Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

ДОСЛІДЖЕННЯ ТА РОЗРОБКА МЕТОДІВ АНАЛІЗУ ЗОБРАЖЕНЬ ДЛЯ ПРОТИДІЇ СПУФІНГУ У СИСТЕМАХ РОЗПІЗНАВАННЯ ОБЛИЧ
(тема)

Виконав:
здобувач 2 року навчання,
групи ІНФм-24-2
Богун Р.С.
(прізвище, ініціали)
Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Інформатика
(повна назва освітньої програми)
Науковий керівник ст. викл. Путятіна О. Є.
(посада, прізвище, ініціали)

Допускається до захисту

Завідувач кафедри інформатики _____ Кобилін О. А.
(підпис) (прізвище, ініціали)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту

Кафедра Інформатики

Рівень вищої освіти другий (магістерський)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Богуну Руслану Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження та розробка методів аналізу зображень для протидії спуфінгу у системах розпізнавання обличчя

затверджена наказом університету від 14 листопада 2025 року № 1045Ст
2. Термін подання здобувачем роботи до екзаменаційної комісії 30 листопада 2025 р.
3. Вихідні дані до роботи методи класифікації зображень, літературні джерела щодо застосування методів класифікації, програмні засоби для реалізації вибраних методів класифікації, зображення людей та їх імітації для тренування та тестування моделей.

4. Перелік питань, що потрібно опрацювати в роботі
1. Аналіз сучасних методів класифікації об'єктів на зображеннях.
2. Аналіз літературних джерел щодо апробації методів класифікації об'єктів на зображеннях.
3. Формування алгоритму для класифікації зображень.
4. Візуалізація сформованих алгоритмів та їх результатів.
5. Розробка програмного застосунку, що надасть змогу класифікувати зображення імітації людей та справжніх людей згідно вимог до безпеки та комфорту потенційних користувачів.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) актуальність проблеми класифікації об'єктів на зображеннях, об'єкт та мета дослідження, постановка задачі, блок-схеми алгоритмів вибраних методів класифікації, матриці плутанини згідно з заданих налаштувань системи навчання, ілюстрації розробленого модуля протидії спуфінгу з елементами керування для запуску навчання або виведення результатів, ілюстрація результатів класифікації обраної моделі або ансамблю, висновки, перспективи та апробація роботи.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	29.09.2025	
2	Аналіз завдання, підбір літератури	30.09.25-07.10.25	
3	Аналіз літератури з досліджуваної проблеми	08.10.25-14.10.25	
4	Особливості методів аналізу зображень	15.10.25-20.10.25	
5	Дослідження методів аналізу протидії спуфінгу	21.10.25-27.10.25	
6	Програмна реалізація	28.10.25-05.11.25	
7	Обґрунтування отриманих результатів	06.11.25-11.11.25	
8	Оформлення пояснювальної записки	12.11.25-14.11.25	
9	Перевірка на нормоконтроль	19.11.25-10.12.25	
10	Перевірка на плагіат	20.11.25-10.12.25	
11	Рецензування	21.11.25-10.12.25	
12	Підготовка презентації та доповіді	21.11.25-22.12.25	
13	Занесення роботи в електронний архів	21.11.25-22.12.25	
14	Попередній захист кваліфікаційної роботи	01.12.25-22.12.25	

Дата видачі завдання 29 вересня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ ст. викл. Путятіна О. Є.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи: 65 с., 2 табл., 15 рис., 1 дод., 25 джерел.

БІБЛІОТЕКА PYTORCH, БІБЛІОТЕКА TORCHVISION, БЛОК-СХЕМА, ДЕСКТОП-ЗАСТОСУНОК, КЛАСИФІКАЦІЯ ЗОБРАЖЕНЬ, ENSEMBLE CLASSIFICATION, FAS MODEL, MOBILENETV2, SPOOFING, TRANSFER LEARNING.

Об’єктом дослідження є зображення людей та зображення які імітують людей.

Метою дослідження є розробка та дослідження комплексної архітектури протидії спуфінгу, яка на основі ансамблевого навчання забезпечує високу ефективність виявлення атак.

Використано методи аналізу зображень CNN, зважений завантажувач даних пакетів та зважену функцію втрат, та ансамблеву агрегацію. Проведено аналіз сучасних методів протидії спуфінгу та літературних джерел за темою. Сформовано та візуалізовано алгоритм методів блок-схемами.

Наукова новизна роботи полягає у розробці та експериментальному обгрунтуванні комплексної методології навчання для подолання викликів FAS-систем.

Ця робота має взаємозв’язок з іншими роботами з методології PAD, слідуючи тренду аналізу текстур та посилює теоритичну базу обробки незбалансованих даних.

Результати кваліфікаційної роботи, отримані в ході досліджень мають місце для подальшого розвитку точності для комфорту користувачів.

У результаті дослідження розроблено CLI-застосунок для протидії спуфінгу, що надає можливість порівняльної оцінки ефективності трьох моделей та ансамблю за показчиком точності, матриці плутанини.

ABSTRACT

Explanatory note to the qualification work: 65 pages, 2 table, 15 figures, 1 appendix, 25 sources.

BLOCK DIAGRAM, CNN, ENSEMBLE CLASSIFICATION, FAS MODEL, IMAGE CLASSIFICATION, MOBILENETV2, PYTORCH LIBRARY, SPOOFING, TRANSFER LEARNING.

The object of the study is images of people and images that imitate people.

The purpose of the study is to develop and study a comprehensive architecture for countering spoofing, which, based on ensemble learning, provides high efficiency in detecting attacks.

CNN image analysis methods, weighted packet data loader and weighted loss function, and ensemble aggregation were used. An analysis of modern methods of countering spoofing and literature sources on the topic was conducted. An algorithm of methods was formed and visualized with flowcharts.

The scientific novelty of the work lies in the development and experimental substantiation of a comprehensive training methodology to overcome the challenges of FAS systems.

This work is interconnected with other works on the PAD methodology, following the trend of texture analysis and strengthening the theoretical basis for processing unbalanced data.

The results of the qualification work obtained during the research have a place for further development of accuracy for user comfort.

As a result of the research, a CLI application was developed to combat spoofing, which allows for a comparative assessment of the effectiveness of three models and the ensemble based on the accuracy indicator and confusion matrix.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

CNN – Convolutional Neural Network (згорткова нейронна мережа)

Depth – глибина

DSC – Deepth Separapable Convolution (глибинно-роздільна згортка)

DW – Deepthwise Convolution (глибинна згортка)

FAS – Face Anti Spoofing (протидія спуфінгу облич)

FN – False Negative (хибно негативний)

FNR – False Negative Rate (частка хибно негативного)

FP – False Positive (хибно позитивний)

FPR – False Positive Rate (частка хибно позитивного)

FR – Face Recognition (розпізнавання обличчя)

LBP – Local Binary Pattern (локальні бінарні патерни)

Liveness Cues – ознаки життя

MSE – Mean Square Error (середньоквадратична похибка)

PA – Presentation Attack (підміна біометричних або візуальних даних)

PAD – Presentation Attack Detection (виявлення атак підміни біометрії)

PW – Pointwise Convolution (точкова згортка)

RNN – Recurencal Neural Network (рекурентна нейронна мережа)

Spoofing – Атака у якій зловмисник видає себе за когось іншого

SVM – Separable Vector Machine (метод опорних векторів)

UX – User Expirience (користувацький досвід)

ВСТУП

Комп’ютерна графіка, як міждисциплінарна галузь науки та техніки, традиційно поділяється на три основні напрямки, кожен з яких має свою унікальну методологію, яка часто взаємопов’язана з іншими вузько специфічними галузями по типу візуалізації, обробки зображень та розпізнавання образів [1].

Основне завдання розпізнавання образів – це отримання семантичного опису зображень об’єктів. Мета розпізнавання цих образів може бути різною, як виділення та сегментація окремих елементів на зображенні або аспектів зображень, так і класифікація зображення в цілому [2]. У нашому випадку зображення класифікується в цілому на класи «атака» та «справжній об’єкт». Слушно буде зауважити, що саме в галузі розпізнавання образів і зокрема в його найбільш чутливій частині – біометричній автентифікації, лежить предмет даного дослідження. Розпізнавання обличчя (надалі FR) стало ключовою технологією, однак її ефективність напряму залежить від можливості адаптації до зовнішніх втручань у роботу моделі.

Актуальність роботи полягає у необхідності забезпечення надійності та цілісності сучасних систем біометричної автентифікації обличчя в умовах зростаючої загрози атаки підміни обличчя. Попри високу точність таких моделей побудованих за допомогою згорткових нейронних мереж (CNN), зловмисники все одно знаходять прості засоби для обходу подібних систем та їх біометричних сенсорів [3,4]. Оскільки сучасні системи біометрії не здатні розрізнити вектор отриманий від живої особи та вектор отриманий від імітації людини, то побудова нової системи автентифікації або допоміжного модуля системи протидії спуфінгу просто необхідна для сучасних систем біометрії. Сучасні системи мають доволі високі вимоги до безпеки, особливо у фінансовому секторі, проте подібні системи стикаються з явними проблемами у своїй роботі та суперечливими вимогами: система вимагає виявити максимальну кількість атак, зберігаючи свою ефективність, та водночас система повинна бути комфортною

для користувачів через мінімізацію хибно позитивних спрацювань. Вирішення цього виклику є головним інженерним викликом даного дослідження.

Наукова задача дослідження полягає у розробці та експериментальному підтвердженні роботоспроможності комплексної, багатофакторної методології навчання, що зможе одночасно вирішити проблему дисбалансу класів та забезпечити робустність системи через диверсифікацію шуканих ознак. Це передбачає створення архітектури, яка здатна до ефективного поєднання зваженого навчання та асиметричні покарання з ансамблевою агрегацією для підвищення точності та можливості до узагальнення з подальшим калібруванням порогу рішення та покарання відповідно до жорстких вимог комфорту користувачів та безпеки.

1 ОГЛЯД ОСНОВНИХ МЕТОДІВ АНАЛІЗУ ЗОБРАЖЕНЬ ДЛЯ ПРОТИДІЇ СПУФІНГУ

Біометричні системи, особливо технології розпізнавання обличчя засновані на розпізнаванні за допомогою штучного інтелекту, постійно еволюціонують через більшу довіру зі сторони роботодавців та інвесторів, що призвело до їх широкого впровадження у сферах безпеки, фінансових транзакцій та контролю доступу [5]. Коли біометричні системи вже стали стандартом індустрії, особливо у мобільних фінансових застосунках, це було лише питанням часу коли ці самі системи будуть атаковані. Цими атаками можуть бути атаки презентації (надалі PA), відомих загальною назвою як спуфінг.

Актуальні дослідження визначають PA як атаку на біометричний сенсор завдяки передачі підробленого зразка існуючого користувача, що може бути як вирізкою фото, фото-маскою, відео користувача, тощо [6]. Сучасні PAD зосереджені на розрізненні біометричних даних, отриманих від живої особи, від даних отриманих від зловмисника. Методи PAD поділяються на апаратні (сенсори, інфрачервоні або 3D-камери) та програмні. Апаратні методи протидії є високоточними та неймовірно ефективні, проте ці підходи не є релевантними поза межами високотехнологічних або захищених державних об’єктів через високу ціну таких елементів та недоступність подібних пристроїв для середньостатистичного користувача, тому програмний підхід є набагато більш цікавим та розповсюдженим, бо цей підхід більш гнучкий та економічно вигідний, оскільки використовує інформацію отриману з 2D камер. Через обмеження звичайних камер існують різні методи обробки та аналізу зображень які зосереджені на виявлені артефактів зображення які не власні живій людині.

1.1 Аналіз текстур та артефактів зображення (LBP та CNN)

Аналіз текстур зображення є одним з найбільш ранніх підходів до систем протидії спуфінгу. Його ефективність базується на базовому і непорушному

фізичному принципі: «Жоден 2D об’єкт не може ідеально імітувати мікротекстуру живої шкіри». Натомість, найчастіша форма PA – це роздрукована фотографія, або відтворення відео на екрані монітора, що вносить специфічні артефакти, яких неможливо позбутись, та є їх невід’ємною ознакою таких атак.

На початкових етапах розвитку PAD домінували класичні методи зі статичними та фіксованими дескрипторами. Основним елементом цих систем були локальні бінарні шаблони, розроблені для ефективного кодування мікротекстур зображення шляхом порівняння інтенсивності центрального пікселя з його сусідами. LBP та його розширення, які додавали часову шкалу для аналізу відео довели свою здатність виявляти такі дефекти як зернистість або лінії аберації (рис.1.1). Однак, ефективність LBP була обмежена його наднизькою здатністю до узагальнення, оскільки ідентифікація атаки можна було обійти новим типом паперу, принтеру або екраном відтворення [7].

Рисунок 1.1 – Приклад сферичної та хроматичної аберації на зображенні

Цей метод надає всю свою увагу виявленню мікроскопічних дефектів які мають носії атаки, але не сама атака, що робить цей метод неймовірно вразливим до зміни методології атаки. Так класичний підхід використовує дескриптори LBP для кодування мікротекстури зображення до бінарного коду. Метод LBP побудований на гіпотезі, що шкіра людини має унікальну мікротекстуру, тоді як поверхня атаки буде мати передбачувану текстуру або дефекти (екран, папір, маска). Частина артефактів виникає через ефект пікселізації, ліній аберації або ліній Муару який виникає через накладання сітки пікселів екрана та сітку сенсора камери [8], інша ж частина артефактів виникає через зернистість паперу яка порушує текстуру шкіри.

LBP це попередник CNN для вирішення задач зі знаходження та аналізу цих мікротекстур та фіксування локальних змін інтенсивності. Система LBP працює наступним чином (1.1, 1.2):

Для центрального пікселя g_c і P сусідніх пікселівg_c на радіусі R, LBP обчислюється як сума порогових різниць, зважених степенями двійки.

$$LBP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c) * 2^p, \tag{1.1}$$

де функція $s(g_p - g_c)$ – бінарний поріг.

$$s(x) = \begin{cases} 1, if\ x \geq 0 \\ 0, if\ x < 0 \end{cases}. \tag{1.2}$$

Результатом цієї функції (1.1, 1.2) є гістограма, яка є статистичним описом локальної текстури зображення. У контексті FAS системи яку ми розробляємо ця гістограма демонструє аномалії у розподілі яскравості зображення, що є характерним для штучних поверхонь під час атак.

Якщо раніше всюди використовували статичний LBP-дескриптору, то на даний момент все більше використовують CNN моделі, які автоматично виявляють найбільш ефективні текстурні ознаки.

Для цього методу попередження атак ключовим елементом є операція згортки. Ця операція застосовує ядро K до вхідного зображення (або карти ознак) X для отримання вихідної карти ознак Y. Це дає виявляти локальні просторові закономірності у вхідному зображені (це можуть бути лінії, кути або текстурні дефекти). У цьому випадку вихідний елемент $Y_{i,j}$ обчислюється як (1.3):

$$Y_{i,j} = (X * K)_{i,j} = \sum_m \sum_n X_{i-m,j-n} K_{m,n} + b, \quad (1.3)$$

де K – фільтр, який CNN оптимізує для виявлення текстурних дефектів;

b – зміщення.

Цей процес (1.3) дозволяє вивчити низькорівневі ознаки текстур, а глибинні шари комбінують ці ознаки для виявленні більш складних патернів спуфінгу, що на відміну від LBP моделі дозволяє аналізувати не лише пікселізацію, зміну яскравості, а й відбиття світла та багато іншого одночасно, що допомагає системі самостійно виявляти ідеальні фільтри для виявлення нових специфічних ознак, що підвищує стійкість системи до нових різноманітних атак. Попри все назване вище, системи на основі локальних бінарних шаблонів все ще можуть бути використані як елемент протидії спуфінгу через їх спеціалізацію на локальній якості зображення, що дозволяє прогнозувати локальну зміну у якості вхідного зображення, що може знайти локальні специфічні шуми або різницю якості зображення у певних областях, що може бути сильною ознакою спуфінгу.

1.2 Аналіз глибини та фізіологічних ознак (Depth та Livenes Cues)

Метод аналізу глибини, який також розглядався як елемент методології для побудови FAS системи, базується на неможливості або надмірній складності імітування 3D моделі обличчя, бо обличчя є складним та нелінійним об’єктом, у той час як більшість атак – це плоска проекція обличчя, тобто фото або відео. Ці методи дозволяють прогнозувати глибину зображення на вхідному 2D

зображенні. При цих умовах глибина обличчя буде градієнтна та нерівномірна, у той час як глибина зображення атаки константна, тобто нульова або дуже мала [9]. Найбільш розповсюджений та стійкий програмний метод розробляється на базисі регресії карти глибини (1.4). CNN буде навчатися відновлювати інформацію про глибину з базового 2D зображення.

$$D = F_{depth}(I), \tag{1.4}$$

де D – карта глибини;

F_{depth} – мережа яка навчається відображенню глибини зображення;

I – вхідне зображення.

За узагальненою формулою (1.4), якщо вхідне зображення буде плоским, то модель спрогнозує матрицю глибини близькою до нуля, бо плоска атака не має природних глибинних градієнтів, у той час як для справжнього обличчя глибина зображення буде відповідати тривимірній структурі у зображенні. Проте для цього навчання моделі необхідна пара з вхідних даних – зображення та мапа глибини, оскільки атаки мають майже нульову мапу глибини, а моделювання істинних мап глибини для моделі є важким та трудоємним процесом.

Попри очевидні мінуси, метод прогнозування карти глибини є ідеальним прикладом використання регресії для вирішення задач класифікації зображень, часто цю модель реалізовують як мультитаск-навчання [10,11]. У цьому випадку модель буде оптимізуватися за регресійною втратою глибини, що буде зваженою сумою втрат системи знаходження підміни особистості та втрат мапи глибини, що може бути потужним регуляризатором системи, проте як зазначено вище, знаходження бази даних для навчання подібної системи є задачею складнішою за саме навчання.

У цей час аналіз Liveness Cues використовує, замість карти глибини, великий перелік біологічних ознак, що можуть підтвердити або спростувати живість об’єкту. Ці системи зазвичай використовують RNN для аналізу послідовних кадрів для аналізу мікрорухів, пов’язаних з кровообігом, диханням,

кліпанням, тощо. Це ефективно проти відеоатак, у тому числі тому що ці відео зазвичай зациклені. Проте для аналізу зображень частіше досліджують артефакти світла та відбиття, у тому числі відношення візуальних аспектів обличчя, у тому числі очей [12].

1.3 Ансамблеве та калібрувальне навчання

Ансамблеве навчання є однією зі стратегій у сфері машинного та глибокого навчання. Суть цього методу полягає у колективному використанні множини індивідуальних класифікаторів або базових моделей для отримання фінального прогнозу, що є набагато більш надійним і більш точним способом класифікації, ніж прогноз окремої моделі. Ця стратегія симулює принципи дипломатичного підходу у прийнятті рішень, де слабкі сторони однієї з моделей, тобто члена ансамблю, компенсуються іншим членом ансамблю. Ці принципи побудови системи дозволяють компенсувати слабкі сторони, що призводить до зниження загальної кількості помилок та їх дисперсії [13,14].

У більшості випадках при такому навчанні використовується найпростіше та найнадійніше усереднення ймовірностей, отриманих від усіх елементів ансамблю, тобто всі незалежні моделі цього ансамблю приносять однаковий вклад до передбачення. У цьому випадку фінальна ймовірність P_{final} того, що певний вхід I буде атакою, обчислюється як середнє арифметичне ймовірностей отриманих від незалежних систем (1.5).

$$P_{final}(I) = \frac{1}{K}\sum_{k=1}^{K} P_k(I), \tag{1.5}$$

де I – вхід до системи;

K – окрема незалежна система для передбачення;

P – передбачення.

Не можу не зауважити, що у випадку використання ансамблевого передбачення необхідно окремо розглядати методологію навчання кожної моделі. Різниця методології може бути у архітектурі моделі яка навчається, це може бути інший набір даних, або фокусування на певних аспектах наявного датасету. Ансамблеве навчання та прогнозування не буде успішним у випадку якщо помилки окремих моделей є корельованими між собою. Якщо перша модель помиляється у текстурі, то для другої моделі важким прогнозуванням має бути освітлення. Саме такі помилки моделей забезпечать як найбільшу диверсифікацію вхідних даних для забезпечення найбільш вдалого передбачення для більшої точності моделі. У випадку, якщо кількість моделей невелика, то можливо гарним рішенням буде встановити ваги для окремих елементів ансамблю, так якщо моделі всього дві, то можливе встановлення вагів 0,7 та 0,3 для моделей відповідно до їх важливості для дослідження або точності моделі, проте сума вагів обов’язково повинна дорівнювати одиниці.

Основна мета цього методу – підвищення робустності, точності та можливості до узагальнення. Кожен окремий алгоритм навчається на обмеженому наборі даних та не може ідеально охопити всі наявні у зображенні ознаки. Ансамблі подібних моделей у свою ж чергу нівелюють «сліпі зони» кожної окремої моделі, використовуючи статистичні переваги агрегації. Навіть якщо базові моделі є нестабільними, або демонструють високу дисперсію помилок, їх усереднення призводить до більш стабільного, менш чутливого до змін вхідних даних фінального рішення, що критично підвищує можливості до узагальнення на нових та наявних даних моделі [15].

Як вже було зазначено вище, успіх ансамблю повністю залежить від диверсифікації використаних моделей, так ансамблеве навчання не буде успішним у випадку якщо помилки цих окремих моделей корельовані між собою [16]. Якщо перша модель помиляється при зміні освітлення, то інша модель має бути спроможна адаптуватися до таких змін або помилятися у іншому аспекти при аналізі зображення. Саме такі некорельовані помилки забезпечать

максимальну точність ансамблю. Для досягнення диверсифікації можна використовувати наступні методи:

– навчання даних на різних підмножинах даних, або використовувати різне представлення ознак;

– використовувати різні архітектури класифікаторів окремих систем (наприклад ансамбль з CNN, ResNet, SVM);

– навчання моделей з різними гіперпараметрами, такими як швидкість навчання, ваги або змінені функції втрат.

Але перечислене вище це лише базові способи для забезпечення роботоспроможності ансамблю. У концепції роботи комп'ютерного зору для PAD системи виникають особливі виклики зі своїми специфічними рішеннями. Так абсолютна більшість доступних баз даних має значний дисбаланс класів, де реальний клас значно перевищує клас фейку, у цьому випадку навчання має бути спеціалізовано під ці обмеження вхідних даних, що можна досягти за допомогою зваженої функції втрат або зваженого завантажувача вхідних даних. Ці методи гарантують збалансування завантажених даних та асиметричний штраф за помилки, збільшуючи важливість помилок певного характеру (FP або FN).

1.4 Комфорт користувачів та безпека

У сучасних системах високо ціниться комфорт користувачів через надвисоку конкуренцію у сучасній коммерційній сфері. Комфорт користувачів, або User Experience (UX), є критично важливим, хоча й часто недооцінюваним, параметром у розробці систем біометричної аутентифікації, особливо в контексті протидії спуфінгу (PAD). Сучасні системи безпеки не можуть бути ефективними, якщо вони не є зручними, користувачі просто перейдуть до конкурентів. Високий рівень безпеки, досягнутий ціною значного дискомфорту, неминуче призводить до саботажу системи з боку кінцевих користувачів [17]. У випадку

систем розпізнавання обличчя, комфорт прямо залежить від здатності PAD-модуля мінімізувати хибнопозитивні спрацювання (False Positives, FP).

Комфорт користувача в контексті біометрії визначається трьома ключовими факторами: швидкість, прозорість та надійність.

– швидкість та час очікування: Це стосується того, як швидко вся система працює, від моменту, коли людина підходить до камери, до моменту, коли двері відчиняються або транзакція завершується. Будь-яке зайве зволікання, спричинене надто складним і тривалим аналізом того, чи є обличчя реальним, чи це підробка, негативно впливає на сприйняття. Люди очікують миттєвого результату, і будь-яка затримка, що перевищує одну чи дві секунди, починає викликати роздратування. У сучасному цифровому світі швидкість доступу є прямим показником зручності;

– непомітність процесу: Цей фактор описує, наскільки ненав'язливим є процес підтвердження особи. Найбільш зручними є ті системи, які працюють пасивно – це означає, що вони автоматично сканують обличчя і приймають рішення без будь-яких вимог до користувача. Якщо системі потрібно, щоб людина робила спеціальні рухи, як-от кліпала, посміхалася чи повертала голову, це робить процес обтяжливим і незручним. Комфорт максимальний тоді, коли технологія працює у фоновому режимі;

– надійність: Це внутрішня впевненість користувача в тому, що система завжди коректно його розпізнає та дозволить увійти. Ця впевненість прямо залежить від кількості помилкових блокувань. Коли справжнього користувача помилково приймають за зловмисника і блокують йому доступ, це швидко руйнує довіру до технології. Часті помилки такого роду призводять до того, що люди починають уникати використання системи, вважаючи її ненадійною та несправедливою.

З економічної точки зору, низький комфорт призводить до зменшення рівня впровадження технології, що є прямими фінансовими втратами для розробника або інтегратора. У сфері мобільного банкінгу або електронної комерції, високий дискомфорт може змусити клієнта відмовитися від транзакції.

У сфері протидії спуфінгу, комфорт користувача інженерно кваліфікується через метрику Хибнопозитивних Спрацювань «False Positive Rate» (1.6).

$$FPR = \frac{False\ Positives\ (FP)}{True\ Negatives\ (TN) + FalsePositives(FP)}. \quad (1.6)$$

False Positive (FP) – це ситуація, коли справжній, легітимний користувач (True Negative) помилково класифікується системою як атака, тобто особа ідентифікована як спуф. З точки зору користувача, це неправомірне блокування доступу. Це викликає фрустрацію, недовіру до технології та, у кінцевому підсумку, відмову від її використання. У критичних системах (наприклад, аеропортах чи медичних установах) це може мати значні операційні наслідки.

Для більшості комерційних застосунків у сфері біометрії, рівень FPR має бути екстремально низьким, часто встановлюється на рівні 1% або менше. У високочутливих фінансових або державних системах цей показник може вимагати рівня 0,1% або навіть 0,01%. Якщо система блокує одного зі ста легітимних користувачів, її комфорт вважається неприйнятним. Таким чином, інженерна задача полягає в тому, щоб розробити систему, яка може забезпечити високу чутливість або ж «Recall» системи до атак, не жертвуючи при цьому комфортом користувача. Для задоволення вимог низького FPR, застосовуються спеціальні інженерні та архітектурні рішення, які буде приведено далі.

Калібрування порогу рішення це найбільш прямий метод контролю комфорту. Далі буде продемонстровано використання цього методу, налаштування порогу рішення дозволяє безпосередньо керувати співвідношенням FP/FN. Підвищення рівня каліброваного порогу (наприклад, з 0,5 до 0,8) робить систему консервативною, змушуючи її вимагати вищої впевненості для прогнозу «Spoof», що прямо знижує FPR.

Ансамблі є архітектурним рішенням для підвищення надійності, що також сприяє комфорту. Об’єднання прогнозів від кількох диверсифікованих моделей може знижувати дисперсію помилок, що призводить до більш стабільних та надійних прогнозів. Це знижує ймовірність випадкового блокування легітимного

користувача, що покращує UX. Також можливе використання пасивних методів аналізу (наприклад, аналіз мікротекстур, прогнозування карти глибини), що є більш зручним, ніж активні методи (вимога моргнути або рухати головою). У цьому випадку, PAD-модуль працює у фоновому режимі, роблячи процес ав тентифікації прозорим і швидким.

Комфорт користувачів та безпека є взаємозалежними, але протилежними цілями. Зниження FPR (підвищення комфорту) часто призводить до збільшення FNR (False Negative Rate, пропуск атаки), що знижує безпеку, і навпаки. Завдання, яке необхідно буде вирішити, полягає у знаходженні оптимального балансу, де FPR відповідає суворим вимогам UX, а TPR (Recall) залишається максимально високим. Це досягається шляхом інтеграції передових методів – від зваженого навчання до багатомодельного ансамблю та точного калібрування порогу, що дозволить повноцінно оглянути перспективи проекту та дослідити отримані результати.

1.5 Проблеми оптимізації в сучасних FAS-системах

Ефективність систем протидії спуфінгу, незважаючи на значні успіхи, досягнуті завдяки глибокому навчанню, залишається обтяженою низкою фундаментальних, взаємопов’язаних викликів, які лежать на перетині статистики, комп’ютерного зору та вимог реальної експлуатації. Ці проблеми, які ми розглядаємо як основні напрямки оптимізації, які можуть бути згруповані навколо статистичного дисбалансу, архітектурного узагальнення та експлуатаційного комфорту користувачів.

Найпершою і найбільш критичною перешкодою для моделі є проблема дисбалансу класів, яка є статистичною аксіомою в галузі біометричної безпеки. Сутність цього виклику полягає у критичній нерівності кількості зразків у більшості наявних навчальних наборів: кількість зображень справжніх облич (Real, негативний клас) завжди значно перевищує кількість зразків атак спуфінгу

(Spoof, позитивний клас). Алгоритми глибокого навчання прагнуть мінімізувати загальну помилку. Коли модель бачить клас «Real» у переважній більшості випадків (наприклад, 95%), вона може досягти високої загальної точності, просто прогнозуючи «Real» щоразу, ігноруючи клас меншості, як ми це побачимо далі. Наслідком цієї упередженості є катастрофічно висока кількість False Negatives, коли атака, тобто Spoof, помилково класифікується як справжнє обличчя. З точки зору безпеки, це означає, що система пропускає абсолютну більшість атак, що є неприпустимим, попри високу загальну метрику точності моделі, яка виявляється оманливою. Відповідно, метрика «Recall» (True Positive Rate, TPR), яка показує здатність системи виявляти атаки, падає до неприйнятно низьких значень, оскільки модель не навчається ознакам Spoof-класу достатньо часто через абсолютну контрпродуктивність спроб виявити клас несправжніх облич (Spoof), який призведе до критичного спаду метрики точності через очевидне передбачення хоча б деякої кількості справжніх елементів як несправжніх. Для подолання проблеми дисбалансу сучасна FAS-інженерія використовує двокомпонентну стратегію, яка впливає як на дані, так і на процес навчання. Перший компонент – це балансування на рівні даних (Oversampling/Undersampling) через «Weighted Random Sampler». Цей метод призначає кожному зразку вагу, обернену до його частоти. Вага міноритарного класу (Spoof) стає значно вищою, що гарантує, що у кожному з пакетів даних обидва класи представлені приблизно рівномірно. Другий компонент – балансування на рівні функції втрат, тобто імплементації зваженої функції втрат. Функція втрат ініціалізується з параметром ваги помилки > 1.0 (наприклад, 5.0). Це створює асиметричний штраф, який значно збільшує покарання за помилки, пов’язані з міноритарним класом (FN), що є найсильнішою формою мотивації для моделі підвищувати чутливість до атак. Ці методи є необхідною умовою для того, щоб модель перейшла від прогнозування на основі частоти до прогнозування на основі фізичної сутності об’єкта [18-20].

Другим фундаментальним викликом є проблема узагальнення та стійкість до Unknown Attacks (атак, які модель ніколи не бачила під час навчання).

Модель, навчена на одному датасеті з фіксованими умовами (одна камера, одне освітлення), часто демонструє різке падіння точності при тестуванні на зовнішньому датасеті, (рис.1.2).

Рисунок 1.2 – Різниця між 2 фотографіями 1 об’єкту за різних обставин

Це відбувається тому, що модель навчається не справжнім фізичним ознакам спуфінгу (наприклад, текстура паперу), а конкретним артефактам, пов’язаним з умовами тренувального набору (специфічний шум сенсора чи фіксована кольорова гама). Для підвищення здатності до узагальнення застосовуються архітектурні стратегії, які забезпечують диверсифікацію ознак. Найефективнішим є ансамблеве навчання, що об’єднує моделі, навчені на некорельованих ознаках. Наприклад, об'єднання фокусу на локальній текстурі з фокусом на глобальному контексті та освітленні гарантує, що атака, яка обійшла один аналізатор, буде виявлена іншим. Цей підхід створює колективну робустність [21]. Додатковою технікою є аугментація даних, яка штучно розширює простір ознак під час навчання, роблячи модель менш чутливою до варіацій умов зйомки.

Третій, але не менш важливий виклик, полягає у компромісі між безпекою та комфортом користувачів. У реальній експлуатації, ефективність системи визначається комфортом користувачів, який безпосередньо вимірюється кількістю хибно позитивних спрацювань, оскільки підвищення чутливості до атак часто призводить до збільшення FP. Інженерне рішення полягає у калібруванні порогу рішення моделі. Замість стандартного порогу у 0.5 коефіцієнту впевненості, поріг підвищується. Це робить систему консервативною, вимагаючи вищої впевненості для прогнозу «Spoof», що прямо знижує FPR (підвищує комфорт). Аналіз цього компромісу візуалізується на ROC-кривій, де метою є знайти оптимальну робочу точку, що задовольняє вимозі низького FPR при максимальному збереженні TPR.

Таким чином, успіх сучасної FAS-системи, в контексті даної роботи, визначається інтеграцією цього комплексного підходу, який є необхідним для створення дійсно робустної та надійної системи.

1.6 Постановка задачі дослідження

Сучасний розвиток технологій розпізнавання обличчя зробив їх ключовим елементом систем безпеки та автентифікації. Однак, як було встановлено в попередньому огляді, критичною вразливістю залишається проблема спуфінгу (Presentation Attack), що вимагає інтеграції надійного модуля протидії PAD. Актуальність даного дослідження зумовлена необхідністю подолання двох основних викликів, які заважають впровадженню FAS-систем у реальні експлуатаційні умови: по-перше, дисбаланс класів у навчальних наборах даних, що призводить до низької ефективності виявлення атак (Recall), і по-друге, необхідність забезпечення комфорту користувачів, що вимагає мінімізації хибно позитивних спрацювань.

Метою даної кваліфікаційної магістерської роботи є розробка та дослідження комплексної, робустної архітектури протидії спуфінгу, заснованої

на ансамблевому навчанні, яка забезпечує високу ефективність виявлення атак в умовах дисбалансу класів при дотриманні жорстких вимог до зручності користувача (низький FPR).

Для досягнення поставленої мети було визначено наступні завдання, які послідовно вирішуються у ході роботи:

– розробка та впровадження механізму зваженого навчання, що включає використання Weighted Random Sampler для балансування навчальних батчів та застосування зваженої функції втрат для асиметричного покарання за пропущені атаки (False Negatives) для боротьби з упередженістю моделі до мажоритарного класу;

– побудова незалежних моделей для навчання, у рамках якої буде створено три незалежні моделі (BBOX Cropped, Full Image, ALT Image), навчених на різних стратегіях вхідних даних, з метою визначення їхніх індивідуальних сильних та слабких сторін щодо текстурних та контекстних ознак;

– розробка та оцінка ансамблевого класифікатора, що передбачає створення та оцінку ансамблевого класифікатора на основі усереднення прогнозів від цих трьох моделей, що є архітектурним рішенням для підвищення узагальнення системи;

– калібрування порогу рішення ансамблевої системи, спрямоване на знаходження рішення між комфортом потенційних користувачів та безпекою системи.

Теоретична цінність роботи полягає в обґрунтуванні комплексної моделі, яка може об’єднати статистичні інструменти боротьби з дисбалансом даних та ансамблевої агрегації для підвищення точності моделі, а практична цінність – у наданні набору інженерних рішень, готових до впровадження у реальні FAS-системи з забезпеченням достатнього комфорту користувачів та вимог до безпеки.

2 МАТЕМАТИЧНІ МОДЕЛІ ФІЛЬТРАЦІЇ ЗОБРАЖЕНЬ

Цей розділ буде присвячений формалізації та детальному математичному опису використаних методів для зваженої бінарної класифікації з асиметричним штрафом за помилки та необхідний для подальшого розуміння роботи та забезпечення надійності та ефективності системи виявлення атак підміни обличчя. У цьому розділі буде розглянуто не лише архітектурні рішення для ієрархічного вилучення ознак з подальшим прогнозуванням, а й будуть розглянуті способи подолання проблем асиметрії вхідних даних та ефективності протидії класу меншості.

2.1 Модель згорткової нейронної мережі

Виходячи з написаного у першому розділі, оптимальним рішенням для побудови моделі рішення цієї задачі було обрано згорткову нейронну мережу, яка слугує основним апаратом для вилучення ієрархічних ознак з вхідних зображень обличчя, з трансформацією сирих пікселів у високорівнений вектор ознак. Архітектура згорткової нейронної мереж (CNN) оптимізована для роботи з сітчастою топологією зображень, що забезпечить як ефективність, так і надійність побудованої моделі зі здатністю до трансляційної інваріантності – це властивість мережі, яка дозволяє мережі розпізнати об’єкт незалежно від його положення на кадрі зображення. Згорткова нейронна мережа, як зрозуміло з назви, працює за рахунок операції згортки. Згорткові шари є фундаментальними будівельними блоками CNN-моделі, що виявляють локальні просторові закономірності на зображенні, такі як краї або кути форм, текстури об’єктів, так ці шари можуть виявити загальні закономірності мікроартефактів при спробі спуфінгу (як зазначено вище, зернистість, лінії аберації, тощо).

Сама же операція згортки є фактичним дискретним згортанням вхідного тензора з тензором вагів (фільтром або ж ядром). Кожен фільтр сам по собі є детектором певної ознаки, наприклад фільтр може детектувати вертикальні або

горизонтальні краї зображень. Фільтр згортається (або послідовно перевіряє) вздовж всієї довжини вхідного зображення. У такому випадку, вихідний елемент карти ознак $Y_{i,j}$ отриманої після згортки буде обчислений наступним чином (2.1)

$$Y_{i,j} = (X * K)_{i,j} = \sum_{m=1}^{M} \sum_{n=1}^{N} X_{i-m,j-n} K_{m,n} + b, \quad (2.1)$$

де X – вхідний тензор, що може бути зображенням або результатом попереднього шару (розміром $H_{in} \times W_{in} \times C_{in}$);

K – фільтр з розмірами $M \times N \times C_{in}$, де M та N зазвичай малі. Ваги цього фільтру змінюються бо мережа навчається їх оптимізації;

Y – вихідна карта ознак;

b – зміщення, яке додається до кожного елемента вихідної мапи ознак, надаючи додатковий рівень свободи для зміщення активації.

При використанні згорткової нейронної мережі у проектах необхідно розуміти, що ефективність згорткових нейронних мереж полягає у спільному використанні ваг, тобто фільтр K буде застосований до всіх просторових позицій вхідного тензору, що драматично знижує кількість параметрів. Це значно збільшує швидкість обчислень та забезпечує ту саму інваріантність, що дозволяє мережі розпізнати ознаку, незалежно від її положення на зображенні.

Коли ми маємо вхідний сигнал та операції перетворення, то нам необхідна певна функція яка буде визначати вихідний сигнал для зменшення обчислювальної ваги та підготовки даних для використання різними нейронами нейронної мережі. Такими функціями у згорткових нейронних мережах виступають функції активації. Ці функції, як вже зауважено, визначають вихідний сигнал, вносять нелінійність до навчання для вирішення складних та комплексних задач та перетворюють вхідні дані на певний ряд даних з окресленими межами у той же час визначаючи чи буде нейрон взагалі активованим, залежно від отриманої інформації. Більшість функцій активацій виконують однакову роботу і фактично визначають нелінійні взаємозв’язки між даними і особливо вирізняються між собою типами, так функція сигмоїди (2.2),

так же відома логістична функція – одна з перших функцій активації, яку використовували в багатошарових персептронах. Фактичною особливістю функції сигмоїди є стиснення вхідних даних у жорстко обмежений діапазон, що центрується близько нуля.

$$f(x) = \sigma(x) = \frac{1}{1+e^{-x}}. \quad (2.2)$$

Вихідний діапазон цієї функції (2.2) лежить строго між 0 та 1, що робить її гарним вибором при використанні у фінальних вихідних шарах для бінарної класифікації, де вихідний сигнал може бути відразу представлений як ймовірність належності до того чи іншого класу, однак при використанні функції сигмоїди у рамках згорткової нейронної мережі зі зваженим порогом є не зовсім таким гарним через проблеми з виходом та градієнтом. Вихід який центрований навколо нуля може зменшити швидкість навчання коли для підтвердження позитивного спрацювання моделі необхідно досягти певного порогу, але більш критичною є проблема зникнення градієнта, тобто у випадках коли абсолютне значення входу стає занадто великим у позитивному чи негативному полі, то функція перенасичується і градієнт стрімко прямує до нуля, що призводить до поширення помилки ваг.

Іншим вибором для функції активації може бути функція гіперболічного тангенса або «Tanh» (2.3) яка фактично є еволюційним продовженням функції сигмоїди, спрямована на усунення головних проблем сигмоїди, проте не вирішуючи її повністю.

$$f(x) = \tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}. \quad (2.3)$$

Вихідний діапазон цієї функції активації (2.3) лежить у рамках від -1 до 1 та основною перевагою цієї функції є певний ефект стабілізації градієнту, що може прискорити збіжність моделі через центрованість до нуля та більш

рівномірні ваги поміж нейронів у сусідніх шарах. Проте ця функція активації страждає ж від тієї самої проблеми зникнення градієнта при надмірних значеннях. Саме це стало критичною проблемою для імплементації у моделях згорткових нейронних мереж на користь наступної функції, що зберігає постійний градієнт позитивних значень.

У роботі використано прості шари активації ReLU (вбудовані у інфраструктуру MobileNetV2), які створюють нелінійність моделі, що допомагає моделі вивчати взаємозв'язки між ознаками, що критично важливо для подібних задач, і особливо важливо при виявленні спуфінгу. У моделі після операції згортки, функція активації застосовується до кожного елемента вихідної карти ознак (2.4) та приводить до нуля всі негативні значення і залишає лише позитивні значення незмінними.

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

Функція ReLU надзвичайно проста для обчислення (бо це лише операція порівняння) і у той же час вона прискорює навчання забезпеченням збереження градієнту, але не слід забувати і про шари об'єднання (Pooling), що розміщується між послідовними згортковими шарами для зменшення розміру карти ознак після згортки. Це дозволяє згортковій нейронній мережі знову скоротити кількість параметрів та обчислювальні вимоги, у той же час це підвищує стійкість моделі до спотворень та зсувів у вхідному зображенні.

Для виконання роботи було обрано архітектуру MobileNetV2 через її обчислювальну ефективність, що дозволить значно прискорити дослідження та зменшити навантаження на систему в цілому, при цьому ця система доволі швидко розгортається та використовує нововведення у системах згорткових нейронних мереж, а саме глибинно-сепарабельну згортку (Depthwise separable Convolution). Традиційні системи CNN одночасно фільтрують ознаки та комбінують канали, проте MobileNetV2 декомпозує ці елементи на два окремі

легші кроки. У випадку використання цієї архітектури ми окремо маємо операції глибинної згортки та точкової згортки.

У вище зазначеній архітектурі глибинна згортка – це перший етап згортки, який відповідає за просторову фільтрацію, де кожен окремий фільтр застосовується до кожного вхідного каналу незалежно одне від одного. Якщо ми має звичайне зображення, то ми зазвичай маємо 3 вхідні канали (RGB), а після обробки фільтром ми отримаємо 3 вихідні карти ознак, що дозволяє нейронній мережі виявити локальні патерни, тобто кути, лінії або текстури у межах кожного окремого каналу, але не створює нових комбінацій між ними, що робить інформацію фрагментованою на кожному фільтрі, що може здаватись мінусом при відсутності тих самих зв’язків між каналами.

Другий етап – точкова згортка яка вирішує проблему комбінування цих каналів. По суті, це звичайна згортка але з фільтром розміру 1×1. Цей фільтр досліджує глибину всіх 3 каналів отриманих на попередньому кроці і обчислює їх зважену суму, створюючи нову карту ознак. Цей етап не дивиться на просторових сусідів, а змішує інформацію між каналами кожного окремого пікселя. Разом ці два етапи – глибинна фільтрація та точкове змішування отримують функціонально такий же результат, що й звичайна згортка, тобто отримують складні та комплексні ознаки, але з драматично меншою кількістю параметрів та обчислень, що оптимізує систему. Розділення процесу дозволило досягти колосальної різниці параметрів та обчислень, що створює ймовірність запуску й роботи подібної моделі як і на машині в 8-9 разів потужнішої, яка використовує звичайну згорткову нейронну мережу. Така оптимізація дозволить запускати модель навіть на смартфоні або вбудованих мережах інтернету речей без великого навантаження на сервер, де ресурси батареї, якщо така є, звісно, та процесора суворо обмежені.

Але додатковим бонусом до оптимізації MobileNetV2 не можна не зауважити, що більшість систем CNN використовують один з 2 типів шарів об’єднання для зниження обчислювального навантаження у наступних шарах та підвищенні інваріатності моделі. Фактично, це операція у якій певну область замінюють репрезентативним значенням для зменшення розмірності даних. Перший тип – Max Pooling (рис. 2.1) обирає максимальне значення локального блоку, що допомагає зберігати найбільш виражені ознаки та підвищує робустність системи до зсувів у зображенні, а другий тип – Average Pooling, при якому замість вибору найбільшого значення локального блоку, система репрезентує середнє арифметичного цього блоку, що допомагає моделі з узагальненням ознак для отримання представлення всієї області та часто зустічається у останніх шарах перед подачею даних до класифікатору для отримання глобального контексту. Проте, подібне зменшення розмірності даних може призводити до незворотньої втрати інформації, бо, зазвичай, спуфінг це комплекс незначних ознак, а не один показчик, тому архітектура MobileNetV2 глобально переосмислює подібний крок. У цій моделі імплементована система лінійних вузьких шийок (bottlenecks), що прямо стосується проблеми втрати інформації при застосуванні нелінійних функцій активації у стисненні низьковимірних просторів. При використанні звичайних способів, наприклад функції активації ReLU (2.4),ми безповоротньо втрачаємо від’ємні значення, що як зазначено вище, може значно скорити обчислювальне навантаження, проте ReLU ефективно працює з високорозмірними просторами, навіть якщо він руйнує від’ємну частину даних. У цій втраченій інформації може бути цінна інформація низьковимірних ознак, наприклад мікротекстур або тонкі градієнти кольорів за якими система може передбачити атаку та захистити систему від спуфінгу [22]. Так у системі MobileNetV2 використовується лінійна функція активації на виході шарів ($f(x) = x$) стиснення.

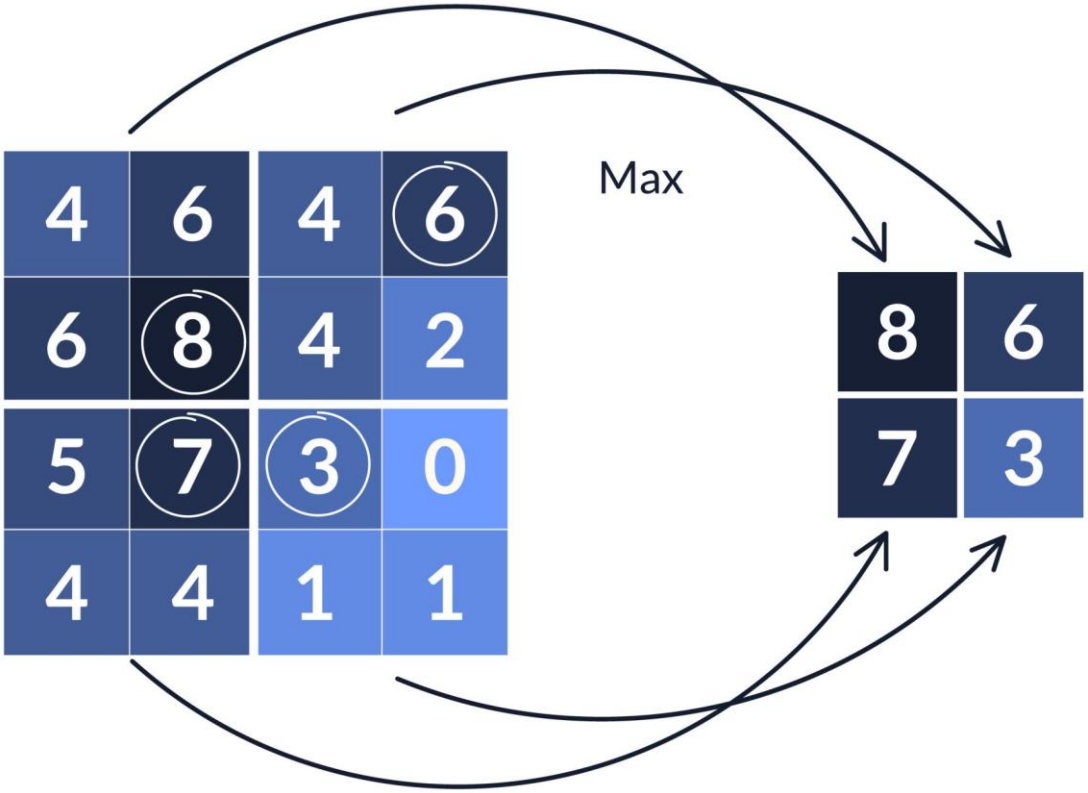

Рисунок 2.1 – Принцип роботи об’єднання Max Pooling

Це гарантує, що інформація буде збережена без незворотних перетворень, що допомагає нам у вилученні ознак, а нелінійні активатори використовуються лише у високорозмірних просторах, де втрата інформації не є такою важливою [23].

2.2 Порівняння ефективності MobileNetV2 та CNN

У роботі використовується саме архітектура MobileNetV2, але для обгрунтування існує необхідність розглянути чому ця архітектура більш ефективна, а саме перевірити гіпотезу про радикальне зменшення операцій множення-додавання в порівнянні з класичною моделлю.

Припустимо, що розміри карт ознак однакові, у такому випадку ми маємо наступні дані:

– розмір вхідної карти ознак $H \times W \times C_{in}$, де H – висота зображення, W – ширина зображення, а C_{in} – кількість вхідних каналів;

– розмір вихідної карти ознак: $H \times W \times C_{out}$, де навідміну від попередньої вхідної карти C_{out} – кількість вихідних каналів;

– розмір ядра: $K \times K$, зазвичай розмір ядра сягає 3×3.

Одночасно, виходячи з цього обчислена вартість стандартної моделі буде виглядати як (2.5):

$$Cost(Conv) = H \times W \times K^2 \times C_{in} \times C_{out}. \quad (2.5)$$

При розрахункові загальної кількості обчислювальних операцій (MAdds), які будуть необхідні для обробки одного стандартного згорткового шару необхідно буде брати до уваги наступні фактори: Фільтр, який має просторові ознаки які ми приймемо за стандартні, що дає $K \times K = 3 \times 3 = 9$, проте фільтр повинен мати ту ж саму глибину, що й вхідний тензор, тобто повний розмір фільтра буде становити $K \times K \times C_{in}$. Це означає, що для генерації одного вихідного пікселя для одного вихідного каналу ми повинні отримати скалярний добуток фільтра та ділянки вхідного зображення, що вимагає $K^2 \times C_{in}$ операцій множення-додавання.

Проте не слід забувати, що у реальній задачі ми обчислюємо не просто один піксель, а карту ознак або вхідне зображення (у цьому випадку глибина тензору буде дорівнювати 3 у RGB зображення), що означає, що для повного обчислення вхідного тензору нам необхідно повторити попередню операцію $H \times W$ разів, тобто на добуток висоти та ширини вихідної карти ознак, що робить нашу вартість обчислень для 2D карти однак до $(H \times W) \times (K^2 \times C_{in})$, але це лише для однієї карти ознак, а ми зазвичай отримуємо певну C_{out} кількість різних ознак з власним фільтром, тобто ми повинні повторити попередню операції кількість разів рівну кількості необхідних ознак, іншими словами наша

загальна вартість – це скалярний добуток вартості обчислення 1 пікселя, кількості пікселів карти або зображення та кількості карт або фільтрів.

У цей же час вартість обчислень для глибинної сепарабельної згортки буде виглядати як (2.6):

$$Cost(DW) = H \times W \times K^2 \times C_{in}. \quad (2.6)$$

Як ми можемо помітити, у цій формулі (2.6) відсутній показник C_{out}, це через те що, вище описується першие етап глибинно-сепарабельної згорти, де відбувається лише просторова фільтрація без змішування каналів, проте слід зауважити, що на відміну від стандартного способу проведення цієї операції, у даному випадку ми маємо пласкі 2D фільтри, тобто C_{in} буде рівним 1. Тут ми вже можемо побачити спрощення стандартної формули, бо для обчислення одного пікселя на 1 каналі вартість обчислення буде сягати K^2, а вартість 1 карти сягає $(H \times W) \times K^2$. При цьому доречним буде зауважити, що кожен канал обчислюється окремо одне від одного, тому надана вище формула буде обчислена C_{in} разів. Цей етап не створює нових зв'язків.

Другий етап цієї згортки (2.7) виконує лише комбінування каналів, не дивлячись на своїх просторових сусідів.

$$Cost(PW) = H \times W \times C_{in} \times C_{out}. \quad (2.7)$$

Напевно, ви же помітили, що глибинно-сепарабельна згортка дуже схожа на звичайну згортку але розбиту на 2 етапи, де на першому етапі вихідні канали та глибина дорівнюють 1 та є константними, то на другому етапі фіксується розмір ядра, тобто вартість обчисленні одного пікселя буде сягати 1 через те, що квадрат константи все одно буде одиницею, це по суті зважена сума всіх вхідних каналів у одному конкретному пікселі. У такому випадку вартість 1 карти ознак $(H \times W \times C_{in})$, та цей процес скалярного добутку розміру карти або зображення

та кількості вхідних фільтрів буде повторено C_{out} разів для отримання C_{out} вихідних каналів.

Для обчислення цільної вартості глибинно сепарабельної згортки нам потрібно отримати суму 2 етапів згортки (2.8), тобто суму глибинної та точкової згортки.

$$Cost(DSC) = Cost(DW) + Cost(PW). \quad (2.8)$$

Якщо ж говорити про загальну вартість глибинно сепарабельної нейронної мережі, то ми можемо для простоти вигляду формули (2.9) ми можемо виділити спільний множник ($H \times W \times C_{in}$), це дає нам формулу, згідно якої вартість цієї мережі – це добуток вхідних пікселів та вхідних каналів на суму розміру просторового ядра.

$$Cost(DSC) = H \times W \times C_{in} \times (K^2 + C_{out}) \quad (2.9)$$

Наша кульмінаційна формула (2.10), яка допоможе нам довести математичну зверхність та ефективність моделі MobileNetV2 (DSC) над звичайними моделями згорткових нейронних мереж (Conv) та розрахувати їх обчислювальну вартість за умов однакового розміру зображення та вхідних і вихідних каналів.

Це буде відносний показник ефективності архітектури в порівнянні зі звичайною структурою згорткових мереж, там наш чисельник – це вартість розрахунку DSC моделі (2.8), а знаменник – вартість звичайної моделі (2.5) відповідно. В око відразу кидаються спільні показники $H \times W \times C_{in}$які можна скоротити.

$$Ratio = \frac{Cost(DSC)}{Cost(Conv)} = \frac{H \times W \times C_{in} \times (K^2 + C_{out})}{H \times W \times K^2 \times C_{in} \times C_{out}}. \quad (2.10)$$

Це показує, що система не залежить від кількості вхідний каналів, тобто перевага буде зберігатися як на початку, так і впродовж проходження всього циклу моделей, а також видна незалежність від розміру зображень або карт ознак. У цьому випадку ми розділимо дріб на суму двох окремих дробів, що матиме наступний вигляд (2.11):

$$Ratio = \left(\frac{K^2}{K^2 \times C_{out}}\right) + \left(\frac{C_{out}}{K^2 \times C_{out}}\right). \quad (2.11)$$

Після скорочення цього дробу (2.11) ми отримуємо суму квадрату фільтру та вихідного каналу у чисельнику, а у знаменнику ми маємо добуток вже названих елементів, проте ми очевидно можемо скоротити квадрат фільтру у першому елементу суми та вихідні канали у другому елементі, отримуючи фінальну формулу (2.12):

$$Ratio = \frac{1}{C_{out}} + \frac{1}{K^2} \ . \quad (2.12)$$

Як ми бачимо (2.12), фінальне спрощення нашої відносної переваги становить що найменше у 9 разів, оскільки MobileNetV2 переважно використовує згорткові ядра розміром 3, а кількість вихідних каналів зазвичай достатньо велика, щоб нівелювати цей показник (64, 128 чи 256 і т. д.), але говорячи мовою розрахунків $\frac{1}{128} + \frac{1}{9}$ приблизно сягає 0.0078 та 0.111 відповідно, тобто Глибинно-сепарабельна згортка з ядром 3×3 та 128 вихідними потоками зробить лише 11.9% розрахунків у порівнянні з класичною моделлю, що є неймовірно важливо для швидкодії моделі, особливо якщо ми будемо обчислювати великі потоки інформації, під які підпадає обраний датасет у якому приблизно 270 тисяч зображень для навчання та валідації.

2.3 Функція втрат та оптимізації

Якщо ми вже маємо наш процес навчання, то його необхідно вимірювати певним чином, саме цю функцію і виконує функція втрат. Її часто називають критерієм чи критерієм помилки або цільовою функцією і є наріжним каменем будь-якої моделі глибокого навчання. Ця функція забезпечує числове вимірювання розбіжності між результатом роботи моделі та істинними значеннями, і без цієї функції модель просто не може працювати об’єктивно, бо не було би впевненості навчається модель чи просто обирає випадкові ваги.

Функція втрат є вирішальним елементом будь-якої моделі та допомагає оптимізатору у процесі навчання для послідовної зміни вагів заради покращення результатів. Саме цей елемент є надмірно важливим для навчання, яке фактично не просто видає вірний або частково вірний результат, а у мінімізації цієї самої функції втрат, що перетворює помилку на сигнал моделі для зміни ваг. Вибір конкретної функції втрат є не особистим бажанням чи технічним рішенням, вибір моделі функції втрат повинен бути методологічно обгрунтованим. Наприклад, якщо буде обрано середньоквадратичну функцію втрат для цієї задачі(2.13).

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{p}_i)^2, \tag{2.13}$$

де N – загальна кількість зразків;

y_i – істинна бінарна мітка, тобто 0 або 1;

$\hat{p}_i$ – прогнозована ймовірність за допомогою сигмоїди $\hat{y}_i$.

Середньоквадратична функція втрат (2.13) є фундаментальною у сфері машинного та глибокого навчання, проте вона була розроблена у першу чергу для задач регресії, де основна мета прогнозування – це отримання неперервної величини. Ця функція вимірює середньоквадратичну різницю між прогнозованим та справжнім значенням. Не можна сказати, що вибір цієї функції втрат для задач бінарної класифікації буде правильним, попри розповсюдженість

цієї функції у системах глибокого і машинного навчання, через принципову різницю типу задач для яких був розроблений цей критерій та необхідних для дослідження, проте якщо використовувати її для бінарної класифікації, то ми можемо зіткнутися з серйозними проблемами. Середньоквадратична похибка фундаментально не передбачена для мінімізації розбіжності між двома розподілами. Функція середньоквадратичної похибки вимірює відстань між числами. Якщо модель буде класифікувати об'єкти з низькою впевненістю, то буде отримано невелику втрату, проте якщо буде зроблена помилка з великою впевненістю, то ми отримаємо дуже велику втрату точності, що значно знижує ефективність і рівномірність оптимізаційного процесу. Крім того, як вже було зазначено вище, проблема зі зникненням градієнту дуже важлива для даного дослідження. Фактичним недоліком функції середньоквадратичної похибки є градієнт функції втрат, який є похідною з сили оновлення ваг. Середньоквадратична похибка використовує сігмоїду на виході, який обчислюється ланцюгом, і саме цей ланцюг сильно залежить від похідної функції сігмоїди подібної до описаної вище (2.1). Тобто, якщо впевненість функції сягає значень близьких до 0 або 1, то впевненість моделі стає занадто високою, що можна побачити далі (2.14).

$$(\hat{p}_i - y_i) * \hat{p}_i * (1 - \hat{p}_i) \, . \tag{2.14}$$

У випадку такої надмірної впевненості $\hat{p}_i * (1 - \hat{p}_i)$ стає надзвичайно малим, що призводить до зникнення градієнта. Тобто якщо функція повертає значну похибку, то ваги оновлюються повільно, що значно сповільнює навчання або навіть може його повністю зупинити через занадто слабкий сигнал від функції помилки, що робить збіжність моделі згорткової нейронної мережі, особливо для проблем класифікації занадто нестабільним для постійного використання

Для задач бінарної класифікації традиційно використовують бінарну крос-ентропію, яка вимірює різницю між справжнім розподілом класів та

передбаченим розподілом ймовірностей класів, бо вихід таких моделей може бути лише 0 або 1 і часто отримуються через функцію активації сигмоїди, проте для нашої роботи більше підходить бінарна крос-ентропія з логітами (2.15).

$$\text{Pred} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log\left(\sigma(\widehat{y_i})\right) + (1 - y_i)\log(1 - \sigma(\widehat{y_i}))\right], \quad (2.15)$$

де $\widehat{y_i}$ – логіт (вихід моделі до застосування сигмоїди);

$\sigma(\widehat{y_i})$ – функція сігмоїди, що перетворює логіт на ймовірність;

N – кількість зразків у пакеті.

Цю функцію можна вважати стандартом, через чисельну стійкість, поєднуючи сигмоїду та звичайну крос-ентропію в одному виразі, і за використанням цієї формули ми можемо спостерігати як відхилення прогнозу $\sigma(\widehat{y_i})$ від істинної мітки y_i стимулює функцію втрат Pred, що сприяє коригування вагів межі. Проте у межах цієї роботи ми стикнулись з дисбалансом класу, при якому моделі математично ефективно не навчатись, через очевидну перевагу виділення всіх елементів як один клас без намагань до фактичної класифікації класів, до прикладу, у датасеті CelebASpoof частина класу спуф сягає приблизно лише 4-5% від загальної бази даних. За даних умов ця проблема є очевидною.

Для математичного вирішення проблеми дисбалансу класів та забезпечення високого показника виявлення атак ми можемо ввести систему асиметричного покарання за неправильне передбачення класів. У такому випадку нам необхідно замінити модель бінарної крос-ентропії на модель де втрати будуть зважені для асиметричного покарання за похибку (2.16). Для боротьби з упередженістю системи до класу більшості, тобто, справжнього класу, ми можемо ввести параметр W для збільшення ваги передбачення на певний показчик лише тоді, коли істина мітка дорівнює 1 (тобто спуф).

$$\text{WPred} = -\frac{1}{N}\sum_{i=1}^{N}\left[W * y_i \log\left(\sigma(\widehat{y_i})\right) + (1 - y_i)\log(1 - \sigma(\widehat{y_i}))\right]. \quad (2.16)$$

Як ми бачимо (2.16) ця функція WPred змушує градієнти сильніше впливати на ваги мережі у напрямку підвищення точності що до міноритарного класу, бо штраф отриманий за неправильне передбачення впливає на систему лише за умови передбачення класу 1(Spoof) як справжнього і буде помножений у W разів, але необхідно розуміти, що самостійна робота цієї функції недостатня, через те, що у пакеті даних може не бути елементу 1 класу взагалі, а тому необхідна додаткова функція, яка допоможе у помірному, або ж збалансованому завантаженні даних до моделі.

Тобто ми бачимо упередженість моделі до класу більшості через значний дисбаланс класів, але навіть зважена функція втрат не зможе допомогти, якщо у пакеті немає даних 1 класу, роблячи функцію зважених втрат п'ятим колесом системи, проте системі необхідно виявляти, хоч і рідкісні, проте дорогі з точки зору безпеки, атаки. Виходячи з цього у кваліфікаційній роботі використовується стратегія зваженого завантажувача даних, яка забезпечить наявність обох класів у кожному пакеті для більшої ефективності зваженої функції втрат (2.17). Стратегія зваженого завантажувача призначає вагу для кожного окремого елементу в наборі даних, а потім вибирає зразки пропорційно цим вагам.

$$w_i = \frac{1}{\text{Кількість зразків класу } c} = \frac{1}{N_c}. \tag{2.17}$$

Якщо клас справжніх об'єктів буде мати 10000, то вага кожного зразку справжнього об'єкту буде мати вагу $w_i = \frac{1}{10000} = 0{,}0001$, якщо ми маємо лише 2000 об'єктів класу атаки, то ймовірність обрати елементи атаки буде у 5 разів вища, ніж елементи справжніх облич, що балансує ці класи для завантаження у модель для навчання, що надає нам можливість відійти від статистичної упередженості, що дає нам більш надійну та робустну систему захисту від атак, оскільки модель бачить атаки значно частіше і при цьому помилка на атаці карається суворіше.

3 КОМП'ЮТЕРНА МОДЕЛЬ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ

3.1 Обґрунтування вибору середовища програмної реалізації

Модель коду було розроблено у середовищі Visual Studio Community (надаліVS Community) 2022. Це середовище багатофункціональне та комплексне, що дозволяє використовувати професійні інструменти для роботи з Python, особливо у роботі з великими або складними проектами. VS Community є приємним для мене та звичним, проте це не лише особисте бажання. Це середовище має комплексний налагоджувач, який стає нативним через годину користування, цей налагоджувач більш надійний та інтегрований у саме середовище, ніж відомі мені альтернативи. Налагоджувач дозволяє глибоко слідкувати та інспектувати пам'ять, аналізувати вийнятки та знаходити неочевидні процеси у коді та помилки, що виникають через ці неочевидні нюанси, крім того можливисті бачити повний стек викликів допомагає у роботі зі складними та ресурсомісткими програмами.

Крім прекрасного налагоджувача, структура VS Community ідеальна для організації великих проектів, де важливо керувати вихідними, вхідними файлами в рамках одного файлу рішення. Це допомагає підтримувати академічну та інженерну дисципліну, чітко розділяючи різні модулі проекту. У додатку VS Community легко інтегрується у середовище Windows, що дозволяє подальший розвиток проекту.

Якщо говорити про переваги VS Community, то слід говорити і про переваги роботи з мовою програмування Python у цьому середовищі. Це середовище пропонує надійні та безпечні інструменти для рефакторингу, що дуже допомогло у моментах зі зміною архітектури проекту, про які буде згадано нижче. Також система IntelliSense вважається однією з найкращих, що статично глибоко перевіряє код під час набору тексту, виявляючи потенційні помилки, навіть якщо це нюанси стилю або звичайні рекомендації ще до запуску програми (особливо у комбінації з розширенням «Sonarqube»). Надзвичайно важливим

була можливість керувати середовищами та пакетами, що дозволило налаштувати середовище «cuda» для роботи зі своєю графічною картою та окремо підключити кожен з пакетів бібліотек або їх елементів для пришвидшення роботи. Також це середовище просто ідеальне для подальшої інтеграції у будь-який проект гібридного стилю, тобто з використанням Python для високорівневої логіки та C++ для критично важливих у своїй ефективності обчислень, але якщо говорити про ефективність, то VS Community пропонує вбудовані інструменти профілювання, яке надає можливість обрати функції для інспектування, що надасть інформацію про споживання ресурсів окремими елементами для подальшої діагностики, що є важливим при роботі з тензорами зображень.

Саме тому вибір VS Community є виправданим для виконання цієї роботи, бо це середовище надає стабільність та можливість глибокого аналізу коду, що дає зусередитись на продуктивності та отриманні результатів.

3.2 Програмна реалізація

Програмна реалізація системи для виявлення атак виду підміни обличчя у системах розпізнавання обличчя представляє собою модульний багатошаровий застосунок на базі фреймворку PyTorch. Ця структура дозволяє досягти високої точності та достатньої гнучкості для експериментів з вагами для проведення порівняльних досліджень та боротьби з дисбалансом. Так основним фреймворком став PyTotch, який було обрано через динамічність графу обчислень, що спрощує налагоджування та можливість більш гнучко працювати з тензорами. Це було критичною частиною через швидкість його роботи та ефективності роботи з NVIDIA CUDA, що дозволило значно пришвидшити роботу. Прискорення завдяки CUDA та додатковому налагодженню сягнуло від 100+годин, до 20-30 годин, що пришвидшило роботу від 3.5 до 5 разів.

Для базової архітектури було обрано модель MobileNetV2 через глибинні роздільні згортки та використанню залишкових інвертованих блоків, що

збільшує ефективність моделі у використанні ресурсів. Ця модель відносно невелика у своїх параметрах, проте вона зберігає свою потужність вилучення ознак, що може бути важливим для наступного інтегрування у серверну частину застосунку або мобільних пристроїв, відомих за їх відносно слабкими характеристиками. Через доступ до аннотацій датасету для управління данними було обрано бібліотеки NumPy, Pandas, JSON. Остання бібліотека це розширення файлів метаданих, у яких зберігаються шляхи до файлів, мітки, координати для обрамлення обличчя. NumPy є доволі базовим для будь-яких математичних дій у коді, у нашому випадку ця бібліотека використовується для перетворення міток на етапі оцінки ефективності моделі.

Для роботи з Computer Vision було обрано бібліотеки Pillow (PIL). Ця бібліотека використовується для ефективного завантаження даних та обрізання зображень відповідно до рамок обличчя. Також у роботі є бібліотеки tqdm, Scikit-learn, які допомагають не втратити надію у людство. Бібліотека tqdm додає так званий лоадбар, що дозволяє бачити приблизний час навчання у відносно комфортному форматі, а не очікувати початку чи закінчення епохи для відображення результатів епохи навчання, що дає контролювати час необхідний для навчання, а інша бібліотека забезпечує стандартизовану реалізацію метрик, таких як матриця плутанини або точність моделі.

Після пояснення всіх ключових моментів проекту слід переходити до самої реалізації проекту. Цей застосунок є дещо базовим, він складається з 3 частин, де при ініціалізіції відбувається завантаження даних у послідовності про яку написано нижче (рис 3.1), а після завантаження даних з’являється інтерфейс керування про який слід розповісти пізніше.

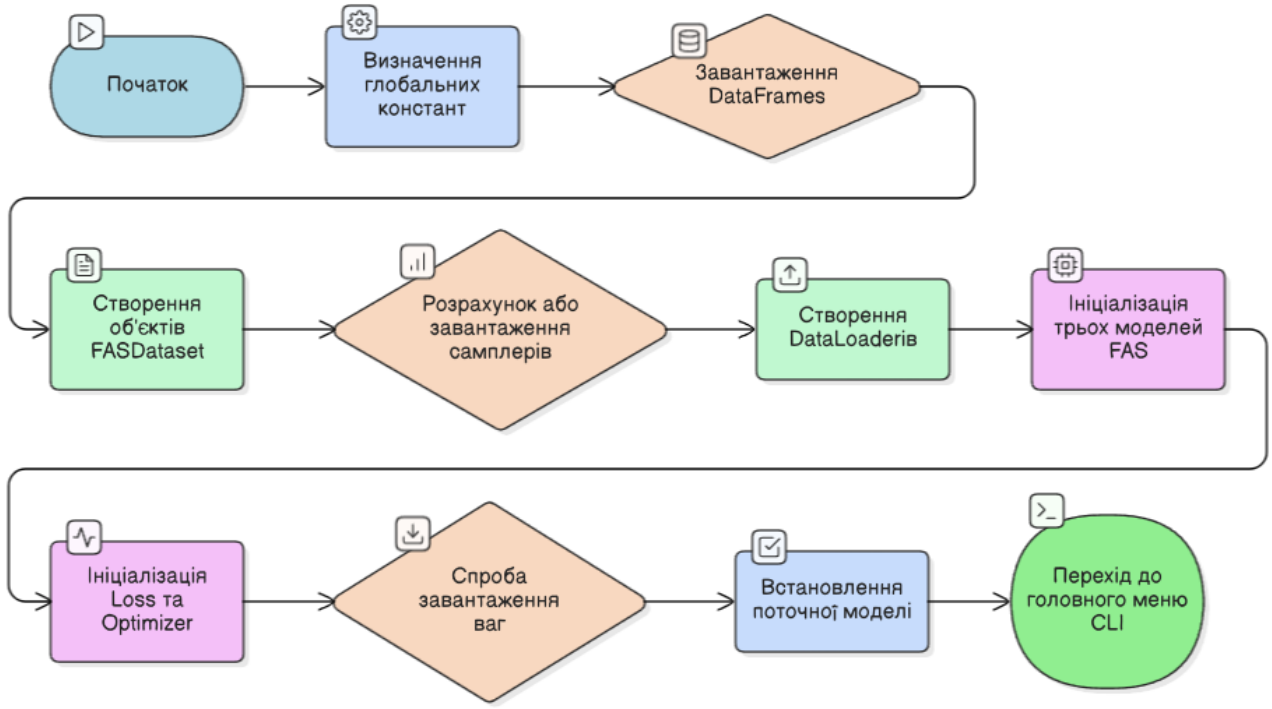

Рисунок 3.1 – Блок схема процесу ініціалізації коду

Після запуску коду, як ми бачимо на рисунку (рис 3.1), є аж 9 блоків перед переходом до наступного етапу. Спочатку визначаються глобальні константи, серед яких ключові гіперпараметри, шляхи до даних та важливі константи для боротьби з дисбалансом датасету який симулює реалістичну атаку (спуфінг сягає близько 4-5% всього датасету). Глобальні константи забезпечують використання єдиного набору налаштувань для вього коду. Наступні два блоки завантажують та зчитують конфігураційні файли у розширенні JSON, які містять мітки файлів, шляхи до зображень та координати анотованих рамок обличчя, а після цього застосунок створює об’єкти класів для 3 режимів навчання про які детально буде детальніше пізніше. Створюється 3 об’єкти класу, відповідно до кожного режиму навчання, який потребує індивідуального завантаження даних та попередньої обробки цих даних. Наступний блок був цікавим вирішенням проблеми дизбалансу датасету. Цей блок викликає функцію самплеру. Якщо вже створений самплер не знайдено, то відбувається повний розрахунок ваг самплеру на основі частоти появи класів у навчальному наборі. Після розрахунку або завантаження ваг створюються об’єкти завантаження даних які використовують самплери для збалансування потоку даних до моделі. Після цього застосунок ініціалізує вище

згадані незалежні моделі MobileNetV2 і переміщує їх на обчислювальний пристрій, що займає доволі багато об’єму оперативної пам’яті, а слід за цим створюється функції оптимізаціїї та підрахунку втрат. Це дозволяє сформувати ансамбль та проводити порівняння у якості роботи моделей, а останній блок, який за замовчуванням встановлює модель «best_fas_bbox_model» дозволяє відразу перейти до тестування або донавчання моделі від її найкращого стану(рис. 3.2).

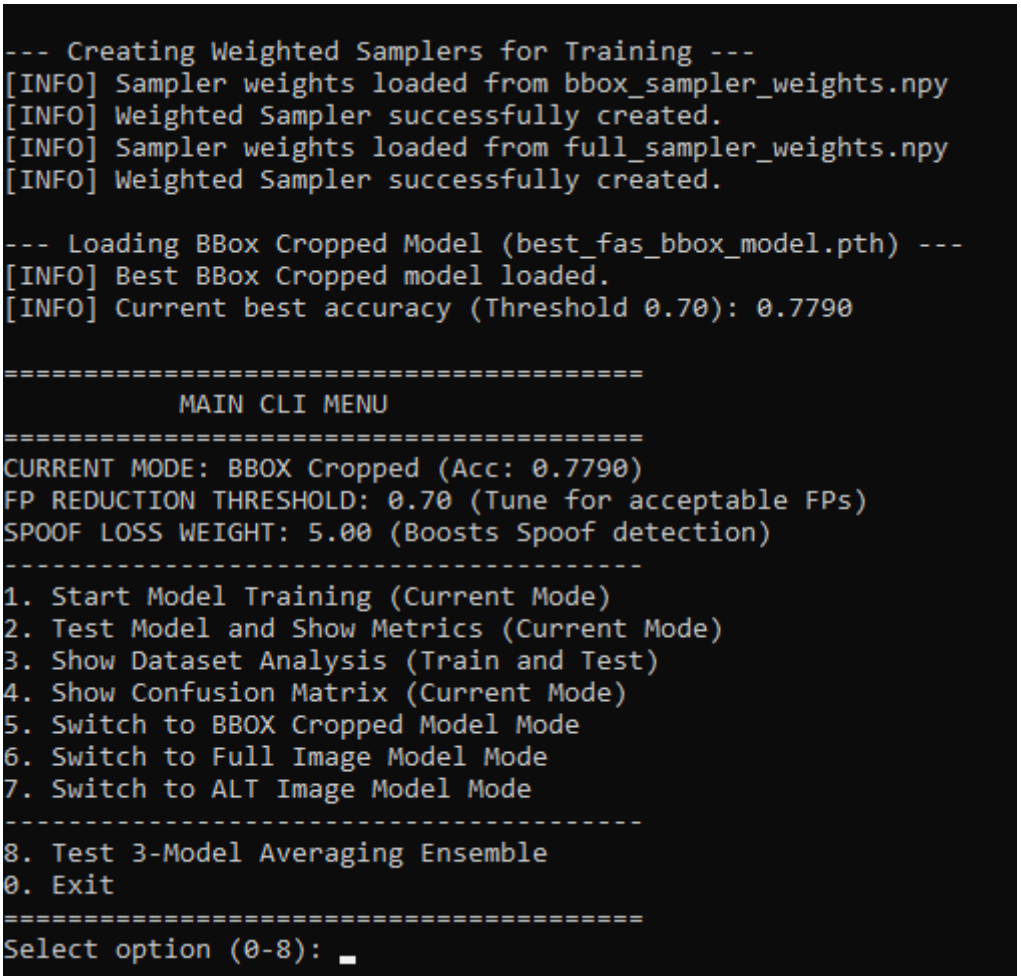

Рисунок 3.2 – CLI інтерфейс застосунку при його запуску

Під час запуску застосунку після ініціалізації ми бачимо (рис. 3.2) CLI інтерфейс, в якому показана інформація про завантаження даних або про їх відсутність з необхідністю їх створення, так самплери будуть створені автоматично, проте навчання моделі необхідно запускати самостійно. Також інтерфейс показує поточну модель з яку буде викликано при навчанні, аналізі

або тестуванні. Всього обрати можна три моделі, а саме: BBOX, Full image, alt image. Також інтерфейс показує поріг при якому зображення буде передбачено як справжнє, а нижче вагу передбачень зображень які є спуффінгом. Сам інтерфейс складається з 9 пунктів (табл 3.1).

Таблиця 3.1 – Елементи керування CLI інтерфейсу

Опція	**Назва**	**Призначення**	**Категорія**
1	**2**	**3**	**4**
1	Start Model Training (current Mode)	Запускає цикл тренування «start_training» для поточної активної моделі. Зберігає модель при покращенні точності	Навчання та керування моделі
2	Test Model and Show Metrics	Виконує оцінку моделі, використовуючи встановлений поріг впевненості моделі	Аналіз та оцінка
3	Show Dataset Analysis	Аналізує тестовий набір, показуючи кількість зразків та їх розподіл по класам	Аналіз та оцінка
4	Show Confusion Matrix	Запускає тестування поточної моделі з відображенням матриці плутанини	Аналіз та оцінка
5	Switch to BBOX Cropped Model Mode	Встановлює поточну модель на вибрану зі спробою завантаження збережених ваг	Навчання та керування моделі
6	Switch to Full Image Model Mode	Встановлює поточну модель на вибрану зі спробою завантаження збережених ваг	Навчання та керування моделі

Продовження таблиці 3.1

1	2	3	4
7	Switch to ALT Image Model Mode	Встановлює поточну модель на вибрану зі спробою завантаження збережених ваг	Навчання та керування моделі
8	Test 3-Model Averaging Ensemble	Перевірка робустності моделію Викликає 3 моделі та використовує їх для фінального рішення за середнім прогнозуванням	Ансамбль
0	Exit	Зупиняє роботу застосунку	Навчання та керування моделі

Як вже було зазначено , програма керується через CLI інтерфейс, що означає повну можливість керування застосунком через інтерфейс, так при виборі опції 1 – початку навчання, застосунок почне тренування активної моделі (рис 3.3), що включає в себе перевірку активної моделі моделі щоб детермінувати який завантажувач даних використовувати та ініціалізує оптимізатор та функцію втрат перед початком навчання. Після встановлення перших параметрів, застосунок переходить до зовнішнього циклу епох який запускає лоадбар для спостереження за прогресом внутрішнього циклу навчання за допомогою пакетів які проходять через шари моделі та обчислюють кінцевий вихідний прогноз. Після обчислення пакету система переходить до режиму оцінки.

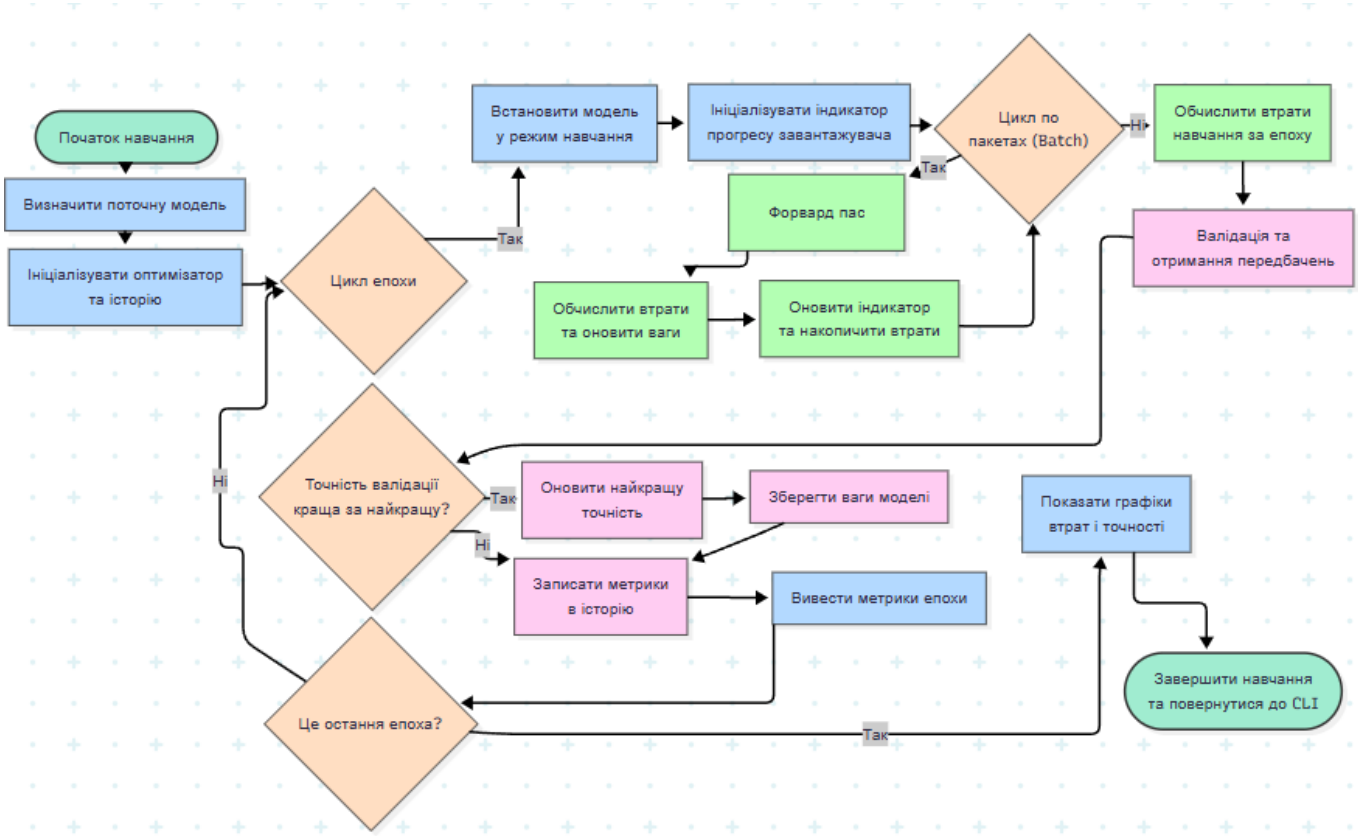

Рисунок 3.3 – Блок схема процесу навчання

Система обчислює втрати та градієнти з подальшим оновленням ваг для подальшого навчання та оновленням лоадбару з оновленням змінної поточних втрат за епоху. Після обчислення всіх пакетів система повертає фінальне середнє значення втрат за всю епоху з початком валідації яка переводить систему у режим оцінки з подальшим обчисленням точності та перевіркою точності з попередньою (якщо це перша епоха, то початкова точність 0.00), та якщо точність вища за попередню, то ваги моделі будуть збережені, встановлюючи чекпоінт для моделі, якщо ж точність не змінилась або зменшилась, то збереження моделі пропускається (рис. 3.4).

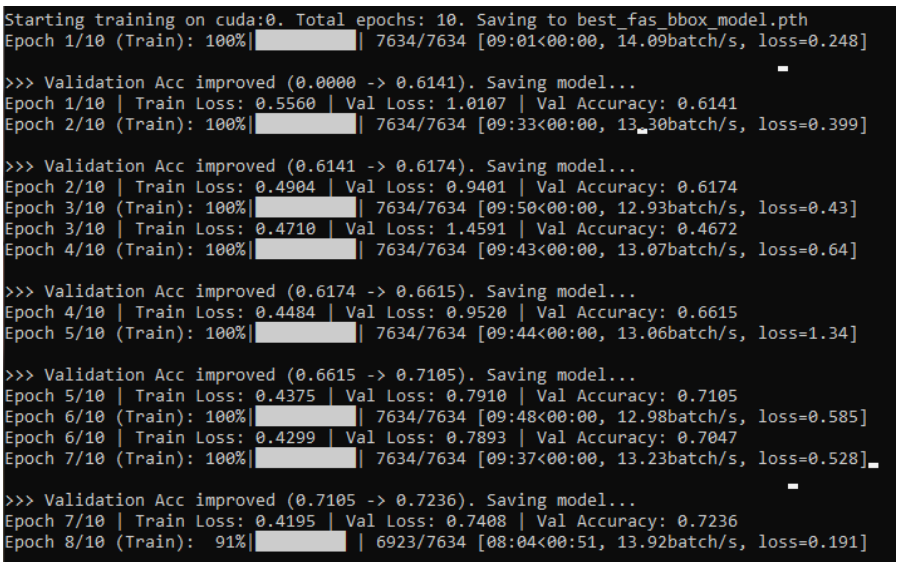

Рисунок 3.4 – Приклад процесу навчання моделі та її збереження

При завершенні навчання всіх епох система закінчує навчання та повертає користувачу графік з побудованою точністю та втратами кожної епохи навчання для візуалізації результатів навчання, що допомагає аналізувати процес навчання, а не лише його результати. Після відображення графіків навчання система повертається до CLI з оновленою збереженою моделлю. Крім того Dropout встановлений на 0,2, що допомагає системі не перенавчатися та підвищує робустність системи.

Також користувачу доступні функції для аналізу датасету, який являє собою варіацію «CelebA», а саме «CelebA-Spoof», який є у вільному доступі у вигляді 75 томів архіву розміром у 1000МБ кожен. Що дозволяє відтворити результати цієї роботи з мінімальними труднощами, а також у користувача є доступ до матриці плутанини, що допомагає зрозуміти у чому саме проблема та деякі неочікувані труднощі, що будуть показані нижче.

3.3 Інструкція користувача

Застосунок призначений для інженерів та дослідників, які займаються розробкою та порівняльним аналізом моделей глибокого навчання для виявлення

атак виду підміни обличчя. Основний функціонал включає керування трьома незалежними моделями та тестування їх ансамблю

Для коректної роботи застосунку необхідно відповідати наступним вимогам:

– операційна система: Windows 10/11 або Linux;

– Python 3.9+;

– бібліотеки PyTorch, torchvision, Numpy, Matplotlib, scikit-learn, tqdm;

– відеокарта з підтримкою NVIDIA CUDA та її середовище (рекомендовано для значного пришвидшення роботи).

Перед початком роботи необхідно провести початкові налаштування шляхів, а саме ROOT_DIR, TRAIN_JSON, TEST_JSON, також бажано переконатись, що у вас працює середовище CUDA. Якщо ж ви хочете використати інший датасет, то повинен підійти будь-який з наявними анотаціями облич та структурою, яка буде надана у додатку (або можливий варіант написання власного коду для знаходження та виділення обличчя).

Для початку роботи відкрийте «.sln» файл застосунку у будь-якому середовищі розробки або через термінал. Застосунок перевірить наявність коректно встановленого датасету та моделей, після чого відобразить CLI меню з інформацією про поточну активну модель та налаштування, які будуть впливати на результат (Threshold та Loss Weight). Опис головного меню та функціоналу можна знайти вище (табл. 3.1).

Рекомендований порядок роботи:

– оберіть опцію 3 для підтвердження наявності дисбалансу класів та коректності завантажених даних;

– оберіть опцію 5. Завантаження BBOX моделі, та оберіть опцію 1 і дочекайтесь закінчення навчання моделі;

– повторіть попередній пункт з опцією 6 та 7. Усі моделі повинні бути навчені та збережені;

– після навчання моделей використайте опції 2 та 4 для опцій 5, 6 та 7 для отримання інформації про їх точність та матриці плутанини;

– якщо результат незадовільний знайдіть у коді «FP_REDUCTION_THRESHOLD» та змініть його значення, чим вище значення, тим більше елементів будуть відмічені як «Spoof», тому обирайте між безпекою та зручністю можливих користувачів з розумом;

– повторіть попередні кроки;

– якщо результат вас влаштовує, обирайте опцію 8. Ця опція запускає ансамбль моделей для фінальної оцінки найбільш робустної системи.

3.4 Дослідження та тестування розробленої моделі

Розроблена модель складається з 3 варіацій моделі на архітектурі MobileNetV2.

Ця модель представник згорткової нейронної мережі, що використовує глибинно-роздільну згортку. Вибір вже готової архітектури обгрунтований недоцільністю самостійної розробки мережі, а архітектура MobileNetV2 достатньо легка у своїх обчисленнях через значно меншу кількість параметрів та FLOPS порівняно з іншими моделями при відносно однаковвій точності для цього завдання. Ця мережа ефективно використовує ресурси пристрою, а структура моделі, зосереджена на виявлення тонких артефактів, мікротекстурах об’єктів є майже ідеальною для цієї задачі.

Для аналізу датасету ми можемо використати вбудовану у застосунок функцію (рис.3.5), яка демонструє нам більше 270 тис. зображень, з яких лише 10856 є хибними зображеннями людей, тобто спуфами. Це лише 4.02% від всього датасету. Хоч такий розподіл даних симулює реальний потік, де спуфи – абсолютна меншість зображень, проте кожний спуф, який може проникнути у систему – загроза для безпеки як користувача, так і всього застосунку.

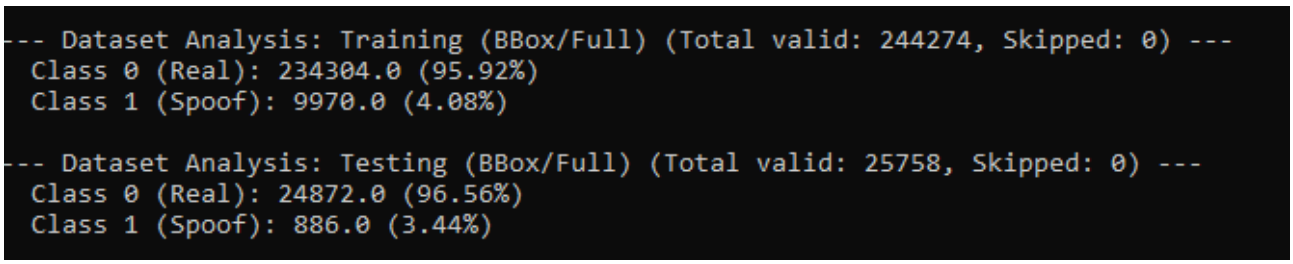

Рисунок 3.5 – Інформація про базу даних у застосунку

Дослідження будуть описані у вигляді точності, повноти розпізнавання атак, специфічності (правильно виявлені справжні обличчя) та частки хибно позитивних спрацювань моделі. Це дозволить повноцінно зрозуміти що відбувається під час досліджень та дозволить порівняти моделі. Через наявність чекпоїнтного зберігання моделі ми можемо не турбуватись про перенавчання моделі через фінальну роботу з найкращим з навчених варіантів моделі, крім того слід знайти баланс між захистом та комфортом користувача. У цьому випадку буде порівняно відносну кількість хибно позитивних спрацювань моделі до відносної кількості спуфів, що змогли обійти модель. Це дасть нам зрозуміти які показники є найефективнішими для знаходження цього балансу, проте для цього показнику приймемо до уваги, що слід вважати, що безпека теоретично впливає на якість моделі у 10 разів більше, ніж незадоволення користувачів, що не змогли отримати доступ через хибнопозитивне спрацювання моделі.

Тобто нашою цільовою метрикою буде зважена частка помилок, що ми досягнемо підсумуванням всіх помилок системи, помножених на їх коефіцієнт важливості і діленням на загальну кількість зразків, формула для підрахунку буде виглядати як(3.1):

$$\text{Зважена вартість} = \left(W_{fn} * FN\right) + (W_{fp} * FP), \tag{3.1}$$

де FN – хибнонегативні зразки;

FP – хибнопозитивні зразки;

W – вага помилки.

У такому випадку зважена частка помилок (WER) буде виглядати як: WER сягає зваженої вартості поділеної на загальну кількість з оцінкою точності у 1 – WER.

Це формулювання метрики точності дозволить нам точно та точково оцінити компроміс, який ми маємо між чіткими та високими вимогами до безпеки та надважливим комфортом користувачів. Це надзвичайно важливо для розробки та застосування подібної моделі у застосунку, і у випадку якщо даний компроміс не буде збережено, то система буде дуже швидко скомпрометована (рис. 3.6), як вже зазначалося у вступі до кваліфікаційної роботи, але якщо модель буде занадто агресивна, що повністю вилучить можливість вторгнення зловмисника до системи, то користувачі будуть занадто незадоволені складністю входу до системи і швидко знайдуть заміну застосунку, роблячи його не конкуренто спроможним.

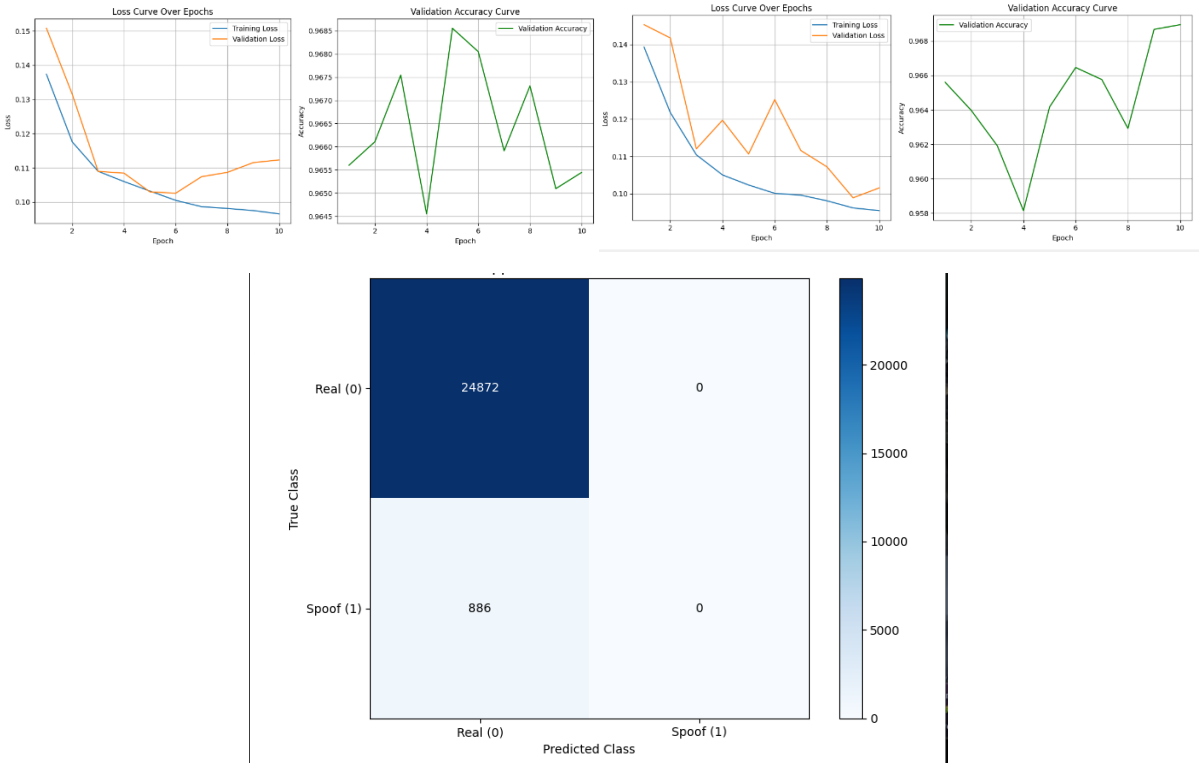

Рисунок 3.6 – Графіки втрат та матриця плутанини без використання додаткових засобів (а – графіки моделі BBOX, б – графіки моделі Full, в – матриця плутанини ансамблю)

Перші спроби запуску моделі були відносно вдалими. Після процесу оптимізації для прискорення моделі від 10 годин на 1 епоху навчання до 20 хвилин, перший запуск моделей був неймовірно вдалий на перший погляд (рис 3.6 а,б), точність моделей сягає 96%, що може здивувати, тому було проведено повторне навчання і коли результати моделей виявились схожими, то запуск ансамблю у відношенні 70% моделі BBOX та 30% моделі Full_image з виведенням матриці плутанини (рис.3.6 в) показав, що модель не могла покращитись через зупинку на чекпоїнті в 96% точності, що призвело до передбачень всіх елементів як справжніх зображень, бо будь-які зміни у вагах були неймовірно контр продуктивні через те, що відносна кількість spoof зображення сягає лише близько 4%.

Після цього дослідження було вирішено, що система буде збільшена до 3 моделей для збільшення стійкості (рис. 3.7), а також було необхідно додати допоміжні системи. Однією з допоміжних систем є функція «weighted_sampler», яка забезпечить поступовий збір пакетів з 2 класами зображень, бо існує ймовірність того, що система навчається розрізняти лише 1 клас, і не може розрізнити інший.

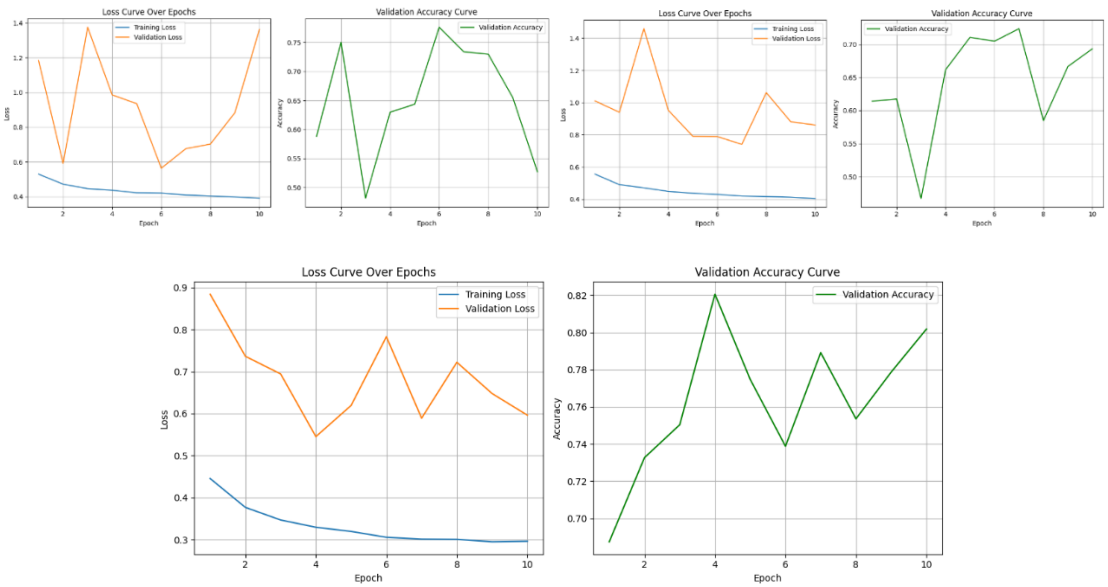

Рисунок 3.7 – Графіки навчання при вагах впевненості 0.7 та штрафові за помилку у розмірі 5 (а – графік навчання Full_Image, б – графік навчання Alt_Image, в – графік навчання BBOX_cropped)

Після навчання ми більше не бачимо 96% точності, що є добре. Під час цього дослідження було додано самплери, які рівномірно наповнюють пакети для навчання за вагою кожного елементу, яка є оберненим індексом частоти. Це змушує модель бачити обидва класи однаково часто, а не лише мажоритарний клас «Real». Також додано поріг впевненості моделі, при якому модель повинна бути хоча б на 70% впевнена, щоб присвоїти зображенню клас 0, тобто «Real», у інших випадках зображенню буде присвоєно клас «Spoof», але в той же час через важливість недопущення спуфів у систему було додано функцію втрат «nn.BCEWithLogitsLoss(pos_weight=torch.tensor([SPOOF_POS_WEIGHT]))».
Якщо перед цим ми могли залишити сигмоїду для звичайної бінарної класифікації, то необхідність балансування навчання та датасету змушує нас використовувати бінарну крос-ентропію з логітами. Вона більш стабільна та дозволяє балансувати вагу передбачень. Для виправлення дисбалансу «SPOOF_POS_WEIGHT» було втановлено на 5. Це фактично множник штрафу за помилку, де відношення реального зображення як спуфу буде вартувати 1 пункт точності, у той час як відношення спуфу як реального зображення буде вартувати системі 5 пунктів. Така реалізація карає модель за пропущену атаку, у той час як хибне спрацювання карається значно м’якше. Така система дозволяє системі навчатися у більш керованому ключі (рис 3.7), а точність залишається високою попри агресивність системи, хоч і існує ймовірність накопичення помилки з часом та швидкого перенавчання моделі.

Ансамбль цього дослідження (рис 3.8) виглядає набагато краще, так Правильно обрано було 21142 об’єкти класу справжнього зображення, а 842 об’єкти правильно обрані як спуф, що дає нам близько 85.35% точності моделі. Так 95.03% зображень атаки правильно оцінюються як спуф, проте лише 18.42% випадків зпрацювання моделі на атаку є справжньою атакою. Після зважання оцінки точності ми отримуємо 83.81% точності ансамблю. Це гарна точність моделі, проте можливо модель є занадто агресивною до атак, необхідно спробувати більш м’яку модель.

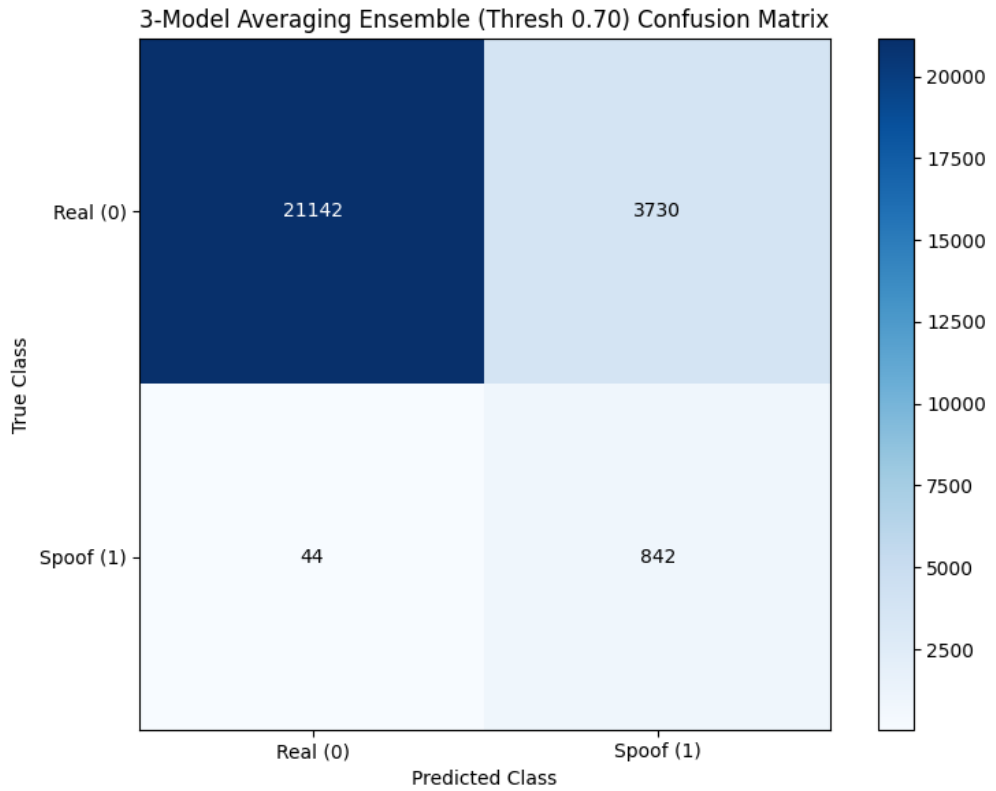

Рисунок 3.8. – Матриця плутанини порогу 0.7 та штрафу розміром в 5

Наступне дослідження встановило поріг впевненості моделей у розмірі 0.55, а штраф за пропуск атаки лише на 2. У цьому випадку моделі повних зображень (рис 3.9 б, в) виглядають більш хаотично. Модель все ще залишається відносно стабільною, з точністю яка коливається від 82% до 69% точності для моделей повного зображення, у той час як на момент закінчення навчання модель обрізаного зображення з рамками обличчя (рис 3.9 а) виглядає наче точність все ще може зростати, бо втрати моделі поступово зменшуються, проте ми бачимо нестабільність для стандартного каскадного навчання декількох моделей, що ще раз підкреслює необхідність чекпоїнтів для подібних моделей.

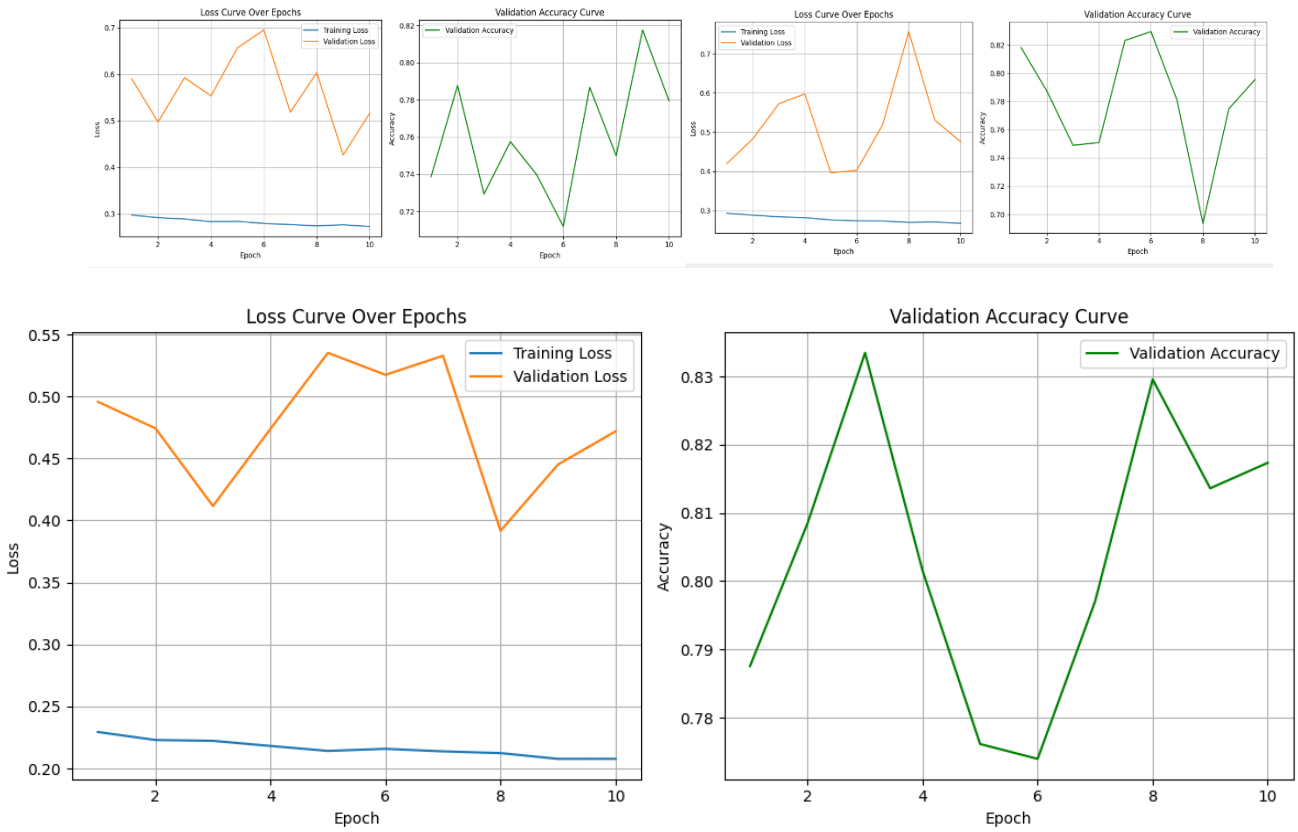

Рисунок 3.9 – Графіки навчання моделей з порогом у 0.55 та штрафом за похибку у розмірі 2 (а – графік моделі BBOX_Cropped, б – графік моделі FULL_Image, в – графік моделі Alt_Image)

Результати ансамблю цієї моделі доволі дивуючі (рис.3.10). Точність відбиття атак складає приблизно 96,15%, а абсолютна точність моделі склала 85,46%. При менш агресивній моделі кількість пропущених атак зменшилась на 10, як кількість блокованих користувачів зменшилась на 8, що вже може говорити нам про покращення, але якщо бути говорити показниками, то зважена точність склала 84,23%, що краще на 0,42% ніж агресивна модель. На подив, ця модель є краща щодо зважених критеріїв. Зменшення порогу призвело до зменшення похибок обох типів. Це рідкісне явище, проте воно свідчить, що модель з порогом 0,55 є кращою за модель з порогом 0,7, проте необхідно протестувати важливість штрафу за помилки, бо у випадку, якщо необхідна впевненість для моделі знайдена, необхідно побачити інші варіанти штрафування за похибку.

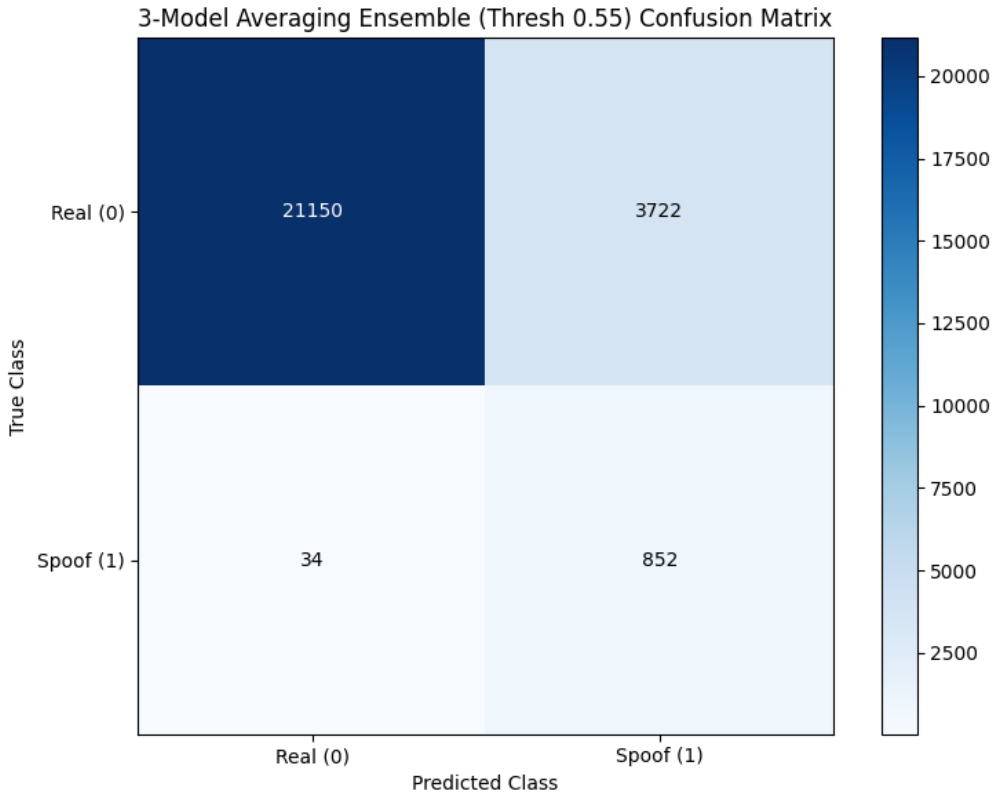

Рисунок 3.10 – Матриця плутанини для ансамблю з порогом 55% впевненості та штрафом у розмірі 2

При спробі збільшення показника штрафу за хибнопозитивне передбачення до 10 (рис 3.11 а), на подив значно збільшився показчик втрат, а точність моделі сягнула 72% у пікові, у той час як точність попередніх моделей сягала близько 80 відсотків. Лише з тренування 1 моделі ми вже можемо бачити, що штраф розміром у 10 занадто великий, та лише зменшує ефективність моделі через занадто велике покарання за невдале навчання. З цих причин наступна спроба навчання була з показчиком штрафу у розмірі 4, що призвело до доволі хороших результатів. Перша модель, яка навчалась з цим показчиком (рис 3.11 б), досягла точності близько 79%, що не є значним зниженням та було вирішено продовжити дослідження з цим показчиком.

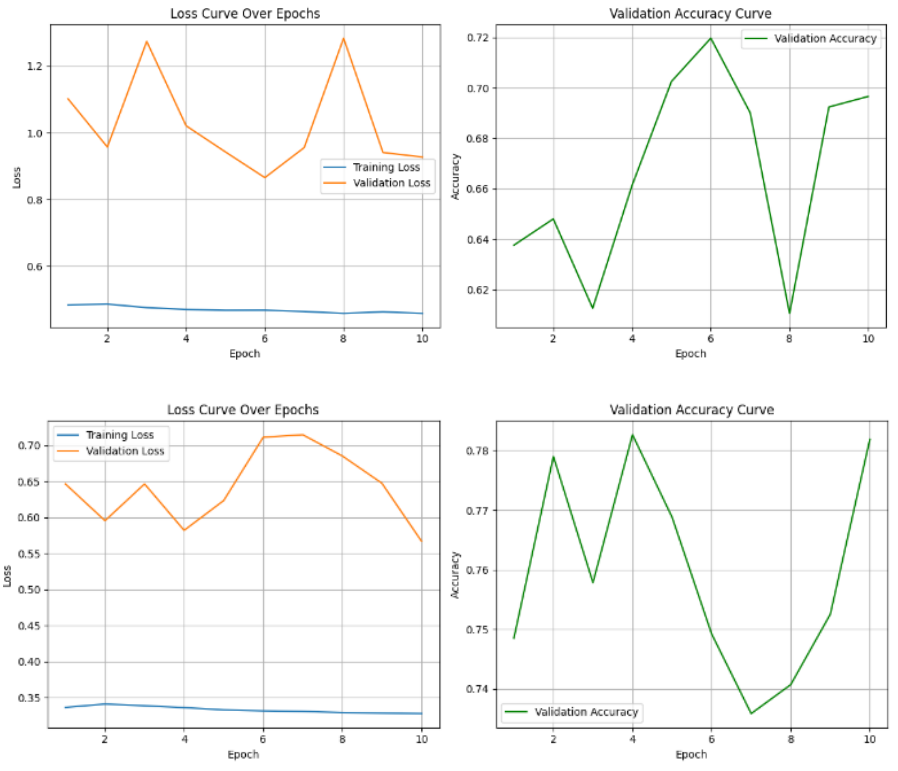

Рис 3.11 – Графіки навчання моделі BBOX_Cropped (а – зі штрафом 10, б – зі штрафом 4)

Закінчення навчання з цими показчиками (рис 3.12) просто неймовірно зменшило кількість фейків, які змогли проникнути повз модель, це вдвічі менше за попередню модель, але також можна помітити збільшення кількості користувачів, які були хибно визначені як фейки. Хоча ця зміна не така відносно велика як зменшення фейків, все одно необхідно обчислити ефективність. Згідно формули виведеної на початку етапу досліджень зважена оцінка точності сягає 82,10%. Хоча ми змогли захистити систему набагато краще з цими показчиками, 720 користувачів не змогли потрапити досистеми, хоча це відносно невелика кількість (маємо на увазі майже 25тис користувачів), проте через те як були встановлені ваги обчислення доречно буде сказати, що якість моделі знизилась на 2,13%. Як ми можемо бачити (табл. 3.2) ми провели експериментальне дослідження розробленої системи протидії спуфінгу, проте ці результати можна визнати незадовільними.

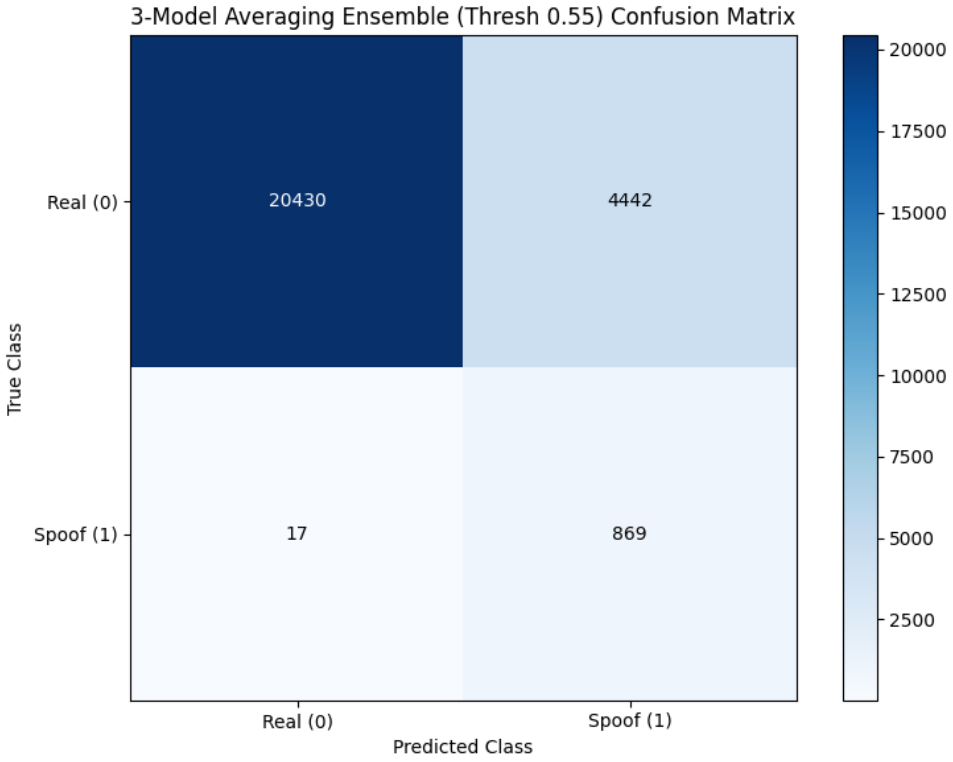

Рисунок 3.12 – Результат ансамблю з порогом 55 та ціною похибки 4

Найкращий результат точності сягнув лише 84,23%, що не можна вважати достатнім для використання у комерційних системах, через те що, при значному зниженні кількості пропущених атак кількість користувачів, що не змогли отримати доступ до системи значно збільшується, але головною метою експерименту було демонстрування ефективності та життєздатності подібного методу для протидії спуфінгу, бо навіть при низькому штрафі система демонструє здібність до боротьби з упередженістю до реального класу та виявлення атак. Система ансаблю продемонструвала свою робустність у повній мірі, бо зміна порогу рішення (від 0,55 до 0,70) та штрафу не сильно впливають на точність моделі, яка коливається в діапазоні 82,10%-84,23%, що може свідчити про низьку дисперсію помилок.

Таблиця 3.2 – Результати ансамблю всіх досліджень з однаковими показчиками

Конфігурація ансамблю	FN (Хибно негативний)	FP (Хибно позитивний)	Зважена похибка FP+(10*FN)	Зважена точність
Поріг 0,70 Штраф 5	44	3730	4170	83,81%
Поріг 0,70 Штраф 4	42	3722	4142	83,67%
Поріг 0,70 Штраф 2	26	4000	4260	83,48%
Поріг 0,55 Штраф 2	34	3722	4062	84,23%
Поріг 0,55 Штраф 4	17	4442	4612	82,10%
Поріг 0,62 Штраф 4	25	3975	4225	83,60%
Поріг 0,62 Штраф 5	20	4119	4369	83,60%

Найкращий баланс було досягнуто при конфігурації Поріг 0,55 та штраф 4. Ця комбінація дала результат у 84,23%, проте для подальших досліджень необхідно експериментувати з різними конфігураціями окремих моделей та поєднання їх у ансамбль з цим порогом, що зможе поєднати в собі системи різної консервативності, що можливо зможе підвищити точність системи, іншим же варіантом підвищення точності може бути використання іншої системи для «ALT_Image model», бо на даний момент вона копіює модель повних зображень.

ВИСНОВКИ

Таким чином, у кваліфікаційній роботі досліджено методи обробки зображення та розроблено модель для протидії спуфінгу, а також вирішено такі завдання:

– проведено аналіз методів та принципів знаходження дефектів зображення та проблематики і переваг даних методі, що дало можливість визначити архітектуру для розробки моделі протидії;

– сформовано та програмно реалізовано покроковий алгоритм для кожного з обраних методів навчання, що дало можливість створення функціонального базису моделі для подальшого проведення експерименту;

– розроблено та реалізовано стратегію балансування даних, що дало можливість вирішити проблему упередженості моделі до класу більшості;

– розроблено стратегію зваженого навчання, що дозволило збільшити чутливість моделі до атак;

– проведено калібрування порогу рішень та асиметричного покарання, що дозволило збалансувати кількість пропущених атак та заблокованих користувачів для знаходження балансу між безпекою та комфортом.

У рамках кваліфікаційної роботи було проведено комплексне дослідження стратегій виявлення атак підміни обличчя, які охоплюють диверсифіковані підходи до аналізу вхідних даних. Ці підходи охоплюють як дослідження текстур локальних ознак обличчя, так і аналізу контексту зображення та ансамблевої агрегації для підвищення робустності та точності системи, що було програмно реалізовано у вигляді модуля з консольним інтерфейсом для налагоджування та навчання і перенавчання моделей з можливістю оцінки ефективності моделей за відображення точності, матриці плутанини та ROC-кривої.

Для всіх процесів навчання та роботи алгоритмів було побудовано блок-схеми для візуалізації їх логіки, щоб наочно зобразити послідовність операцій.

Реалізовано механізм боротьби зі статистичними викликами бази даних. Для подолання цього було використано стратегії зваженого завантаження даних

та асиметричного покарання за помилки у навчанні. Також проведене тестування розроблених моделей та порівняння їх результатів у контексті компромісу між безпекою та комфортом користувача.

Розроблено зручний та зрозумілий користувацький інтерфейс для надання змоги самостійного дослідження кожної окремої моделі та їх ансамблю з виведенням ключових метрик для їх аналізу.

Наукова новизна роботи полягає у створенні та експериментальному підтверджені ефективності методології диверсифікації ансамблевої моделі прийняття рішень з бінарної класифікації з математичними інструментами для забезпечення робустного виявлення атак при дотриманні вимог до комфорту користувачів та безпеки.

Результати роботи апробовано у вигляді 2 тез доповідей під час Міжнародної студентської наукової конференції «Модернізація та сучасні українські і світові наукові дослідження» [24], VIII Всеукраїнська студентська конференція «Формування сучасної науки: методика та практика» [25].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Творошенко І.С., Зеленський М.О. Дослідження гібридних методів для класифікації складноструктурованих зображень. Сучасний рух науки: тези доповідей VIII міжнародної науково-практичної інтернет-конференції (Дніпро, 3–4 жовтня 2019 р.). Дніпро, 2019. Т. 3. С. 382-387.

2. Daradkeh, Y.I., Gorokhovatskyi, V., Tvoroshenko, I., Gadetska, S., and Al-Dhaifallah, M. (2021) Methods of Classification of Images on the Basis of the Values of Statistical Distributions for the Composition of Structural Description Components, IEEE Access, 9, pp. 92964-92973.

3. Ridley, J., & Bentley, J. (2025, July 25). Brits can get around Discord's age verification thanks to Death Stranding's photo mode, bypassing the measure introduced with the UK's Online Safety Act. We tried it and it works—thanks, Kojima. PC Gamer. https://www.pcgamer.com/hardware/brits-can-get-around-discords-age-verification-thanks-to-death-strandings-photo-mode-bypassing-the-measure-introduced-with-the-uks-online-safety-act-we-tried-it-and-it-works-thanks-kojima.

4. Gamers, VPNs, and Norman Reedus: how the UK Is fighting back against online ID checks. (2025, August 9). Cybernews. https://cybernews.com/security/norman-reedus-vpn-surge-rebellion-uks-online-safety-act.

5. Shijie Rao, Yidong Huang, Kaiyu Cui, and Yali Li, "Anti-spoofing face recognition using a metasurface-based snapshot hyperspectral image sensor," Optica 9, 1253-1259 (2022).

6. Khairnar, S., Gite, S., Kotecha, K., & Thepade, S. D. (2023). Face Liveness Detection Using Artificial Intelligence Techniques: A Systematic Literature Review and Future Directions. Big Data and Cognitive Computing, 7(1), 37. https://doi.org/10.3390/bdcc7010037.

7. 19. Tvoroshenko I., and Tkachenko D. (2020) Mechanisms of image classification based on descriptors of local features, Abstracts of IV International

Scientific and Practical Conference «Integration of scientific bases into practice» (October 12-16, 2020). Stockholm, Sweden, pp. 443-448.

8. Wikipedia Contributors. (2019, October 14). Moir. Wikipedia; Wikimedia Foundation. https://en.wikipedia.org/wiki/Moir.

9. Integration of scientific bases into practice. (2020). International Science Group. 443-448.

10. Kresimir Delac, Grgic, M., & Grgic, S. (2008). Recent Advances in Face Recognition. BoD – Books on Demand. Стр109-143

11. GeeksforGeeks. (2018, November 14). Introduction to MultiTask Learning(MTL) for Deep Learning. GeeksforGeeks. https://www.geeksforgeeks.org/deep-learning/introduction-to-multi-task-learningmtl-for-deep-learning.

12. Suyel Namasudra. (n.d.). Data Science and Network Engineering. Springer Nature.

13. Жадан, О. В. (2022). Дослідження класифікатора зображень із використанням ансамблевих засобів аналізу складу структурного опису.

14. Gorokhovatskyi, V., Tvoroshenko, I., & Chmutov, Y. (2022). Застосування систем ортогональних функцій для формування простору ознак у методах класифікації зображень. Advanced Information Systems, 6(3), 5-12.

15. Jain, A. K., Ross, A., & Prabhakar, S. (2004). An introduction to biometric image processing and recognition. IEEE Transactions on Circuits and Systems for Video Technology, 14(1), 4–20.

16. Tumer, K., & Ghosh, J. (1996). Error correlation and error reduction in ensemble classifiers. Applied Intelligence, 6(3), 193-204.

17. Mansfield, A., Maaten, R. J., & Van Den Bosch, P. (2018). The usability of biometric systems. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 48(4), 633–645.

18. Гончаров, Д. М. (2025). Дослідження підвищення точності класифікації зображень за допомогою методів семантичного аналізу та вагових коефіцієнтів

[Кваліфікаційна робота здобувача вищої освіти, пояснювальна записка, Харківський національний університет радіоелектроніки].

19. Daradkeh, Y.I., Gorokhovatskyi, V., Tvoroshenko, I., Gadetska, S., and Al-Dhaifallah, M. (2021) Methods of Classification of Images on the Basis of the Values of Statistical Distributions for the Composition of Structural Description Components, IEEE Access, 9, pp. 92964-92973.

20. Gorokhovatskyi, V., Tarasenko, A., & Trokhymchuk, S. (2020). Застосування логічних форм функцій вибору у задачах розпізнавання образів. Вісник Національного технічного університету «ХПІ». Серія: Математичне моделювання в техніці та технологіях, (1 (1355)), 16-23.

21. Guo, X., Jiang, Q., Pimentel, A. D., & Stefanov, T. (2024). Model and system robustness in distributed CNN inference at the edge. Integration, 100, 102299. https://doi.org/10.1016/j.vlsi.2024.102299.

22. Singh, D. (2024, October 30). Understanding ReLU: The Activation Function Driving Deep Learning Success. Medium; AI-Enthusiast. https://medium.com/ai-enthusiast/understanding-relu-the-activation-function-driving-deep-learning-success-1cbe58eb555a.

23. Корякін, І. М. (2021). Розроблення застосунку для виявлення та класифікації подібності рис людських облич.

24. Богун, Р. С. (2025, листопад 4). Актуальність загрози та гнучкість спуфінгу. Матеріали IX Міжнародної студентської наукової конференції “Модернізація та сучасні українські і світові наукові дослідження”, Житомир, Україна. Матеріали конференцій МНЛ.

25. Богун Р. С. (2025, жовтень 17). Використання стратегії диверсифікованого ансамблю для підвищення робустності систем протидії спуфінгу (PAD). Матеріали VIII Всеукраїнської студентської наукової конференції “Формування сучасної науки: методика та практика”, Львів, Україна. Матеріали конференцій МНЛ.