пацієнтів.

Метою цієї роботи є дослідити наявні методи штучного інтелекту для аналізу дерматоскопічних зображень при діагностуванні захворювань шкіри, порівняти їх та знайти ефективну архітектуру для побудови системи штучного інтелекту який дозволить швидко та ефективно аналізувати й обробляти зображення захворювання шкіри й буде забезпечувати високу точність у виявленні наявних захворювань.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Аналіз наявних методів діагностики захворювань шкіри

За останній час медицина зробила багато кроків з вдосконалення й винаходу різних методів для аналізу захворювань шкірі. Також у наш час зі стрімким розвитком медичних технологій постійно вдосконалюються старі методі та винаходяться нові, оскільки шкіра це найбільший орган людського тіла. Шкіра не лише виконує функцію захисту від зовнішніх впливів, але і є важливим індикатором нашого здоров'я. Різноманітні захворювання шкіри можуть виявлятися через різні симптоми, від покрасніння та свербіжу до серйозних патологій, таких як рак шкіри [1].

На даний чай існує дуже багато методів для аналізу захворювань шкіри, ці методи використовуються залежно від типу захворювання, його характеристик та етапу розвитку [2]. Розглянемо деякі з найбільш поширених методів аналізу захворювань шкіри:

– дерматоскопія. дерматоскопія є методом для дослідження пігментних плям, судин, волосистої частини шкіри та дермальних структур. Цей метод дозволяє виявляти ознаки різних дерматологічних захворювань, включаючи рак шкіри;

– біопсія шкіри. це процедура, під час якої знімається невеликий зразок тканини з ураженої області для подальшого вивчення під мікроскопом. Біопсія дозволяє отримати точну діагностику та визначити характер патології;

– клінічний огляд. лікар проводить візуальний огляд пацієнта та вивчає характеристики висипів, плям, вузлів чи інших симптомів. Клінічний огляд може включати в себе пальпацію (дотик) та огляд під великим збільшенням;

– кров'яні тести. деякі захворювання шкіри, такі як автоімунні або системні захворювання, можуть супроводжуватися змінами в аналізах крові, таких як підвищений рівень запальних маркерів;

– мікробіологічні тести. для виявлення бактерій, грибків чи вірусів, які можуть бути причиною інфекційних захворювань шкіри;

– дерматогістопатологічний аналіз. дослідження тканин, взятих під час біопсії, під мікроскопом дозволяє отримати інформацію про зміни в клітинах та структурах, що може бути важливим для точної діагностики [3];

– імунологічні тести. для деяких аутоімунних захворювань шкіри важливим є вивчення імунологічних параметрів, таких як антитіла.

Як можна побачити, сучасна медицина має дуже багато методів для діагностики захворювань шкіри, але варто відзначити, що методи не використовуються по одному, гарна практика це комбінація кількох методів для повноцінного аналізу та діагностики захворювань шкіри [4]. Але варто пам’ятати, що точний підхід для вибору методів для дослідження захворювання шкіри залежить від конкретних обставин і симптомів, які представлені у пацієнта.

1.1.1 Дерматоскопія у діагностиці шкірних захворювань

Дерматоскопія – це неінвазивний метод обстеження, спрямований на детальне вивчення пігментованих утворень на поверхні шкіри. Цей метод став ключовим для діагностики захворювань шкіри, зокрема злоякісних новоутворень [5]. Завдяки використанню спеціального пристрою – дерматоскопа, лікар може отримати збільшене та освітлене зображення шкіри, що надає детальний огляд її структури.

Метод дерматоскопії базується на використанні польового мікроскопа(дерматоскопа), який дозволяє лікарю бачити структурні деталі шкіри, недоступні для звичайного огляду. Оптична система дерматоскопа забезпечує збільшення та освітлення зображення, а фільтри дозволяють

виділити певні характеристики, такі як пігмент, кров'яні судини чи рогові частки. На рисунку 1.1 можна побачити дерматоскоп.

Рисунок 1.1 – Дерматоскоп

Також варто зазначити важливість дерматоскопії в ранньому виявленні захворювань шкіри. Дерматоскопія визначається як ключовий елемент у ранньому виявленні та відстежуванні різноманітних захворювань шкіри, зокрема меланом – агресивного та підступного злоякісного утворення. Ранні стадії захворювань часто характеризуються змінами в структурі та колірному відтінку пігментованих утворень, що може бути виявлено лише за допомогою дерматоскопії [6].

Дерматоскопія дозволяє лікарям не тільки визначити наявність патології, але й класифікувати її тип та визначити оптимальний підхід до лікування. Застосування цього методу стає важливим етапом у

впровадженні індивідуалізованої медицини та підвищує ефективність профілактичних заходів.

Даний метод є одним з найефективніших методів для вивчення пігментованих утворень та ранньої діагностики раку шкіри. Вона дозволяє лікарю детально оглядати структури шкіри, сприяючи виявленню навіть малих змін. Цей метод може бути використаний для вивчення судин, фолікулів, різних плям і вузлів.

1.1.2 Біопсія шкіри у діагностиці шкірних захворювань

Біопсія шкіри – це процедура, під час якої лікар відбирає невеликий зразок тканини з ураженої області шкіри для подальшого лабораторного аналізу [7]. Цей метод використовується для отримання точної діагностики та визначення природи або причини патології шкіри. Процедуру біопсії можна побачити на рисунку 1.2.

Рисунок 1.2 – Біопсія шкіри

Також варто відзначити, що зазвичай виділяють три типи біопсії шкіри, а саме:

– суперфіціальна (поверхнева) біопсія. лікар відбирає зразок тканини з верхнього шару шкіри, що може включати епідерміс та частину дерми. Цей метод часто використовується для дослідження пухлин, висипів та інших областей змін шкіри;

– пучкова (коркова) біопсія. відбирається кілька зразків шкіри з різних областей за допомогою спеціальних інструментів або біопсійного пучка. Це дозволяє лікарю дослідити різні частини ураженої області;

– діагностична біопсія. відбирається зразок з центральної частини ураженої області для точної діагностики та визначення природи патології.

Біопсія шкіри визначається своєю ключовою роллю в діагностиці та розумінні різноманітних захворювань шкіри, пропонуючи цінні відомості для подальшого лікування та управління здоров'ям пацієнта. Її застосування включає в себе точне встановлення діагнозу пухлин, таких, як рак шкіри, що стає особливо важливим у роботі з онкологічними пацієнтами. Крім того, біопсія шкіри розкриває свою важливість у диференціації захворювань зі схожими симптомами, а також у вивченні запальних та інфекційних процесів, надаючи цільову інформацію для ефективного лікування [8]. Додатково, вона використовується для визначення причин алергічних реакцій, що сприяє розкриттю повного спектра факторів, що впливають на стан шкіри. Загалом, біопсія шкіри не лише відкриває нові горизонти у глибинному розумінні структурних та клітинних аспектів шкіри, але й стає ключовим інструментом для персоналізованого підходу до діагностики та лікування різних захворювань, забезпечуючи якісну медичну допомогу.

1.2 Аналіз методів отримання дерматоскопічних зображень

Для того, щоб провести аналіз дерматоскопічних зображень потрібно ці зображення отримати, зазвичай для отримання якісних знімків використовують спеціальні пристрої, а саме дерматоскопи.

Для отримання якісних зображень шкіри за допомогою дерматоскопів, які відображають деталі та структури на поверхні шкіри, є декілька ключових аспектів, які слід враховувати. Перш за все, важливо забезпечити належне освітлення та фіксацію при використанні дерматоскопа для отримання чітких та детальних зображень. Крім того, налаштування параметрів зображення, таких як контрастність та насиченість, може вплинути на якість отриманих знімків. Вони служать для того, щоб отримувати якісні знімки високого розширення шкіри для подальшого дослідження лікарем на наявність різних утворень, патологій та зміни на поверхні шкіри які можуть бути не помічені при звичайному огляді. На рисунку 1.3 зображено дерматоскопічний знімок зроблений за допомогою дерматоскопа.

Рисунок 1.3 – Дерматоскопічний знімок зроблений за допомогою професійного дерматоскопа

Дерматоскопи, такі як UM039 Mini й Mobail Microscope ROHS, мають компактний дизайн та є переносними, що дуже сприяє для застосування в різних клінічних умовах. Вони дозволяють лікарям проводити як загальний,

так і детальний огляд шкіри для виявлення патологій, утворень та інших змін, що допомагає вчасно постановці діагнозу та прийнятті відповідних медичних заходів. На рисунку 1.4 зображено трихоскоп/дерматоскопUM039 Mini UM039 Mini, а на рисунку 1.5 зображено портативний дерматоскоп Mobail Microscope ROHS.

Рисунок 1.4 – Трихоскоп/дерматоскоп UM039 Mini UM039 Mini

Рисунок 1.5 – Портативний дерматоскоп Mobail Microscope ROHS

Також дерматоскопічні знімки або огляд можна робити за допомогою спеціальних електронних луп якщо нема дерматоскопів. Наприклад як

аналогом дерматоскопа використовують Visum Ecare 3,5, дана електронна портативна лупа має автономну підсвітку, наближення з 2.5 до 17 разів й змінну корекцію кольору. Visum Ecare 3,5 зображено на рисунку 1.6

Рисунок 1.6 – Електронна лупа Visum Ecare 3,5

1.3 Штучний інтелект у медицині

Для точної діагностики захворювань необхідно пройти роки медичної освіти. Однак, навіть за наявності такої освіти, діагностика залишається трудомістким і довготривалим процесом. Крім того, лікарська помилка продовжує залишатися однією з найпоширеніших причин смерті пацієнтів. В багатьох галузях попит на фахівців перевищує наявну кількість. Це ставить лікарів у складне положення та часто затримує час, який можна було б витратити на більш ретельну діагностику пацієнта. Таким чином, на допомогу приходить штучний інтелект.

У сучасному світі штучний інтелект швидко трансформує різні галузі, і медицина не є винятком. Впровадження штучного інтелекту у медичну практику відкриває перед нами несхожі можливості для революції у діагностиці, лікуванні та управлінні здоров'ям. Здійснення високоточних діагнозів, розробка персоналізованих планів лікування, автоматизація рутинних завдань та прогнозування захворювань – це лише кілька аспектів, де роль штучного інтелекту виявляється надзвичайно важливою. У цьому

контексті дослідження можливостей та викликів, пов'язаних зі злагодженим злиттям медицини та інноваційних технологій, надає нам можливість поглиблено розглянути, як штучний інтелект революціонізує сферу медичної науки та надає високотехнологічні інструменти для вдосконалення догляду за здоров'ям пацієнта.

1.3.1 Штучний інтелект у діагностуванні захворювань

Штучний інтелект грає ключову роль у сфері діагностики захворювань, надаючи швидкі та точні результати, а також покращуючи можливості раннього виявлення та лікування [9]. Ось деякі способи, які ШІ використовується у діагностичній медицині:

– зображення медичних досліджень. ШІ допомагає аналізувати зображення, отримані за допомогою рентгенів, магнітно-резонансної томографії (МРТ), комп'ютерної томографії (КТ) та інших методів. Алгоритми глибинного навчання можуть виявляти аномалії та визначати їх характер, допомагаючи лікарям у точній діагностиці;

– аналіз медичних даних та історій захворювань. ШІ може обробляти великі обсяги медичних даних, включаючи результати аналізів, біометричні показники та інші клінічні дані. Аналіз цих інформаційних потоків допомагає виявляти закономірності та асоціації, сприяючи швидшому та точнішому діагностуванню;

– електрокардіограми (ЕКГ) та моніторинг серцевої діяльності. алгоритми ШІ можуть аналізувати ЕКГ та інші дані з моніторів серцевої діяльності для виявлення аритмій, ішемії та інших порушень;

– біомаркери та геноміка. ШІ використовується для аналізу генетичних даних та ідентифікації біомаркерів, які можуть служити ознаками певних захворювань чи ризиків їх розвитку;

– допоміжний аналіз симптомів та історій пацієнтів. в системах штучного інтелекту можуть використовуватися алгоритми обробки

природної мови для аналізу симптомів, які вводять пацієнти, та порівняння їх історій з великими базами даних для виявлення можливих діагнозів.

Алгоритми штучного інтелекту вивчають виявлення закономірностей, що характерні для різних захворювань, так само як це роблять лікарі (наприклад, шкірні захворювання, пухлини тощо). Основна різниця полягає в тому, що для навчання алгоритмам потрібно безліч тисяч конкретних цифрових прикладів, проте вони здатні зробити висновки за долі секунди та виділяти ті дрібниці, на які людина може і не звернути увагу [10].

Алгоритми використовуються для виявлення раку легенів або інсульту за допомогою комп'ютерної томографії, оцінки ризику раптової серцевої смерті чи інших серцевих захворювань на основі електрокардіограм та зображень МРТ серця, класифікації шкірних захворювань на зображеннях шкіри, виявлення індикаторів діабетичної ретинопатії на фотографіях очей. На рисунку 1.7 зображено як можуть бути використані дані для класифікації захворювань.

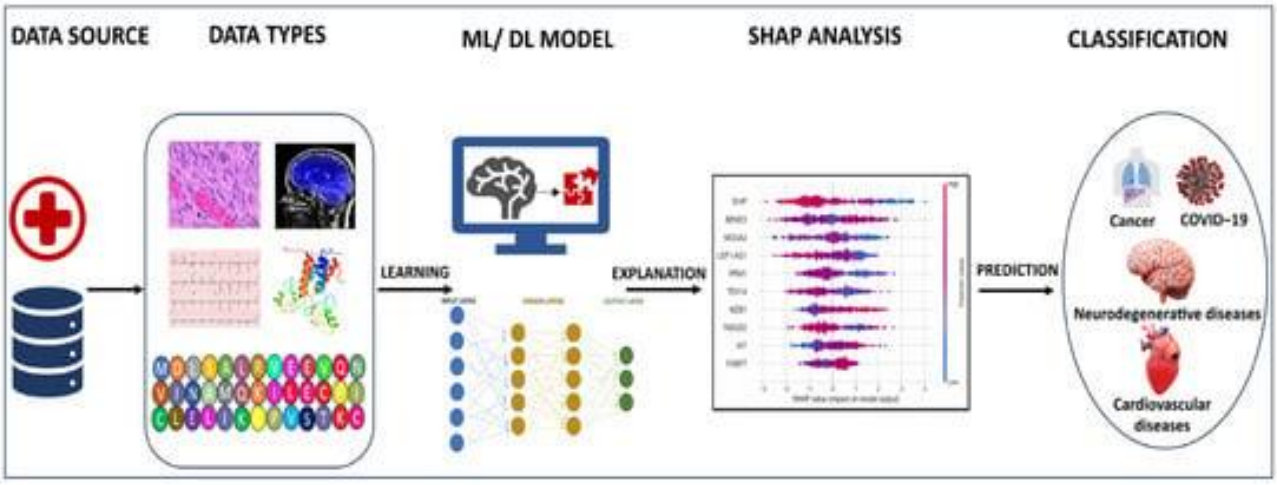


Рисунок 1.7 – Використання даних для діагностування хвороб

Вхідні дані для навчання моделі ML/DL можна отримати з баз даних і лікарень. Ці дані можуть включати дані секвенування наступного

покоління, послідовність і структурну інформацію про білки, скановані зображення, ЕКГ, гістопатологічні зображення та електронні медичні картки пацієнтів [11]. Модель, навчена з використанням цих наборів даних, може бути здатна ідентифікувати або діагностувати стан хвороби для невідомого зразка. Далі для визначення ознак, що відповідають за прогноз, використовуються зрозумілі методи ШІ, зокрема SHAP-аналіз.

Використання штучного інтелекту в діагностиці дозволяє покращити точність та швидкість визначення хвороби пацієнта, що може значно полегшити роботу лікарів, підвищити якість медичної допомоги та зробити медичні послуги більш доступними та дозволяти лікарям приймати більше людей.

1.3.2 Штучний інтелект та телемедицина

Штучний інтелект також використовуються у телемедицині, ШІ та телемедицина взаємодіють для поліпшення якості та доступності медичних послуг. Існує дуже багато різних напрямків, які використовують ШІ в телемедицині, наприклад для діагностики за допомогою обробки зображень. Алгоритми ШІ можуть аналізувати медичні зображення [12] на телефоні або у спеціальному додатку, виявляючи аномалії та надаючи точні діагнози, що особливо корисно для областей з обмеженим доступом до спеціалістів.

Також штучний інтелект дуже поширено використовуються для моніторингу стану пацієнтів за допомогою носимих пристроїв. ШІ оброблює дані з носимих пристроїв, дозволяючи в реальному часі моніторити показники здоров'я та реагувати на можливі проблеми. Це допомагає лікарям мати обширну статистику про пацієнта та слідити за життьовими показниками, й, наприклад назначати ліки в залежності від показників, або сам ШІ може аналізувати дані та присилати лікарю пропозиції по альтернативнім способам лікування або новим лікам які потрібно призначити пацієнту [13].

У останній час також можна побачити дуже різкий ріст чат-ботів та віртуальних асистентів в медицині [14]. За допомогою ШІ тепер можна створювати чат-ботів та віртуальних асистентів, які можуть надавати консультації, надсилати пригоди для медичних тестів чи нагадувати про лікування.

ШІ також допомагає працювати з великими даними й проводити аналіз, штучний інтелект оброблює великі об'єми медичних даних, знаходячи зв'язки та патерни для прогнозування епідемій, виявлення факторів ризику та оптимізації лікування.

Такі великі об’єми даних допомагають ШІ підвішувати точність рекомендації й встановлення діагнозу та мати додаткову вибірку для до навчання. Також це дозволяє будувати великі експертні системи для діагностики та лікування. Ці експертні системи, які базуються на медичних знаннях та алгоритмах, для допомоги у діагностиці та прийнятті рішень щодо лікування.

Незабаром настане ера індивідуалізованої телемедицини на основі штучного інтелекту. У майбутньому медичне зображення можна буде отримати за допомогою офтальмологічного інструменту, а потім передати на телемедичну платформу через Інтернет. Телемедична платформа на основі штучного інтелекту проаналізує зображення, щоб поставити діагноз і запропонувати користувачеві (спеціалісту або пацієнту) ліки для персональної прецизійної медицини.

В залежності від системи, та складності телемедійної системи передача інформації може бути реалізована по різному, але на рисунку 1.8 можна побачити як приклад передачі інформації від пацієнта до штучного, а й дали до лікаря, де він може приймати подальші рішення для постановлення діагнозу [15].

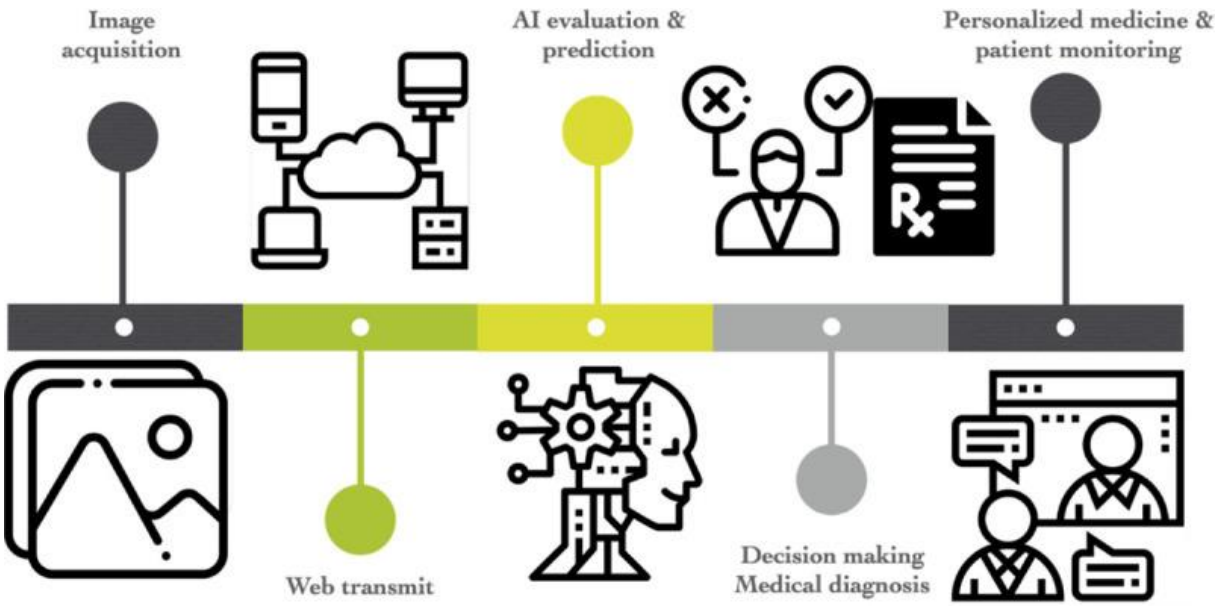


Рисунок 1.8 – Передача інформації у телемедичній системі

ШІ вже глибоко увійшов у наше повсякденне життя, у медичній сфері все більше алгоритмів ШІ розробляються і застосовуються в клініці [16]. З впровадженням і поєднанням з даними про ставки та Інтернетом речей доступність і застосовність ШІ, безумовно, збільшиться. Цілком можливо, що за допомогою ШІ пацієнти в майбутньому зможуть швидко і точно отримувати цінні медичні поради щодо своїх захворювань, можна вже вважати що ера індивідуалізованої телемедицини на основі ШІ вже на порозі.

Телереабілітація та навчання на відстані: ШІ використовується для розвитку телереабілітаційних програм та систем навчання на відстані, що дозволяє пацієнтам отримувати медичну допомогу та інструкції, не виходячи з дому [17].

Всі ці застосування ШІ в телемедицині сприяють покращенню доступності та ефективності медичних послуг, особливо у віддалених або важкодоступних регіонах [18].

2 АНАЛІЗ СУЧАСНИХ МЕТОДІВ ДЛЯ КЛАСИФІКАЦІЇ ТА АНАЛІЗУ ЗОБРАЖЕНЬ

Останнім часом штучний інтелект здійснив значні революції в різних галузях, особливо в обробці зображень. Стрімкий розвиток технологій і глибокого навчання призвів до значного поліпшення здібностей ШІ в розпізнаванні та класифікації зображень. Це призвело до широкого застосування таких технологій у різних сферах, включно з медициною, автомобільною промисловістю, безпекою та багато іншого.

Одним із найпомітніших досягнень у сфері класифікації зображень є розвиток згорткових нейронних мереж (CNN). Ці мережі, натхненні біологічними процесами в мозку, здатні автоматично витягувати ознаки із зображень і використовувати їх для точної класифікації. Наприклад, CNN були успішно застосовані для розпізнавання об'єктів на зображеннях, діагностики медичних станів на рентгенівських знімках і скануваннях МРТ, а також для автоматичного аналізу відеоматеріалів у системах безпеки.

Іншим прикладом значного прогресу в галузі класифікації зображень є використання методів глибокого навчання для знаходження та класифікації об'єктів на зображеннях у реальному часі. Технології такого роду активно застосовуються в системах спостереження і безпеки, автономних транспортних засобах, а також у робототехніці для різних завдань, таких як навігація і маніпуляція об'єктами.

Дані приклади показують на потенціал і вплив, який штучний інтелект чинить у сфері класифікації зображень, демонструють його широкий спектр застосування та показує нам те, що ШІ розвиваються зараз дуже швидкими темпами й за останньою статистикою тенденцією можна очікувати що у наступні роки ШІ здійснить ще дуже великі кроки у різних сферах й у сфері класифікації зображень

2.1 Класифікація зображень за допомогою CNN

Згорткова нейронна мережа (CNN) – це тип штучної нейронної мережі, який переважно використовується для обробки даних з сітчастою топологією, наприклад, для розпізнавання та класифікації зображень [19]. У порівнянні з іншими алгоритмами й методами класифікації зображень, основною перевагою згорткових нейронних мереж є те, що вони є формою неконтрольованого навчання і не потребують маркованих зразків для вивчення ознак з даних. Завдяки цій перевазі згорткові нейронні мережі стали однією з найпопулярніших мереж глибокого навчання. В останні роки було доведено, що згорткові нейронні мережі також є цінним інструментом в інших галузях науки про дані, включаючи аналіз великих даних.

Згорткові нейронні мережі – це категорія нейронних мереж, які довели свою ефективність у таких сферах, як розпізнавання та класифікація зображень. «Згортковий» в CNN належить до математичної операції, яку ці мережі використовують для обробки даних. Ця операція значно зменшує кількість параметрів у моделі, що робить її легшою для навчання і менш схильною до перенавчання порівняно з повністю зв'язаними мережами.

ШНМ відіграють важливу роль в аналізі зображень, забезпечуючи вирішення таких завдань, як розпізнавання зображень, розпізнавання облич та виявлення об'єктів. Дизайн CNN натхненний структурою та функціями зорової кори головного мозку людини, де різні нейрони реагують на різні візуальні стимули в полі зору.

CNN здійснили революцію в галузі комп'ютерного зору, особливо у сфері класифікації зображень. Завдяки своїй здатності автоматично вивчати ознаки з необроблених піксельних даних, CNN досягли значного успіху в широкому спектрі застосувань, включаючи розпізнавання об'єктів, виявлення облич та аналіз медичних зображень. Розглянемо структуру ШНМ, обговоримо їхні переваги та обмеження, дослідимо етапи використання ШНМ для класифікації зображень, висвітлимо поточні

досягнення, розглянемо практичні застосування і на завершення визначимо майбутні напрямки досліджень ШНМ.

Традиційні нейронні мережі не є ідеальними для обробки й роботи з зображеннями і можуть зіткнутися зі значними різними проблемами. Вони не дуже добре справляються з різницею в зображеннях, наприклад, різницею в орієнтації, розмірі або розташуванні об'єкта на зображенні яку людина природно не помічає. Це відбувається тому, що вони не враховують просторову ієрархію пікселів на зображенні. З іншої сторони штучні нейронні мережі спеціально розроблені для обробки піксельних даних і можуть приймати вхідне зображення, присвоювати важливість (ваги й зсуви, які можна вивчити) різним аспектам або об'єктам на зображенні й бути здатними відрізнити один від іншого.

Також варто відзначити, що попередня обробка зображень або даних, необхідна для ШНМ, набагато нижча порівняно з іншими алгоритмами класифікації. У той час як у примітивних методах фільтри створюються вручну, при достатньому навчанні, ШНМ здатні вивчати ці фільтри й характеристики.

Після ознайомлення з CNN, розглянемо як CNN виглядає з точки архітектури. Як і у повністю зв'язані нейронні мережі, згорткова нейронна мережа має вхідний шар, вихідний шар і один або кілька прихованих шарів.

Вхідний шар є першим шаром нейронної мережі й містить вхідні дані, вхідний шар вводить дані в нейронну мережу, щоб дані могли бути передані наступним шарам для обробки. Вихідний шар є останнім шаром нейронної мережі, він передає вихідні дані з нейронної мережі. Приховані шари – це один або декілька шарів, які існують між вхідним і вихідним шарами. Кожен шар також має один або декілька нейронів. Нейрони є основною одиницею нейронної мережі. Кожен нейрон має вхід і виробляє вихідні дані.

У повністю пов'язаній нейронній мережі всі її приховані шари є повністю пов'язаними шарами, так само як і необхідні активні функціональні шари. Через обмеженість цієї конструкції, повністю зв'язані

нейронні мережі мають два основні недоліки при використанні для обробки великих обсягів даних з топологією, подібною до сітки:

– інформація про координати втрачається, коли матриця даних перетворюється на одновимірний масив.

– велика кількість змінних ускладнює навчання, а також робить її схильною до перенавчання.

Однак архітектура згорткових нейронних мереж може легко розв’язати ці дві проблеми. На відміну від більш відомої повнозв'язної нейронної мережі, приховані шари згорткової нейронної мережі можуть бути не тільки повнозв'язними шарами та активною функцією, але також можуть бути згортковими шарами, що об'єднує інші шари. Як правило, приховані шари згорткової нейронної мережі є комбінацією згорткових шарів, об'єднаних шарів, повністю зв'язаних шарів, а також активної функції. На рисунку 2.1 можна побачити архітектуру CNN.

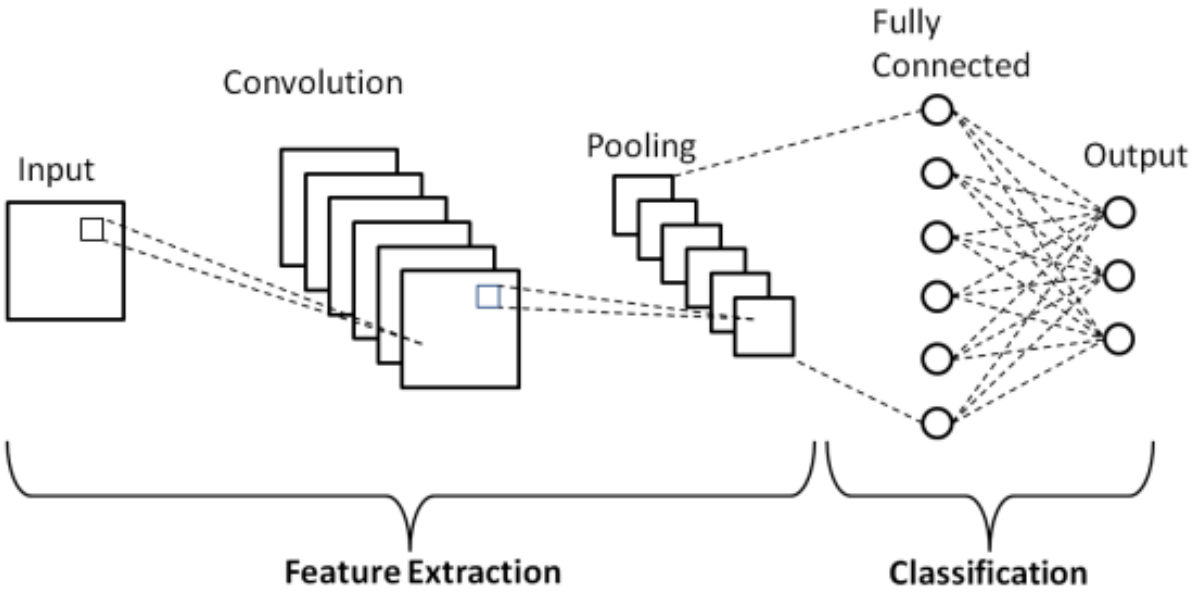


Рисунок 2.1 – Архітектура нейронної мережі CNN

Поговоримо про шари у мережі CNN. На рисунку 2.1 можна побачити що, першим шар у мережі є вхідний шар. Він отримує необроблені значення пікселів зображення і слугує початковою точкою мережі.

Наступним шаром йде шар згортки. Згорткові шари є основними будівельними блоками згорткової нейронної мережі, і саме на них відбувається більшість обчислень у згортковій нейронній мережі. Нейрон на згорткових шарах отримує вхідні дані від нейрона на попередньому шарі й виробляє єдиний вихід. Однак, на відміну від повністю пов'язаних шарів, кожен нейрон згорткової нейронної мережі не з'єднується з кожним нейроном попереднього шару, а лише з тими нейронами, які знаходяться в рецептивному полі. Згорткові шари дозволяють нейронній мережі вивчати особливості в кожній невеликій локальній області, а не в усьому наборі даних. Ось чому згорткові нейронні мережі такі потужні в класифікації зображень: вони можуть вивчати локальні особливості більш ефективно, ніж традиційні нейронні мережі. Крім того, це також зменшує кількість змінних під час навчання, що робить згорткову нейронну мережу простішою в навчанні й менш схильною до перенавчання.

Даний шар виділяє особливості шляхом застосування фільтрів до вхідного зображення. Кожен фільтр навчається виявляти певні шаблони або особливості, такі як краї, кути або текстури. Згорткові шари використовують операцію згортки, коли фільтр ковзає по вхідному зображенню, виконуючи по елементні множення та підсумовування для створення карт особливостей.

Щоб краще зрозуміти шар згортки, потрібно знати важливі терміни, які пов'язані з шаром згортки, й які зазвичай використовують, почнемо з receptive field. У нейронній мережі рецептивним полем кожного нейрона є область попереднього шару, з якої він отримує вхідні дані. У повністю зв'язаному шарі сприйнятливим полем кожного нейрона є весь попередній шар. У шарі, що згортається, сприйнятливе поле зазвичай є квадратом (наприклад, n*n прямокутник пікселів на матриці зображення), а

розмір сприйнятливого поля визначається розміром ядра. Іноді вхідні дані бувають не двовимірними, а тривимірними.

У цьому випадку, додавши до ядра додатковий вимір глибини, рецептивні поля можуть стати 3-вимірними і вивчати тривимірні ознаки, так само як і у випадку з двовимірним набором даних. Під час навчання кожен нейрон шару згортки вивчає особливості свого рецептивного поля.

Також варто зазначити про Kernel, він використовується для виділення ознак у шарі згортки. Зазвичай це квадратна матриця.

У кожному шарі згортки ядро цього шару сканує вхідні дані з попереднього шару, рухаючись зліва направо, а потім зверху вниз. Як проходять обчислювання згортки зображено на рисунку 2.2.

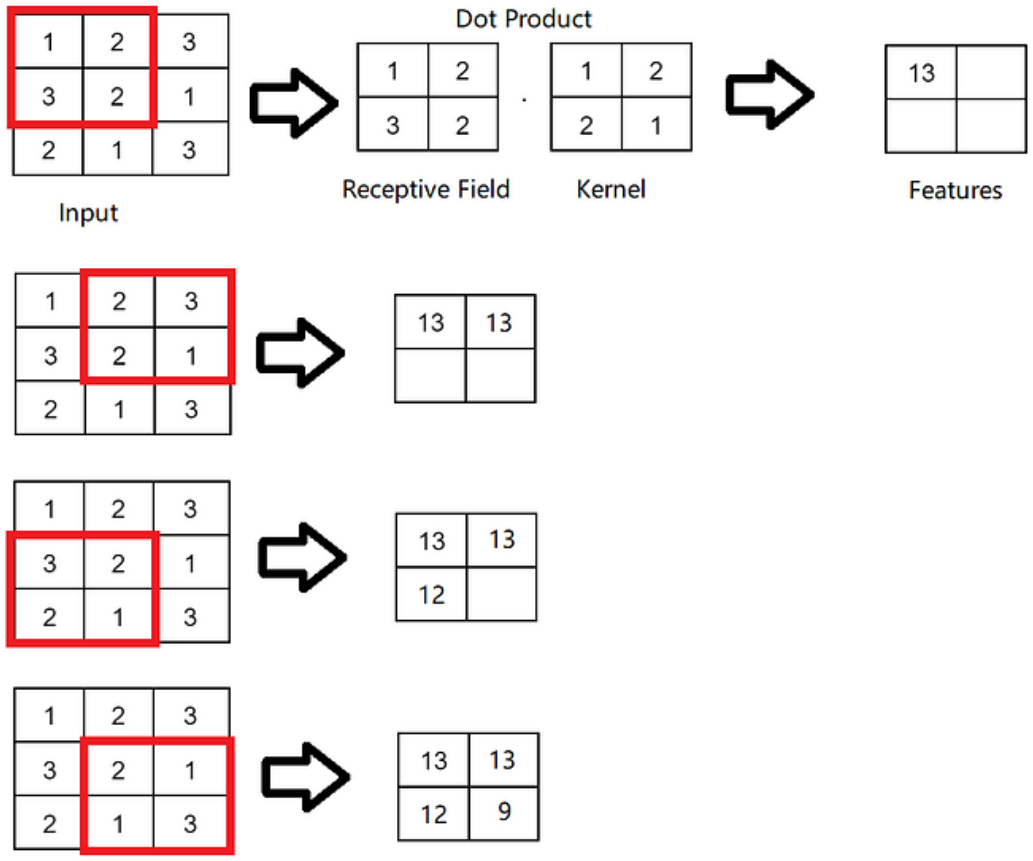


Рисунок 2.2 – Обчислювання згортки за допомогою Kenrel

Кожного разу Kenrel витягує дані з області (яка є рецептивним полем нейрона) на попередньому шарі, яка має такий самий розмір, як і ядро, і виробляє єдиний вихід на основі вхідних даних і ваг ядра. Результуючий набір даних після шару згортки називається ознаками згортки.

Ваги в кожній комірці ядра являють собою коефіцієнт множення для відповідної точки даних на сприйнятливому полі й використовуються для обчислення значення відповідної точки даних на матриці ознак згортки.

Також варто відзначити ще й padding, через те, що Kernel може сканувати лише область набору даних і не може виходити за межі набору даних, точки даних на межі набору даних не можуть бути вивчені належним чином, як точки в центрі набору даних. Крім того, потрібно переконатися, що після обробки шаром згортки розмір набору даних залишається незмінним.

Щоб розв'язати ці проблеми, можна використати прокладку, яка створює додаткові точки даних за межами набору даних, щоб збільшити його розмір, ці точки даних зазвичай мають значення 0. Як проходить обчислення за допомогою padding можна побачити на рисунку 2.3

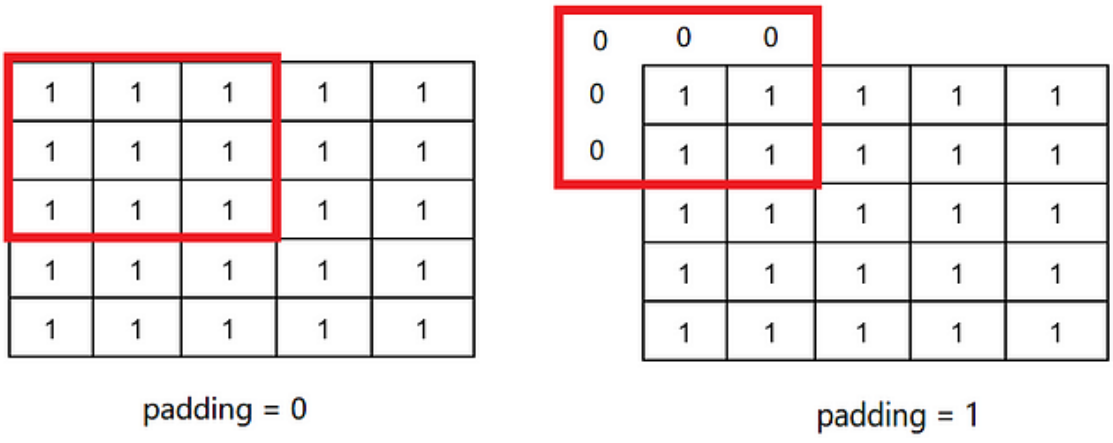


Рисунок 2.3 – Обчислювання згортки за допомогою Padding

Наступним шаром після шару згортки йде шар об'єднання. Зазвичай даний шар зменшує розмір наборів даних, зводячи кілька вхідних даних до одного вихідного. Об'єднувальний шар зазвичай додається після кількох неперервних шарів згортки. Мета додавання об'єднувальних шарів дуже проста, це потрібно для того, щоб зменшити розмір даних, а також кількість змінних, таким чином зменшуючи кількість обчислень, які необхідно виконати.

Існує декілька типів алгоритмів об'єднання даних:

– максимальне об'єднання. вхідні дані з максимальним значенням будуть вихідними.

– середнє об'єднання. середнє значення всіх вхідних даних буде вихідним.

– об'єднання за L2-нормою. значення L2-норми всіх вхідних даних буде виходом.

Передостаннім шаром йде Fully Connected Layer, також відомий як Dense layer, у даному шарі кожен нейрон з'єднується з кожним нейроном попереднього шару. У згортковій нейронній мережі основною метою повністю пов'язаного шару є вивчення особливостей всього вхідного сигналу, а не особливостей локального кластера. Наприклад, в задачах класифікації повністю зв'язані шари можуть бути використані для навчання для визначення типу тварини на вхідному зображенні.

Останнім шаром йде output layer де ми отримаємо остаточні прогнози або класифікації на основі вивчених репрезентацій.

Згорткові нейронні мережі мають кілька переваг, які роблять їх добре придатними для задач класифікації зображень, а саме:

– навчання локальних особливостей. ШНМ чудово навчаються ієрархічно представляти візуальні ознаки, та зосереджуватись на локальних ділянках вхідних даних. Це дозволяє їм вловлювати значущі патерни, незалежно від їхнього розташування на зображенні;

– спільне використання параметрів. CNN використовують спільне використання параметрів у різних просторових точках. Використовуючи однакові ваги фільтрів у різних просторових точках, ШНМ досягають інваріантності перекладу, що дозволяє їм розпізнавати об'єкти незалежно від їхнього положення на зображенні;

– ієрархічна структура. Завдяки кільком рівням абстракції ШНМ можуть вивчати складні об'єкти ієрархічно. Низькорівневі шари фіксують прості об'єкти, такі як краї, тоді як більш глибокі шари поступово вивчають більш складні патерни та представлення об'єктів;

– автоматичне виділення ознак. На відміну від традиційних ручних методів вилучення ознак, CNN автоматично вивчає відповідні ознаки безпосередньо з даних. Це усуває потребу в експертних знаннях про предметну область і ручній інженерії ознак.

Незважаючи на свої дивовижні можливості, CNN також мають певні обмеження:

– велика потреба у навчальних даних. ШНМ потребують великої кількості маркованих навчальних даних для хорошого узагальнення. Недостатня кількість даних може призвести до перенавчання, коли мережа не зможе узагальнити невидимі приклади;

– обчислювальна складність. ШНМ можуть бути дорогими в обчислювальному плані, особливо для великих наборів даних і складних архітектур. Навчання ШНМ часто вимагає потужних графічних процесорів або розподілених обчислювальних ресурсів;

– відсутність інтерпретованості. ШНМ часто вважають «чорними скриньками», що ускладнює інтерпретацію міркувань, які лежать в основі їхніх прогнозів. Зрозуміти, чому ШНМ приймає те чи інше класифікаційне рішення, може бути складно, що обмежує їх використання в областях, де інтерпретованість має вирішальне значення.

2.1.1 LeNet

LeNet була однією з перших і найпростіших згорткових нейронних мереж. Вона була представлена в 1998 році Яном Лекуном та ін. у статті «Навчання на основі градієнта, застосоване до розпізнавання документів» (Gradient-Based Learning Applied to Document Recognition). В основному його використовували для розпізнавання простих рукописних символів і цифрових зображень на основі набору даних MNIST.

На рисунку 2.4 показано архітектуру LeNet-5.

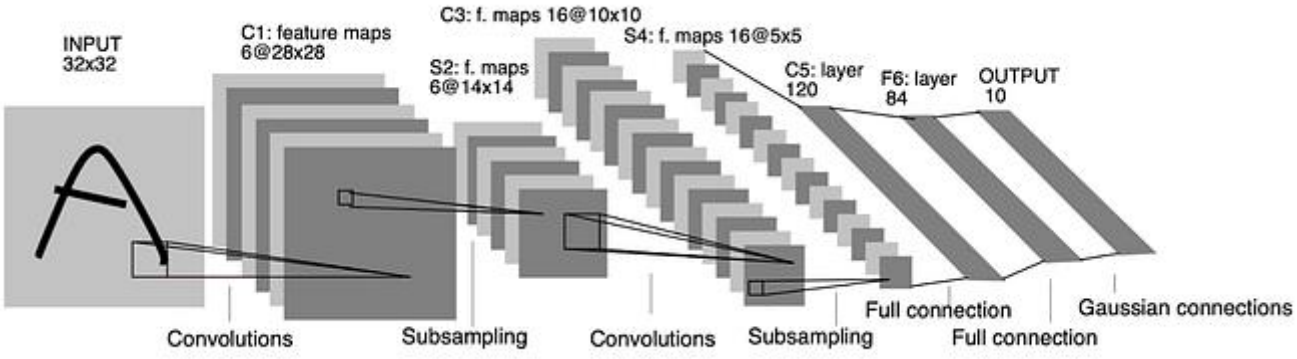


Рисунок 2.4 – Архітектура LeNet-5

Як можна побачити, LeNet-5 має основні компоненти згорткової нейронної мережі: 3 згорткових шари, 2 шари субдискретизації та 2 повністю з'єднаних шари [20]. Функція активації Tanh використовується в кожному шарі, а функція активації softmax використовується в останньому шарі. LeNet-5 є базовим, простим і зрозумілим програмним продуктом. Вона є основою для всіх майбутніх моделей, однак вона не дуже ефективна в розпізнаванні зображень порівняно з більш досконалими моделями, такими як AlexNet або GoogLeNet. Проте це найкраща модель для початківців, які хочуть дізнатися про згорткові нейронні мережі.

2.1.2 AlexNet

AlexNet – це архітектура згорткової нейронної мережі, розроблена Алексом Крижевським та його колегами у 2012 році, і вважається однією з найвпливовіших робіт у галузі комп'ютерного зору. Модель навчалася на базі даних ImageNet, яка містить понад 14 мільйонів зображень у 20 000 категоріях.

На рисунку 2.5 можна побачити архітектуру мережі AlexNet.

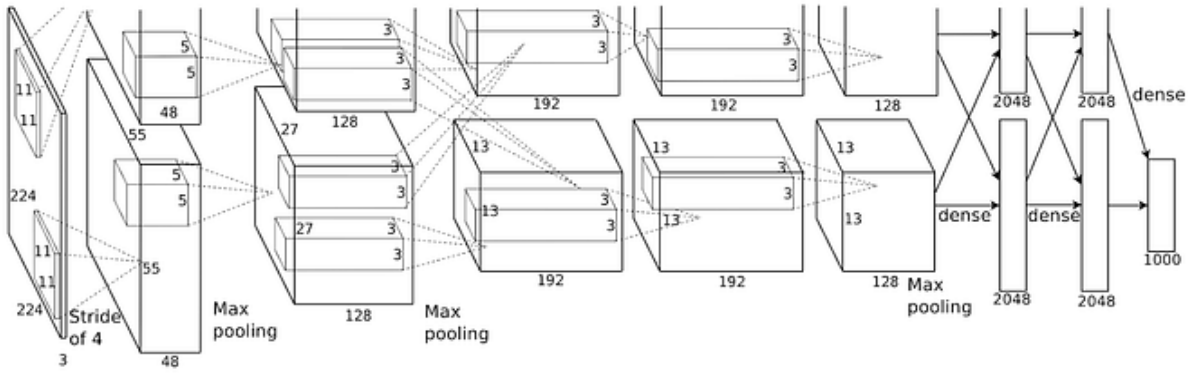


Рисунок 2.5 – Архітектура AlexNet

AlexNet містить 8 шарів: 5 згорткових шарів і 3 повністю з'єднаних шари. Функція активації ReLU використовується в кожному шарі, окрім вихідного шару. Щоб уникнути надмірної підгонки, AlexNet використовувала шари доповнення даних і відсікання. Особливістю AlexNet було використання ReLU замість тангенціальних або сигмоїдних функцій, що дозволило скоротити час навчання в 6 разів. Крім того, це була перша велика модель, яка використовувала кілька графічних процесорів для навчання, що також значно прискорило процес. Результат AlexNet був вражаючим: модель досягла рівня помилок у топ-1 та топ-5 тестових наборів 37,5% та 17,0%, що перевершило найкращі моделі на той час.

2.1.3 VGGNet

VGGNet згорткова нейронна мережа, розроблена Карен Сімонян та Ендрю Зіссерман з Оксфордського університету у 2014 році. VGG дуже ефективна в розумінні та вилученні особливостей із зображень і широко використовується в глибокому навчанні [21]. Він був навчений на наборі даних ImageNet. Він складається з п'яти конфігурацій, які містять від 11 до 19 вагових шарів. Він має кілька згорткових шарів з меншими фільтрами розміром 3x3 замість фільтрів великого розміру. У всіх прихованих шарах використовується функція активації ReLU, а в останньому шарі – softmax. Процес навчання, що використовується у VGG, подібний до AlexNet. Архітектуру VGGNet зображено на рисунку 2.6.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Рисунок 2.6 – Архітектура VGGNet

2.2 Класифікація зображень за допомогою SVM

Машина опорних векторів (SVM) – це потужний алгоритм машинного навчання, який використовується для задач класифікації та регресії. SVM особливо добре підходить для задач класифікації, де метою є розділення точок даних на різні категорії або класи [22]. SVM працює шляхом пошуку оптимальної гіперплощини, яка найкраще розділяє дані на окремі класи, максимізуючи маржу (відстань) між гіперплощиною та найближчими точками даних кожного класу.

Щоб далі поглиблюватись у роботу SVM потрібно поговорити про основні моменти роботи SVM, а саме про основні поняття:

– hyperplane (гіперплощина). У задачі бінарної класифікації (поділ даних на два класи) гіперплощина є межею рішення. Це лінія (у 2D), площина (у 3D) або гіперплощина (у вищих вимірах), яка розділяє точки даних на різні класи. SVM прагне знайти гіперплощину, яка максимізує відстань між гіперплощиною і найближчими точками даних кожного класу;

– support Vectors (опорні вектори). Це точки даних, які знаходяться найближче до гіперплощини й мають найменший запас. Опорні вектори мають вирішальне значення для визначення положення та орієнтації гіперплощини;

– margin (відступ). Відступ – це відстань між гіперплощиною і найближчими точками даних кожного класу. SVM прагне максимізувати цю відстань, оскільки більша відстань зазвичай призводить до кращого узагальнення нових, ще не бачених даних;

– kernel Trick. SVM може обробляти як лінійно відокремлювані, так і нелінійно відокремлювані дані. Для нелінійних даних він використовує трюк ядра для відображення даних у простір вищої розмірності, де можливе лінійне розділення. Найпоширеніші ядра включають лінійні, поліноміальні та ядра радіально-базисних функцій (RBF);

– C Parameter. SVM має гіперпараметр «C», який контролює компроміс між максимізацією маржі та мінімізацією помилок класифікації. Менші значення C підкреслюють більший запас, але можуть призвести до помилкової класифікації, тоді як більші значення C надають перевагу точній класифікації, але можуть призвести до вужчого запасу.

SVM працює, знаходячи гіперплощину в просторі ознак, яка найкраще розділяє точки даних на два класи. Гіперплощина – це плоска поверхня, яка ділить простір на дві частини. У випадку SVM гіперплощина – це лінія, якщо дані мають дві ознаки, площина, якщо дані мають три ознаки, і так далі.

Загальну архітектуру SVM можна побачити на рисунку 2.7.

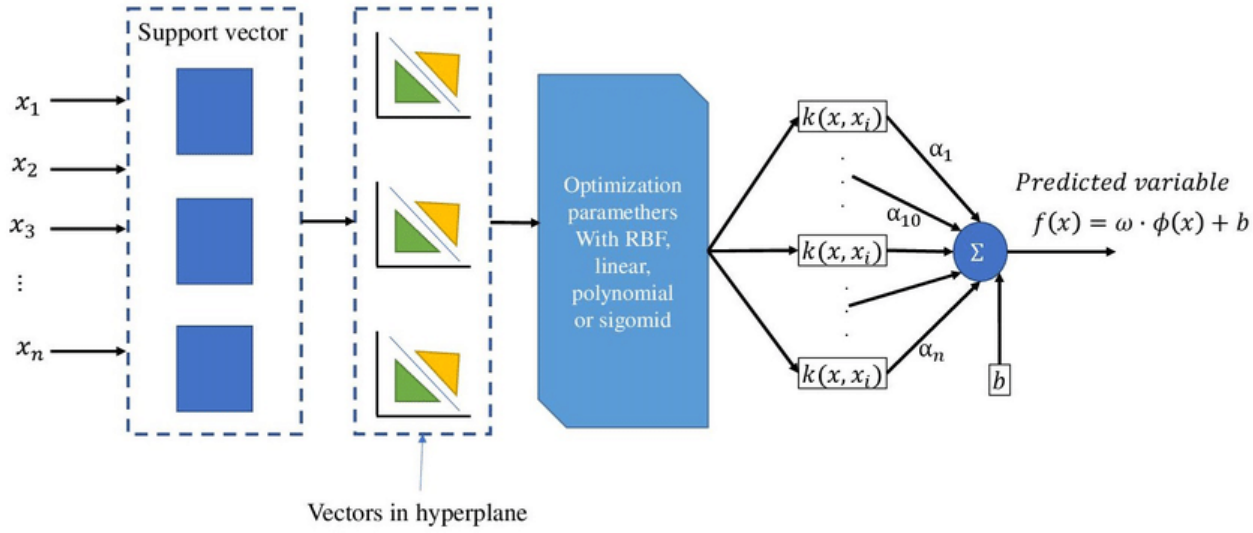


Рисунок 2.7 – Загальна архітектура моделі обробки опорних векторів (SVM)

Мета SVM знайти гіперплощину, яка максимізує маржу між двома класами. Максимізація відриву допомагає гарантувати, що гіперплощина зможе добре узагальнювати нові дані. Щоб знайти оптимальну гіперплощину, SVM використовує алгоритм квадратичного програмування. Квадратичне програмування – це тип алгоритму оптимізації, який можна

використовувати для знаходження мінімального або максимального значення функції за умови дотримання набору обмежень. У випадку SVM обмеження полягають у тому, що всі точки даних повинні бути правильно класифіковані з одного боку гіперплощини [23].

Після знаходження оптимальної гіперплощини SVM можна використовувати для класифікації нових точок даних, просто визначаючи, на який бік гіперплощини вони потрапляють. На рисунку 2.8 можна візуалізація методу опорних векторів для класифікації.

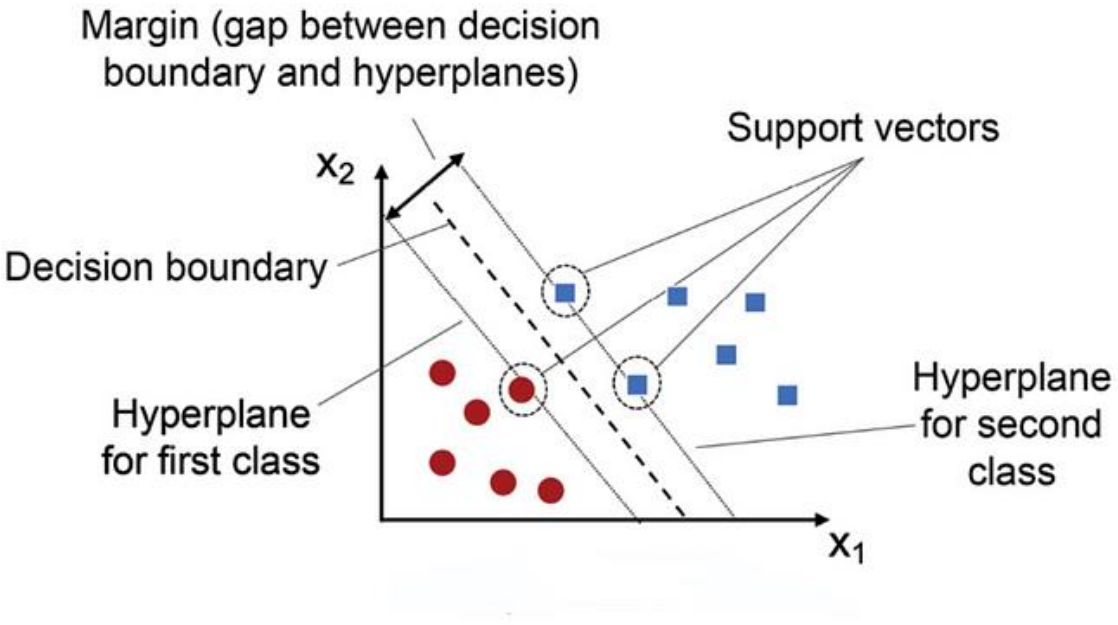


Рисунок 2.8 – Візуалізація методу опорних векторів (SVM) для класифікації

Було визначено, що SVM є дуже потужним алгоритм машинного навчання, який широко застосовується в різних галузях науки й техніки. В основі SVM лежить ідея пошуку оптимальної розділювальної гіперплощини між класами даних. Цей метод має низку переваг, хороша узагальнювальна здатність, здатність працювати з даними та ефективність у разі обмежених обсягів навчальних даних.

Метод опорних векторів широко використовується в різних сферах, включаючи класифікацію, регресію та виявлення аномалій. У бінарній класифікації, SVM ефективно вирішує завдання виявлення спаму, аналізу настроїв та медичної діагностики, такої як виявлення раку. Для многокласової класифікації, використовуються методи, такі як «один проти всіх» або «один проти одного», і застосовуються для розпізнавання рукописних цифр та категоризації документів. У регресії, SVM використовується для прогнозування числових значень, що застосовується у фінансовому прогнозуванні та оцінці цін на нерухомість. Також SVM ефективно виявляє аномалії в наборах даних, використовуючись у кібербезпеці та контролі якості.

У сфері обробки зображень та комп'ютерного зору, SVM використовується для класифікації об'єктів на зображеннях та розпізнавання облич. Додатково, SVM відіграє важливу роль у розпізнаванні рукописного тексту, в системах оптичного розпізнавання символів (OCR), що застосовується у процесі автоматизованої обробки документів та цифруванні історичних матеріалів. Також SVM знаходить своє застосування в дистанційному зондуванні та геопросторовому аналізі для класифікації земельного покриття, картографування використання земель та виявлення рослинності в дистанційних дистанціях. І у загальному підсумку, SVM використовується в широкому спектрі областей, від біомедицини та біоінформатики до фінансового прогнозування та систем рекомендацій.

3 АНАЛІЗ НАЯВНИХ МЕТОДІВ ОБРОБКИ Й АНАЛІЗУ ДЕРМАТОСКОПІЧНИХ ЗОБРАЖЕНЬ

Обробка й аналіз дерматоскопічних зображень дуже важливий етап для підвищення точності класифікації захворювань шкіри. Зазвичай дерматоскопічні зображення отримуються за допомогою дермоскопії. Автоматичне розпізнавання дерматоскопічних зображеннях все ще залишається складним завданням, оскільки має кілька проблем, а саме низький контраст між ураженими ділянками шкіри і нормальними ділянками шкіри ускладнює точну сегментацію ділянок ураження [24]. Також деякі хвороби можуть мати високий ступінь візуальної схожості, що призводить до труднощів у відрізненні. Також варто пам’ятати про варіації стану шкіри, наприклад, колір шкіри, природне волосся або вени, у різних пацієнтів призводять до різного зовнішнього вигляду меланоми, з точки зору кольору й текстури.

Було з’ясовано, що сегментація ураження шкіри є важливим кроком для більшості класифікаційних підходів. Точна сегментація може вплинути на точність подальшої класифікації ураження, було проведено багато досліджень спрямованих на отримання якісних результатів сегментації вогнищ ураження й було доказано, що сегментація ураження шкіри дає підвищення класифікації дерматоскопічних зображень. Наприклад, як що подивитися на все запропоновані алгоритми, а саме на некерований алгоритм, названий Independent Histogram Pursuit (IHP), який використовується для сегментації ураження шкіри. Даний алгоритм був протестований на багатьох різних наборів даних, і показав точність 97%, що є дуже гарним показником, також було проведено велику кількість досліджень, де порівнювалися різні методі для сегментації ураження шкіри на дерматоскопічних зображеннях, й усі дослідження підвередили, що різні методи сегментації ураження шкіри допомагають підвищити точність класифікації [25].

На основі результатів сегментації можна витягти створені вручну ознаки для розпізнавання різних захворювань шкіри. Наприклад, дослідник Шефер використовував підхід автоматичного визначення меж для сегментації області ураження, а потім зібрав витягнуті ознаки, тобто форму, текстуру і колір, для розпізнавання різних захворювань шкірі. З іншого боку, в деяких дослідженнях було зроблено спробу безпосередньо використовувати створені вручну ознаки для розпізнавання меланоми без етапу сегментації. На відміну від підходів, що використовують створені вручну функції, мережі глибокого навчання використовують ієрархічні структури для автоматичного вилучення ознак. У зв'язку з проривами, досягнутими глибоким навчанням у дедалі більшій кількості завдань обробки зображень, деякі дослідження почали застосовувати підходи глибокого навчання для розпізнавання меланоми. Codella et al. запропонував гібридний підхід, що об'єднує згорткову нейронну мережу (CNN), розріджене кодування і метод опорних векторів (SVM) для виявлення й класифікації різних захворювань шкіри.

Незважаючи на те, що було запропоновано багато робіт й зроблено багато досліджень, все ще існує потенціал для покращення ефективності як сегментації, так і класифікації уражень шкіри. Міжнародна співпраця з візуалізації шкіри (ISIC) – це співпраця, що зосереджується на автоматичному аналізі уражень шкіри, і з 2016 року постійно розширює свої набори даних. У 2017 році ISIC випустила анотовані набори даних для трьох завдань обробки зображень уражень шкіри, включаючи сегментацію уражень, виділення дерматологічних ознак і класифікацію уражень, щоб допомогти дослідникам підвищити точність автоматичних методів виявлення меланоми. На відміну від широко вивчених сегментації та класифікації уражень, виділення дерматологічних ознак є новим завданням у цій галузі, відповідно, для вирішення цієї проблеми було запропоновано мало досліджень. Можна зробити висновки, що попередня обробка дуже

потрібна для підвищення точності класифікації захворювань, її можна використовувати ще й для підвищення продуктивності системи.

3.1 Зміна роздільного розширення зображення

Наприклад, представимо, що до нас приходять картинки розміром 1000 на 700 пікселей або більше. Дана роздільна здатність зображення дуже велика, що може сповільнити систему й можливо нам потрібно тільки 50% від оригінальної роздільності зображення. Тому необхідно змінити масштаб зображень для мережі глибокого навчання, оскільки пряма зміна розміру зображення може спотворити форму ураження шкіри, спочатку потрібно обрізати центральну область зображення ураження, а потім пропорційно змінити розмір цієї області до нижчої роздільної здатності. Розмір центрального квадрата можна встановити на рівні 0,8 висоти зображення і автоматично обрізати відносно центру зображення. Приклад можна побачити на рисунку 3.1, такий підхід не тільки збільшує площу ураження для виявлення ознак, але й зберігає форму ураження шкіри.

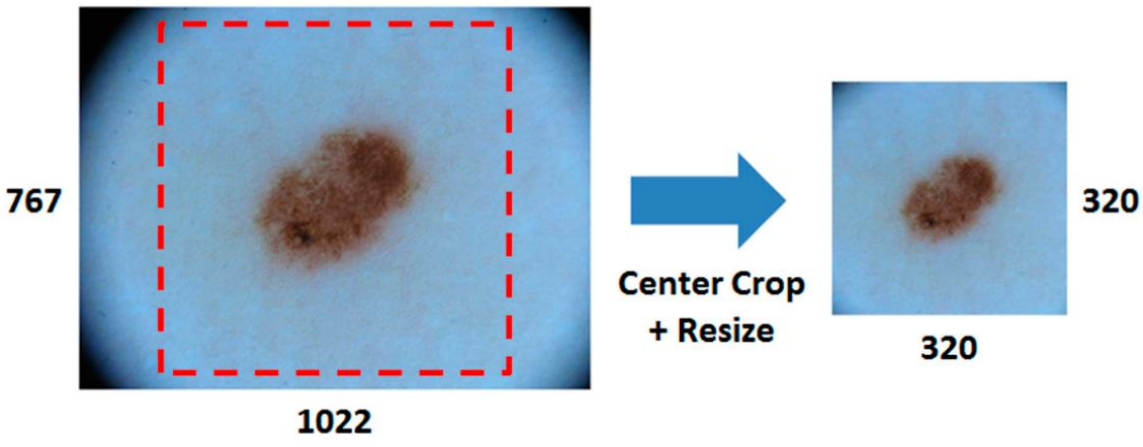


Рисунок 3.1 – Попередня обробка зображення ураження шкіри за допомогою ресайзу початкового зображення

3.2 Виділення карти ймовірності ураження шкіри

З точними картами ймовірності різних категорій уражень на зображенні шкіри пов'язані численні можливості в галузі дерматології та медичного дослідження. Ці дані надають корисну інформацію для автоматичної індексації, раннього виявлення потенційно злоякісних уражень та підтримки медичних фахівців у прийнятті рішень. Вони також відіграють важливу роль у медичному навчанні та дослідженнях, допомагаючи розуміти характеристики різних типів уражень та їх вплив на практику дерматології.

Оскільки точні карти ймовірності різних категорій ураження на зображенні ураження шкіри надають корисну інформацію, то потрібно детальніше розібрати потенційний компонент, який буде використовуватися для уточнення карт ймовірності ураження шкіри. Для того, щоб це зробити, потрібно нормалізувати отримані значення з зображення до [0, 1], далі $vi(x,y)$ значення (x, y) на і-й карті, нормалізовану ймовірність ураження шкіри, а саме pi можна вирахувати за наступною формулою 2.1.

$$\mathrm{p_i(x,y)} = \frac{\mathrm{v_i(x,y)} - \min(\mathrm{v_i(x,y)})}{\sum_{\mathrm{i}=1}^{3}(\mathrm{v_i(x,y)} - \min(\mathrm{v_i(x,y)}))} \ \mathrm{i} \in 1,2,3 \qquad (2.1)$$

де (x, y) – нормалізована ймовірність ураження шкіри по пікселям.

Кожен піксель в області ураження має різну важливість для класифікації ураження. На рисунку 3.2 видно, що область біля межі ураження на деяких зображеннях ураження шкіри має більш схожий вигляд, тобто колір/текстуру, на шкіру, ніж центральна область, сині лінії на рисунку це виявлені межі уражень. Зона ураження з подібними до шкіри характеристиками може надавати менше інформації для розпізнавання

ураження. Тому відстані від пікселів до найближчої межі використовуються для представлення важливості пікселів для класифікації ураження.

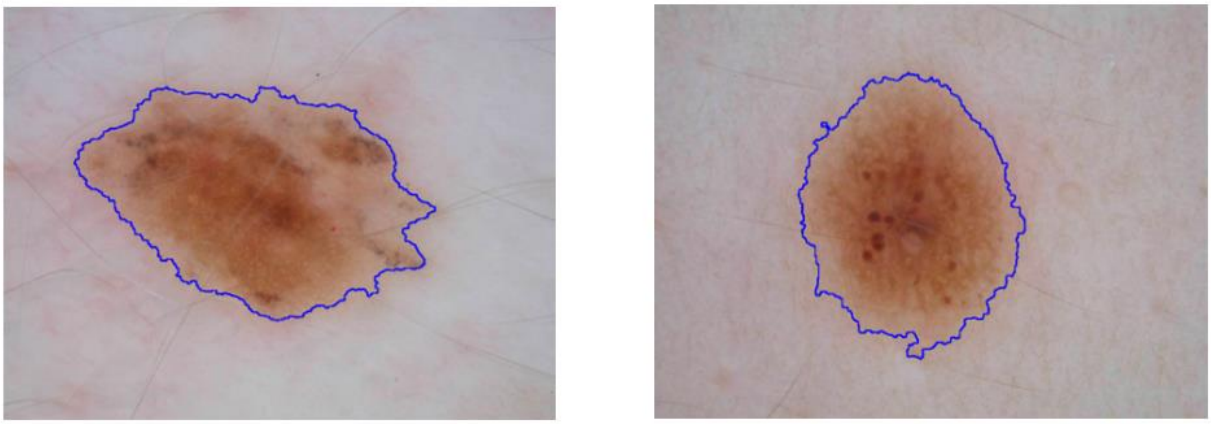


Рисунок 3.2 – Виділена потенційно уражена ділянка шкіри

Приклади карт ймовірності захворювання можна побачити на рисунку 3.3. Кольори на карті відображають вагу відповідних пікселів. Карта відстаней множиться на кожну з нормалізованих можливостей для створення уточнених карт, далі потрібно усереднити можливості в зоні ураження на уточнених картах, щоб отримати індекси для різних категорій ураження шкіри.

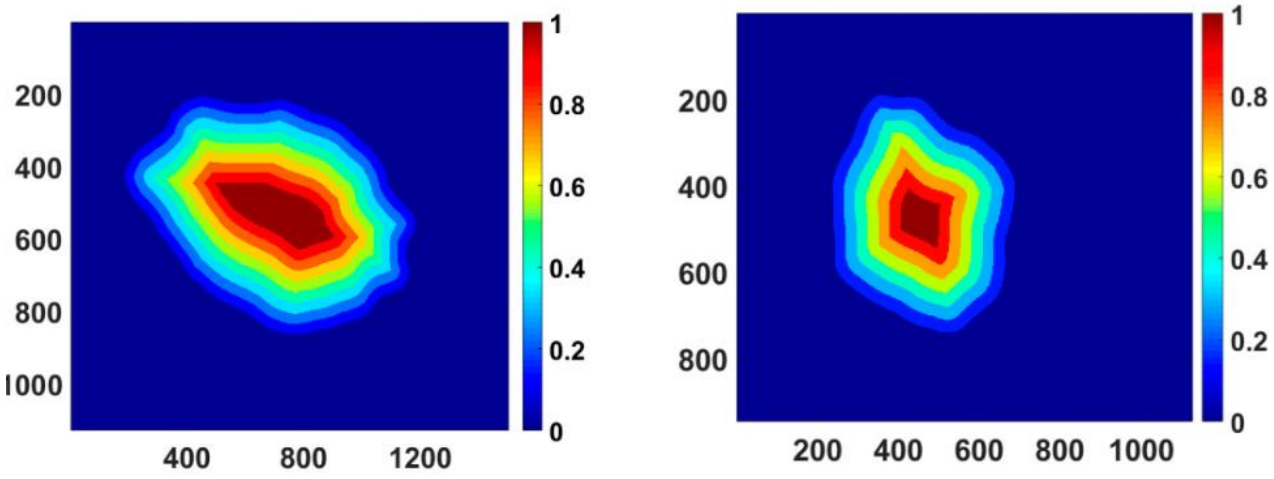


Рисунок 3.3 – Карта ймовірності потенційно ураженої ділянки шкіри

3.4 Фільтр Гаусса

Фільтр Гаусса застосовується для видалення небажаних пікселів із зображень, зазвичай це можуть бути артефакти, або шуми на зображенні. Шуми або артефакти можуть бути, спричинені природними факторами, такими як волосся, бульбашки повітря або зовнішніми факторами, такими як освітлення, тіні та різні пристрої захоплення. Цей шум присутній на медичних зображеннях.

Даним методом небажані пікселі замінюються шляхом обчислення середнього значення сусідніх або навколишній пікселів небажаних пікселів, які залежать від гаусівського розподілу.

Гауссівський фільтр належить до лінійних фільтрів. Він використовується для зменшення або усунення шуму та покращення зображень. Згладжування зображень за допомогою методу Гауса призводить до розмиття зображення. Ступінь згладжування визначається середньоквадратичним відхиленням за Гауссом. Фільтр Гаусса використовується для видалення небажаних пікселів, таких як волосся і бульбашки повітря, шляхом заміни небажаних пікселів усередненням пікселів сусідніх пікселів, це можна вирахувати за наступною формулою 2.2

$$h(x,y) = \frac{1}{2\pi\,\sigma^2}\, e\, \frac{x^2-y^2}{2\sigma^2} \qquad (2.2)$$

де x – відстань від початку координат по горизонтальній осі;

y – відстань від початку координат по вертикальній осі;

σ – стандартне відхилення гаусівського розподілу.

На рисунку 3.4 й 3.5 можна побачити процес покращення зображень за допомогою фільтра Гаусса.

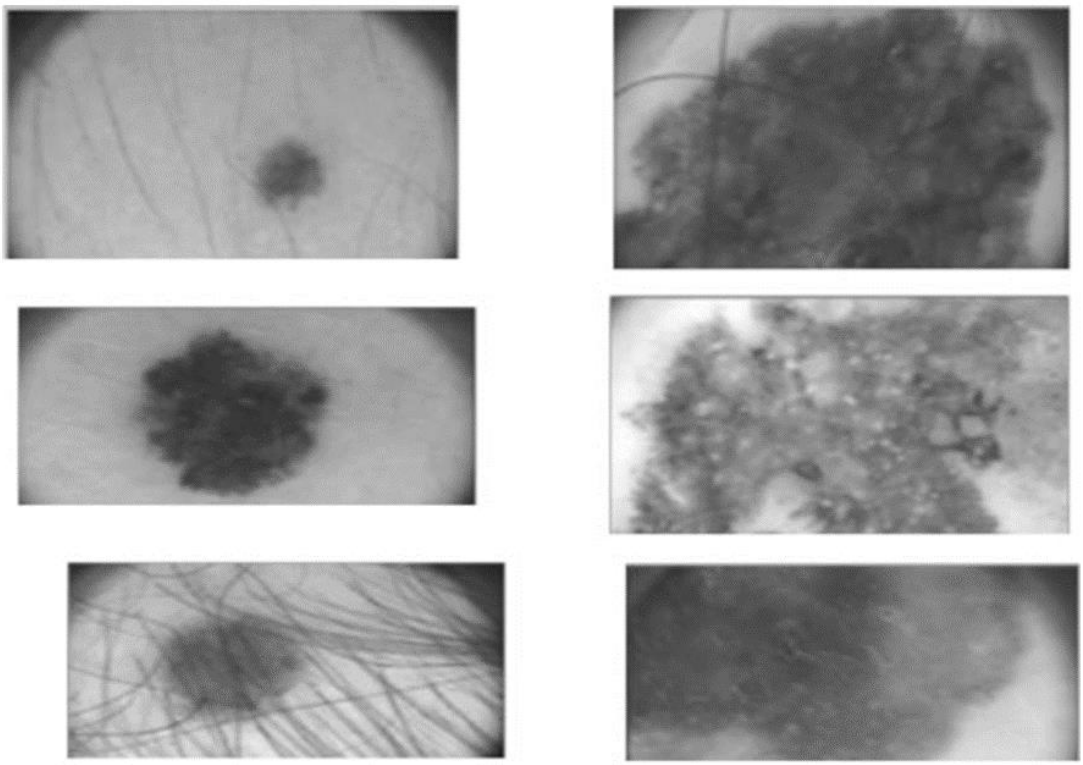


Рисунок 3.4 – Результат використання фільтра Гаусса

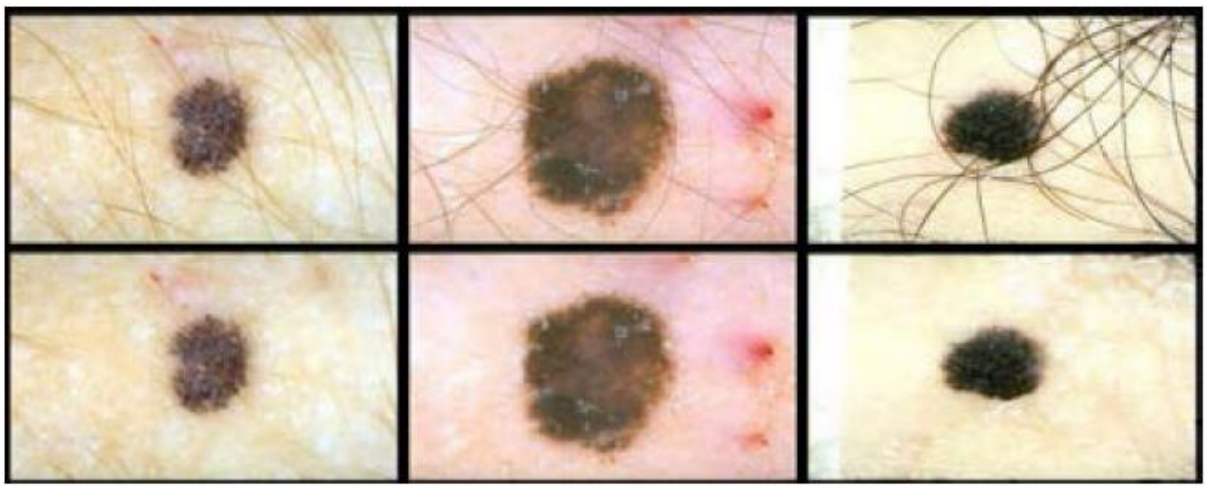


Рисунок 3.5 – Результат використання фільтра Гаусса

4 ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА ПРОВЕДЕННЯ ЕКСПЕРИМЕНТІВ

4.1 Обґрунтування вибраної мови програмування

Для програмної реалізації було вибрано мову програмування Python. Дана мова програмування використовуються у різних сферах, але найчастіше її використовують у сфері штучного інтелекту. Python має дуже велику кількість бібліотек, які направлені, наприклад на машинне навчання, комп'ютерний зір й глибоке навчання. Для написання програми були використані наступні бібліотеки: pytorch, numpy, pandas, torchvision й sklearn. Для візуалізації результатів була використана бібліотека matplotlib.

4.2 Огляд набору даних

Для проведення практичних експериментів було обрано набір даних HAM10000. Особливістю цього набору даних є те, що у цьому набору даних є дерматоскопічні зображення з різних популяцій. Кінцевий набір даних складається з 10015 дерматоскопічних зображень, які можуть використовуватися для машинного навчання [26]. Також даний набір даних включає репрезентативну колекцію всіх важливих діагностичних категорій у сфері пігментних уражень. Детальну інформацію можна побачити у таблиці 4.1.

Таблиця 4.1 – Інформація про набор даних HAM10000

Повна назва	Скорочена назва	Кількість екземплярів
actinic keratoses and intraepithelial carcinoma / Bowen's disease	akiec	327
basal cell carcinoma	bcc	514
benign keratosis-like	bkl	1099

Продовження таблиці 4.1

dermatofibroma	df	115
melanoma	mel	1113
melanocytic nevi	nv	6705
angiomas, angiokeratomas, pyogenic granulomas and hemorrhage	vasc	142

Приклад дерматоскопічного зображення з набору даних, а саме з категорії basal cell carcinoma [27] зображено на рисунку 4.1

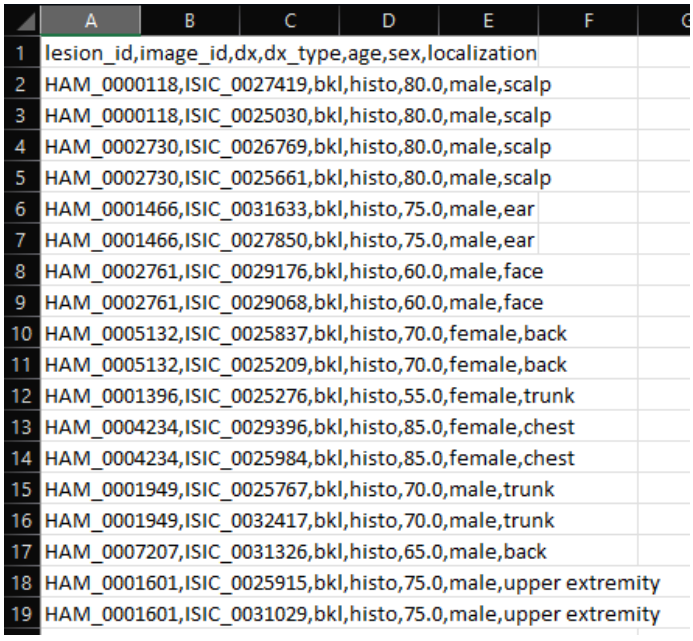
Рисунок 4.1 – Зображення захворювання шкіри basal cell carcinoma

Також, основною проблемою з пошуком набору даних полягала у тому, що потрібна не тільки велика кількість зображень, а ще потрібно мати помічені дані, вибраний набір даних має файл де можна побачити детальну інформацію про кожне зображення [28]. У таблиці 4.2 можна побачити з чого складається файл з метаданими.

Таблиця 4.2 – Метадані набору даних

Назва параметру	Приклад параметру	Пояснення
lesion_id	HAM_0000118	Унікальний ідентифікатор у наборі даних
image_id	ISIC_0027419	Унікальна назва зображення
dx	Bkl(Усі типи ми розглядали у таблиці 4.1)	Вид захворювання
dx_type	histo, follow_up й consensus	Як поставили діагноз
age	80.0	Вік пацієнта
sex	Male й female	Стать пацієнта
localization	Scalp, back, upper extremity	Де знаходиться уражений ділянка шкіри

На рисунку 4.2 можна побачити як метадані оригінально збережені у файлі с метаданими.



	A	B	C	D	E	F	G
1	lesion_id,image_id,dx,dx_type,age,sex,localization						
2	HAM_0000118,ISIC_0027419,bkl,histo,80.0,male,scalp						
3	HAM_0000118,ISIC_0025030,bkl,histo,80.0,male,scalp						
4	HAM_0002730,ISIC_0026769,bkl,histo,80.0,male,scalp						
5	HAM_0002730,ISIC_0025661,bkl,histo,80.0,male,scalp						
6	HAM_0001466,ISIC_0031633,bkl,histo,75.0,male,ear						
7	HAM_0001466,ISIC_0027850,bkl,histo,75.0,male,ear						
8	HAM_0002761,ISIC_0029176,bkl,histo,60.0,male,face						
9	HAM_0002761,ISIC_0029068,bkl,histo,60.0,male,face						
10	HAM_0005132,ISIC_0025837,bkl,histo,70.0,female,back						
11	HAM_0005132,ISIC_0025209,bkl,histo,70.0,female,back						
12	HAM_0001396,ISIC_0025276,bkl,histo,55.0,female,trunk						
13	HAM_0004234,ISIC_0029396,bkl,histo,85.0,female,chest						
14	HAM_0004234,ISIC_0025984,bkl,histo,85.0,female,chest						
15	HAM_0001949,ISIC_0025767,bkl,histo,70.0,male,trunk						
16	HAM_0001949,ISIC_0032417,bkl,histo,70.0,male,trunk						
17	HAM_0007207,ISIC_0031326,bkl,histo,65.0,male,back						
18	HAM_0001601,ISIC_0025915,bkl,histo,75.0,male,upper extremity						
19	HAM_0001601,ISIC_0031029,bkl,histo,75.0,male,upper extremity						

Рисунок 4.2 – Зображення як зберігаються метадані

4.3 Попередній аналіз даних

Після поверхневого огляду набору даних, проаналізуємо увесь набір даних, щоб мати загальну картину у представлених даних. Основну увагу приділимо кількості захворювань, щоб зрозуміти розподіл по категоріях захворювань у наборі даних. Розподіл захворювань можна побачити на рисунку 4.3.

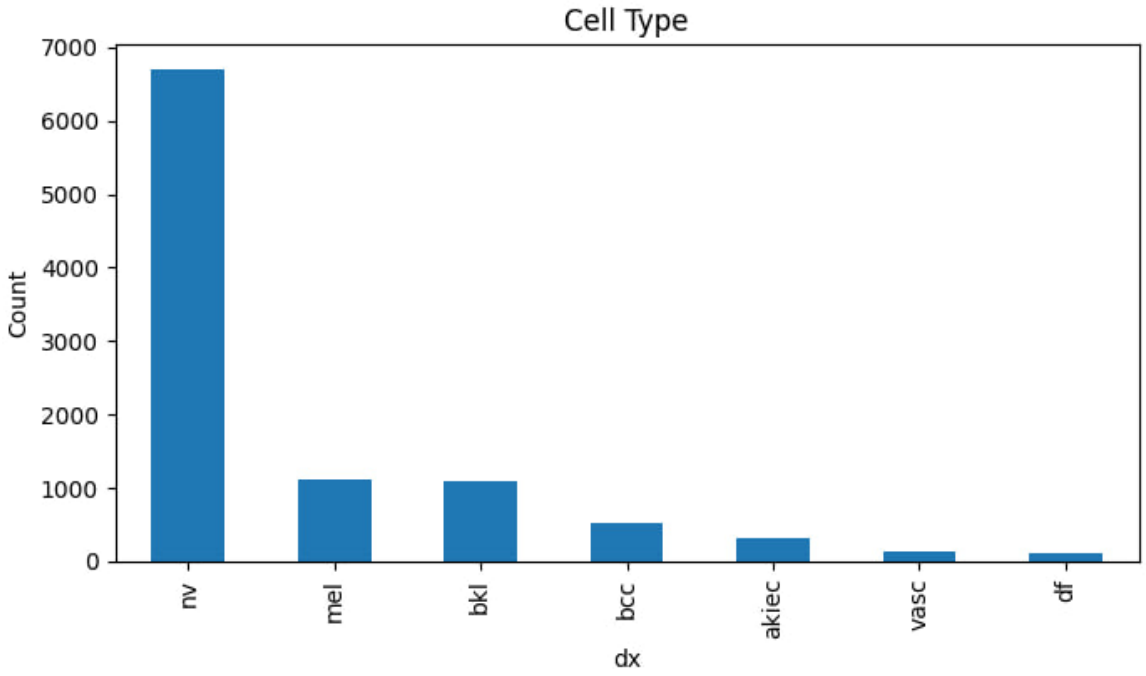


Рисунок 4.3 – Розподіл хвороб у наборі даних

З розподілу захворювань на рисунку 4.3, можна побачити, що існує значний дисбаланс у кількості зображень для кожного типу клітин. Для типу ураження melanocytic nevi(nv) 6705 з 10015. Це набагато більше зображень, ніж для інших типів, таких як dermatofibroma(df) 115 з 10015.

На жаль, але це звичайне явище для більшості наборів даних. У даному прикладу можна побачити брак деяких захворювань, через обмежену кількість пацієнтів. Можна зробити висновок, що буде гарною

практикою, попередньо проаналізувати дані, з якими буде йти подальша робота.

Варто зробити увагу на те, що під час аналізу набору даних можна побачити дисбаланс класів. Цю проблему дуже важливо вирішити, тому, що в ході тренування й отримання результатів є вірогідність отримати неоптимальні результати, через те, що мережа буде упередженою до представлених класів і не матиме можливості вивчити розподіл недостатньо до інших представлених класів. Тому потрібно використати функцію втрат, для того, щоб призначити вагу кожному класу й забезпечити збалансоване навчання між класами. Кастомна функція зображена на рисунку 4.4, а на рисунку 4.5 можна побачити оновлені ваги для класів.

```
2 usages
def estimate_weights_mfb(label):
    import warnings

    warnings.filterwarnings( action: "ignore", category=FutureWarning)

    class_weights = np.zeros_like(label, dtype=np.float64)
    counts = np.zeros_like(label)
    for i, l in enumerate(label):
        counts[i] = metadata[metadata['dx'] == str(l)]['dx'].value_counts()[0]
    counts = counts.astype(np.float64)
    median_freq = np.median(counts)
    for i, l in enumerate(label):
        class_weights[i] = median_freq / counts[i]
    return class_weights

classweight = estimate_weights_mfb(label)
for i in range(len(label)):
    print(label[i], ":", classweight[i])
```

Рисунок 4.4 – Функція для обчислення нових коефіцієнтів ваги для класів

```
akiec : 1.5718654434250765
bcc : 1.0
bkl : 0.467697907188353
df : 4.469565217391304
mel : 0.4618149146451033
nv : 0.07665920954511558
vasc : 3.619718309859155
```

Рисунок 4.5 – Нові коефіцієнти ваги для класів

Також був проведений аналіз віку пацієнтів й де находиться захворювання у пацієнта. Детальний розподіл по віку пацієнтів зображений на рисунку 4.6, а на рисунку 4.7 можна побачити розподіл де знаходиться уражена шкіра.

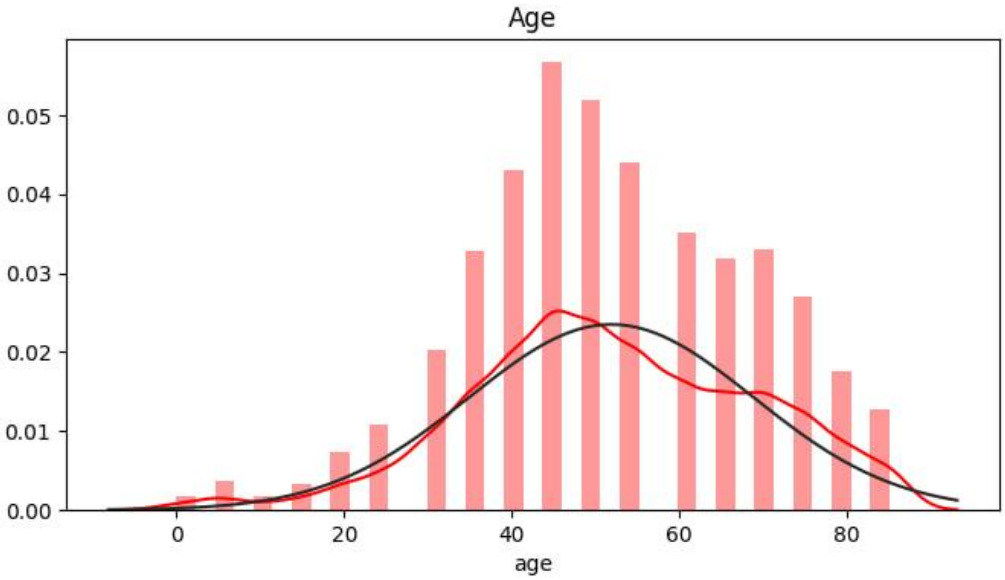


Рисунок 4.6 – Розподіл віку пацієнтів у наборі даних

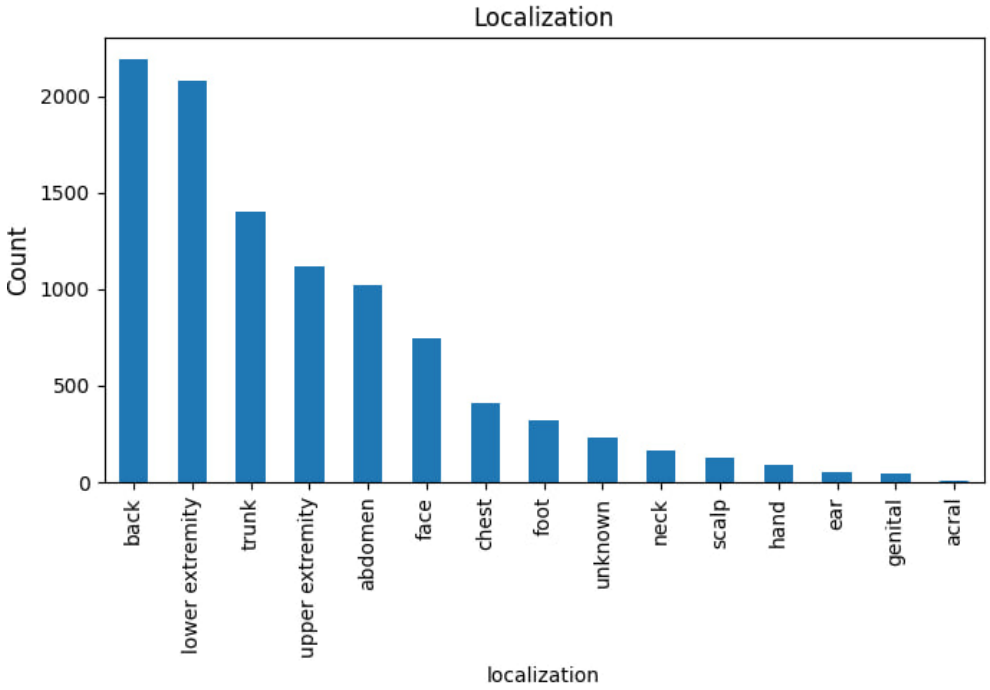


Рисунок 4.7 – Розподіл місцезнаходження ураженої шкірі у наборі даних

4.4 Попередня обробка й розподіл даних

Після попереднього й поверхневого аналізу набору даних, потрібно згрупувати картинки з нашого набору до їх класів. Було з’ясовано що, наш набір даних це картинки й окремо йде файл з метаданими, тож потрібно розробити код, який розподілить зображення по класах за допомогою метаданих [29]. Розроблений код зображений на рисунку 4.8.

```
dest_dir = "C:/Users/atom5/PycharmProjects/D1/HAM/HAM10K/"

if not os.path.exists(dest_dir):
    # A path to the folder which has all the images:
    data_dir = os.getcwd() + "/HAM/HAM10000"

    # A path to the folder where you want to store the rearranged images:


    # Read the metadata file:
    metadata = pd.read_csv(os.getcwd() + '/HAM/HAM10000_metadata.csv')
    label = ['bkl', 'nv', 'df', 'mel', 'vasc', 'bcc', 'akiec']
    label_images = []

    # Copy the images into new folder structure:
    for i in label:
        os.mkdir(dest_dir + str(i) + "/")
        sample = metadata[metadata['dx'] == i]['image_id']
        label_images.extend(sample)
        for id in label_images:
            shutil.copyfile((data_dir + "/" + id + ".jpg"), (dest_dir + i + "/" + id + ".jpg"))
        label_images = []
```

Рисунок 4.8 – Розроблений код для розподілу зображень по класам

Розроблений код автоматизує процес розподілу зображень відповідно до їх класів у новій структурі папок. Він перебирає метадані, що містять інформацію про класи зображень, та копіює кожне зображення відповідно до його класу у відповідну папку.

Це створює більш зручну та організовану структуру для подальшого аналізу та використання зображень у різних обчислювальних завданнях.

Перед тим, як почати вже тренувати мережу й проводити тестування, можна ще зробити збагачення даних. Збагачення даних, це допоміжний, але важливий процес й інструмент для наповнення набору даних більшою кількістю зображень і збільшення дисперсії, на яку мережа наражається під час навчання. Зазвичай використовують такі методи, як обертання, зміни точку зору, зміна освітлення або комбінацію даних методів [30].

Таки методи можуть допомоги мережі стати біль стійкий до змін у вхідних зображеннях. З усіх згаданих практик будуть використовуватися тільки повертання зображення на 180 градусів, й на 60 градусів. Приклад зображення яке було повернуто на 60 градусів зображено на рисунку 4.9.

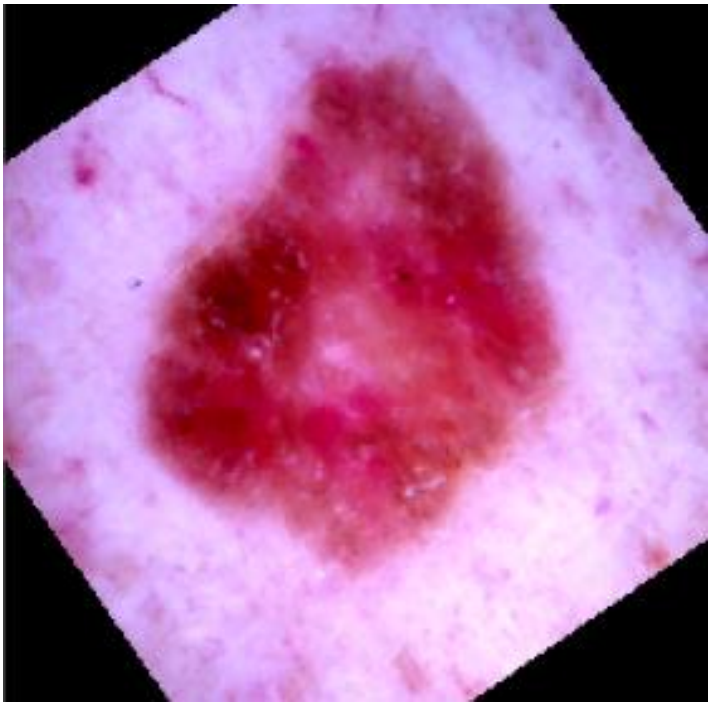


Рисунок 4.9 – Зображення яке повернули на 60 градусів

Також варто пам’ятати про те, що варто нормалізувати наші вхідні дані, в якості даних у нас це пікселі. Нормалізація даних відбувається шляхом віднімання середнього значення інтенсивності колірного каналу з кожного пікселя, а потім ділення результату на стандартне відхилення того

ж каналу. Код який відповідає за обробку тестових даних зображений на рисунку 4.10.

```
transform_train = transforms.Compose([
                    transforms.Resize((224,224)),
                    transforms.RandomHorizontalFlip(),
                    transforms.RandomRotation(degrees=60),
                    transforms.ToTensor(),
                    transforms.Normalize(norm_mean, norm_std),
                    ])
```

Рисунок 4.10 – Код який відповідає за попередню обробку тестових даних

Після аналізу даних й попередньої обробки, перейдемо до розділення набору даних на частини, а саме на тренувальну, валідуючу й тестову. Завдяки розподілу даних на три різні категорії, ми уникнемо надмірного пристосування та покращимо узагальнення моделі.

– тренувальний набір даних. Частина набору даних, яка використовується для навчання кінцевої моделі;

– валідуючий набір даних. Частина набору даних, яка використовується для забезпечення неупередженої оцінки відповідності моделі навчальному набору даних при налаштуванні гіперпараметрів моделі, наприклад швидкості й навчання;

– тестовий набір даних. Частина набору даних, яка не використовується в процесі навчання. Даний набір даних забезпечує неупереджені дані для остаточної оцінки навченої моделі.

У наборах медичних зображень важливо розділяти дані на рівні пацієнтів, тобто зображення одного пацієнта мають бути або у тренувальному, або у тестовому наборі, але не можуть бути використані одночасно в обох.

У випадку нерівномірного розподілу класів важливо переконатися, що однаковий відсоток кожного класу включений у кожен з розділів.

Наприклад, якщо у нас є лише 10 зображень для класу A, й розподіл визначений як 70% для навчання, 10% для валідації й 20% для тестування, забезпечимо, щоб 7 зображень класу A були використані для навчання, 1 для валідації й 2 для тестування. На рисунку 4.11 можна побачити розподіл зображень по різних вибірках.

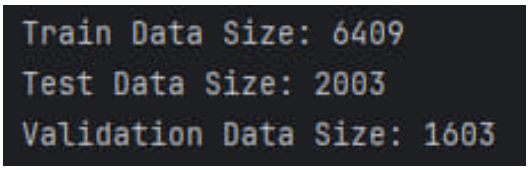


Рисунок 4.11 – Розподіл зображень по різним вибірках

4.5 Тренування мереж й проведення експериментів

Після проведення усіх попередніх обробок даних й розподілу на різні вибірки, потрібно ініціалізувати кастомну мережу. Мережу буде збудована на основі простої архітектури LeNet. Архітектуру мережі LeNet обговорювали у пункті 2.1.1. На рисунку 4.12 зображений код, який ініціалізує кастомну мережу. Характеристики комп'ютера на якому проходило тренування можна побачити у таблиці 4.3.

Таблиця 4.3 – Характеристики комп’ютеру

Компонент комп’ютеру	Опис компоненту
Процессор	Intel Core i5-13600KF
Відеокарта	NVIDIA GeForce RTX 4070 Ti 12 GB GDDR6X
Оперативна пам’ять	Kingston FURY DDR5-6000 32768MB
SSD – 1	Kingston KC3000 1TB M.2 2280 NVMe PCIe Gen 4.0 x4 3D TLC NAND (SKC3000S/1024G)
SSD – 2	KINGSTON SA400S37480G
HDD	WDC WD10EZEX-21WN4A0 1TB

```
2 usages
class LeNetCustom(nn.Module):
    def __init__(self):
        super(LeNetCustom, self).__init__()
        self.conv1 = nn.Conv2d( in_channels: 3,  out_channels: 6,  kernel_size: (5, 5), padding=2)
        self.conv2 = nn.Conv2d( in_channels: 6,  out_channels: 16,  kernel_size: (5, 5))
        self.fc1 = nn.Linear(16 * 54 * 54,  out_features: 120)
        self.fc2 = nn.Linear( in_features: 120,  out_features: 84)
        self.fc3 = nn.Linear( in_features: 84, num_classes)
        self.dropout = nn.Dropout(0.5)  # Додано Dropout шар
        self.bn1 = nn.BatchNorm2d(6)  # Додано Batch Normalization після першого згорткового шару
        self.bn2 = nn.BatchNorm2d(16)  # Додано Batch Normalization після другого згорткового шару

    def forward(self, x):
        x = self.bn1(F.max_pool2d(F.leaky_relu(self.conv1(x)),
                                  kernel_size: (2, 2)))  # Змінено ReLU на Leaky ReLU та додано Batch Normalization
        x = self.bn2(F.max_pool2d(F.leaky_relu(self.conv2(x)),
                                  kernel_size: (2, 2)))  # Змінено ReLU на Leaky ReLU та додано Batch Normalization
        x = x.view(-1, self.num_flat_features(x))
        x = F.leaky_relu(self.fc1(x))  # Змінено ReLU на Leaky ReLU
        x = self.dropout(x)  # Додано Dropout
        x = F.leaky_relu(self.fc2(x))  # Змінено ReLU на Leaky ReLU
        x = self.fc3(x)
        return x
```

Рисунок 4.12 – Код для ініціалізації кастомної мережі

Вибриємо оптимізатор втрат, метою якого є пошук мінімуму для нашої функції втрат. Існує багато різних методів мінімізації функції втрат, які в більшості випадків базуються на градієнті моделі. У нашому випадку будемо використовувати оптимізатор Adam.

Важливим параметром оптимізатора є швидкість навчання, якщо швидкість навчання буде дуже мала, параметри мережі будуть змінюватися дуже незначно, а пошук мінімуму займе дуже багато часу.

При дуже високій швидкості навчання оптимізатор може змінювати параметри занадто різко, що може призвести до того, що він ніколи не знайде мінімум.

Тому було прийняте рішення використати швидкість навчання у 1e-5. Ініціалізацію оптимізатора та застосування швидкості навчання у 1e-5, код зображений на рисунку 4.13.

На рисунку 4.14 зображено як була зроблена ініціалізація функції втрат. Також використовуємо спеціальні ваги, які мають допомогти забезпечити збалансоване навчання між класами.

```
# Adam optimizer
optimizer = optim.Adam(net.parameters(), lr=1e-5)
```

Рисунок 4.13 – Код для ініціалізації оптимізатора Adam й застосування швидкість навчання у 1e-5

```
# Criterion (loss function)
criterion = nn.CrossEntropyLoss(weight=class_weights)
```

Рисунок 4.14 – Код для ініціалізації оптимізатора функції втрат

Для класифікації було вирішено використовувати різноманітні моделі нейронних мереж, а саме: LeNetCustom, LeNet, ResNet18, VGG19, AlexNet, GoogLeNet та RegNet_Y_32GF.

Для покращення швидкості збільшення точності моделі та зменшення часу навчання, всі моделі, крім LeNetCustom і LeNet, були вже з наявними натренованими вагами, які були попередньо навчені на великому наборі даних. Це дозволить моделям здійснити перехід від вже вивчених особливостей до нових даних, що допоможе покращити точність класифікації.

Усі моделі були навчені протягом 30 епох, що є достатньою кількістю для поверхневого оцінювання потенціалу моделі на нашому наборі даних.

Деякі моделі можуть виявити свій потенціал на ранніх етапах навчання, тоді як інші можуть вимагати більш тривалого навчання для досягнення оптимальної точності.

Розберемо як буде проходити підготовка та розподіл даних для тренування, валідації та тестування нейронної мережі. Спочатку створимо датасет зображень, які зберігаються в директорії набору даних, і до них застосовується попередня обробка даних, який був обговорений раніше.

Потім дані розділяються на навчальні, валідаційні та тестові підмножини. Для кожної підмножини створюється відповідний завантажувач даних (DataLoader), який забезпечує подачу даних під час тренування та валідації. Потім датасет створюється знову з іншим набором попередньої обробки (transform_test) для тестових даних, і для них також створюється завантажувач даних. Це забезпечує правильне розділення та подачу даних для різних етапів навчання моделі. Код який розподіляє дані зображений на рисунку 4.15.

```
if __name__ == '__main__':
    SubsetRandomSampler = torch.utils.data.sampler.SubsetRandomSampler

    dataset = torchvision.datasets.ImageFolder(root=data_dir, transform=transform_train)

    train_samples = SubsetRandomSampler(train_indices)
    val_samples = SubsetRandomSampler(val_indices)
    test_samples = SubsetRandomSampler(test_indices)

    train_data_loader = torch.utils.data.DataLoader(dataset, batch_size=batch_size,
                                                    shuffle=False, num_workers=1, sampler=train_samples)
    validation_data_loader = torch.utils.data.DataLoader(dataset, batch_size=validation_batch_size,
                                                         shuffle=False, sampler=val_samples)

    dataset = torchvision.datasets.ImageFolder(root=data_dir, transform=transform_test)
    test_data_loader = torch.utils.data.DataLoader(dataset, batch_size=validation_batch_size,
                                                   shuffle=False, sampler=test_samples)
```

Рисунок 4.15 – Код для підготовка та розподілу даних для тренування, валідації та тестування

Розробимо метод для обчислення точності передбачень моделі. Метод get_accuracy повинен обчислювати точність моделі, визначаючи кількість правильних передбачень у наборі даних.

Також він має порівнювати передбачені значення з фактичними мітками, і повертати загальну кількість зразків у поточному наборі та кількість правильних передбачень. Це дозволить нам оцінити, наскільки добре модель передбачає класи.

Такий підхід допоможе зрозуміти, які класи модель розпізнає добре, а де їй ще потрібне покращення. Розроблений метод get_accuracy зображений на рисунку 4.16.

```
def get_accuracy(predicted, labels):
    batch_len = labels.size(0)
    correct = (predicted == labels).sum().item()
    return batch_len, correct
```

Рисунок 4.16 – Розроблений метод get_accuracy

Також розробимо метод, який буде використовуватися для оцінки продуктивності моделі на валідаційному наборі даних. Цей метод буде називатися evaluate.

Розроблений метод evaluate має переводити модель у режим оцінки, обчислювати втрати та точність для кожного батчу валідаційних даних, підсумовувати результати для всіх батчів і повертати середнє значення втрат та точності.

Цей процес дозволить нам визначити, наскільки добре модель передбачає класи на валідаційних даних, допомагаючи оцінити її загальну продуктивність. Розроблений метод evaluate зображений на рисунку 4.17.

```python
def evaluate(model, val_loader):
    losses = 0
    num_samples_total = 0
    correct_total = 0
    model.eval()
    for inputs, labels in val_loader:
        inputs, labels = inputs.to(device), labels.to(device)
        out = model(inputs)
        _, predicted = torch.max(out, 1)
        loss = criterion(out, labels)
        losses += loss.item()
        b_len, corr = get_accuracy(predicted, labels)
        num_samples_total += b_len
        correct_total += corr
    accuracy = correct_total / num_samples_total
    losses = losses / len(val_loader)
    return losses, accuracy
```

Рисунок 4.17 – Розроблений метод evaluate

Розберемо вже розроблений код, для тренування наших мереж. Розроблений код призначений для тренування нейронної мережі протягом заданої кількості епох. На кожній епосі він проходить через всі батчі навчальних даних, обчислює втрати (loss) та точність для кожного батчу, оновлює ваги моделі за допомогою зворотного поширення помилки (backpropagation) та оптимізатора. Після кожної епохи він також оцінює модель на валідаційних даних, обчислюючи середні значення втрат і точності на валідаційному наборі.

Результати тренування та валідації (втрати і точність) зберігаються у відповідні списки для подальшого аналізу або візуалізації. В кінці кожної епохи результати виводяться на екран, що дозволяє стежити за прогресом тренування моделі. Розроблений код для тренування усіх мереж зображений на рисунку 4.18.

```python
for epoch in range(num_epochs):
    running_loss = 0.0
    correct_total = 0.0
    num_samples_total = 0.0
    for i, data in enumerate(train_data_loader):
        # get the inputs
        inputs, labels = data
        inputs, labels = inputs.to(device), labels.to(
            device)  # Move inputs and labels to the same device as the model
        optimizer.zero_grad()

        # forward + backward + optimize
        outputs = net(inputs)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()

        # compute accuracy
        _, predicted = torch.max(outputs, 1)
        b_len, corr = get_accuracy(predicted, labels)
        num_samples_total += b_len
        correct_total += corr
        running_loss += loss.item()

    running_loss /= len(train_data_loader)
    train_accuracy = correct_total / num_samples_total
    val_loss, val_acc = evaluate(net, validation_data_loader)
```

Рисунок 4.18 – Код для тренування мереж протягом заданої кількості епох

Розберемо дуже важливий метод get_pretrained_model. Даний метод був розроблений, щоб було простіше конфігурувати наші моделі й швидко замінювати моделі для тренування. Даний метод виконує отримання попередньо навчених моделей глибокого навчання, які можна використовувати для різних завдань класифікації. Він дозволяє обрати модель за її ім'ям model_name та налаштувати кількість класів total_num_classes, до яких потрібно адаптувати модель.

Даний метод включає умовну логіку для обробки різних моделей. Наприклад, якщо обрано модель ResNet18, ResNet152, VGG19, AlexNet або RegNet_Y_32GF, вона завантажується з попередньо навченими вагами для набору даних ImageNet. Потім останній зв'язаний шар моделі замінюється новим шаром, який має кількість вихідних класів, вказаних у total_num_classes. Код розробленого методу зображений на рисунку 4.19.

```python
def get_pretrained_model(model_name, total_num_classes):
    if model_name == 'LeNetCustom':
        net = LeNetCustom()
    elif model_name == 'LeNet':
        net = LeNet()
    elif model_name == 'ResNet18':
        net = models.resnet18(weights=ResNet18_Weights.IMAGENET1K_V1)
        net.fc = nn.Linear(net.fc.in_features, total_num_classes)
    elif model_name == 'ResNet152':
        net = models.resnet152(weights=ResNet152_Weights.IMAGENET1K_V2)
        net.fc = nn.Linear(net.fc.in_features, total_num_classes)
    elif model_name == 'VGG19':
        net = models.vgg19(weights=VGG19_Weights.IMAGENET1K_V1)
        net.classifier[6] = nn.Linear(net.classifier[6].in_features, total_num_classes)
    elif model_name == 'AlexNet':
        net = models.alexnet(weights=AlexNet_Weights.IMAGENET1K_V1)
        net.classifier[6] = nn.Linear(net.classifier[6].in_features, total_num_classes)
    elif model_name == 'GoogLeNet':
        net = models.googlenet(weights=GoogLeNet_Weights.IMAGENET1K_V1)
    elif model_name == 'RegNet_Y_32GF':
        net = models.regnet_y_32gf(weights=RegNet_Y_32GF_Weights.IMAGENET1K_SWAG_LINEAR_V1)
        net.fc = nn.Linear(net.fc.in_features, total_num_classes)
    else:
        raise ValueError("Unknown model name")

    print('Selected model:', model_name)

    return net
```

Рисунок 4.19 – Метод для отримання налаштованих моделей

Для початку, детально розглянемо результати тренування кастомної мережі на основі LeNetCustom. Всі графіки та додаткові результати будуть представлені у додатку А.

На рисунку 4.20 зображений графік втрат на тренувальних та валідаційних даних. Цей графік дозволяє оцінити динаміку зменшення втрат під час навчання моделі та визначити її загальну продуктивність.

Рисунок 4.20 – Графік втрат на тренувальних та валідаційних даних

Можна побачити, що мережа покроково навчалася на даних й справлялась з новими зображеннями, але отриманий результат не дуже задовільний.

На рисунку 4.21 зображений графік точності класифікації на тренувальних та валідаційних даних. З рисунка 4.21 можна побачити, що мережа навчається трохи повільно, але свої 0.7 точності на тренувальних даних досягла, але точність на валідаційних даних є дуже низькою. Можливо, створена модель не дуже підходить для класифікації багатьох зображень, або потрібно тренувати модель більше ніж 30 епох й ще доповнити наш набір даних, для того, щоб уникнути такого дисбалансу класів, який у нас є зараз. На рисунку 4.22 зображена остання епоха тренування мережі.

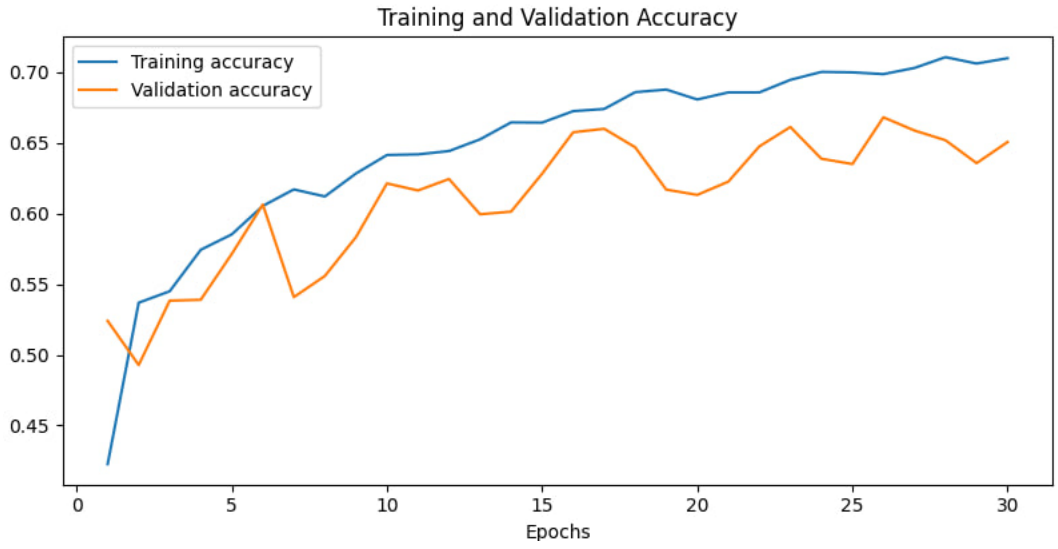


Рисунок 4.21 – Графік точності класифікації на тренувальних та валідаційних даних

```
Epoch: 30
Loss: 0.863  Accuracy:0.710
Validation Loss: 1.138  Val Accuracy: 0.651
```

Рисунок 4.22 – Інформація про останню епоху тренування моделі

Більш детальну інформацію як модель класифікувала хвороби можна побачити на рисунках 4.23, 4.24.

```
Test data
Accuracy of actinic keratoses : 30 %
Accuracy of basal cell carcinoma : 55 %
Accuracy of benign keratosis-like lesions : 51 %
Accuracy of dermatofibroma : 99 %
Accuracy of melanoma : 49 %
Accuracy of melanocytic nevi : 73 %
Accuracy of vascular lesions : 83 %
```

Рисунок 4.23 – Детальна інформація точності класифікації по захворюванням на тестових даних

```
Validation data
Accuracy of actinic keratoses : 84 %
Accuracy of basal cell carcinoma : 51 %
Accuracy of benign keratosis-like lesions : 49 %
Accuracy of dermatofibroma : 33 %
Accuracy of melanoma : 53 %
Accuracy of melanocytic nevi : 71 %
Accuracy of vascular lesions : 99 %
```

Рисунок 4.24 – Детальна інформація точності класифікації по захворюванням на валідаційних даних

Також варто згадати про матрицю невідповідностей. Матриця невідповідностей являє собою підсумок результатів прогнозування для задачі класифікації. Вона може допомогти нам зрозуміти, які класи важко розрізнити за допомогою нашої моделі. По осі X візуалізуємо передбачення нашої моделі, а по осі Y – мітки базової істини. В ідеальній матриці невідповідностей всі високі значення були б зосереджені вздовж діагоналі. Але даний метод не був розроблений, тож потрібно його розробити, для більш детальної та нагальної візуалізації точності результатів тренування.

Розроблений код для візуалізації матрицю невідповідностей зображений на рисунку 4.25.

```
confusion_matrix = torch.zeros(len(classes), len(classes))
with torch.no_grad():
    for data in test_data_loader:
        images, labels = data
        images, labels = images.to(device), labels.to(device)
        outputs = net(images)
        _, predicted = torch.max(outputs, 1)
        for t, p in zip(labels.view(-1), predicted.view(-1)):
            confusion_matrix[t.long(), p.long()] += 1

cm = confusion_matrix.numpy()

fig, ax = plt.subplots(figsize=(7, 7))
sns.heatmap(cm / (cm.astype(float).sum(axis=1) + 1e-9), annot=False, ax=ax)

# labels, title and ticks
ax.set_xlabel('Predicted', size=25);
ax.set_ylabel('True', size=25);
ax.set_title('Confusion Matrix(Test data)', size=25);
ax.xaxis.set_ticklabels(['akiec', 'bcc', 'bkl', 'df', 'mel', 'nv', 'vasc'], size=15); \
        ax.yaxis.set_ticklabels(['akiec', 'bcc', 'bkl', 'df', 'mel', 'nv', 'vasc'], size=15);
```

Рисунок 4.25 – Код для візуалізації матриці невідповідності

Розроблений код призначений для обчислення та візуалізації матриці помилок (confusion matrix) для тестових даних моделі класифікації. Розберемо даний метод, тому, що дуже важливо розуміти побудову даних матриць. Спочатку створюємо матрицю нулів заданого розміру, що відповідає кількості класів. Потім робимо ітерацію по тестовому набору даних, де кожне зображення подається на модель, і для кожного передбаченого класу збільшується відповідний елемент матриці помилок. Після цього конвертуємо матрицю помилок в масив numpy для використання у бібліотеці візуалізації. Далі, матриця помилок візуалізується за допомогою теплової карти (heatmap), де кожна комірка показує відносну частоту помилок для відповідного передбаченого класу в порівнянні з фактичним класом.

Результат роботи розробленого коду для матриці невідповідностей й результаті відображені даним кодом зображені на рисунку 4.26

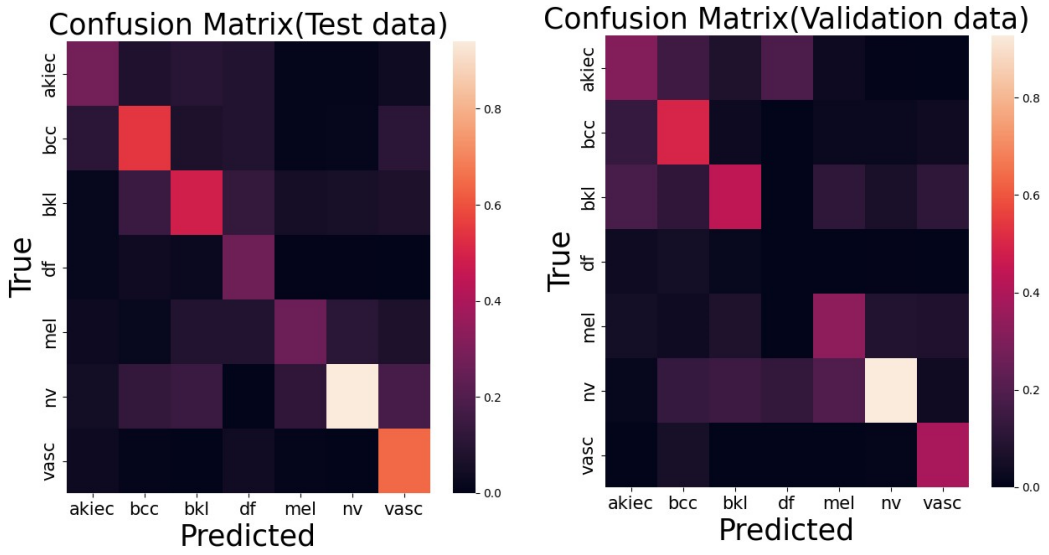


Рисунок 4.26 – Матриця невідповідностей мережі LeNetCustom

Наступний етап, це навчання усіх наших мереж, та порівняння їх у точності класифікації на нашому наборі даних. Точність класифікації мереж можна побачити у таблиці 4.4

Таблиця 4.4 – Результати класифікації мережами

Назва моделі	Загальна точність	Точність на тренувальних даних	Точність на валідаційних даних
LeNet	0.61	0.59	0.59
LeNetCustom	0.65	0.71	0.65
ResNet18	0.78	0.91	0.77
ResNet152	0.84	0.97	0.85
VGG19	0.85	0.95	0.85
AlexNet	0.81	0.96	0.81
GoogLeNet	0.81	0.87	0.81
RegNet_Y_32GF	0.84	0.91	0.84

На основі результатів отриманих з експериментів, можна зробити висновок, що мережа LeNetCustom яка була побудована на основі LeNet справилася краще, ніж стандартна LeNet, що є позитивним сигналом для подальшого вдосконалення цієї архітектури.

Однак, більш складні мережі показали ще кращі результати, що свідчить про їхню більшу потужність та здатність до глибокого навчання. У цілому, результат мереж непоганий, але можна побачити те, що точність на тестових даних значно більше, ніж на валідаційних. Це може свідчити про те, що кількісний дисбаланс у класах дуже сильний, та потрібно розробляти окремий код, для помірного розподілу даних, спробуємо зробити інакше.

Виберемо моделі, які себе добре показали й приберемо кастомні ваги для класів, які зображені на рисунку 4.5 й подивимося на результат. Результат буде зображений у таблиці 4.5.

Таблиця 4.5 – Результати класифікації мережами без вагів класів

Назва моделі	Загальна точність	Точність на тренувальних даних	Точність на валідаційних даних
ResNet18	0.84	0.975	0.83
ResNet152	0.87	0.99	0.88
VGG19	0.85	0.98	0.86
AlexNet	0.84	0.99	0.84

Виходячи з таблиці 4.5, можна побачити, що мережа ResNet152 показала дуже гарний результат, але варто зазначити, що інші мережі також продемонстрували достойні результати [31]. На рисунку 4.27 можна побачити матрицю невідповідностей, яка надає візуальне представлення про те, як добре модель здатна класифікувати дерматоскопічні зображення. Також, на рисунку 4.28 зображена остання епоха навчання мережі, що дозволяє детально проаналізувати процес навчання та оцінити стабільність і точність моделі на завершальному етапі тренування [32].

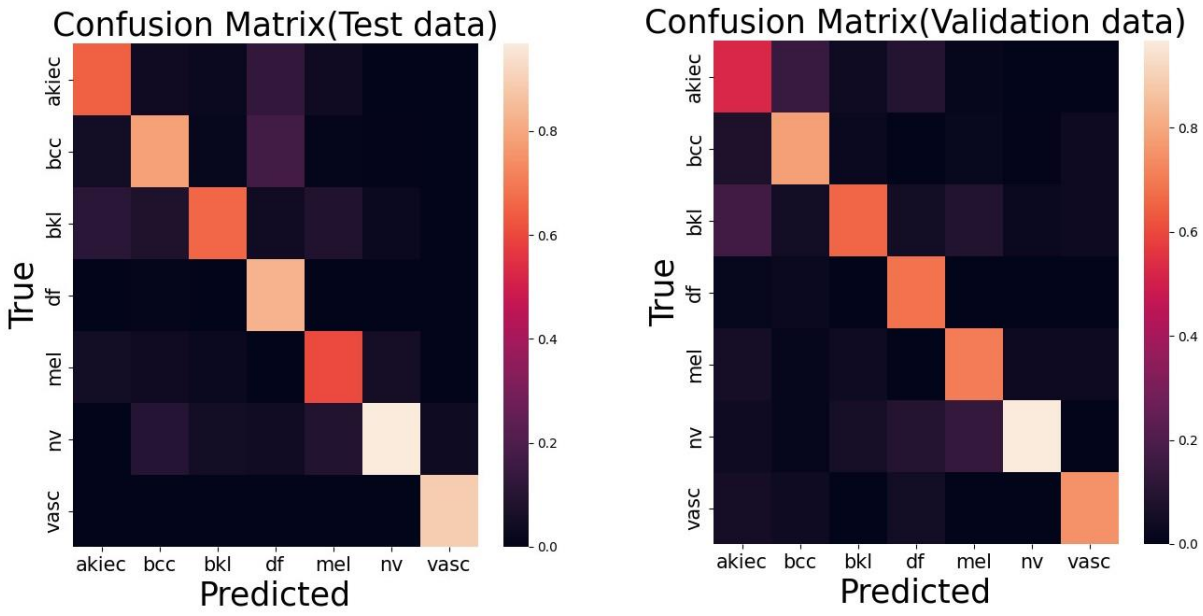


Рисунок 4.27 – Матриця невідповідностей мережі ResNet152

```
Epoch: 30
Loss: 0.050  Accuracy:0.982
Validation Loss: 0.683  Val Accuracy: 0.857
```

Рисунок 4.28 – Інформація про останню епоху навчання моделі ResNet152

Рисунок 4.29 – Графік точності на тренувальних та валідаційних даних

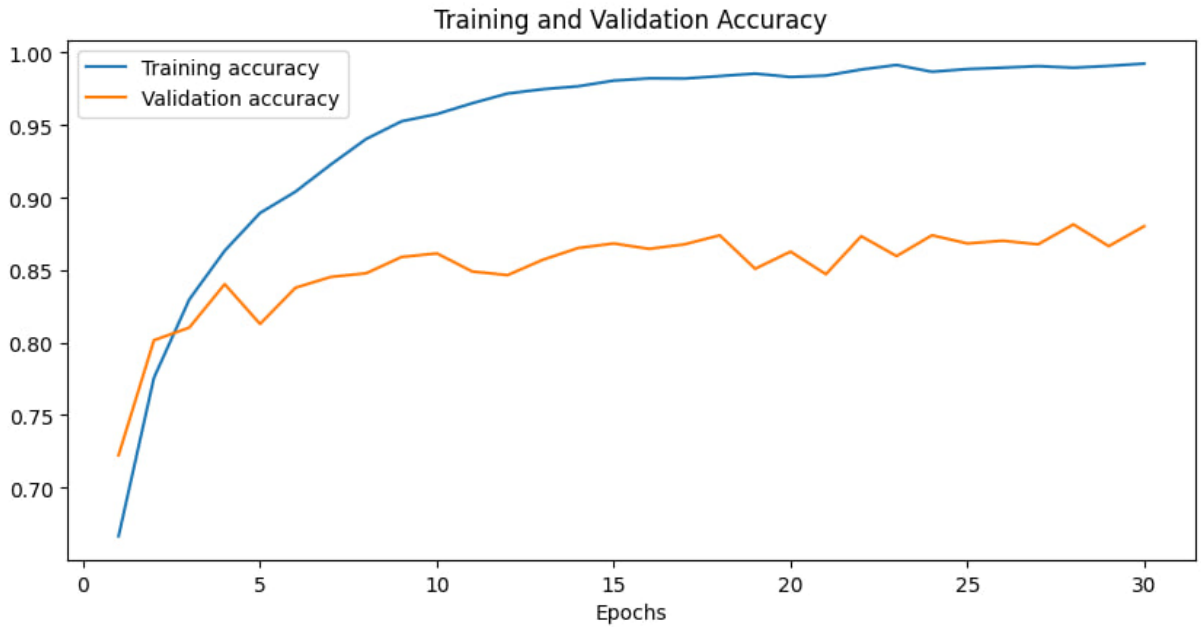


Рисунок 4.30 – Графік втрат на тренувальних та валідаційних даних.

4.6 Порівняння результатів

Порівняння результатів які були отримали при виконанні практичної частини наведені у таблиці 4.6.

Таблиця 4.6 – Порівняння результатів різних мережах

Назва моделі	Загальна точність	Точність на тренувальних даних	Точність на валідаційних даних
1	2	3	4
LeNet	0.61	0.59	0.59
LeNetCustom	0.65	0.71	0.65
ResNet18	0.78	0.91	0.77
ResNet18(без вагів)	0.87	0.99	0.88
ResNet152	84	0.97	0.85
ResNet152(без вагів)	0.87	0.99	0.88
VGG19	0.85	0.95	0.85

Продовження таблиці 4.6

1	2	3	4
VGG19(без вагів)	0.85	0.98	0.86
AlexNet	0.81	0.96	0.81
AlexNet(без вагів)	0.84	0.99	0.84
GoogLeNet	0.81	0.87	0.81
RegNet_Y_32GF	0.84	0.91	0.84

Найбільшу точність показали ResNet18(без вагів) й ResNet152(без вагів), але е ще ResNet152, VGG19, AlexNet(без вагів) й RegNet_Y_32GF не сильно відстають. Не сильно відстають по точності класифікації. Після отриманих кінцевих результатів з різних мереж, можна побачити, що на якість розпізнавання впливає дуже велика кількість факторів, розмір даних, баланс класів, ваги класів й використання мереж з наявними вагами. Також, деякі моделі розпізнають деякі хвороби краще ніж інші, що також треба пам’ятати, та мати це на увазі. Деякі окремі мережі будуть краще класифікувати специфічні хвороби, тому має сенс об’єднувати мережі в ансамблі, та комбінувати прогнози, для того, щоб прийняти правильне рішення.

ВИСНОВКИ

В ході роботи була досліджена класифікація й аналіз дерматоскопічних зображень різними способами, а саме побудовою моделей на основі штучного інтелекту. В ході дослідження даної проблеми було запропоновано побудувати моделі на основі штучного інтелекту з різними підходами й імплементацію інтелектуальних моделей для аналізу й класифікації зображень захворювань шкіри.

При розв'язанні цієї проблеми були проведені теоретичні дослідження для розробки методології та підходів до побудови мережевих моделей класифікації на основі дерматоскопічних зображень. Акцент був зроблений на обрання оптимальних алгоритмів та параметрів для побудови найефективніших моделей класифікації.

В результаті проведених практичних експериментів були розроблені та навчені моделі, які демонстрували високий рівень точності та надійності у класифікації різних захворювань шкіри на основі дерматоскопічних зображень.

Результати практичних експериментів можна побачити в таблиці 4.5, на них можна побачити як різні мережі справилися з класифікацією дермаскопічних зображень. Деякі моделі, наприклад як ResNet152 без вагів класів показала дуже гарний результат класифікації, але є деякі моделі, які показали не дуже втішний результат. Також варто взяти в увагу, що для досягнення ще кращої класифікації, потрібно використовувати мережеві ансамблі, це значно підвищить точність класифікації й дасть більшу вибірку оцінок для винесення фінальної оцінки

Отримані результати свідчать про потенційну ефективність застосування штучного інтелекту в області дерматології для покращення діагностики та лікування шкірних захворювань.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Carmel S., Shani E., Rosenberg L. Skin cancer protective behaviors among the elderly: Explaining their response to a health education program using the health belief model. 1996. No. 22(7). P. 651–668p.

2. Малихін О.С. Застосування методів штучного інтелекту для аналізу дерматоскопічних зображень при діагностуванні захворювань шкіри. URL: https://doi.org/10.30837/IYF.ELBE.2024.085 (дата звернення: 22.05.2024)

3. Al-Nuaimi Y., Sherratt M. J., Griffiths C. E. Skin health in older age . Maturitas , 79 , 256 – 264p.

4. A J Bailey. Molecular mechanisms of ageing in connective tissues. URL: https://pubmed.ncbi.nlm.nih.gov/11322995/ (дата звернення: 08.03.2024).

5. Bilgili S. G. Karadag A. S. Ozkol H. U. Calka O. , & Akdeniz N. The prevalence of skin diseases among the geriatric patients in Eastern Turkey 535 – 539p

6. Russo T, Piccolo V, Lallas A, Argenziano G. Recent advances in dermoscop. URL: https://f1000research.com/articles/5-184/v1(дата звернення: 08.03.2024).

7. Argenziano G, Fabbrocini G, Delfino M. Epiluminescence microscopy: a new approach to in vivo detection of Sarcoptes scabiei. 751–753p

8. Piraccini BM, Balestri R, Starace M, Rech G. Nail digital dermoscopy (onychoscopy) in the diagnosis of onychomycosis. 509-513p.

9. Bauer J., Forschner A., Garbe C., Röcken M. Dermoscopy of tungiasis. Arch Dermato. 761-763p.

10. Milad Mirbabaie, Stefan Stieglitz, Nicholas R. J. Frick. Artificial intelligence in disease diagnostics: A critical review and classification on the current state of research guiding future direction. URL:https://link.springer.com/article/10.1007/s12553-021-00555-5 (дата звернення: 08.03.2024).

11. Panch T., Szolovits P., Atun R. Artificial intelligence, machine learning and health. URL: https://jogh.org/documents/issue201802/jogh-08-020303.pdf (дата звернення: 08.03.2024).

12. Ольга А. Исаева., О. Г. Аврунин., И. И. Моро.з, О. В. Столяренко., Р. Б. Аксельрод., В. С. Киливник., Конрад Громашек., Айгуль Искакова., С. Рахметуллина., А. Козбакова. Possibilities of automated image processing at optical capillaroscopy URL: https://doi.org/10.1117/12.2569772 (дата звернення: 22.05.2024).

13. F. Amato, A. Lopez, E.M. Pena-Mendez, P. Vanhara, A. Hampl, J. Havel. Artificial neural networks in medical diagnosis. 45-58p

14. Chen X., Xie H., Cheng G., Poon L.K.M., Leng M., Wang F.L. Trends and features of the applications of natural language processing techniques for clinical trials text analysis. URL: https://www.mdpi.com/2076-3417/10/6/2157 (дата звернення: 08.03.2024).

15. Ma J., Porter A.L. Analyzing patent topical information to identify technology pathways and potential opportunities. 811-827p

16. Selivanova, K. G., Trubitcin, A. A., Avrunin, O. G. Development of a comprehensive method for the dermatoscopic images analysis of the facial skin with acne. Biophysical Bulletin, (46), 34-45. URL: https://doi.org/10.26565/2075-3810-2021-46-03 (дата звернення: 08.03.2024).

17. Селіванова К. Г., Тимкович М. Ю. Проєктування телемедичної системи об’єктивізованої оцінки тремору рук із зовнішнім кінестетичним впливом. С. 255-257.

18. Boursalie O., Samavi R., Doyle T. E. Machine learning and mobile health monitoring platforms: A case study on research and implementation challenges. p 179-203

19. Mayank Mishra. Convolutional Neural Networks, Explained. URL: https://doi.org/10.26565/2075-3810-2021-46-03 (дата звернення: 18.05.2024).

20. Abis Hussain Syed. Hands on guide to LeNet-5 (The Complete Info). URL: https://syedabis98.medium.com/hands-on-guide-to-lenet-5-the-complete-info-b2ae631db34b (дата звернення: 18.05.2024).

21. Mateen, M., Wen, J., Nasrullah; Song, S., Huang, Z. Fundus Image Classification Using VGG-19 Architecture with PCA and SVD. Symmetry 2019, 11, 1. URL: https://doi.org/10.3390/sym11010001 (дата звернення: 18.05.2024).

22. Anupam Sharma., Deep Dive into Support Vector Machine., URL: https://towardsdatascience.com/deep-dive-into-support-vector-machine-654c8d517103 (дата звернення: 18.05.2024).

23. Sohaib Zafar. Choosing the Right Pre-Trained Model: A Guide to VGGNet, ResNet, GoogleNet, AlexNet, and Inception. URL: https://medium.com/@sohaib.zafar522/choosing-the-right-pre-trained-model-a-guide-to-vggnet-resnet-googlenet-alexnet-and-inception-db7a8c918510 (дата звернення: 20.05.2024).

24. Місоченко С.Ю., Селіванова К.Г., Аврунін О.Г. Дослідження використання вірогіднісних методів у сфері обробки біомедичних зображень.URL:https://openarchive.nure.ua/server/api/core/bitstreams/5c6de871-beb7-44f5-bc3a-fe01a5ef83c2/content (дата звернення: 22.05.2024).

25. Черкасова, Є. О., Селіванова, К. Г. Програмний модуль аналізу дерматоскопічних зображень шкіри обличчя людини з акне. URL: http://openarchive.nure.ua/handle/document/12007 (дата звернення: 25.05.2024)

26. Philipp Tschandl., Cliff Rosendahl., Harald Kittler. The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. URL: https://www.nature.com/articles/sdata2018161 (дата звернення: 20.05.2024).

27. O. A. Isaieva., O. G. Avrunin. Segmentation of ultraviolet-dermatoscopic images. Dynamics of the development of world science. Abstracts of the 6th International scientific and practical conference. Perfect Publishing. Vancouver, Canada. 2020. Pp. 26-30.

28. Rosendahl, C., Tschandl, P., Cameron, A. Kittler, H. Diagnostic accuracy of dermatoscopy for melanocytic and nonmelanocytic pigmented lesions. J Am Acad Dermatol 64, 1068–1073 (2011).

29. SohamMazumder. Skin Lesion Classification — An Educational Guide. URL:https://medium.com/miccai-educational-initiative/skin-cancer-image- classification-an-educational-guide-2a043a1beb59 (дата звернення: 25.05.2024).

30. Stevenson A. D., Mickan S., Mallett S., Ayya M. Systematic review of diagnostic accuracy of reflectance confocal microscopy for melanoma diagnosis in patients with clinically equivocal skin lesions. Dermatol Pract Concept 3, 19–27p (2013).

31. Akay, B. N., Kocyigit, P., Heper, A. O. & Erdem, C. Dermatoscopy of flat pigmented facial lesions: diagnostic challenge between pigmented actinic keratosis and lentigo maligna. Br J Dermatol 163, 1212–1217p (2010).

32. Accuracy vs. precision vs. recall in machine learning: what's the difference?. Evidently AI – Open-Source ML Monitoring and Observability. URL: https://www.evidentlyai.com/classification-metrics/accuracy-precision-recall (дата звернення: 20.05.2024).