МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«До захисту допущено»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)
18 грудня 2023 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня магістр

зі спеціальності 122 - Комп'ютерних наук,
освітньо-професійної програми «Інформатика»
на тему: «Інформаційна технологія забезпечення робастності системи розпізнавання медичних діагностичних зображень»
здобувача групи ІН.м-24 Сущка Івана Олександровича

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ Іван СУЩОК
(підпис)

Керівник,
кандидат технічних наук, доцент В'ячеслав МОСКАЛЕНКО _____
(підпис)

Суми – 2023

Сумський державний університет

Факультет електроніки та інформаційних технологій

Кафедра комп'ютерних наук

«Затверджую»

В.о. завідувача кафедри

_____ Ігор ШЕЛЕХОВ

(підпис)

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня магістр

зі спеціальності 122 - Комп'ютерних наук, освітньо-професійна програми «Інформатика»

здобувача групи ІН.м-24 Сущка Івана Олександровича

1. Тема роботи: «Інформаційна технологія забезпечення робастності системи розпізнавання медичних діагностичних зображень»
затверджую наказом по СумДУ від «06» грудня 2023 року № 1412-VI.
2. Термін здачі здобувачем кваліфікаційної роботи до 18 грудня 2023 року_____
3. Вхідні дані до кваліфікаційної роботи _____
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)
1) Аналіз проблеми предметної області, постановка й формування завдань дослідження.
2) Огляд технологій, що використовуються в інформаційних технологіях та інструментах активних дій.
3) Виконання практичної частини.
4) Аналіз результатів.
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)_____
6. Консультанти до проекту (роботи), із значенням розділів проекту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «____» _____ 20 ___ р.

Завдання прийняв до виконання _____ Керівник _____

(підпис) (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	*Аналіз проблеми предметної області, постановка й формування завдань дослідження*	*08.12.2023*	
2	*Огляд методів забезпечення робастності системи розпізнавання медичних діагностичних зображень*	*09.12.2023*	
3	*Реалізація програмного забезпечення для вирішення задачі*	*10.12.2023*	
4	*Аналіз отриманих результатів*	*15.12.2023*	
5	*Оформлення пояснювальної записки до кваліфікаційної роботи*	*16.12.2023*	

Здобувач вищої освіти _____ Керівник _____

(підпис) (підпис)

АНОТАЦІЯ

Записка: 63 стор., 19 рис., 6 формул, 24 джерела.

Обґрунтування актуальності теми роботи – Розвиток та застосування інформаційних технологій для забезпечення робастності системи розпізнавання медичних діагностичних зображень є вкрай важливим для забезпечення ефективного функціонування систем розпізнавання медичних діагностчних зображень незалежно від умов отримання зображень.

Об’єкт дослідження — інформаційні технології забезпечення робастності системи розпізнавання медичних діагностичних зображень.

Предмет дослідження – реалізація інформаційних технологій для забезпечення робастності системи розпізнавання медичних діагностичних зображень.

Мета роботи — реалізація робастної системи розпізнавання медичних діагностичних зображень стійкої до протиборчих атак.

Методи дослідження — аналіз наукових публікацій, аналіз методів забезпеченння робастності, порівняльний аналіз систем розпізнаванння медичних діагностичних зображень.

Результати — реалізована інтелектуальна система забезпечення робастності системи розпізнавання медичних діагностичних зображень.

МАШИННЕ НАВЧАННЯ, РОБАСТНІСТЬ, МЕДИЧНІ ДІАГНОСТИЧНІ ЗОБРАЖЕННЯ, ПРОТИБОРЧІ АТАКИ, ЗНЕШУМЛЮЮЧИЙ АВТОЕНКОДЕР

ЗМІСТ

ВСТУП

Актуальність. Зростання попиту на використання штучного інтелекту (ШІ) в медичній діагностиці, яка є однією з ключових сфер сучасної медицини. ШІ має потенціал значно модернізувати підходи до діагностичних процесів, зокрема шляхом розпізнавання медичних діагностичних зображень. Це сприяє підвищенню точності, швидкості та об'єктивності діагностики, зменшенню помилок та витрат, а також розвантаженню лікарів.

Об'єкт дослідження. Медичні діагностичні системи, які використовують ШІ для обробки та аналізу медичних зображень.

Предмет дослідження. Розробка та оптимізація алгоритмів ШІ для підвищення робастності систем розпізнавання медичних діагностичних зображень.

Гіпотеза. Інтеграція передових алгоритмів ШІ може значно покращити точність та надійність систем діагностики медичних зображень, зокрема шляхом підвищення їхньої робастності до різноманітних умов отримання зображення.

Наукова новизна. Розробка інноваційних підходів до використання ШІ у медицині, що дозволяє ефективно вирішувати складні задачі розпізнавання медичних діагностичних зображень. Дослідження спрямоване на покращення робастності таких систем, що є ключовим аспектом у підвищенні якості та ефективності медичної діагностики.

Структура. Аналіз сучасних тенденцій використання ШІ в медичній діагностиці, огляд інструментів ШІ для розпізнавання зображень, оцінку ефективності різних алгоритмів, розробка методології підвищення робастності, реалізація та тестування результатів роботи робастної системи розпізнавання медичних діагностичних зображень.

1. АНАЛІЗ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Сучасний стан та тенденції розвитку інтелектуальних систем аналізу медичних діаностичних даних

Однією з областей штучного інтелекту, яка найбільше користується попитом, є розпізнавання зображень, яке використовується в багатьох сферах людської діяльності, таких як медицина, безпека, освіта та розваги. Комп'ютерні алгоритми та моделі використовуються для розпізнавання зображень, визначення елементів зображення, включаючи сцени, об'єкти, особи, дії, емоції, текст та інші елементи. Глибокі згорткові нейронні мережі (CNN) є однією з найпоширеніших і найефективніших методів розпізнавання зображень; вони здатні ідентифікувати складні та абстрактні характеристики зображень виходячи з великої кількості даних.

ШІ може аналізувати великі обсяги медичних даних, виявляти закономірності та аномалії, що дозволяє прогнозувати епідемічну ситуацію, виявляти ознаки патологій тощо.

Найефективнішою медичною стратегією є профілактика, яка спрямована на запобігання, раннє виявлення, лікування захворювань та попередження ускладнень. Вона дозволяє підвищити якість життя, знизити смертність та витрати на охорону здоров'я. Одним з основних інструментів профілактики є скринінгові дослідження, які полягають у масовому обстеженні населення або груп ризику на наявність певних захворювань. Скринінгові дослідження можуть використовувати різні методи, такі як анкетування, лабораторні та інструментальні обстеження.

Однак, скринінгові дослідження не є досконалими і мають ряд проблем та викликів, серед яких одним з найважливіших є інтерпретація результатів, що впливає на чутливість та специфічність методу. Це часозатратна та суб'єктозалежна лікарська задача, а отже існує висока залежність інтерпретації результатів від людського фактору. Неправильна інтерпретація результатів може

призвести до гіпо- та гіпердіагностики, несвоєчасного лікування, втрати довіри пацієнтів до надійності традиційних медичних знань.

Тому, одним з можливих шляхів покращення інтерпретації результатів скринінгових досліджень є застосування штучного інтелекту. Він може аналізувати великі обсяги даних, виявляти закономірності та взаємозв'язки, робити прогнози та рекомендації, навчатися від свого досвіду та набувати нових знань.

Розпізнавання медичних зображень може допомогти виявити ранні стадії захворювань, таких як рак, серцево-судинні, неврологічні, інфекційні тощо, і вчасно призначити оптимальне лікування, збільшивши шанси пацієнтів на виживання та одужання. Розпізнавання медичних зображень також може допомогти вивчати структуру та функції організму, аналізувати генетичні та молекулярні механізми захворювань, розробляти нові методи діагностики та лікування, а також створювати персоналізовані медичні рішення. Розпізнавання медичних зображень є одним з найпотужніших інструментів медичної науки, який може покращити якість життя та здоров'я людей.

Розпізнавання медичних діагностичних зображень полягає у виявленні, класифікації, локалізації, сегментації та інтерпретації патологічних змін, органів, тканин, клітин, молекул і т. п. на зображеннях, отриманих за допомогою різних методів візуалізації.

Застосування ШІ в медицині має дві основні гілки: віртуальну та фізичну. Віртуальний компонент представлений машинним навчанням (також називається глибоким навчанням), яке представлено математичними алгоритмами, які покращують навчання через досвід [1].

Фізична форма застосування ШІ в медицині включає фізичні об'єкти, медичні пристрої та все більш складні роботи, які беруть участь у наданні допомоги (carebots) [2].

Мабуть, найбільш перспективним підходом є використання роботів як помічників; наприклад, робот-компаньйон для людей похилого віку з

когнітивними порушеннями або обмеженою мобільністю. Японські керботи є найдосконалішою формою цієї технології. Роботи використовуються в хірургії як асистенти-хірурги або навіть як сольні виконавці. У медицині хірургічна система, створена американською компанією Intuitive Surgical , була названа Да Вінчі на знак визнання його надихаючого впливу. Він був схвалений Управлінням з контролю за якістю харчових продуктів і медикаментів (FDA) у 2000 році, і кількість установок, що працюють у всьому світі, наразі перевищує 5000. Хірургічні системи Da Vinci полегшують складні операції з використанням мінімально інвазивного підходу та можуть контролюватись хірургом з консолі. Система зазвичай використовується для простатектомії та гінекологічних хірургічних процедур. Його починають використовувати для відновлення серцевих клапанів. в Україні такі операції проводяться з 2019 року [3].

Основою доказової медицини є встановлення клінічних кореляцій та уявлень за допомогою розробки асоціацій і шаблонів з існуючої бази даних інформації. Традиційно ми використовували статистичні методи для встановлення цих закономірностей і асоціацій. Комп'ютери навчаються мистецтву діагностики пацієнта за допомогою двох широких методів – блок-схем і підходу до бази даних.

Підхід, заснований на блок-схемі, передбачає трансляцію процесу збору анамнезу, тобто лікар ставить серію запитань, а потім приходить до вірогідного діагнозу шляхом поєднання представленого комплексу симптомів. Це вимагає передачі великої кількості даних у хмарні мережі на базі машин, враховуючи широкий спектр симптомів і хворобливих процесів, які зустрічаються в рутинній медичній практиці. Результати цього підходу обмежені, оскільки машини не можуть спостерігати та збирати сигнали, які може спостерігати лише лікар під час зустрічі з пацієнтом [4].

Навпаки, підхід до бази даних використовує принцип глибокого навчання або розпізнавання образів, який передбачає навчання комп'ютера за допомогою повторюваних алгоритмів у розпізнаванні того, як виглядають певні групи

симптомів або певні клінічні/радіологічні зображення. Прикладом такого підходу є проект штучного мозку Google, запущений у 2012 році. Ця система навчилася розпізнавати котів на основі 10 мільйонів відео на YouTube, ефективність підвищилась завдяки перегляду все нових і нових зображень. Після 3 днів навчання він міг передбачити зображення кота з точністю 75%.

Роботи також можуть бути корисні в оцінці змін у продуктивності людини в таких ситуаціях, як реабілітація. Інша сфера, де штучний інтелект може бути корисним, це моніторинг керованої доставки ліків до цільових органів, тканин або пухлин. Наприклад, нещодавня розробка нанороботів, призначених для подолання проблем доставки, які виникають, коли виникають труднощі з дифузією терапевтичного агента в цільове місце [5].

Застосування штучного інтелекту в медицині відкриває нові можливості для покращення діагностики та лікування хвороб. Віртуальні та фізичні компоненти ШІ, такі як машинне навчання та роботи-помічники, вже активно використовуються в медичній практиці.

Розпізнавання зображень, зокрема медичних діагностичних зображень, є одним з найбільш перспективних напрямків застосування ШІ. Воно допомагає виявити ранні стадії захворювань та призначити оптимальне лікування, а також допомагає вивчати структуру та функції організму.

Однак, необхідно враховувати, що ефективність застосування ШІ в медицині залежить від якості вхідних даних та алгоритмів машинного навчання. Тому подальші дослідження та розробки в цій галузі є важливими для досягнення більш точних та надійних результатів.

1.2 Аналіз існуючих методів забезпечення робастності моделей розпізнавання зображень

Робастність моделей розпізнавання зображень означає здатність моделей витримувати різні види збурень, таких як шум, спотворення, зміна освітлення, перспективи, масштабу тощо, і продовжувати правильно розпізнавати об’єкти на зображеннях. Робастність є актуальною, оскільки в умовах реальної зйомки

зображення не можуть бути абсолютно стандартизованими, і, з високою ймовірністю, матимуть певні аберації. Тому моделі повинні бути гнучкими і стійкими до таких варіацій візуальних даних. Робастність також є важливою для захисту моделей від атак зі змагальними прикладами, які є спеціально підібраними зображеннями, що вводять модель в оману. Такі атаки можуть мати серйозні наслідки, особливо у критичних сферах, таких як безпека, медицина тощо. Через що важливо розробляти та вдосконалювати методи забезпечення робастності моделей розпізнавання зображень, які б дозволяли підвищити їхню стійкість до різних збурень та атак.

Основні групи методів забезпечення робастності моделей розпізнавання зображень можна поділити на попередню обробку зображень, покращення моделей розпізнавання та післяобробку результатів розпізнавання. Попередня обробка зображень полягає в застосуванні різних алгоритмів для покращення якості зображень, таких як нормалізація, фільтрація, зменшення шуму, сегментація, видалення фону тощо. Покращення моделей розпізнавання зводиться до застосуванні різних технік для підвищення загальної здатності моделей до навчання та адаптації до нових даних, таких як глибоке навчання, ансамблеві методи, перенос навчання тощо. Післяобробка результатів розпізнавання полягає в застосуванні різних критеріїв для оцінки та вибору найкращих результатів розпізнавання, таких як порівняння, виявлення аномалій, ранжування, голосування тощо [6].

Перевагами цих методів є покращення ефективності та точності розпізнавання зображень, а недоліками є збільшення складності та часу обчислень, а також можливість недонавчання або перенавчання моделей. У наступних розділах ми розглянемо детальніше приклади застосування та аналізу деяких існуючих методів забезпечення робастності моделей розпізнавання зображень.

Моделювання фону або об’єкта є методом попередньої обробки зображень, який дозволяє виділити об’єкти інтересу від фону, або навпаки, що може

дозволити поліпшити точність та ефективність подальшого аналізу зображень, такого як розпізнавання об’єктів, сегментація, класифікація тощо. Основні кроки цього методу описані далі.

Першим кроком проводиться визначення моделі фону або об’єкта, яка може бути статичною або динамічною. Статична модель фону або об’єкта припускає, що фон або об’єкт не змінюється протягом часу, тому модель може бути побудована на основі одного або декількох початкових кадрів відео. Динамічна модель фону/об’єкта припускає, що фон або об’єкт може змінюватися протягом часу, тому модель повинна бути адаптивною та здатною враховувати зміни освітлення, погоди, руху тощо. Для побудови моделі фону/об’єкта можна використовувати різні методи, такі як статистичні, нейромережеві, генеративні, глибинні тощо.

Наступним кроком порівнюється поточне зображення з моделлю фону або об’єкта. Цей крок полягає в обчисленні різниці між кожним пікселем поточного зображення та відповідним пікселем моделі фону/об’єкта. Різниця може бути виміряна за допомогою різних метрик, таких як абсолютна різниця, квадратична різниця, відстань Магаланобіса, відстань Геллінґера тощо.

Третім кроком є виявлення різниці між поточним зображенням та моделлю фону або об’єкта. Цей крок полягає в виділенні пікселів, які мають значну різницю з моделлю фону/об’єкта, та віднесенні їх до переднього плану або заднього плану. Для цього можна використовувати різні методи порогування, такі як глобальне, локальне, адаптивне, Оцу, Ніблека, Сауволи тощо.

Надалі відбувається класифікація пікселів зображення на фоні або об’єкті. Цей крок полягає в присвоєнні пікселям зображення міток, які вказують, чи належать вони до фону або об’єкта. Для цього можна використовувати різні методи класифікації, такі як логістична регресія, дерева рішень, нейронні мережі, опорні вектори, к-найближчих сусідів тощо.

Останнім кроком є покращення результату за допомогою морфологічних операцій, таких як ерозія, дилатація, відкриття, закриття тощо. Цей крок полягає

в застосуванні різних операцій до зображення, які дозволяють видалити шум, заповнити діри, згладити краї, з'єднати компоненти тощо. Морфологічні операції виконуються за допомогою структурного елементу, який є невеликим зображенням, яке визначає форму та розмір операції [7,8].

Розглянемо основні методи моделювання фону/об'єкта, які використовуються для виявлення рухомих об'єктів на відео. Моделювання фону/об'єкта полягає в тому, що кожен піксель зображення класифікується як належний до фону або до об'єкта, який відрізняється від фону. Це дозволяє виділити об'єкти інтересу від нерелевантного фону та полегшити подальшу обробку зображень. Методи моделювання фону/об'єкта можуть бути розділені на п'ять основних категорій, залежно від того, як вони вибудовують та оновлюють модель фону або об'єкта, яка використовується для порівняння з поточним зображенням.

Методи на основі середнього значення, які використовують середнє значення інтенсивності пікселів для моделювання фону/об'єкта, є найпростішими та найшвидшими методами, але вони не здатні адаптуватися до змін у фоні або об'єкті, таких як рух, освітлення, тіні тощо. Прикладом такого методу є Running Average, який оновлює середнє значення кожного пікселя за допомогою експоненційного згладжування.

Методи на основі гаусівського розподілу, які використовують гаусівську функцію для моделювання фону/об'єкта, є більш гнучкими та робастними, ніж методи на основі середнього значення, але вони вимагають більше обчислень та пам'яті. Функція Гауса у контексті машинного навчання є функцією ядра, яка використовується для вимірювання схожості між двома об'єктами або точками даних. Функція Гауса зображена формулою (1.1) має вигляд:

$$K(x, x') = \exp\left(-\frac{\|x-x'\|^2}{2\sigma^2}\right) \quad (1.1)$$

де x і x′ є векторами, а σ є параметром ширини ядра. Функція Гауса приймає значення від 0 до 1, причому більші значення означають більшу схожість [9].

Прикладом такого методу є Single Gaussian, який моделює кожен піксель за допомогою однієї гаусівської функції, параметри якої оновлюються за допомогою алгоритму Калмана. Алгоритм Калмана - це алгоритм, який використовується для оцінки стану динамічної системи за допомогою неповних та зашумлених даних. Цей алгоритм працює в два етапи: на Е-кроці (expectation) він обчислює очікуване значення функції правдоподібності, вважаючи приховані змінні за спостережувані, а на М-кроці (maximization) він обчислює оцінку максимального правдоподібності, таким чином збільшуючи очікуване правдоподібності, обчислене на Е-кроці [10].

Методи на основі змішування гаусівських розподілів, які використовують суму гаусівських функцій для моделювання фону або об’єкта, є найбільш універсальними та потужними методами, які можуть враховувати різні режими фону/об’єкта, такі як мульти-модальність, динаміка, зміна освітлення тощо. Прикладом такого методу є Gaussian Mixture Model (GMM), який моделює кожен піксель за допомогою суми K гаусівських функцій, параметри яких оновлюються за допомогою алгоритму EM. Алгоритм EM - це алгоритм, який використовується в математичній статистиці для знаходження оцінок максимального правдоподібності параметрів ймовірнісних моделей, в разі, коли модель залежить від деяких прихованих змінних. Він схожий на алгоритм Калмана, але застосовується до статичних, а не динамічних систем. Алгоритм EM також складається з двох кроків: на Е-кроці він обчислює математичне сподівання логарифму правдоподібності, використовуючи поточні оцінки параметрів, а на М-кроці він максимізує це математичне сподівання, знаходячи кращі оцінки параметрів. Алгоритм EM часто використовується для розділення суміші гаусівських розподілів [11].

Методи на основі текстур, які використовують текстурні ознаки для моделювання фону або об’єкта, є більш відповідними для сцен, де фон або об’єкт має виражену текстуру, таку як трава, листя, вода тощо. Прикладом такого методу є Local Binary Pattern (LBP), який використовує бінарний код, який відображає

відносну яскравість сусідніх пікселів, для моделювання фону або об’єкта. Методи на основі нейронних мереж, які використовують нейронні мережі для моделювання фону або об’єкта, є найбільш сучасними та перспективними методами, які можуть навчатися з великої кількості даних та вирішувати складні задачі моделювання фону або об’єкта. Прикладом такого методу є Background Matting, який використовує глибинну нейронну мережу, яка приймає на вході поточне зображення, початкове зображення фону та зображення маски, яке вказує об’єкт інтересу, та виводить високоякісне зображення переднього плану, заднього плану та альфа-маски [12].

Ефективність та точність моделювання фону або об’єкта залежать від різних факторів, серед яких виділяють: тип зображення, тип об’єкта, тип фону, рівень шуму, рівень освітлення та рівень контрасту.

Тип зображення - це характеристика зображення, яка визначає його формат, якість, джерело та змінність. Тип зображення може бути статичним або динамічним, одноканальним або багатоканальним, реальним або синтетичним. Кожен тип зображення має свої переваги та недоліки для моделювання фону або об’єкта.

Статичне зображення - це зображення, яке не змінюється в часі, наприклад, фотографія. Статичне зображення є простим та стабільним для моделювання фону або об’єкта, оскільки воно не містить руху, який може спотворювати або затемнювати об’єкт. Однак статичне зображення також може бути обмеженим та недостатнім, оскільки воно не відображає динаміку та зміну об’єкта в різних умовах.

Динамічне зображення - це зображення, яке змінюється в часі, наприклад, відео. Динамічне зображення є багатим та повним для моделювання фону або об’єкта, оскільки воно відображає рух, який може показувати або приховувати об’єкт. Також динамічне зображення може використовувати часову інформацію, яка може допомогти визначити та відслідковувати об’єкт. Однак динамічне

зображення також може бути складним та непостійним, оскільки воно містить рух, який може збивати з толку або змішувати об’єкт.

Одноканальне зображення - це зображення, яке має тільки один канал кольору, наприклад, сіре зображення. Одноканальне зображення є простим та економним для моделювання фону або об’єкта, оскільки воно має менше даних, ніж багатоканальне зображення. Однак одноканальне зображення також може бути недостатнім та неяскравим, оскільки воно не відображає кольорову інформацію, яка може допомогти розрізняти або виділяти об’єкт.

Багатоканальне зображення - це зображення, яке має декілька каналів кольору, наприклад, RGB зображення. Багатоканальне зображення є багатим та яскравим, оскільки воно відображає кольорову інформацію, яка може допомогти розрізняти або виділяти об’єкт. Однак багатоканальне зображення також може бути складним та ресурсомістким, оскільки воно має більше даних, ніж одноканальне зображення.

Реальне зображення - це зображення, яке було зняте в реальному світі, наприклад, фотографія або відео. Реальне зображення є реалістичним та достовірним, оскільки воно відображає реальну сцену, в якій знаходиться об’єкт. Однак реальне зображення також може бути непередбачуваним та неконтрольованим, оскільки воно може містити різні фактори, які можуть впливати на якість зображення, такі як освітлення, шум, тіні, відблиски тощо.

Синтетичне зображення - це зображення, яке було створене штучно, наприклад, за допомогою комп’ютерної графіки. Синтетичне зображення є ідеальним та контрольованим для моделювання фону або об’єкта, оскільки воно може бути створене за певними правилами та параметрами, які визначають об’єкт. Однак синтетичне зображення також може бути нереалістичним та недостовірним, оскільки воно може не відповідати реальній сцені, в якій знаходиться об’єкт, на рисунку 1.1 зображено приклад такого синтетчного зображення.

Рисунок 1.1 – Синтетичне зображення створене за допомогою комп’ютерної графіки

Тип об’єкта - - це характеристика об’єкта, яка визначає його форму, розмір, колір, текстуру, рух та інші властивості. Тип об’єкта може бути рухомим або нерухомим, однорідним або неоднорідним, великим або малим. Кожен тип об’єкта має свої переваги та недоліки для моделювання фону або об’єкта.

Об’єкт може бути рухомим або нерухомим. Рухомий об’єкт переміщається в просторі в часі, наприклад, автомобіль, людина, тварина тощо. Нерухомий об’єкт залишається на тому ж місці в часі, наприклад, будинок, дерево, статуя тощо. Рухомий об’єкт є більш складним та динамічним , оскільки він може змінювати свою форму, розмір, напрямок, швидкість тощо. Однак рухомий об’єкт також може бути більш видимим та розпізнаваним , оскільки він може виділятися від фону або мати характерні ознаки.

Об’єкт може бути однорідним або неоднорідним. Однорідний об’єкт має рівномірну текстуру або колір, наприклад, куля, куб, плитка тощо. Неоднорідний об’єкт має різноманітну текстуру або колір, наприклад, квітка, торт, картинка

тощо. Однорідний об’єкт є більш простим та стабільним , оскільки він має менше деталей, які можуть змінюватися або втрачатися. Однак однорідний об’єкт також може бути більш невиразним та невідокремленим , оскільки він може зливатися з фоном або не мати відмінних рис.

Об’єкт може бути великим або малим. Великий об’єкт займає більше простору на зображенні, ніж малий об’єкт. Розмір об’єкта впливає на те, як багато інформації міститься на зображенні, а також на те, як багато уваги приділяється об’єкту. Великий об’єкт може бути більш детальним та розпізнаваним , оскільки він може мати більше характеристик, які можна визначити або відслідковувати. Однак великий об’єкт також може бути більш складним та нестійким , оскільки він може мати більше змін, які можуть впливати на його форму, колір, текстуру тощо.

Фон - це те, що знаходиться за об’єктом на зображенні. Він може мати різні властивості, які впливають на те, як легко виділити об’єкт від нього. Фон може бути рухомим або нерухомим. Рухомий фон змінюється з часом, наприклад, хмари, трафік, люди тощо. Нерухомий фон залишається статичним, наприклад, стіна, дерево, будівля тощо. Рухомий фон ускладнює виділення об’єкта, оскільки він може перекривати або змішуватися з ним.

Також фон може бути однорідним або неоднорідним. Однорідний фон має однакову текстуру або колір, наприклад, небо, поле, стеля тощо. Неоднорідний фон має різну текстуру або колір, наприклад, гори, місто, кімната тощо. Неоднорідний фон ускладнює виділення об’єкта, оскільки він може мати схожі або контрастні елементи з ним.

Також має значення розмір фону. Великий фон займає більше простору на зображенні, ніж малий фон. Розмір фону впливає на те, як багато інформації міститься на зображенні, а також на те, як багато уваги приділяється об’єкту. Великий фон може бути більш складним та різноманітним, ніж малий фон, але також може допомогти виділити об’єкт, якщо він контрастує з ним [13, 14].

Рівень шуму - це характеристика зображення, яка визначає, наскільки чистим та якісним є зображення. Шум - це випадкові збурення, які додаються до зображення під час його створення, передачі або зберігання, і які погіршують його якість. Шум може бути високим або низьким. Високий рівень шуму означає, що на зображенні є багато випадкових відхилень від очікуваного значення, тоді як низький рівень шуму означає, що на зображення містить мало випадкових відхилень. Рівень шуму впливає на складність та точність моделювання фону або об’єкта, оскільки він може затемнювати, спотворювати або приховувати деталі, які необхідні для розпізнавання або виділення об’єкта. Наприклад, високий рівень шуму вимагає більш робастної та чутливої моделі фону або об’єкта, ніж низький рівень шуму, оскільки він може призводити до помилок або невизначеності в моделюванні.

Рівень освітлення - це характеристика зображення, яка визначає, наскільки яскравим та чітким є зображення. Освітлення - це світло, яке падає на зображення від різних джерел, таких як сонце, лампи, вогні тощо. Освітлення може бути високим або низьким, рівномірним або нерівномірним. Рівень освітлення впливає на складність та точність моделювання фону або об’єкта, оскільки він може змінювати контраст, кольори, тіні, відблиски та інші ефекти, які можуть впливати на якість зображення. Наприклад, низький рівень освітлення вимагає більш чутливої та високороздільної моделі фону або об’єкта, ніж високий рівень освітлення, оскільки він може призводити до втрати деталей або видимості об’єкта.

Рівень контрасту - це характеристика зображення, яка визначає, наскільки відрізняються світлі та темні області зображення. Контраст - це відношення між найсвітлішим та найтемнішим значенням яскравості на зображенні. Контраст може бути високим або низьким, рівномірним або нерівномірним. Високий контраст означає, що є велика різниця між світлими та темними областями зображення. Рівень контрасту впливає на складність та точність моделювання фону або об’єкта, оскільки він може змінювати видимість, чіткість, деталізацію

та інші аспекти, які можуть впливати на якість зображення. Наприклад, високий контраст може допомогти виділити об’єкт від фону або підкреслити його характеристики, тоді як низький контраст може призводити до втрати деталей або розрізнення об’єкта [15].

Всі ці фактори впливають на ефективність та точність моделювання фону або об’єкта. Важливо враховувати ці фактори при виборі відповідної моделі для конкретних задач. Кожна модель має свої сильні та слабкі сторони, і немає універсальної моделі, яка була б найкращою для всіх ситуацій. Тому важливо розуміти, які фактори впливають на ефективність та точність моделювання, і вибирати модель, яка найкраще підходить для конкретного випадку.

1.3 Знешумлюючий автоенкодер як захист від протиборчих атак

Протиборчі атаки є однією з найбільших загроз для безпеки та надійності моделей глибокого навчання, особливо в області комп’ютерного зору. Протиборчі атаки полягають у створенні спеціально підібраних змін до вхідних даних, які можуть змусити модель видавати неправильний результат, не змінюючи при цьому їхнього зовнішнього вигляду про людське око.

Протиборча атака - це метод атаки на модель глибокого навчання, який додає невеликі зміни до вхідних даних, аби спричинити помилки в класифікації або розпізнаванні. Протиборча атака може бути виконана в режимі white-box, коли атакуючий має повний доступ до моделі, її параметрів та архітектури, або в режимі black-box, коли атакуючий має доступ лише до вихідних даних моделі.

Протиборчі атаки можуть використовуватися з різною метою, серед яких шахрайство, дезінформація або саботаж. Наприклад, протиборча атака може використовуватися для обходу системи розпізнавання обличчя, щоб отримати доступ до конфіденційної інформації або здійснити незаконні дії. Протиборча атака також може застосовуватися для впливу на рішення, які приймаються на основі моделей глибокого навчання, таких як медична діагностика, автономне керування або судова експертиза.

Відтак, важливо розробити ефективні методи захисту від протиборчих атак, які можуть забезпечити стійкість та надійність моделей глибокого навчання. Одним з таких методів є використання Denoising Autoencoder (DAE). Це тип нейронної мережі, що навчається відновлювати зображення з пошкоджених або зашумлених зображень.

Як було зазначено вище, DAE є типом нейронної мережі, яка навчається відновлювати чисті зображення з пошкоджених або зашумлених зображень. На рисунку 1.2 наведено архітектуру DAE. Це може бути використано як захист від протиборчих атак, які додають невеликі зміни до вхідних даних, щоб спричинити помилки в класифікації або розпізнаванні.

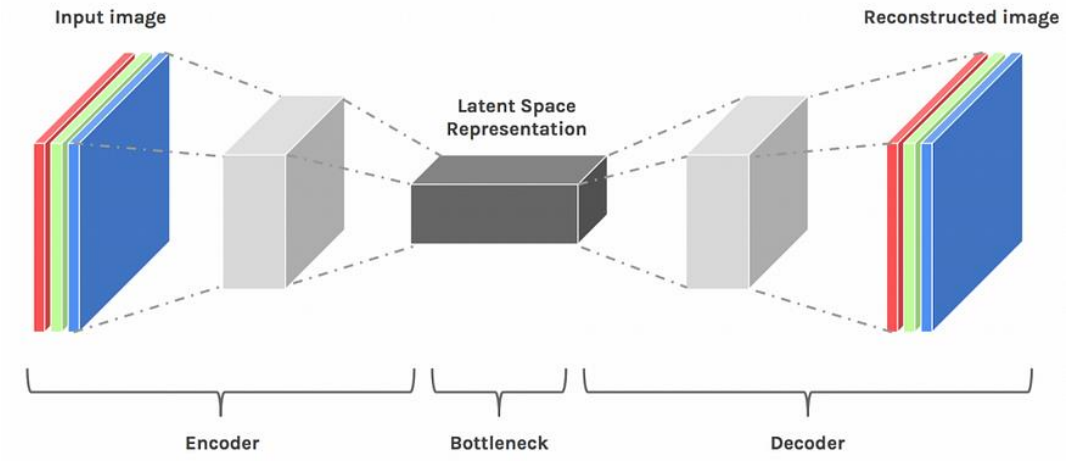

Рисунок 1.2 – Архітектура знешумлюючого автоенкодера

Ідея полягає в тому, що DAE може видалити або зменшити шум, який додається до вхідних даних, і повернути їх до оригінального стану. Таким чином, DAE може слугувати як попередня обробка для моделі глибокого навчання, яка виконує основну задачу, наприклад, класифікацію зображень. Якщо DAE успішно відновлює вхідні дані, то модель глибокого навчання повинна отримати правильну відповідь.

Для того, щоб використати DAE як захист від протиборчих атак, потрібно виконати три наступні кроки.

Перший крок полягає в навчанні DAE парами зашумлених та чистих зображень. Зашумлені зображення можуть бути створені шляхом додавання

різних видів шуму, таких як Гауссівський, сіль-перець, або випадковий шум. Чисті зображення можуть бути взяті з оригінального набору даних або з іншого джерела. Ціль навчання - мінімізувати функцію втрати, яка вимірює різницю між реконструкцією та чистим зображенням.

Другим кроком застосувати DAE до вхідних даних, які можуть бути атаковані протиборчо. Це може бути зроблено шляхом подачі вхідних даних до енкодера, який перетворює їх в код, а потім до декодера, який перетворює їх назад в реконструкцію. На рисунку 1.3 наведено схему роботи знешумлюючого автоенкодера. Реконструкція повинна бути близькою до оригінального зображення, але без шуму або протиборчих змін [16].

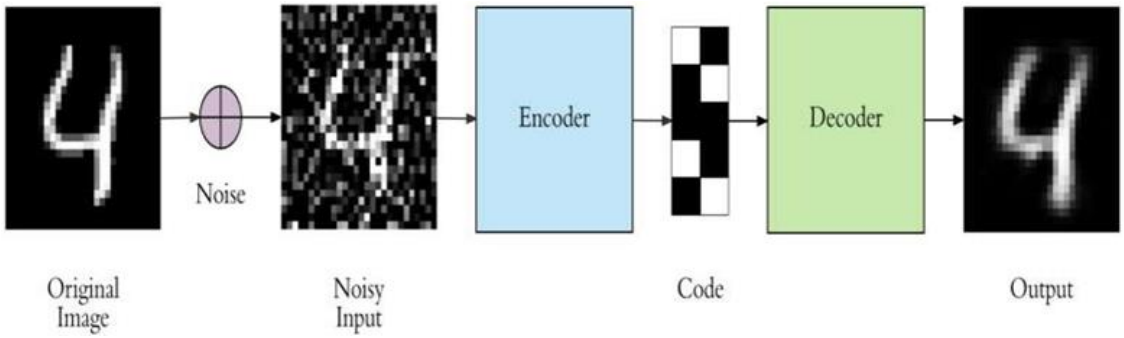

Рисунок 1.3 – Схема роботи знешумлюючого автоенкодера

Третій крок полягає в тому, щоб подати реконструкцію до моделі глибокого навчання, яка виконує основну задачу, наприклад, класифікацію зображень. Модель глибокого навчання повинна видати правильну відповідь, оскільки реконструкція не містить шуму або протиборчих змін, які можуть її заплутати.

Серед переваг використання DAE для захисту від Adversarial Attack, можна виділити наступні:

– DAE є універсальним методом, який може використовуватися для будь-якої моделі глибокого навчання, яка працює з зображеннями, при цьому не потрібно змінювати архітектуру або параметри моделі глибокого навчання, аби застосувати DAE як захист;

– DAE є простим та ефективним методом, що не потребує складних обчислень або додаткових даних, адже навчання DAE може виконуватися на

основі вже наявних даних, а застосування DAE до вхідних даних може бути виконане швидко та легко;

– DAE може видалити або зменшити різні види шуму або змін, що можуть бути додані до вхідних даних (Гауссівський шум, сіль-перець шум, або випадковий шум, які можуть бути використані для протиборчих атак).

Але метод знешумлюючого автоенкодера має також і свої недоліки.

По-перше, DAE не є повністю надійним захистом від Adversarial Attack, оскільки він може бути нездатним до відновлення всіх видів шуму або змін, які можуть додаватися до вхідних даних. Наприклад, якщо шум або зміна є дуже малою або дуже схожою на оригінальне зображення, то DAE може не помітити, або навіть посилити їх. Також, якщо шум або зміна є дуже великою, або дуже відмінною від оригінального зображення, то DAE може не відновити їх або навіть зіпсувати зображення.

По-друге, DAE може втратити деяку інформацію або деталі з вхідних даних, які можуть бути важливими для основної задачі. Наприклад, DAE може видалити деякі текстури, кольори або контури з зображення, які ,можливо, були б корисними для розпізнавання об’єктів. DAE також може внести деякі артефакти або спотворення в реконструкцію, що може знизити якість зображення.

По-третє, DAE також може бути атакований, якщо атакуючий має доступ до його параметрів або архітектури. Наприклад, атакуючий може створити протиборчу зміну, яка буде спеціально підібрана для обходу DAE, або може змінити DAE так, що він буде видаляти не шум, а корисну інформацію з вхідних даних [17].

Як було зазначено вище, DAE може бути використаний як частковий захист від протиборчих атак, але не як повний захист. Для підвищення рівня захисту, DAE поєднують з іншими методами, які спрямовані на зменшення чутливості моделі глибокого навчання до шуму або змін, які можуть бути додані до вхідних даних, або на збільшення різноманітності та стійкості моделі глибокого навчання до різних видів атак.

Defensive Distillation - це метод, який навчає модель глибокого навчання на основі вихідних даних іншої моделі, яка була навчена на тих самих даних, але з використанням високої температури. Це призводить до того, що модель стає менш чутливою до невеликих змін вхідних даних, оскільки вона видає менш різкі ймовірності для різних класів. Таким чином, модель стає стійкішою до протиборчих атак, які намагаються максимізувати різницю між вихідними даними моделі та бажаним висновком [18].

Ensemble Adversarial Training - це метод, який навчає модель глибокого навчання на основі комбінації оригінальних даних та даних, які були протиборчо атаковані. Це призводить до того, що модель стійкішою до різних видів протиборчих атак, оскільки вона навчається розпізнавати як чисті, так і зашумлені зображення. Таким чином, модель стає більш здатною до адаптації до нових або невідомих протиборчих атак, які можуть з’явитися в майбутньому [19].

Universal Adversarial Perturbations - це метод, який створює власну протиборчу зміну, яка може атакувати будь-яку модель глибокого навчання, що працює з зображеннями. Ця протиборча зміна є універсальною, оскільки вона не залежить від конкретного зображення або моделі, а лише від загальних характеристик зображень, таких як текстури, кольори або контури. Ця протиборча зміна може бути використана як засіб виявлення або видалення інших протиборчих змін, оскільки вона може виявити або анулювати їхній вплив на модель [20].

1.4 Формалізована постановка задачі

Метою дослідження є розробка інформаційної технології забезпечення робастності системи класифікації медичних зображень. Об’єктом дослідження є процес класифікації медичних зображень за допомогою нейронної мережі. Класифікація медичних зображень - це процес визначення класу зображення за допомогою нейронної мережі.

Основні завдання, які вирішуються в роботі, є наступними:

– розробка моделі нейронної мережі для класифікації медичних зображень;

– розробка методу навчання робастної моделі класифікації медичних зображень, який використовує протиборчі приклади для збільшення робастності моделі до різних збурень;

– розробка критеріїв оцінювання точнісних характеристик та робастності нейронної мережі, які враховують якість зображення та тип збурення;

– реалізація інформаційної технології забезпечення робастності системи класифікації медичних зображень та аналіз її ефективності на реальних даних.

Обмеження для розробки інформаційної технології забезпечення робастності системи розпізнавання медичних зображень:

– зображення мають однаковий розмір та роздільну здатність.

– зображення мають достатню якість та контрастність для виділення об’єкта інтересу.

– зображення містять лише один об’єкт інтересу, який займає значну частину зображення.

– збурення на зображеннях є незалежними від типу зображення та типу об’єкта.

2. МОДЕЛІ І МЕТОДИ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ЗАБЕЗПЕЧЕННЯ РОБАСТНОСТІ СИСТЕМИ РОЗПІЗНАВАННЯ МЕДИЧНИХ ДІАГНОСТИЧНИХ ЗОБРАЖЕНЬ

2.1 Модель нейромережі для розпізнавання медичних діагностичних зображень

Нейромережі є потужними інструментами для аналізу та класифікації зображень. Їхня здатність навчатися від даних та адаптуватися до різних умов робить їх ідеальними для медичної діагностики. Розглянемо деякі з найпоширеніших та найсучасніших моделей.

2.1.1 Характеристика деяких сучасних моделей

Згорткові нейронні мережі (CNN або ConvNet) - це спеціалізований тип алгоритму глибокого навчання, призначений для завдань, що вимагають розпізнавання об'єктів, включаючи класифікацію, виявлення та сегментацію зображень. Основні компоненти CNN включають згорткові шари, шари активації ReLU, пулінгові шари та повнозв’язні шари. Згорткові шари застосовують фільтри до зображення для виявлення специфічних шаблонів. Пулінгові шари зменшують розміри даних, вибираючи найважливіші особливості, тоді як повнозв’язні шари використовуються на останньому етапі для класифікації зображень. Однак, CNN схильні до перенавчання через високу складність та здатність вчитися детальних шаблонів у великомаштабних даних [21].

RNN - це стан алгоритмів для послідовних даних, які використовуються, наприклад, у Siri від Apple та голосовому пошуку Google. Вони мають внутрішню пам'ять, яка дозволяє їм пам'ятати вхідні дані і робити точні прогнози для наступних даних. Це робить їх ідеальними для роботи з послідовними даними, такими як часові ряди, мова, текст, фінансові дані, аудіо, відео, погода та багато іншого.

В RNN інформація циркулює через петлю, коли приймається рішення, враховуючи поточний вхід та попередні вхідні дані. Таким чином, RNN має два входи: поточний та недавній минулий. RNN застосовує ваги до поточних та

попередніх вхідних даних і коригує ці ваги за допомогою градієнтного спуску та зворотного розповсюдження помилок у часі (BPTT). Однак RNN мають проблеми з вивченням довготривалих залежностей, а також із зникненням градієнтів [22].

CapsNet - це тип штучної нейронної мережі, яка використовується для кращого моделювання ієрархічних відносин. CapsNet вводить структури, названі "капсулами", у згорткові нейронні мережі (CNN), і використовує вихідні дані декількох капсул для формування більш стійких представлень для вищих капсул. Капсула складається з нейронів, які активуються для різних властивостей типу об'єкта, таких як положення, розмір і відтінок.

CapsNet замінюють скалярні детектори ознак на векторні капсули та максимальне пулінг на маршрутизацію за згодою. Коли кілька капсул погоджуються, ймовірність правильного виявлення значно вища. Вихідні дані однієї капсули (дочірньої) направляються до капсул наступного шару (батьківського) відповідно до здатності дочірньої капсули передбачати вихідні дані батьківської [23].

Кожен тип мережі має свої переваги та недоліки, залежно від специфіки задачі. CNN, завдяки їх здатності до точного розпізнавання образів, є найбільш ефективним вибором для медичних діагностичних зображень.

2.1.2 Архітектура обраної моделі

Архітектура обраної нейронної мережі є важливим елементом, який визначає можливості та ефективність моделі. Як показано на рисунку 2.1, дана нейронна мережа має кілька основних шарів.

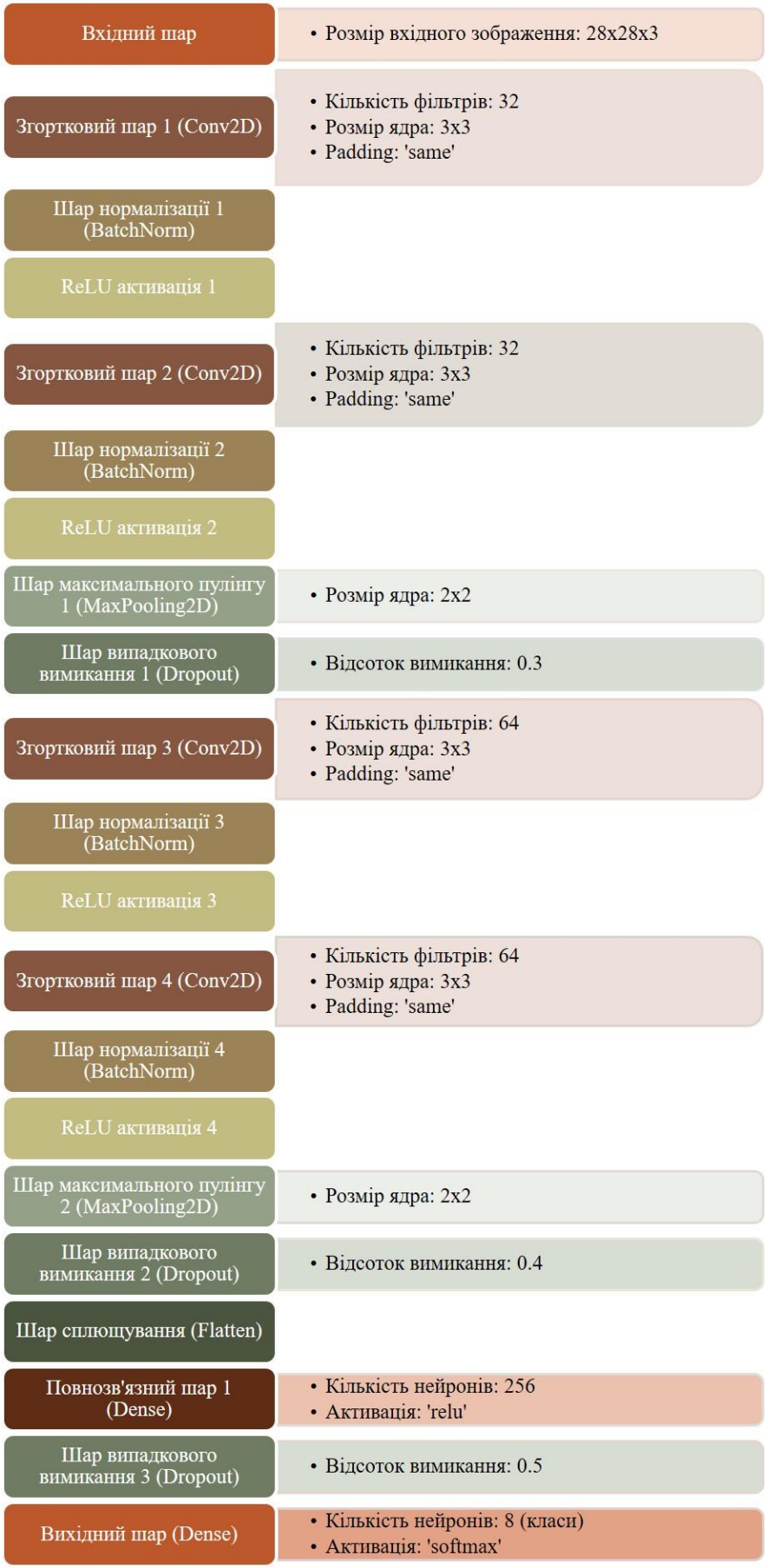

Рисунок 2.1 – Архітектура реалізованої моделі CNN

До складу архітектури моделі входить послідовність згорткових шарів з ядрами 3x3 для екстрагування ознак зображення, за ними розташовані шари нормалізації та функції активації ReLU. Далі представлені шари пулінгу з ядрами 2x2 для зменшення розмірності даних та шари випадкового вимикання нейронів. Після цього розміщені плоскі повнозв'язні шари для класифікації, а на виході знаходиться softmax-шар для отримання ймовірностей класів.

Згорткові шари (Convolutional Layers) відіграють ключову роль у згорткових нейронних мережах, особливо у задачах, пов’язаних із обробкою зображень. Модель використовує два згорткові шари на початку архітектури, кожен з яких містить 32 фільтри з ядром розміром (3, 3). Ці шари служать для виділення важливих ознак із зображень, таких як текстури, краї, кольори та інші візуальні елементи. Математично згортка в моделі CNN описується за формулою 2.1:

$$C(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i+m, j+n) \cdot K(m,n) \qquad (2.1)$$

де:

- $C(i,j)$ - значення пікселя в вихідному зображенні після згортки.
- I - вхідне зображення.
- K - ядро згортки.
- (i,j) - координати пікселя в вихідному зображенні.
- (m,n) - координати пікселя в ядрі згортки.

У процесі згортки, ядро згортки пройде через усе вхідне зображення, застосовуючи згортку в кожній локальній області. Ця операція дозволяє виділити різні ознаки на різних ділянках зображення. Завдяки цьому, згорткові шари відіграють ключову роль у виявленні та аналізі образів.

Кожен згортковий шар у моделі має набір ядер згортки. Коли до вхідного зображення застосовується кожне ядро, отримуються різні карти ознак. Ці карти потім використовуються для глибшого аналізу зображення та ідентифікації об'єктів на ньому. Розмір ядра згортки та крок згортки (strides) мають велике значення, оскільки вони визначають розмір вихідного зображення після згортки.

Згорткові шари в CNN здатні самостійно вчити різноманітні ознаки та шаблони з вхідних зображень протягом процесу навчання. Це робить CNN надзвичайно потужним інструментом для обробки зображень, дозволяючи їм адаптуватися та ефективно вирішувати складні задачі візуального аналізу.

Після кожного згорткового шару в архітектурі згорткової нейронної мережі використовуються функція активації ReLU та шар нормалізації BatchNormalization.

Функція активації ReLU (Rectified Linear Unit), застосовується після згорткових операцій і приймає вихідні значення згорткового шару. Вона визначається формулою 2.2:

$$ReLU(x) = max(0, x) \tag{2.2}$$

де x - вхідне значення.

Функція ReLU активує тільки позитивні значення x, встановлюючи вихідні значення рівними нулю для від'ємних значень. Якщо x більше або дорівнює нулю, то ReLU повертає саме це значення x, в іншому випадку повертає нуль. Ця функція дозволяє проходити позитивні сигнали без змін, а від'ємні обнуляються, створюючи нелінійність у мережі. Це допомагає нейронній мережі навчати складні функції та виявляти нелінійні залежності в даних.

Шар BatchNormalization використовується для нормалізації вихідних значень попереднього шару. Він допомагає стабілізувати процес навчання, швидше досягаючи збіжності і покращує загальну продуктивність. Цей шар нормалізує вхідні дані, вираховуючи їх середнє значення та стандартне відхилення, і перетворює їх таким чином, щоб середнє значення наближалось до нуля, а стандартне відхилення - до одиниці.

Після цих операцій, дані готові для подальшої обробки мережею.

Наступна важлива операція у CNN, це використання пулінгових шарів.

Після кожного парного згорткового шару вставляються пулінгові шари для зменшення розмірності зображення та підвищення обчислювальної ефективності. Максимальний пулінг з розміром пулінгу (2, 2) описується

формулою 2.3. Цей процес для кожної 2x2 області вхідного зображення обирає максимальне значення і створює нову картку з меншими розмірами. Він допомагає зменшити обчислювальні витрати та скоротити кількість параметрів у мережі, зберігаючи при цьому важливі ознаки зображення.

$$M(i,j) = max_{m,n} I(i+m, j+n) \quad (2.3)$$

де $M(i,j)$ відображає максимальне значення в заданій 2x2 області вхідного зображення I. Ці операції допомагають покращити продуктивність та робочий процес нейронної мережі при обробці зображень.

Шари Dropout: Після пулінгових шарів вставляються шари Dropout з ймовірністю вимикання 30% та 40% відповідно. Це допомагає уникнути перенавчання та підвищує стійкість моделі.

Функція ReLU та шар BatchNormalization сприяють ефективному видобуванню ознак, а максимальний пулінг допомагає зменшити розміри зображення без втрати важливої інформації. Ці операції дозволяють CNN ефективно навчатися, адаптуючись до різних візуальних ознак та патернів, що є ключовим аспектом у розпізнаванні та аналізі медичних зображень.

Шари Dropout використовуються як форма регуляризації, що допомагає зменшити перенавчання в нейронних мережах. Принцип роботи полягає у випадковому "вимиканні" частини нейронів у процесі тренування, що змушує мережу не покладатися занадто на будь-які конкретні нейрони та забезпечує розподіл відповідальності серед більшої кількості ознак. Це сприяє розвитку більш загальних ознак та знижує ризик перенавчання на специфічних даних тренувального набору.

У представленій архітектурі моделі застосовано три шари випадкового вимкнення нейронів. Перший шар Dropout, що має 30% ймовірність, розміщений після першої пари згорткових шарів та MaxPooling, цей шар з ймовірністю вимикання 30% допомагає зменшити перенавчання на початкових ознаках, виділених із зображень. Другий шар Dropout з ймовірністю вимикання в 40% допомагає забезпечити, що глибші шари мережі не стануть занадто

спеціалізованими на конкретних ознаках тренувальних даних. Третій шар Dropout, розміщений перед останнім повнозв'язним шаром, шар із ще вищою ймовірністю вимикання 50% забезпечує додатковий рівень регуляризації перед кінцевою класифікацією, зменшуючи ризик перенавчання на складних ознаках.

Застосування послідовних шарів Dropout з різними рівнями ймовірності вимикання сприяє розвитку моделі, яка менш схильна до надмірної адаптації на тренувальний набір. Це покращує загальну здатність моделі до узагальнення та забезпечує більш точні передбачення на нових даних, що є важливим для застосувань у сфері медичної діагностики.

Повнозв'язні шари у представленій моделі згорткової нейронної мережі виконують вирішальну функцію фінального аналізу та класифікації вхідних даних після їх обробки згортковими та пулінговими шарами.

Повнозв’язні шари перетворюють зображення, яке було переведено у вектор після проходження через попередні шари, на вихідний вектор, який може бути використаний для класифікації або інших видів передбачення. Математично ця операція виражається як матричне множення за формулою 2.4.

$$Y = XW + b \tag{2.4}$$

де:

- X - вхідний вектор, представляє оброблені дані зображення.
- W - матриця ваг, визначає ваги зв'язків між нейронами у повнозв'язаному шарі та нейронами у попередньому шарі.
- b - вектор зсуву, додається до результату матричного множення.
- Y - вихідний вектор, містить результати обчислень.

У представленій моделі згорткової нейронної мережі реалізовано один повнозв'язний шар з 256 нейронами. Цей шар розміщено після послідовності згорткових та пулінгових шарів перед вихідним шаром. Кожен нейрон у даному шарі формує зв'язки з усіма елементами вектора, отриманого в результаті операції сплощення, набуваючи ваги та зсуви.

Такий повнозв'язний шар відіграє важливу роль у поєднанні та інтерпретації ознак, вилучених на попередніх етапах. Він здійснює складні нелінійні перетворення даних та дозволяє виявляти залежності між ознаками зображень, що є критичним для точної подальшої класифікації.

Зазначений повнозв’язний шар є ключовим для фінального етапу класифікації, оскільки саме він надає механізм для формування кінцевого передбачення на базі оброблених даних. Зокрема, даний шар дає змогу здійснити класифікацію медичного зображення за відповідною категорією з урахуванням усіх зібраних та опрацьованих ознак.

Отже, повнозв’язний шар у представленій архітектурі є вирішальним для виведення висновків та ухвалення рішень на основі оброблених зображень, що становить фундамент процесу глибокого навчання.

Вихідний шар (Output Layer) має кількість нейронів, рівну кількості класів. Використовується функція активації softmax для отримання ймовірностей приналежності зображення до кожного класу.

Вихідний шар у представленій моделі згорткової нейронної мережі має ключове значення, оскільки саме він забезпечує класифікацію зображень. Кількість нейронів у вихідному шарі відповідає загальній кількості класів і становить 8.

Для перетворення вхідних сигналів вихідного шару у ймовірності застосовується функція активації softmax. Вона трансформує вхідні значення у вихідні, котрі мають властивості ймовірностей: кожне значення лежить у діапазоні від 0 до 1, а їх сума дорівнює 1.

Функція softmax для одного нейрона у вихідному шарі задається формулою 2.5.

$$P(y_i) = \frac{e^{z_i}}{\sum_{j=1}^{8} e^{z_j}} \tag{2.5}$$

де:

– $P(y_i)$ - ймовірність приналежності зображення до класу i.

- z_i- вхідний сигнал для нейрона, який відповідає класу i.
- Сума у знаменнику включає в себе вхідні сигнали для всіх нейронів у вихідному шарі, що забезпечує нормалізацію вихідних значень.

Формула 2.5 дозволяє трансформувати лінійні вхідні сигнали вихідного шару у розподіл ймовірностей, що надає змогу класифікувати вхідне зображення до одного з восьми наявних класів. Використання функції активації softmax є важливим в задачах багатокласової класифікації, як у цьому випадку, де потрібно визначити, до якого саме з класів належить зображення. Тобто за допомогою вихідного softmax-шару реалізується власне класифікація вхідних зображень за приналежністю до однієї з восьми категорій.

2.1.3 Оптимізатор та Функція Втрат

Під час навчання моделі згорткової нейронної мережі (CNN) ключовими компонентами є оптимізатор, функція втрат та метрика оцінки. У моделі, що розглядається, застосовуються оптимізатор Adam, функція втрат Sparse Categorical Crossentropy та метрика точності (accuracy).

Оптимізатор Adam (Adaptive Moment Estimation) є сучасним методом оптимізації, використовуваним для оновлення ваг в мережі під час навчання. Його перевага полягає в адаптивному регулюванні швидкості навчання для кожного параметра моделі, що базується на оцінці перших та других моментів градієнтів. Це робить Adam особливо ефективним в умовах, де градієнти можуть мати різну величину або коли обчислювальні вимоги до налаштування швидкості навчання є високими.

Sparse Categorical Crossentropy є стандартною функцією втрат для задач класифікації, де кожен зразок належить лише одному з декількох класів. Ця функція втрат ефективно обробляє випадки, де класи представлені у формі цілих чисел, уникаючи потреби використання one-hot кодування. Вона вимірює різницю між розподілом ймовірностей, передбаченим моделлю, та фактичним розподілом класів, надаючи моделі інформацію, необхідну для коригування її ваг.

Точність як метрика оцінює відсоток випадків, коли передбачення моделі збігається з фактичним класом зразків. Це пряма та інтуїтивно зрозуміла метрика, яка показує ефективність моделі у класифікації зображень. Висока точність свідчить про те, що модель добре впоралася зі своєю задачею класифікації.

Використання оптимізатора Adam забезпечує ефективне та адаптивне налаштування ваг, що є критично важливим у процесі навчання глибоких нейронних мереж. Функція втрат Sparse Categorical Crossentropy дозволяє точно вимірювати ефективність моделі у класифікації, а метрика точності забезпечує чітке уявлення про її продуктивність. Разом ці компоненти формують основу для ефективного навчання моделі, спрямованого на високу точність та здатність до узагальнення.

У наведеній моделі використовується адаптивне змінення швидкості навчання в нейронних мережах, що є ключовою стратегією для підвищення ефективності та точності процесу навчання. Ця стратегія втілюється через використання функції планувальника швидкості навчання, яка базується на епохах навчання.

На початковому етапі навчання, що включає перші 10 епох, швидкість навчання зберігається незмінною. Такий підхід дозволяє моделі швидко "навчатися" з даних, оскільки на ранніх стадіях навчання моделі зазвичай знаходяться далеко від оптимального розв'язку. Стабільна швидкість навчання сприяє більш швидкому досягненню моделлю певного рівня точності.

Після перших 10 епох швидкість навчання починає зменшуватися з кожною наступною епохою. Цей метод дозволяє моделі вишуканіше налаштовуватися та досягати кращих результатів, уникаючи ризику "перестрибування" через оптимальні значення параметрів. Зменшення швидкості навчання в такий спосіб допомагає уникнути осциляції навколо оптимуму та досягти більш точної збіжності.

Використання адаптивного підходу до змінення швидкості навчання дозволяє досягти більшої точності та ефективності в процесі навчання моделі. Цей метод особливо корисний у випадках, коли простір розв'язків складний та нелінійний, як у більшості задач глибокого навчання, де потрібно знайти найоптимальніші параметри моделі для вирішення задачі класифікації. Адаптація швидкості навчання на основі епох є одним із способів управління процесом навчання, що дозволяє досягти більшої стійкості та надійності моделі, а також уникнути проблем, пов'язаних із занадто швидким чи повільним навчанням.

Запропонована в дослідженні архітектура моделі та процес її навчання дозволяють досягти високої точності класифікації зображень на вісім різних класів типів крові. Ефективність досягається за рахунок поєднання кількох ключових архітектурних рішень.

По-перше, використання згорткових шарів з наступною Батч-нормалізацією та функцією активації ReLU дозволяє ефективно виділяти просторові ознаки зображень, важливі для подальшої класифікації. По-друге, застосування шарів пулінгу зменшує розмірність даних, запобігаючи перенавчанню. По-третє, повнозв'язні шари обробляють векторне подання ознак перед фінальною класифікацією.

Окрім архітектурних рішень, важливу роль відіграє вибір оптимізатора, функції втрат та методів регуляризації, зокрема шарів випадіння. Адаптивне зменшення швидкості навчання також сприяє ефективній оптимізації моделі.

Завершальний аналіз результатів на тестових даних підтверджує здатність моделі досягати високої точності класифікації зображень крові. Таким чином, запропоновані методи демонструють свою ефективність для вирішення поставленого завдання.

2.1.4 Загальний опис моделі

У розглянутій архітектурі моделі та методиці навчання виявляється значний потенціал для досягнення високої точності у класифікації зображень на

восьми різних класах типів крові. Використання згорткових шарів, які ефективно виявляють локальні зразки та ознаки у зображеннях, дозволяє адаптуватися до різних візуальних характеристик зображень крові, включаючи форму клітин, текстури та інші важливі особливості.

Важливу роль у зменшенні розмірності вихідних даних зі згорткових шарів відіграють пулінгові шари. Це сприяє не тільки зниженню обчислювальних витрат, але й зменшує втрату інформації, несуттєвої для класифікації.

Повнозв'язані шари, розміщені на наступному етапі, обробляють векторне представлення даних, отримане після згорткових та пулінгових шарів. Ці шари відіграють ключову роль у завершальному вивченні залежностей між ознаками та їхній класифікації.

Функція активації ReLU та шар BatchNormalization, використовувані після згорткових шарів, підвищують ефективність виокремлення важливих ознак зображень. Ці компоненти поліпшують здатність моделі виявляти ключові особливості зображень крові.

Застосування шарів Dropout є важливим для запобігання перенавчанню, що збільшує стійкість моделі та забезпечує більшу узагальнюваність результатів на невиданих даних.

Оптимізатор Adam та функція втрат Sparse Categorical Crossentropy є вирішальними у процесі навчання моделі, дозволяючи оптимізувати ваги для мінімізації помилок класифікації.

Адаптивне змінення швидкості навчання через використання функції планувальника, яка зменшує швидкість навчання з плином часу, допомагає у покращенні процесу оптимізації.

Загалом, ця архітектура та методика навчання забезпечують можливість досягнення високої точності в класифікації різних типів крові, роблячи модель цінним інструментом для аналізу медичних зображень.

2.2 Метод навчання робастної моделі розпізнавання медичних діагностичих зображень

Для створення робастної моделі розпізнавання медичних діагностичних зображень, використовується згорткова нейронна мережа CNN, описана у попередньому пункті.

Після реалізації зазначеної моделі, та тренуванні її на навчальних даних, використовується генерація Adversarial Examples.

Спочатку реалізується процес створення та використання протиборчих прикладів (adversarial examples) з метою оцінки та підвищення стійкості нейронної мережі до потенційних атак, які є незначно модифікованими версіями оригінальних зображень, створеними з метою введення в оману моделі машинного навчання.

Використання протиборчих прикладів у процесі тренування та оцінки моделі нейронних мереж є ключовим для підвищення їхньої стійкості до потенційних зловмисних атак, що забезпечує більш безпечне та надійне застосування у реальних сценаріях.

Окрім навчання моделі на протиборчих прикладах, важливим аспектом підвищення робастності моделі до шуму та зовнішніх перешкод є використання знешумлюючих автоенкодерів. Він навчається відновлювати оригінальні чисті дані зі спотворених вхідних даних.

Основна мета використання знешумлюючого автоенкодера — підвищити здатність моделі до відновлення чистих даних зі спотворених вхідних даних, що підвищує її загальну робастність. Це особливо важливо у сфері обробки медичних зображень, де точність та якість відновлених даних мають критичне значення. Здатність моделі витягувати корисні ознаки зі спотворених даних може бути корисною у розробці алгоритмів, стійких до протиборчих атак, забезпечуючи більшу безпеку та надійність у медичних додатках.

Реалізація знешумлюючего автоенкодера (DAE), має двошарову архітектуру, що складається з етапів кодування та декодування. На етапі

кодування DAE приймає зашумлені вхідні дані та перетворює їх у високорівневе внутрішнє представлення. Декодер потім намагається відновити оригінальні чисті дані з цього внутрішнього представлення. Навчання DAE відбувається методом зворотного поширення помилки з метою мінімізації втрат між вихідними даними та оригінальними чистими зображеннями.

На подальшому етапі, DAE інтегровано з конволюційною нейронною мережею (CNN) для класифікації зашумлених даних. Архітектура об'єднаної моделі включає в себе послідовне з'єднання DAE та CNN, де DAE спершу очищає зображення від шуму, а потім CNN виконує класифікацію. Модель компілюється з використанням оптимізатора 'adam' та функції втрат SparseCategoricalCrossentropy. Цей підхід дозволяє моделі адаптуватися до можливих збурень в даних, що підвищує її здатність до узагальнення та збереження високої точності класифікації.

Таким чином, впровадження знешумлюючих автоенкодерів у процес обробки медичних зображень є значним кроком у підвищенні точності та робастності моделей машинного навчання. Цей підхід не тільки покращує загальну якість відновлення зображень, але й сприяє розробці стійких до помилок та атак методів обробки даних.

2.3 Критерії оцінювання точністних характеристик та робастності нейронної мережі

Оцінка точностних характеристик та робастності нейронних мереж є ключовою частиною процесу їх розробки та валідації. Вибір відповідних метрик оцінювання дозволяє ефективно визначити здатність моделі до правильної класифікації даних, а також її стійкість до різних видів збурень або протиборчих атак.

Основною метрикою, що використовується для оцінки моделі, є точність. Вона визначає відсоток випадків, коли модель правильно класифікує зразки.

Точність розраховується як відношення кількості правильних передбачень до загальної кількості зразків.

Втрати є індикатором того, наскільки передбачення моделі відрізняються від фактичних міток. Нижчий рівень втрат вказує на кращу здатність моделі до передбачення. Зазвичай використовується функція втрат, така як крос-ентропія.

Для детальнішого аналізу результатів класифікації застосовується матриця помилок, що дозволяє визначити чутливість та специфічність моделі, а також оцінити кількість помилок різних типів.

Оцінка робастності моделі здійснюється шляхом аналізу її поведінки при поданні спотворених або ворожих вхідних даних. Використання протиборчих атак, таких як FGSM (Fast Gradient Sign Method), дозволяє визначити здатність моделі опиратися спотворенням в даних.

Окремим аспектом оцінки робастності є аналіз ефективності знешумлюючих автоенкодерів у відновленні оригінальних даних із спотворених вхідних. Робастність автоенкодера може бути визначена через порівняння оригінальних зображень із відновленими, використовуючи метрики, такі як середньоквадратична помилка (MSE).

3 РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ЗАБЕЗПЕЧЕННЯ РОБАСТНОСТІ СИСТЕМ РОЗПІЗНАВАННЯ МЕДИЧНИХ ДІАГНОСТИЧНИХ ЗОБРАЖЕНЬ

3.1 Формування навчальних та тестових даних

Для навчання будь-якої моделі машинного навчання обов’язково потрібні навчальні дані, адже саме на їх основі відбувається тренування та тестування моделей, завдяки чому, вона набуває здатності до аналізу. Саме оптимально підібрані дані, забезпечують можливість отримання моделлю необхідних знань та залежностей для подальшого застосування на практиці при розв’язанні реальних задач.

Обсяг та якість навчальних даних напряму впливає на якість моделі та її здатність узагальнювати отримані знання. Для створення точної та ефективної моделі машинного навчання, вона має бачити під час тренування репрезентивні та різноманітні приклади реальних даних.

3.1.1 Формування навчальної вибірки

Проект MedMNIST є одним із унікальних відкритих наборів даних для комп’ютерного зору та аналізу медичних зображень. Він містить в собі велику колекцію стандартизованих біометричних зображень, такі як рентгенограми, гістологічні препарати, зображення очного дна, знімки УЗД, дерматологічні зображення тощо.

Усі набори даних, попередньо обробляються приводяться до одного розміру та класифікуються. Збірка цих даних спеціально призначені для виконання завдань класифікації на малих зображеннях. Окрім цього, дані подаються у вигляді стандартного розподілу на тренувальні-валідаційні-тестові набори даних, що можуть легко застосовуватись у машинному навчанні [24].

Для виконання поставленої задачі, було обрано один з наборів даних представленого на ресурсі MedMNIST, це BloodMNIST, що базується на наборі даних окремих клітин крові. Зразки даних були взяті у осіб без інфекційних, гематологічних або онкологічних захворювань і без фармакологічного лікування

в момент збору крові, що забезпечує якість цих даних. Ця колекція містить загалом 17092 зображення, організованих на 8 класів. Вихідний набір даних розділено у співвідношенні 7:1:2 на на тренувальний, валідаційний та тестовий набори. Зображення вихідного набору даних з роздільною здатність 3x360x363 пікселів, було обрізано по центру до 3x200x200, та потім зменшено до 3×28×28.

Розподіл класів у тестовому та тренувальному наборах даних, зазначено на графіку, зображеному на рисунку 3.1.

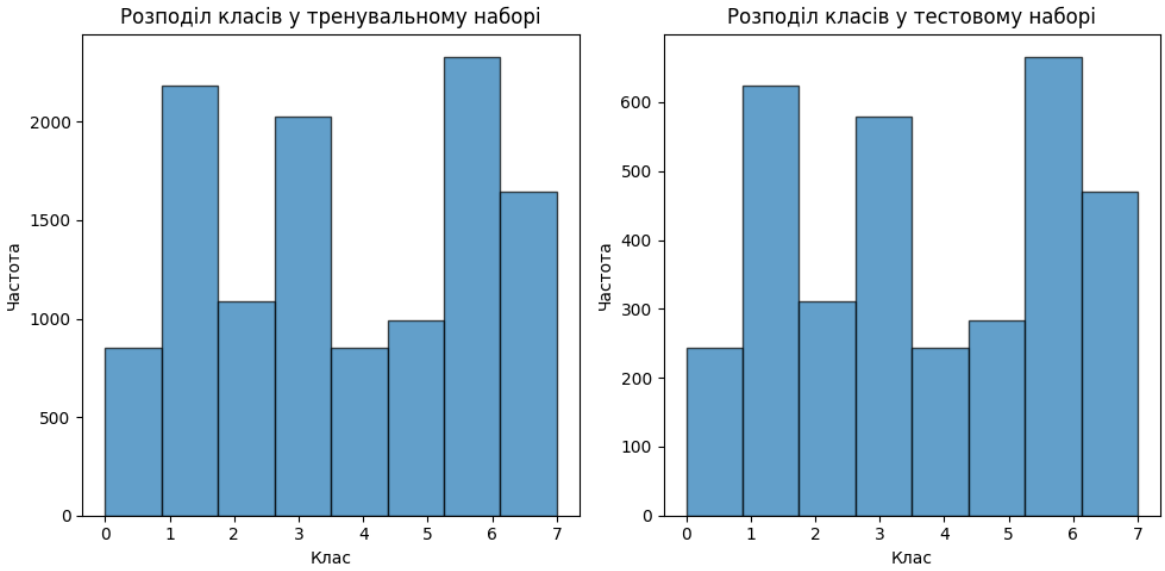

Рисунок 3.1 – Розподіл класів у наборі даних

На рисунках 3.2, 3.3, наведено загальний приклад зображень, зібраних у наборі даних BloodMNIST.

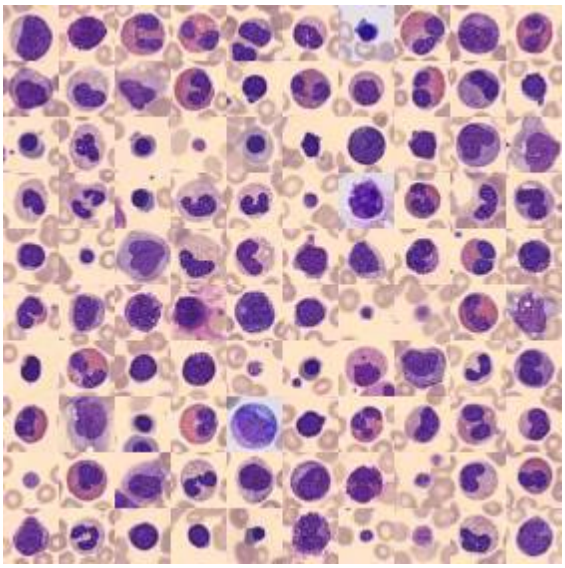

Рисунок 3.2 – Приклад зображень з набору даних

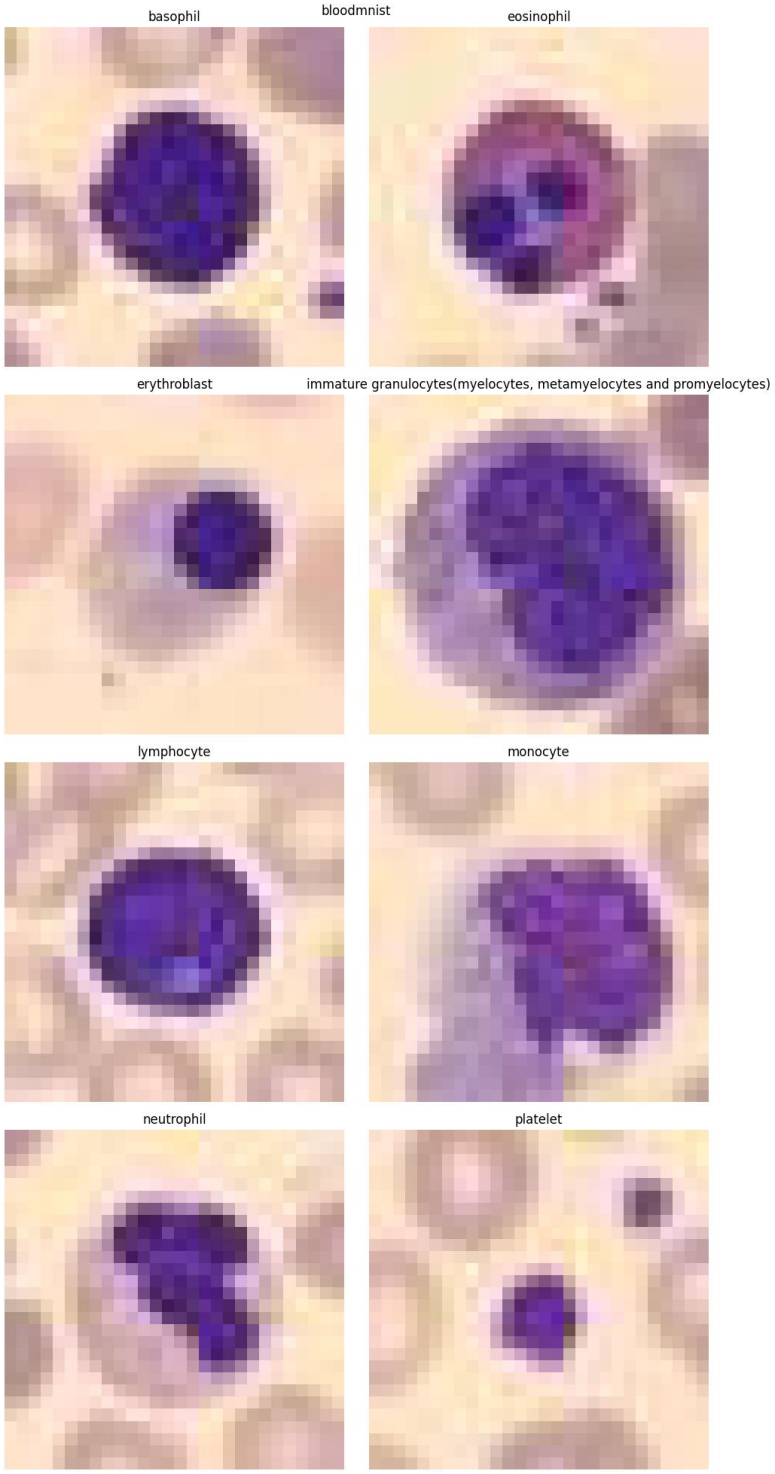

Рисунок 3.3 – Приклад зображень для кожного з класів

3.1.2 Формування протиборчих атак на навчальну вибірку

Для тестування робастності моделі, було впроваджено метод генерації протиборчих прикладів з використанням Fast Gradient Sign Method (FGSM). Цей метод полягає у внесенні маленьких, але цілеспрямованих змін до вхідних даних,

заснованих на градієнтах функції втрати моделі. Додавання цього "шуму" може значно вплинути на вихід моделі, при цьому майже не змінюючи зовнішній вигляд зображення для людського ока.

Функція generate_adversarial створює протиборчі зображення, які потім, для нормалізації, обрізаються до діапазону [0, 1]. Змінна epsilon використовується для контролювання інтенсивності змін, які вносяться у зображення. Для даної моделі, обрано значення epsilon = 0.05, яке є достатньо малим, щоб майже не змінити зовнішній вигляд зображень, але при цьому здатним вплинути на роботу моделі.

На рисунку 3.4 наведено пари оригінальних та згенерованих протиборчих зображень. Завдяки тому що зображення мають малу розмірність (28 на 28 пікселів), атаковані зображення можна відрізнити від оригінальних при значному збільшенні. Без збільшення такі атаки важко помітні людському оку, але при цьому вони можуть вводити модель в оману.

Рисунок 3.4 – Порівняння оригінальних та протиборчих зображень

3.2 Короткий опис програмного забезпечення

Програмне забезпечення для аналізу та обробки медичних зображень було реалізовано за допомогою середовища Google Colab. Google Colab — це хмарний сервіс, який надає можливість виконання коду в Jupyter Notebooks без необхідності локальної установки. Основною перевагою цього середовища є доступ до потужних обчислювальних ресурсів, включаючи графічні процесори (GPU) та тензорні процесори (TPU), що є критично важливим для глибокого навчання та обробки великих наборів даних.

Програмний код був структурований у вигляді Jupyter Notebook, що включає послідовність блоків (cells), розділених на два основних типи: блоки коду та блоки з розміткою (markdown). Блоки з розміткою містять описи та пояснення до коду, що сприяє кращому розумінню логіки роботи та навчального процесу. Код програми наведено у Додатку А

Початкові блоки коду відповідають за інсталяцію необхідних бібліотек та імпортування модулів. Це включає бібліотеки для глибокого навчання (наприклад, TensorFlow), обробки даних (numpy, pandas) та візуалізації результатів (matplotlib).

Виконується налаштування параметрів для обробки даних та архітектури моделі. Функції для завантаження та підготовки даних (MedMNIST) демонструють процес роботи з медичними зображеннями.

Наступний розділ містить код для створення архітектури нейронної мережі. Архітектура мережі, включає конволюційні, пулінгові та повнозв'язні шари, адаптовані для завдань класифікації зображень.

Застосування методу Fast Gradient Sign Method (FGSM) для створення протиборчих прикладів, що використовуються для оцінки стійкості моделі.

Реалізовано процеси навчання та оцінки моделі на різних наборах даних, включаючи стандартні та протиборчі. Використано механізми ранньої зупинки, збереження найкращої моделі та адаптивного змінення швидкості навчання.

Подано графічні представлення динаміки навчання моделі, включаючи точність та втрати, а також матриці помилок для оцінки класифікації.

Використано архітектуру DAE для попередньої обробки вхідних зображень перед їх поданням до конволюційної нейронної мережі, що дозволило підвищити якість та точність класифікації.

Проведено оцінку моделі на стандартних та протиборчих наборах даних, що демонструє ефективність підходу та важливість включення DAE в процес навчання.

Ці елементи програмного забезпечення забезпечують гнучкість та ефективність при роботі з медичними зображеннями, демонструючи важливість застосування передових методів глибокого навчання в медичній діагностиці.

Програмне забезпечення, розроблене в Google Colab, демонструє гнучкість та ефективність у роботі з медичними зображеннями, використовуючи сучасні методи глибокого навчання. Організація коду у вигляді Jupyter Notebook сприяє зручності візуалізації та інтерактивного аналізу результатів, що є ключовим для наукових досліджень та освітнього процесу.

3.3 Постановка та аналіз результатів експериментів

Навчальні дані було завантажено та підготовлено для роботи. Було сформовано 3 набори даних(тренувальний, валідаційний та тестовий), що містять в собі оригінальні зображення, завантажені з ресурсу MedMNIST, що складається з 8 різних класів зображень клітин крові.

На наступному етапі початкова модель тренувалась на навчальному наборі. Під час кожної епохи модель виконує навчання, підлаштовуючи свої ваги і параметри для зменшення функції втрат Sparse Categorical Crossentropy, та підвищення точності передбачень. На рисунку 3.5 зображено графік що відображають залежність точності та втрат від кількості епох навчання. Це допомагає отримати візуальне уявлення про продуктивність моделі та її тренди під час тренування.

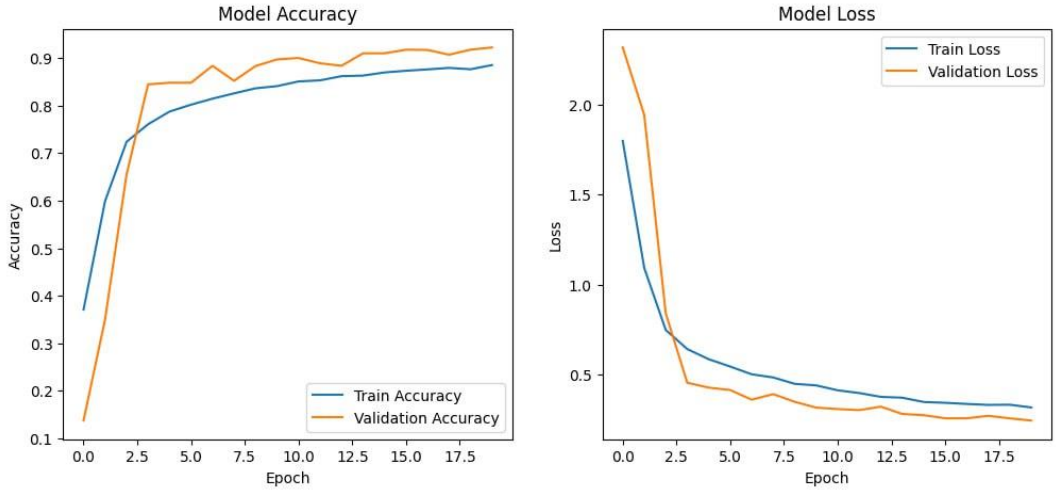

Рисунок 3.5 – Графік точності та графік втрат моделі протягом першого навчання

Графік точності (Accuracy): Цей графік показує, як точність моделі змінювалася на тренувальному та валідаційному наборах даних під час різних епох навчання. Для тренувального та валідаційного наборів наведені окремі криві. За допомогою цього графіка можна визначити, чи спостерігається перенавчання, а також визначити, коли модель досягає найкращої точності

Графік втрат (Loss): Графік втрат відображає, як змінювалася функція втрати моделі під час навчання. Аналогічно до графіка точності, представлені окремі криві для тренувального та валідаційного наборів. Графік дозволяє визначити, як швидко модель зменшує втрати та чи спостерігається перенавчання.

Після цього, модель була протиборчо атакована, щоб оцінити її робастність. Для цього використовувались методи, такі як FGSM (Fast Gradient Sign Method), що дозволили створити спотворені вхідні дані, приклад яких вже було наведено на рисунку 3.4.

Результат оцінки моделі на двох наборах даних: на чистих та протиборчо атакованих зображеннях, наведено на рисунку 3.6. На чистому наборі даних, модель досягає точності 90.1% для тестової вибірки, та 92.2% для валідаційної. Це свідчить про високу ефективність моделі для задачі класифікації клітин крові.

Однак оцінка моделі на зашумлених зображеннях, демонстує досить низький результат: 7.2% для тестової та 6.5% для валідаційної вибірки.

```
--- Standard Model on clean data ---
Validation Loss: 0.2465
Validation Accuracy: 92.2313 %
Test Loss: 0.2617
Test Accuracy: 90.9968 %

--- Standard Model on noisy data ---
Validation Loss: 8.4157
Validation Accuracy: 6.4836 %
Test Loss: 8.429
Test Accuracy: 7.1616 %
```

Рисунок 3.6 – Результат оцінки первинної моделі на чистому та зашумленому наборі даних

Також, на рисунку 3.7, зображено матрицю передбачень для зашумленого набору даних, що дає змогу оцінити, наскільки добре модель класифікує тестові дані, показуючи деталі щодо кожної класифікації (правильної та неправильної), та де саме модель може робити помилки. Зважаючи на результат, можна зробити висновок, що хоч моделі і досить добре справляється з задачею класифікації чистих зображень, вона показує досить низький рівень стійкості до протиборчих атак.

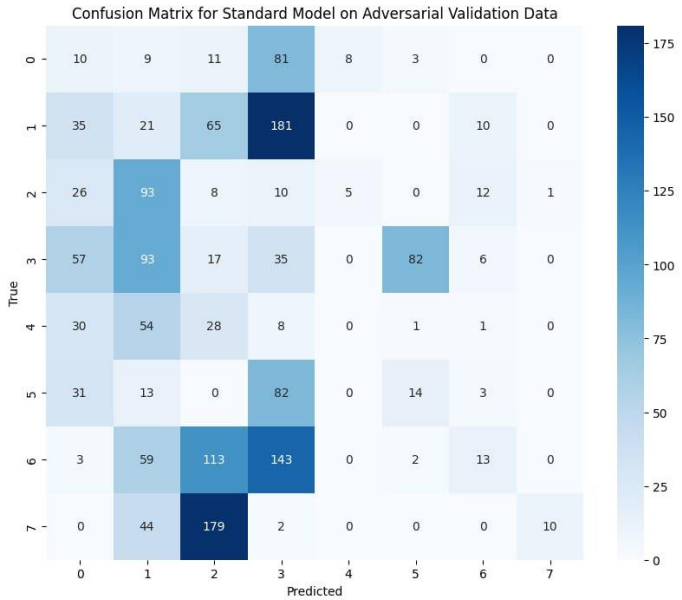

Рисунок 3.7 – Матриця передбачень первинної моделі для протиборчих атак

Після цього модель проходила повторне тренування, отримуючи на вхід протиборчі зображення, що сприяло оцінці її адаптації до можливих збурень та загальному покращенню робастності.

Було проведено повторну оцінку, результати якої, продемонстрували значне покращення точності та робастності для протиборчої вибірки. На графіках (рис. 3.8) можна оцінити зростання точності та зменшення втрат під час другого навчання моделі з використанням протиборчих прикладів.

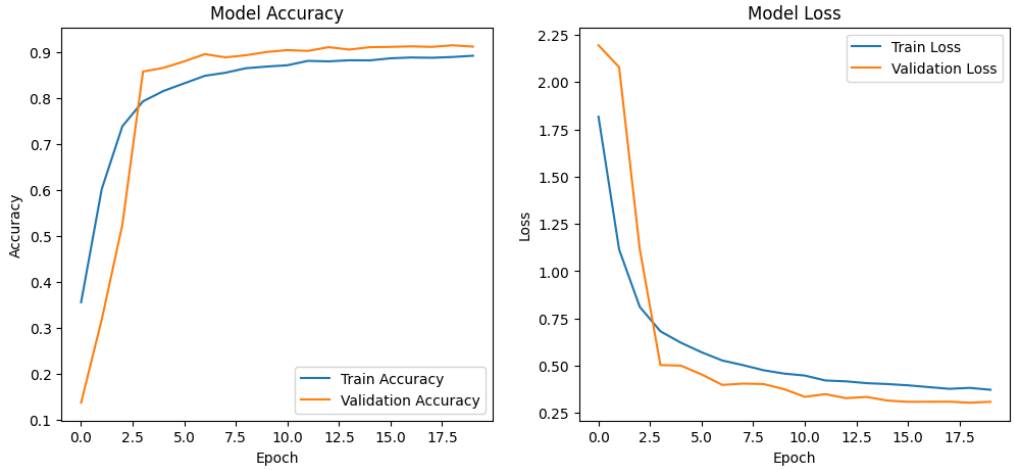

Рисунок 3.8 – Графік точності та графік втрат моделі протягом другого навчання

На рисунку 3.9 наведено результати оцінки моделі для всіх наборів даних. Точність на тестовому зашумленому наборі збільшилася до 89.9%, а на валідаційному - до 91.2%. Це свідчить про підвищення ефективності моделі до класифікації зашумлених зображень. Проте точність оцінки на чистих зображеннях впала до 57.6% для тестового, та 58.4% для валідаційного набору. Отже доцільно реалізувати інший підхід для збереження здатності моделі до аналізу чистих зображень, та класифікації протиборчих зображень.

Також, як і для першого тренування, було реалізовану матрицю передбачень (рис. 3.10), що дозволяє оцінити класифікацію моделі з огляду на конкретні класи.

```
--- Adversarial Trained Model on clean data ---
Validation Loss: 1.0621
Validation Accuracy: 58.3528 %
Test Loss: 1.1114
Test Accuracy: 57.5563 %

--- Adversarial Trained Model on noisy data ---
Validation Loss: 0.3094
Validation Accuracy: 91.2383 %
Test Loss: 0.3608
Test Accuracy: 89.9737 %
```

Рисунок 3.9 – Результат оцінки вторинної моделі

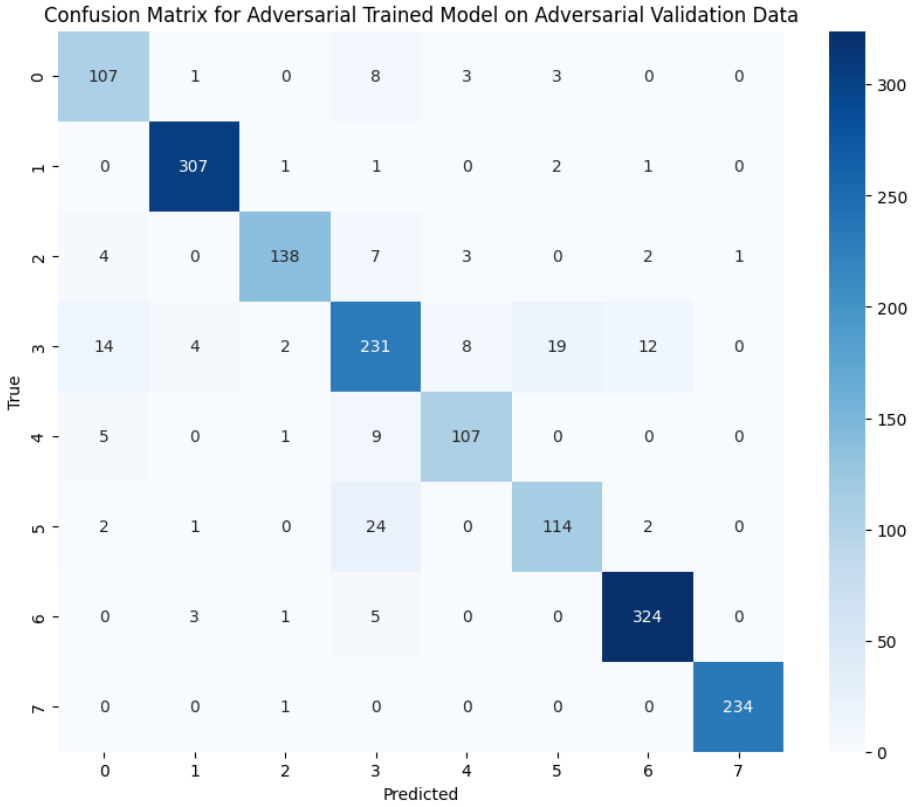

Рисунок 3.10 – Матриця передбачень для вторинної моделі

Наступним етапом було реалізовано знешумлюючий автоенкодер, що навчився відновлювати чисті зображення з шумних вхідних даних. Ефективність автоенкодера оцінювалася шляхом порівняння оригінальних зображень з відновленими, використовуючи метрики, як-от середньоквадратична помилка

(MSE). На рисунку 3.11, можна побачити графік зниження відмінності знешумленого зображення від оригінального

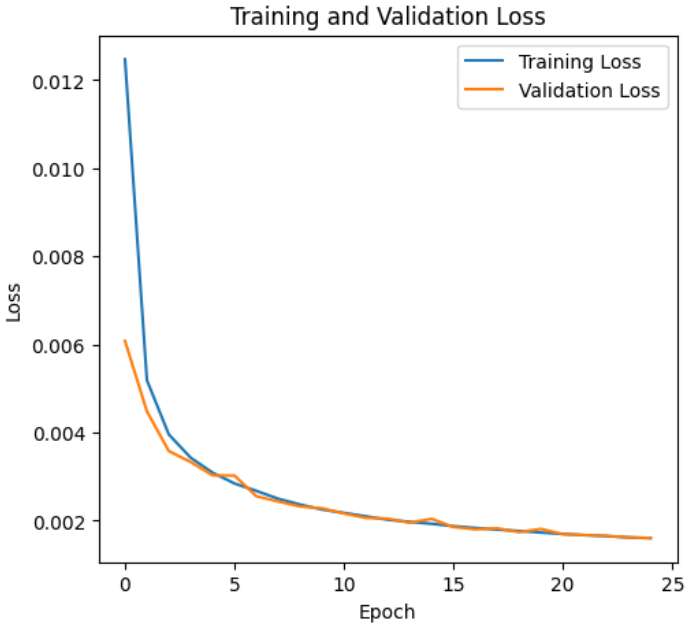

Рисунок 3.11 – Графік зменшення відмінності між знешумленим та оригінальним зображенням

На рисунку 3.12 зображено результат роботи знешумлюючого автоенкодера. Як можна помітити, одель демонструє значний результат по знешумленню вхідних зображень.

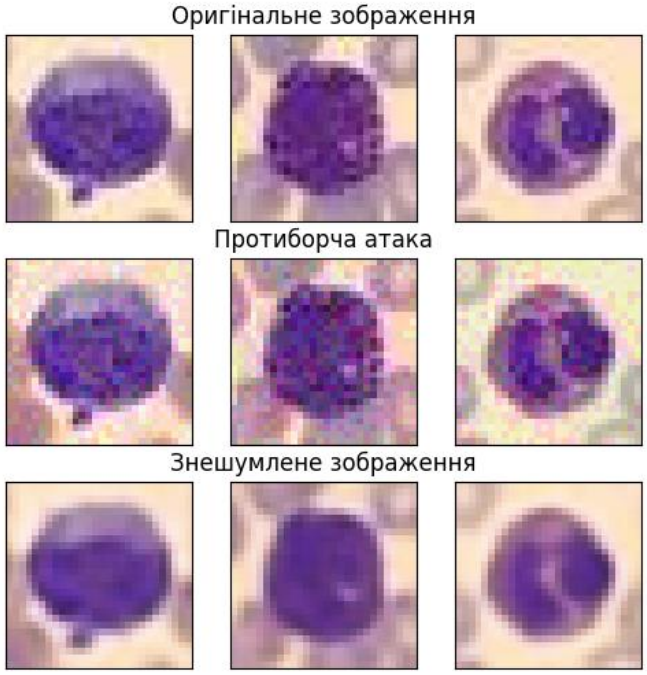

Рисунок 3.12 – Результат роботи знешумлюючого автоенкодера

В подальшому було реалізовано CNN модель з використанням DAE. Принцип роботи полягає у тому, що вхідне зображення піддається попередній обробці перед її поданням до конволюційної нейронної мережі, що дозволило підвищити якість та точність класифікації. Після тренування моделі (рис. 3.13) на протиборчих прикладах, модель продемонструвала покращення загальних результатів до оцінки як чистих зображень, так і протиборчих атак. Як можна побачити з рисунку 3.14, точність моделі на чистих даних становить 85.6% для тестового та 87.9% для валідаційного набору, та для протиборчих атак: 86.2% - тестовий, 87.5%-валідаційний. Це вказує на високу робастність моделі не тільки до стандартних умов використання, але й до потенційних зовнішніх викликів у вигляді зашумлених даних.

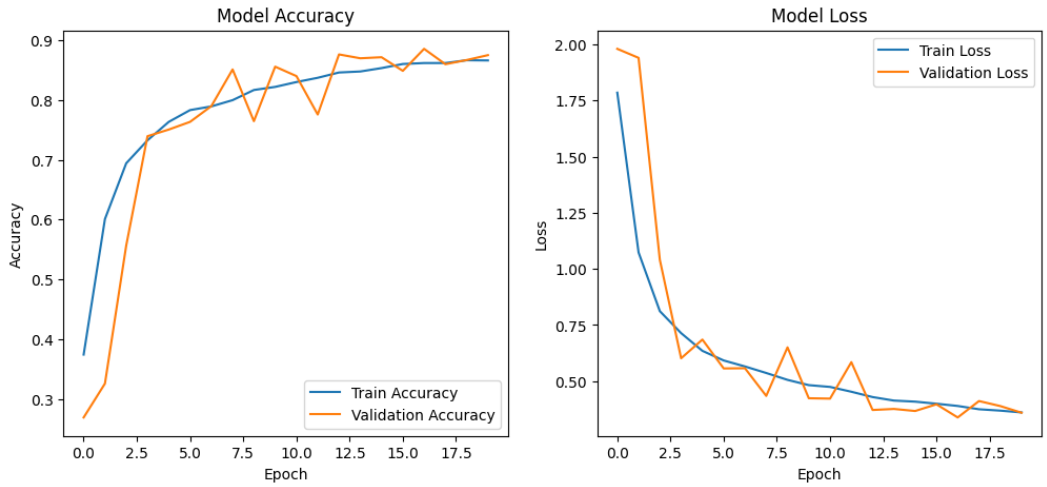

Рисунок 3.13 – Графік точності та графік втрат для третьої моделі

```
--- Adversarial DAE Trained Model on clean data ---
Validation Loss: 0.3448
Validation Accuracy: 87.8505 %
Test Loss: 0.3715
Test Accuracy: 85.9398 %

--- Adversarial DAE Trained Model on noisy data ---
Validation Loss: 0.3595
Validation Accuracy: 87.5 %
Test Loss: 0.3799
Test Accuracy: 86.2321 %
```

Рисунок 3.14 – Оцінка точності третьої моделі

На рисунку 3.15 наведено матрицю передбачень для останньої моделі для протиборчих атак, що дозволяє оцінити якість моделі з огляду на класифікацію конкретних класів зображень.

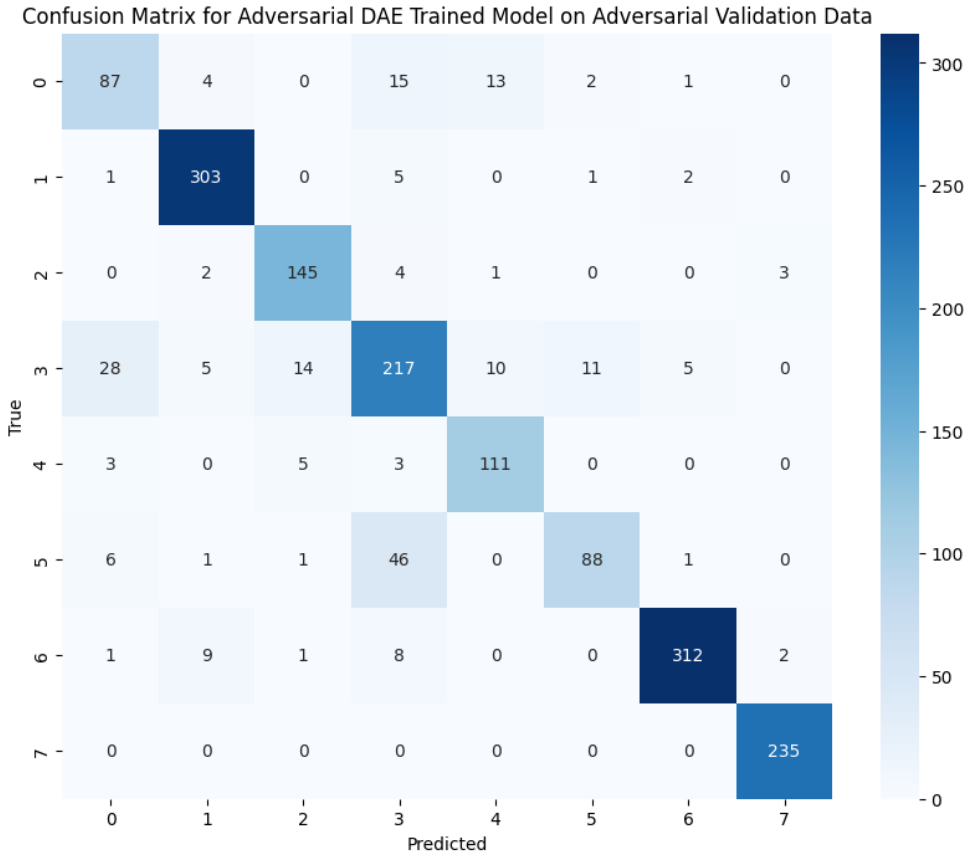

Рисунок 3.15 – Матриця передбачень для третьої моделі

Експерименти з класифікації зображень клітин крові підтвердили, що застосування протиборчих атак та знешумлюючих автоенкодерів ефективно сприяє підвищенню робастності та точності моделі. Використання DAE для попередньої обробки даних дозволило моделі краще "розуміти" та відфільтровувати шум, підвищуючи точність класифікації. Ці результати вказують на необхідність комплексного підходу до тренування та оцінки нейронних мереж, особливо у сфері медичної обробки зображень.

ВИСНОВКИ

У даній роботі було проведено дослідження використання інформаційних технологій для забезпечення робастності системи розпізнавання медичних діагностичних зображень. Основна увага була зосереджена на аналізі та покращенні точності та робастності нейронних мереж у контексті класифікації медичних зображень.

Було проведено експерименти з використанням різних підходів, включаючи протиборчі атаки та знешумлюючі автоенкодери, для визначення їх впливу на робастність моделі. Результати показали, що інтеграція цих методів значно підвищує здатність системи до адаптації до збурень та покращує її точність.

Дослідження також включало аналіз візуальних характеристик оброблених зображень, що дозволило краще зрозуміти поведінку моделі та оцінити її ефективність. Отримані результати демонструють значний потенціал застосування сучасних інформаційних технологій у сфері медичної діагностики та підкреслюють важливість продовження досліджень у цій області.

У підсумку, було реалізовано модель класифікації медичних діагностичних зображень клітин крові та в подальшому покращено її робастність. Отримані висновки можуть бути використані для покращення існуючих систем та розробки нових рішень у галузі медичної діагностики.

СПИСОК ЛІТЕРАТУРИ

1. Hamet, Pavel; Tremblay, Johanne. Artificial intelligence in medicine. Metabolism, 2017, 69: S36-S40.

2. Coates, Ashley Stephen; BA, B. A.; Wareham, Christopher Simon. CAREBOTS AND THE VIRTUE OF CARE. 2021.

3. ТСН У Вінниці робот-хірург виконав першу в Україні операцію, URL: https://tsn.ua/ukrayina/u-vinnici-robot-hirurg-vikonav-pershu-v-ukrayini-operaciyu-1319292.html (дата звернення: 15.11.2023).

4. Malik, Paras, et al. Overview of artificial intelligence in medicine. Journal of family medicine and primary care, 2019, 8.7: 2328.

5. Soto, Fernando, et al. Medical micro/nanorobots in precision medicine. Advanced Science, 2020, 7.21: 2002203.

6. Liu, Chang, et al. A comprehensive study on robustness of image classification models: Benchmarking and rethinking. arXiv preprint arXiv:2302.14301, 2023.

7. Guo, Yanying; Yang, Guoqing; Gao, Qingji. A new approach for the improving result of Edge Detection. In: 2010 International Conference on Measuring Technology and Mechatronics Automation. IEEE, 2010. p. 169-172.

8. Hasan, Mokhtar M.; Mishra, Pramod K. Improving morphology operation for 2D hole filling algorithm. International Journal of Image Processing (IJIP), 2012, 6.1: 635-646.

9. Bishwas, Arit Kumar; Mani, Ashish; Palade, Vasile. Gaussian kernel in quantum learning. International Journal of Quantum Information, 2020, 18.03: 2050006.

10. Kalman Filter Explained Simply - The Kalman Filter, URL: https://thekalmanfilter.com/kalman-filter-explained-simply/ (дата звернення: 23.11.2023).

11. Brownlee, Jason. A gentle introduction to expectation-maximization (EM Algorithm). Machine Learning Mastery, Oct, 2019, 31.

12. Getting started with Image Processing Using OpenCV, URL: https://www.analyticsvidhya.com/blog/2023/03/getting-started-with-image-processing-using-opencv/ (дата звернення: 25.11.2023).

13. IMAGE RECOGNITION: CHOOSING THE RIGHT AI MODEL FOR YOUR PROJECT, URL: https://www.sentisight.ai/image-recognition-choosing-the-right-ai-model-for-your-project/ (дата звернення: 27.11.2023).

14. Xiao, Kai, et al. Noise or signal: The role of image backgrounds in object recognition. arXiv preprint arXiv:2006.09994, 2020.

15. Image Processing: Techniques, Types, & Applications [2023], URL: https://www.v7labs.com/blog/image-processing-guide (дата звернення: 30.11.2023).

16. Denoising Autoencoders, URL: https://medium.com/@harishr2301/denoising-autoencoders-996e866e5cd0 (дата звернення: 01.12.2023).

17. Niu, Zhonghan, et al. On the limitations of denoising strategies as adversarial defenses. arXiv preprint arXiv:2012.09384, 2020.

18. Papernot, Nicolas, et al. Distillation as a defense to adversarial perturbations against deep neural networks. In: 2016 IEEE symposium on security and privacy (SP). IEEE, 2016. p. 582-597.

19. Tramèr, Florian, et al. Ensemble adversarial training: Attacks and defenses. arXiv preprint arXiv:1705.07204, 2017.

20. Moosavi-Dezfooli, Seyed-Mohsen, et al. Universal adversarial perturbations. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017. p. 1765-1773.

21. Sharma, Aditya. Convolutional Neural Networks in Python with Keras. Convolutional Neural Networks in Python DataCamp, 2017.

22. Schmidt, Robin M. Recurrent neural networks (rnns): A gentle introduction and overview. arXiv preprint arXiv:1912.05911, 2019.

23. Xinyi, Zhang; Chen, Lihui. Capsule graph neural network. In: *International conference on learning representations*. 2018.

24. Yang, Jiancheng, et al. MedMNIST v2-A large-scale lightweight benchmark for 2D and 3D biomedical image classification. Scientific Data, 2023, 10.1: 41.

ДОДАТОК А - ФРАГМЕНТИ ПРОГРАМНОГО КОДУ

Інсталяція та імпорт необхідних бібліотек

```
!pip install medmnist
!pip install tensorflow --upgrade
!pip install cleverhans

import medmnist
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt

from medmnist import INFO
from tensorflow.keras import Sequential
from tensorflow.keras.models import Model
from cleverhans.tf2.attacks import fast_gradient_method
from tensorflow.keras.losses import SparseCategoricalCrossentropy
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint,
LearningRateScheduler
from tensorflow.keras.layers import Conv2D, BatchNormalization, ReLU,
MaxPooling2D, Dense, Dropout, Flatten
print("medMNIST: ", medmnist.__version__)
```

Підготовка, завантаження, та нормалізація даних

```
#Prepping dataset
def download_and_prepare_dataset(data_info: dict):
    data_path = tf.keras.utils.get_file(origin=data_info['url'],
md5_hash=data_info['MD5'])

    with np.load(data_path) as data:
        # Get images
        train_images = data['train_images']
        val_images = data['val_images']
        test_images = data['test_images']

        # Get labels
        train_labels = data['train_labels'].flatten()
        valid_labels = data['val_labels'].flatten()
        test_labels = data['test_labels'].flatten()

    return train_images, train_labels, val_images, valid_labels,
test_images, test_labels

train_images, train_labels, val_images, valid_labels, test_images,
test_labels = download_and_prepare_dataset(info)
```

```
#Визначення функцій preprocess і prepare_dataloader
@tf.function
def preprocess(image: tf.Tensor, label: tf.Tensor):
    image = tf.image.convert_image_dtype(image, tf.float32)
    label = tf.cast(label, tf.int32)
    return image, label

def prepare_dataloader(images: np.ndarray, labels: np.ndarray, loader_type:
str = 'train'):
    dataset = tf.data.Dataset.from_tensor_slices((images, labels))
    if loader_type == 'train':
        dataset = dataset.shuffle(1024)
    dataloader = dataset.map(preprocess,
num_parallel_calls=tf.data.AUTOTUNE).batch(configs['batch_size']).prefetch(t
f.data.AUTOTUNE)
    return dataloader
def normalize_images(images):
    # Кліпування значень пікселів
    images_clipped = np.clip(images, 0, 255)

    # Нормалізація значень пікселів до діапазону [0, 1]
    images_normalized = images_clipped / 255.0

    return images_normalized

train_images = normalize_images(train_images)
val_images = normalize_images(val_images)
test_images = normalize_images(test_images)

# Ініціалізація даталоадерів
train_dataloader = prepare_dataloader(train_images, train_labels, 'train')
val_dataloader = prepare_dataloader(val_images, valid_labels, 'valid')
test_dataloader = prepare_dataloader(test_images, test_labels, 'test')
```

Архітектура для навчання CNN моделі

```
# Функція для створення моделі
def cnn_model():
    model = Sequential([
        Conv2D(32, kernel_size=(3, 3), input_shape=(configs['image_height'],
configs['image_width'], configs['image_channels']), padding='same'),
        BatchNormalization(),
        ReLU(),
        Conv2D(32, kernel_size=(3, 3), padding='same'),
        BatchNormalization(),
        ReLU(),
        MaxPooling2D(pool_size=(2, 2)),
```

```
        Dropout(0.3),

        Conv2D(64, kernel_size=(3, 3), padding='same'),
        BatchNormalization(),
        ReLU(),
        Conv2D(64, kernel_size=(3, 3), padding='same'),
        BatchNormalization(),
        ReLU(),
        MaxPooling2D(pool_size=(2, 2)),
        Dropout(0.4),

        Flatten(),
        Dense(256, activation='relu'),
        Dropout(0.5),
        Dense(configs['num_classes'], activation='softmax')
    ])

    return model
```

Функції необхідні для навчання моделі:

```
# Функція для адаптивної зміни швидкості навчання
def scheduler(epoch, lr):
    if epoch < 10:
        return lr
    else:
        return lr * tf.math.exp(-0.1)
# Callbacks
early_stopping = EarlyStopping(monitor='val_loss', patience=5,
restore_best_weights=True)
def create_model_checkpoint(filename):
    return ModelCheckpoint(filename, save_best_only=True)
lr_scheduler = LearningRateScheduler(scheduler)
```

Компіляція та тренування першої моделі

```
# Compile and train the model
model = cnn_model()
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=configs['best
_lr']),
              loss=SparseCategoricalCrossentropy(),
              metrics=['accuracy'])

# Evaluate the model on the validation set before training
val_loss, val_accuracy = model.evaluate(val_dataloader)
print("Validation Loss before training:", val_loss)
print("Validation Accuracy before training:", val_accuracy)
```

```
# Train the model
start_history = model.fit(train_dataloader,
                    epochs=configs['best_epoch'],
                    validation_data=val_dataloader,
                    callbacks=[early_stopping,
create_model_checkpoint('start_model.keras'), lr_scheduler])
```

Генерація adversarial examples

```
def generate_adversarial(model, images, labels):
    adversarial_images = fast_gradient_method.fast_gradient_method(model,
images, configs['epsilon'], np.inf)
    adversarial_images_clipped = np.clip(adversarial_images, 0, 1)
    return adversarial_images_clipped

# Генерація adversarial examples для кожного набору

adversarial_train_images = generate_adversarial(model, train_images,
train_labels)
adversarial_val_images = generate_adversarial(model, val_images,
valid_labels)
adversarial_test_images = generate_adversarial(model, test_images,
test_labels)

# Prepare dataloaders for mixed data
adv_train_dataloader = prepare_dataloader(adversarial_train_images,
train_labels, 'train')
adv_val_dataloader = prepare_dataloader(adversarial_val_images,
valid_labels, 'valid')
adv_test_dataloader = prepare_dataloader(adversarial_test_images,
test_labels, 'test')
```

Оцінка моделі

```
dataloaders = {
    'clean': {'val': val_dataloader, 'test': test_dataloader},
    'noisy': {'val': adv_val_dataloader, 'test': adv_test_dataloader},
}

# @title
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import tensorflow as tf

def evaluate_model(model, model_name, history_fit=None):
    # Evaluating the model on different data types
```

```
    for data_type, dataloader in dataloaders.items():
        val_loss, val_accuracy = model.evaluate(dataloader['val'],
verbose=0)
        test_loss, test_accuracy = model.evaluate(dataloader['test'],
verbose=0)

        print(f"--- {model_name} on {data_type} data ---")
        print("Validation Loss:", round(val_loss, configs['decimals']))
        print("Validation Accuracy:", round(val_accuracy * 100,
configs['decimals']), "%")
        print("Test Loss:", round(test_loss, configs['decimals']))
        print("Test Accuracy:", round(test_accuracy * 100,
configs['decimals']), "%\n")
    if history_fit:
        # Plotting training history
        plt.figure(figsize=(12, 5))
        plt.subplot(1, 2, 1)
        plt.plot(history_fit.history['accuracy'], label='Train Accuracy')
        plt.plot(history_fit.history['val_accuracy'], label='Validation
Accuracy')
        plt.title('Model Accuracy')
        plt.xlabel('Epoch')
        plt.ylabel('Accuracy')
        plt.legend()

        plt.subplot(1, 2, 2)
        plt.plot(history_fit.history['loss'], label='Train Loss')
        plt.plot(history_fit.history['val_loss'], label='Validation Loss')
        plt.title('Model Loss')
        plt.xlabel('Epoch')
        plt.ylabel('Loss')
        plt.legend()
        plt.show()

    # Generating predictions for the adversarial validation dataset
    predicted_labels_val = model.predict(adv_val_dataloader)
    predicted_labels_val = np.argmax(predicted_labels_val, axis=1)

    # Generating the confusion matrix
    confusion_matrix_val = tf.math.confusion_matrix(valid_labels,
predicted_labels_val)

    # Visualizing the confusion matrix
    plt.figure(figsize=(10, 8))
    sns.heatmap(confusion_matrix_val, annot=True, fmt='g', cmap='Blues',
xticklabels=configs['class_names'], yticklabels=configs['class_names'])
    plt.xlabel('Predicted')
    plt.ylabel('True')
```

```
    plt.title(f'Confusion Matrix for {model_name} on Adversarial Validation
Data')
    plt.show()
```

Компіляція та тренування першої моделі

```
# If you need a separate copy of the model
modele_avg = tf.keras.models.clone_model(model)

# Ensure the model is compiled with a fresh optimizer if needed
modele_avg.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=configs[
'best_lr']),
                   loss=tf.keras.losses.SparseCategoricalCrossentropy(from_l
ogits=True),
                   metrics=['accuracy'])

# Training the model on adversarial examples
history_avg = modele_avg.fit(adv_train_dataloader,
                             epochs=configs['best_epoch'],
                             validation_data=adv_val_dataloader,
                             callbacks=[early_stopping,
create_model_checkpoint('adv_model.keras'), lr_scheduler])
```

Створення та тренування знешумлюючого автоенкодера

```
# Denoising Autoencoder Architecture
class DenoisingAutoencoder(Model):
    def __init__(self):
        super(DenoisingAutoencoder, self).__init__()
        self.encoder = Sequential([
            Conv2D(32, kernel_size=(3, 3), padding='same',
input_shape=(configs['image_height'], configs['image_width'],
configs['image_channels']), activation='relu'),
            MaxPooling2D(pool_size=(2, 2)),
            Conv2D(64, kernel_size=(3, 3), padding='same',
activation='relu'),
            MaxPooling2D(pool_size=(2, 2))
        ])

        self.decoder = Sequential([
            Conv2D(64, kernel_size=(3, 3), padding='same',
activation='relu'),
            UpSampling2D(size=(2, 2)),
            Conv2D(32, kernel_size=(3, 3), padding='same',
activation='relu'),
            UpSampling2D(size=(2, 2)),
```

```
            Conv2D(configs['image_channels'], kernel_size=(3, 3),
padding='same', activation='sigmoid')
        ])

    def call(self, x):
        encoded = self.encoder(x)
        decoded = self.decoder(encoded)
        return decoded

dae_model = DenoisingAutoencoder()
dae_model.compile(optimizer='adam', loss='mean_squared_error')


# Training the DAE
history_dae = dae_model.fit(adversarial_train_images,
              train_images,
              epochs=25,
              batch_size=configs['batch_size'],
              validation_data=(adversarial_val_images, val_images))
```

Компіляція та навчання третьої моделі

```
# Combine DAE and CNN for noisy data classification
model_dae_avg = Sequential([
    dae_model,
    cnn_model()  # Assuming cnn_model() is the function that creates the CNN
model
])

model_dae_avg.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=confi
gs['best_lr']),
                      loss=tf.keras.losses.SparseCategoricalCrossentropy(from_l
ogits=True),
                      metrics=['accuracy'])

# Training the DAE model on adversarial examples
history_dae_avg = model_dae_avg.fit(adv_train_dataloader,
                                    epochs=configs['best_epoch'],
                                    batch_size=configs['batch_size'],
                                    validation_data=adv_val_dataloader,
                                    callbacks=[early_stopping,
create_model_checkpoint('model_dae_avg.keras'), lr_scheduler])
```