Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук
(повна назва)

Кафедра програмної інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження моделей виявлення фейкового контенту
на основі технологій машинного навчання
(тема)

Виконав:
студент 2 курсу, групи ІПЗм-22-4

Кіценко Ю.О.
(прізвище, ініціали)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва спеціальності)
Тип програми освітньо-наукова

Керівник проф. Смеляков К.С.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри _____ З.В.Дудар
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерних наук
Кафедра програмної інженерії
Рівень вищої освіти другий (магістерський)
Спеціальність 121 – Інженерія програмного забезпечення
Тип програми освітньо-наукова програма
Освітня програма Інженерія програмного забезпечення
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«___» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

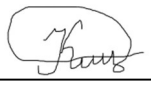
студентові Кіценку Юрію Олександровичу
(прізвище, ім’я, по батькові)

1. Тема роботи «Дослідження моделей виявлення фейкового контенту на основі технологій машинного навчання»
Затверджена наказом по університету від 29.03.2024р. № 250 Ст
2. Термін подання студентом роботи до екзаменаційної комісії 16.06.2024
3. Вихідні дані до роботи інформація щодо згорткових нейронних мереж, існуючих моделей виявлення фейкового контенту та наборів даних, що містять фейковий контент
4. Перелік питань, що потрібно опрацювати в роботі
огляд та аналіз літературних джерел з дослідження, вивчення існуючих моделей виявлення фейкового контенту та наборів даних, що містять фейковий контент, підготовка до проведення експерименту, проведення експерименту та аналіз результатів

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної галузі та постановка задачі	22.01 – 14.02.24	*виконано*
2	Огляд моделей та наборів даних для виявлення фейкового контенту	15.02 – 29.02.24	*виконано*
3	Вибір програмних засобів та архітектури моделей виявлення фейкового контенту	01.03 – 15.03.24	*виконано*
4	Планування експериментів	16.03 – 31.03.24	*виконано*
5	Імплементація моделей виявлення фейкового контенту	01.04 – 10.04.24	*виконано*
6	Експериментальні дослідження	10.04 – 15.04.24	*виконано*
7	Аналіз результатів експериментальних досліджень	15.04 – 18.04.24	*виконано*
8	Написання та оформлення статті та тез доповіді	15.04 – 30.04.24	*виконано*
9	Підготовка пояснювальної записки	01.05 – 20.05.24	*виконано*
10	Підготовка презентації та доповіді	21.05 – 05.06.24	*виконано*
11	Нормоконтроль	05.06 – 08.06.24	*виконано*
12	Рецензування	08.06 – 13.06.24	*виконано*
13	Занесення диплома в електронний архів	14.06 – 15.06.24	*виконано*
14	Попередній захист	15.06.2024	*виконано*
15	Допуск до захисту у зав. кафедри	16.06.2024	*виконано*

Дата видачі завдання 22 січня 2024р.

Студент  Кіценко Ю.О.
(підпис)

Керівник роботи _____ проф. Смеляков К.С.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Пояснювальна записка містить 73 стор., 19 рис., 1 табл., 49 джерел.

ЗАМІНА ІДЕНТИЧНОСТІ, МАШИННЕ НАВЧАННЯ, МАНІПУЛЯЦІЇ З ОБЛИЧЧЯМИ, МОДЕЛІ ВИЯВЛЕННЯ ФЕЙКІВ, ФЕЙКОВИЙ КОНТЕНТ.

Об’єкт дослідження – моделі виявлення фейкового контенту з заміною ідентичності на базі згорткових нейронних мереж.

Мета роботи – аналітичний огляд існуючих моделей виявлення фейкового контенту з заміною ідентичності та наборів даних, які використовуються для виявлення маніпуляцій з заміною ідентичності; побудова та навчання декількох моделей виявлення фейкового контенту на основі нейронних мереж.

Результат роботи – проведено огляд наборів даних маніпуляцій з заміни ідентичності та моделей виявлення фейкового контенту. Виконано навчання декількох моделей виявлення фейкового контенту на основі згорткових нейронних мереж. Навчання проводилося за допомогою сучасного набору даних Deepfake Detection Challenge Dataset.

DEEPFAKE, MACHINE LEARNING, DEEPFAKE DETECTION MODELS, FACE MANIPULATIONS, IDENTITY SWAP.

Object of research – deepfake detection models based on convolutional neural networks.

The aim of the work – analytical review of existing deepfake detection models and identity swap datasets; creation and training of several deepfake detection models based on convolutional neural networks.

Main results – the review of identity swap datasets and deepfake detection models has been conducted. We have built and trained several deepfake detection models based on convolutional neural networks. The training was performed using Deepfake Detection Challenge Dataset.

Я, Кіценко Юрій Олександрович, студент гр. ІПЗм-22-4, здобувач вищої освіти на другому (магістерському) рівні кафедри «Програмна інженерія», заявляю: моя кваліфікаційна робота на тему «Дослідження моделей виявлення фейкового контенту на основі технологій машинного навчання», що буде представлена в екзаменаційну комісію для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElArKhNURE. Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

AE – Автоенкодер (Autoencoder)

AP – Середня точність (Average Precision)

AUC – Площа під кривою ROC (Area Under the ROC Curve)

BCE – Перехресна ентропія при бінарній класифікації (Binary Cross-Entropy)

CNN – Згорткова нейронна мережа (Convolutional Neural Network)

DFDC –Deepfake Detection Challenge

DSP-FWA – Dual Spatial Pyramid for Exposing Face Warp Artifacts

FC – Face Cutout

FWA – Артефакти деформації обличчя (Face Warp Artifacts)

GAN – Генеративно-змагальна мережа (Generative Adversarial Network)

HQ – Висока якість (High Quality)

LQ – Низька якість (Low Quality)

NS – Noisy Student

ROC – Характеристика роботи приймача (Receiver Operating Characteristic)

SGD – (Stochastic Gradient Descent)

SPP – Просторовий пірамідальний пуллінг (Spatial Pyramid Pooling)

SVM – Машина опорних векторів (Support Vector Machine)

ВСТУП

Поява та розповсюдження фейкового контенту в інтернеті є складним явищем, яке має все більший вплив на суспільство, політику та економіку. Зростання доступності технологій глибокого навчання та штучних нейронних мереж робить можливим виготовлення високоякісних фейків. З'явилось багато потужних інструментів, що дозволяють змінювати фотографії, створювати відео з фіктивним контентом і навіть генерувати тексти. Алгоритми соціальних мереж та платформ для рекомендацій контенту сприяють поширенню фейкової інформації. Контент, який викликає емоційну реакцію або підтримку, має великий шанс стати вірусним, незалежно від його достовірності.

В наші дні інформація поширюється в інтернеті дуже швидко, здебільшого через соціальні мережі, блоги та інші платформи. Це дозволяє фейковому контенту вплинути на громадську думку вже до того, як буде виявлено його недостовірність. Різні групи людей та окремі персони можуть створювати фейковий контент за різних мотивів, як політичні агенди, заробіток, псевдожартівливі цілі або намагання вплинути на громадську думку. Використання ботів та автоматизованих систем дозволяє швидко розповсюджувати фейковий контент та створювати враження широкої підтримки чи обговорення в інтернет-середовищі.

Поява моделей виявлення фейкового контенту є відповіддю на зростаючу загрозу фальсифікації інформації в цифровому просторі. З огляду на швидкі та значні зміни в технологіях створення фейків, виникає потреба у високоефективних інструментах та методах для їх виявлення. Тут важливу роль відіграють моделі, засновані на технологіях машинного навчання.

З розвитком глибокого навчання та штучних нейронних мереж моделі виявлення фейкового контенту стають більш потужними та адаптуючись до нових форм фальсифікації. Вони базуються на алгоритмах, які можуть вивчати та розпізнавати патерни у великих обсягах даних, виявляючи аномалії та неправдивості. Ці моделі використовуються для аналізу різноманітних типів

контенту, таких як текст, фотографії та відео. Вони можуть виявляти неправдиві інформаційні статті, графічні обробки чи генеровані штучним інтелектом зображення, а також фейкові відеоматеріали.

Одним із важливих напрямків є використання моделей, які враховують контекст, емоційні аспекти та суб'єктивні нюанси в тексті та мультимедійних даних. Це допомагає уникнути виявлення як фейку легітимної інформації, яка може мати емоційний або суб'єктивний характер.

Загалом, розробка та удосконалення моделей виявлення фейкового контенту залишається важливою актуальною задачею для забезпечення довіри до інформації в цифровому світі, де маніпуляція та розповсюдження фейків стає все більш поширеним явищем. З огляду на вищесказане, дослідження моделей виявлення фейкового контенту на основі технологій машинного навчання є актуальною темою сьогодення.

Метою даного дослідження є проведення аналітичного огляду сучасних моделей виявлення фейкового контенту з заміною ідентичності та відповідних наборів даних, які використовуються для виявлення маніпуляцій з заміною ідентичності; побудова та навчання декількох моделей виявлення фейкового контенту на основі нейронних мереж; порівняння отриманих результатів з даними інших досліджень. Основною задачею дослідження відповідно є створення моделі виявлення фейкового контенту, що відповідає сучасним викликам цифрового світу.

Об’єкт дослідження є моделі виявлення фейкового контенту з заміною ідентичності на базі згорткових нейронних мереж. Предметом дослідження є аналіз та оцінка ефективності згорткових нейронних мереж у виявленні фейкового контенту із маніпуляціями з заміною ідентичності.

При проведенні дослідження було застосовано методи загальної логіки, теоретичні методи та емпіричні методи. Методи загальної логіки допомогли провести аналіз предметної області, теоретичні методи були використані під час аналізу нейронних мереж, тоді як емпіричні методи використовувалися під час проведення експериментального дослідження.

Отримані результати можуть бути використані можуть бути застосовані у боротьба з дезінформацією та фейками, захисті персональної ідентичності, підвищенні довіри до цифрових медіа.

Результати кваліфікаційної роботи було представлено на 28-му Міжнародному молодіжному форумі «Радіоелектроніка та молодь у XXI столітті» та 8-ій Міжнародній конференції “Electrical, Electronic and Information Sciences” (Estream 2024). За результатами доповіді на Міжнародному молодіжному форумі отримано диплом за кращу доповідь секції (наведено у Додатку В) та опубліковано тези доповіді [1]. Результати доповіді на конференції eStream 2024 опубліковані в роботі [2].

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Визначення фейкового контенту

Фейковий контент – це вид інформації, яка створена, розповсюджена або представлена з наміром обдурити аудиторію, вводячи в оману щодо її джерела, автентичності, достовірності або контексту. Це може включати в себе неправдиві новини, маніпулювання фотографіями чи відео, текстові повідомлення, які спотворюють факти або інсценують події, які насправді не відбулися.

Фейкові зображення та відео, створені за допомогою цифрових маніпуляцій, зокрема за допомогою методів DeepFake [3]-[9], нещодавно стали великою громадською проблемою [4]. Дуже популярний термін "DeepFake" відноситься до техніки на основі глибокого навчання, здатної створювати фальшиві відео шляхом заміни обличчя однієї людини на обличчя іншої. Цей термін з'явився після того, як користувач Reddit під ім'ям "deepfakes" заявив у кінці 2017 року, що розробив алгоритм машинного навчання, який допоміг йому перенести обличчя знаменитостей у порно відео [5]. Крім фальшивого порно, деякі з більш шкідливих використань такого фальшивого контенту включають фейкові новини, обмани та фінансові шахрайства.

В даній роботі нас будуть більш за все цікавити маніпуляції з обличчям. Традиційно, кількість та реалізм маніпуляцій з обличчям були обмежені через відсутність складних інструментів редагування, потребу в спеціальних знаннях та складний та часомісткий процес. Наприклад, раннє дослідження цієї теми [10] змінювало рух губ людини для використання іншої звукової доріжки, створюючи зв'язки між звуками аудіодоріжки та формою обличчя. Однак, від цих ранніх робіт до сьогодні, багато речей швидко еволюціонувало за останні роки. На сьогодні стає все більш легко автоматично синтезувати неіснуючі обличчя або маніпулювати реальним обличчям однієї людини на зображенні/відео, завдяки, по-перше, доступності великомасштабних публічних даних, та, по-друге, еволюції технік глибокого навчання, що замінюють багато ручних етапів редагування, таких як автоенкодери (AE) та генеративні змагальні мережі (GAN) [11], [12]. В

результаті, таке відкрите програмне забезпечення та мобільні додатки, як ZAO та FaceApp, відкривають двері для створення фальшивих зображень та відео будь-кому, без необхідного досвіду в цій сфері.

1.2 Типи маніпуляцій з обличчям

Найчастіше виділяють шість типів маніпуляцій з обличчям за їх методами та цілями:

- заміна ідентичності (Identity Swap). Маніпуляція, яка включає заміну обличчя однієї людини на обличчя іншої у відео або на фото;
- повний синтез обличчя (Entire Face Synthesis). Створення цілком нових, неіснуючих облич за допомогою генеративних алгоритмів, таких як GAN (генеративні змагальні мережі);
- зміна атрибутів (Attribute Manipulation). Зміни певних характеристик обличчя, як-от колір волосся, колір шкіри, риси обличчя, додавання чи видалення аксесуарів (окуляри, борода тощо);
- заміна виразу обличчя (Expression Swap). Модифікація виразів обличчя людини, наприклад, зміна емоційного виразу обличчя або рухів губ у відповідності до іншої аудіодоріжки;
- деідентифікація обличчя (Face De-Identification). Процес зміни зображення або відео обличчя таким чином, щоб унеможливити або ускладнити ідентифікацію оригінальної особи;
- морфінг обличчя (Face Morphing). Плавне змішування двох або більше облич для створення нового обличчя, яке містить риси обох або всіх вихідних осіб.

На рисунку 1.1 зображені основні чотири типи маніпуляцій з обличчям [6].

Розглянемо наведені типи маніпуляцій з обличчям більш детально.

Заміна ідентичності (Identity Swap) у контексті обробки зображень та відео є процесом, де обличчя однієї людини замінюється на обличчя іншої. Цей тип маніпуляції став особливо популярним і відомим через розвиток технологій DeepFake.

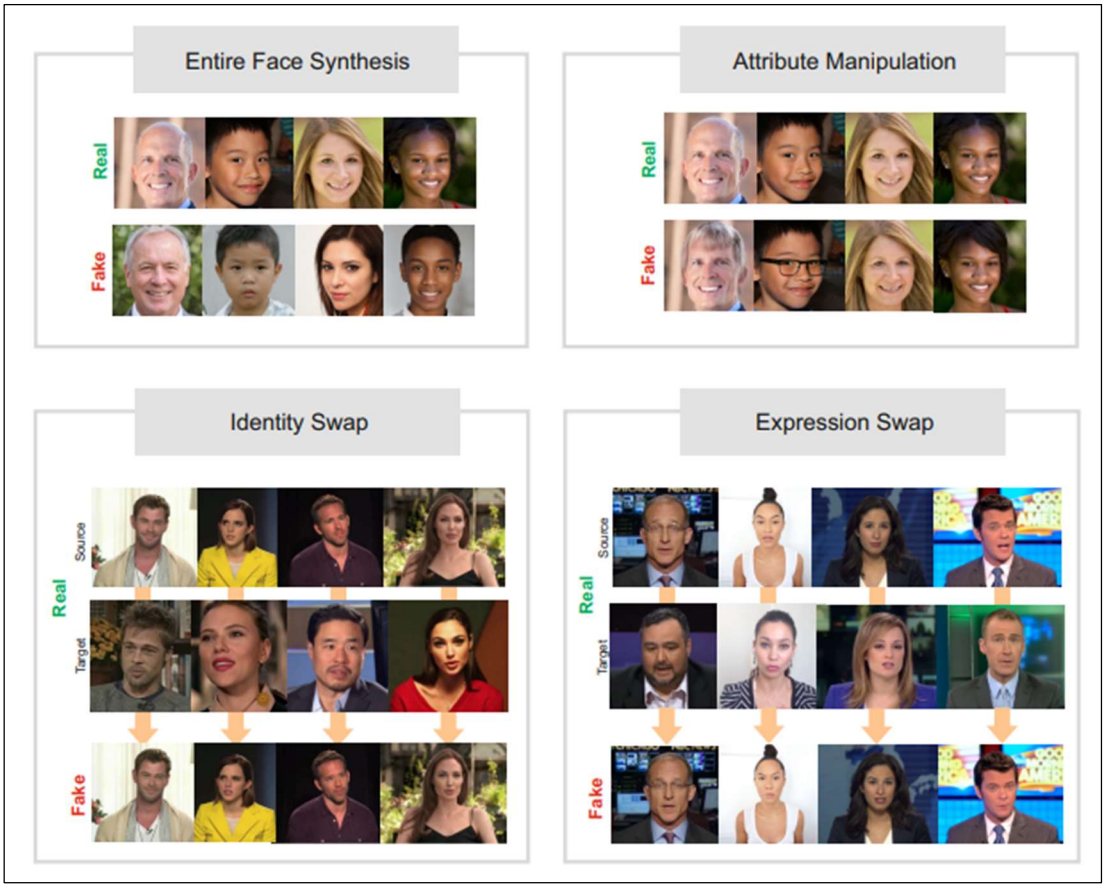


Рисунок 1.1 – Реальні та фейкові приклади основних типів маніпуляцій з обличчям (за даними [6])

Для заміна ідентичності частіш за все використовуються алгоритми глибокого навчання, такі як генеративні змагальні мережі (GAN) та автоенкодери. Ці моделі тренуються на великому наборі даних з обличчями для вивчення способів ефективної заміни обличь. Система спочатку виявляє обличчя на зображенні або відео, а потім аналізує його ключові точки (такі як очі, ніс, рот), щоб точно вирівняти нове обличчя з оригіналом. Нове обличчя накладається на оригінальне, з особливою увагою до вирівнювання рис, освітлення, кольору шкіри та текстур.

Маніпуляції такого типу може бути виконано за допомогою популярних доданків FaceSwap та ZAO . Цей тип маніпуляцій використовується у кіно та телебаченні для створення спецефектів. Також заміна ідентичності може використовуватися для створення політичних фейкових відео, порнографічних відео зі знаменитостями та ін.

Повний синтез обличчя (Entire Face Synthesis) – це процес створення повністю нових, неіснуючих облич за допомогою алгоритмів глибокого навчання, зокрема, за допомогою генеративних змагальних мереж (GANs), зокрема, за допомогою нещодавно розробленої мережі StyleGAN [13]. У наші часи GANs здатні генерувати високоякісні, реалістичні зображення облич, які дуже важко відрізнити від справжніх фотографій. Синтезовані обличчя можуть мати різноманітні риси, такі як різні типи шкіри, риси обличчя, зачіски, вирази обличчя тощо.

Цей тип маніпуляцій використовується у відеоіграх, фільмах та інших медіа для створення унікальних персонажів. За допомогою цієї техніки створюються набори даних з різноманітними обличчями для тренування та тестування алгоритмів розпізнавання облич, вигадані персонажі для рекламних кампаній. Але також цей тип маніпуляцій може бути використаним для шкідливих застосувань, таких як створення дуже реалістичних фейкових профілів у соціальних мережах.

Зміна атрибутів (Attribute Manipulation) в області обробки зображень та комп'ютерного зору є процесом модифікації певних характеристик або рис обличчя на зображенні або відео. Цей процес включає широкий спектр можливих змін: модифікація кольору волосся або очей людини, вікові зміни, модифікація форми або розміру рис обличчя, таких як ніс, губи, щоки, додавання або видалення аксесуарів (окулярів, шапок, сережок тощо), модифікація відтінку шкіри.

Для зміни атрибутів обличчя використовуються GAN, такі як StarGAN, оскільки вони можуть генерувати реалістичні зображення. Використовується програмне забезпечення, таке як Adobe Photoshop, яке дозволяє редагувати та змінювати риси обличчя вручну. Існують популярні додатки, як FaceApp, що надають користувачам можливість змінювати риси обличчя за допомогою кількох натискань.

Цей тип маніпуляцій застосовується для створення забавних або змінених зображень для публікації в соціальних мережах, у моделюванні та графічному дизайні для візуалізації змін у вигляді персонажів, дослідження впливу зміни

атрибутів на системи розпізнавання обличчя.

Заміна виразу обличчя (Expression Swap) – це технологія, яка дозволяє змінити або замінити вирази обличчя людини в зображеннях або відео. Цей процес став можливим завдяки розвитку технологій глибокого навчання і комп'ютерного зору.

Під час створення таких маніпуляцій, як і у випадку заміни ідентичності, спочатку система ідентифікує обличчя на зображенні або відео і визначає ключові риси, такі як очі, ніс, рот, які важливі для виразів обличчя. Далі аналізуються поточні вирази обличчя, включаючи емоції та рухи м'язів. Після цього вирази обличчя змінюються або замінюються на інші, що може включати зміну емоцій, міміки, руху губ тощо. У створенні маніпуляцій такого типу використовуються GAN, а також техніки Face2Face та NeuralTextures [15, 16].

Заміна виразу обличчя застосовується у розважальній індустрії для створення різноманітних виразів обличчя персонажів, вивчення виразів обличчя у аналізі емоцій та поведінки людини. Також цей тип маніпуляції може бути використаний і з більш серйозними наслідками, як наприклад, популярне відео Марка Цукерберга, де він говорить речі, які ніколи не насправді не говорив .

Деідентифікація обличчя (Face De-Identification) – це процес модифікації зображення або відео так, що впізнати конкретну особу було важко або неможливо. Цей процес включає зміну або маскування ключових рис обличчя, зберігаючи при цьому його загальну структуру та вирази, але роблячи ідентифікацію оригінальної особи значно складнішою. Це може бути здійснено за допомогою застосування цифрових фільтрів, зміни особливостей обличчя або навіть заміною обличчя на повністю синтезоване. Цей процес стає все більш важливим у контексті приватності та безпеки даних, оскільки він дозволяє захистити особисту ідентичність людини в цифровому просторі.

Цей тип маніпуляцій застосовується для захисту ідентичності людей у новинах або документальних матеріалах, в наукових дослідженнях, де необхідно зберігати анонімність учасників, у сферах, де важливо зберегти приватність особистих даних, наприклад у банківській справі або охороні здоров'я.

Морфінг обличчя (Face Morphing) – це процес створення плавного переходу або змішування між двома або більше обличчями для створення нового образу, який поєднує риси вихідних облич. Процес морфінгу обличчя починається з вибору двох або більше облич, які будуть використані для створення морфінгу. Система визначає ключові точки на обличчях, такі як куточки очей, носа, рота та інші відмітні риси. Обличчя вирівнюються за розміром та орієнтацією для забезпечення плавного переходу між ними. Далі алгоритми морфінгу плавно змішують риси вихідного обличчя, створюючи нове обличчя з комбінованими характеристиками.

Ця техніка широко використовується у кіно та анімації для створення спецефектів, де персонажі перетворюються з однієї особи на іншу, у графічному дизайні для створення унікальних портретів або арт-проектів.

Відзначимо, що на сьогоднішній день заміна ідентичності, це один з найпопулярніших напрямків досліджень маніпуляцій з обличчям, насамперед це пов’язано зі значними громадськими занепокоєннями навколо DeepFakes [4], які вже згадувались нами раніше. Тому фокус цього проекту буде спрямовано саме на цей тип маніпуляцій та боротьбі з ними.

1.3 Постановка задачі кваліфікаційної роботи

Як випливає з усього сказаного вище об'єктом нашого дослідження є фейковий контент у цифровому середовищі. Хоча це поняття охоплює широкий спектр матеріалів, таких як текстові повідомлення, зображення, відео та аудіофайли, які були сфабриковані або змінені з метою введення в оману або дезінформації, ми сфокусуємо нашу увагу на фейковому контенті, створеному за допомогою методів машинного навчання, включаючи відео та зображення з маніпуляціями заміни ідентичності.

Основною метою даного дослідження є ознайомлення з існуючими методами, моделями та наборами даних, які використовуються для виявлення маніпуляцій з заміною ідентичності, їх всебічний аналіз та оцінка, та створення, та навчання власних моделей виявлення фейкового контенту. Ця мета включає

кілька ключових аспектів.

Огляд існуючих наборів даних, які використовуються при створенні моделей виявлення маніпуляцій із заміною ідентичності. Оскільки у контексті нашого дослідження нас цікавить підхід, заснований на машинному навчанні, є вкрай важливим знайти розмічені набори даних, які можуть бути використані, як під час навчання своїх моделей, так і під час тестування, як своїх так і інших існуючих моделей.

Огляд існуючих моделей виявлення маніпуляцій із заміною ідентичності. Буде проведено вивчення сучасних робіт, опублікованих у наукових журналах, конференціях та інших відомих публікаціях щодо сучасних моделей виявлення маніпуляції із заміною ідентичності, проведено їх порівняльний аналіз та зроблено оцінку ефективності. При цьому також буде проведено ознайомлення з новими дослідженнями та розробками у галузі, з метою виявлення потенційних інноваційних підходів та технологій, які можуть вдосконалити процес виявлення обраного типу маніпуляцій.

Навчання декількох моделей згорткових мереж виявлення фейкового контенту. Буде виконано навчання декількох моделей згорткових нейронних мереж для виявлення фейкового контенту. Планується застосування різноманітних архітектур згорткових мереж, таких як ResNet, EfficientNet та Xception, що дозволить глибше проаналізувати та розпізнавати складні патерни, які можуть вказувати на фальсифікацію зображень або відео. Кожна з цих моделей має унікальні характеристики та підходи до обробки даних, що разом забезпечує більш всебічне та ефективне виявлення фейків.

Цілі цього дослідження перш за все спрямовані на поглиблення розуміння поточного стану в області виявлення маніпуляції із заміною ідентичності за допомогою машинного навчання, а також на підвищення обізнаності щодо викликів та можливостей, які стоять перед науковою спільнотою у цьому напрямку.

2 ОГЛЯД МОДЕЛЕЙ ТА НАБОРІВ ДАНИХ ДЛЯ ВИЯВЛЕННЯ ФЕЙКОВОГО КОНТЕНТУ

2.1 Початковий етап розвитку

Маніпуляції типу "Identity Swap" (заміна ідентичності) привертають значну увагу як серед дослідників, так і в широкої громадськості через свої потенційні впливи на інформаційну безпеку та соціальну довіру [1]-[5]. Технологія, що дозволяє замінювати обличчя людей у відео або на зображеннях, відкриває двері для різноманітних застосувань: від розваг та мистецтва до створення фейкових новин або дезінформаційних кампаній. Ця цікавість посилюється можливістю створення відео, які на перший погляд виглядають переконливо реалістично, що ставить під сумнів автентичність цифрового контенту. Однак, разом із технологічними можливостями, "Identity Swap" порушує важливі питання про етичні межі використання таких маніпуляцій, вимагаючи розвитку нових методів виявлення та верифікації цифрового контенту, щоб зберегти довіру в еру цифрових технологій.

Розглянемо, як розвивалися методи виявлення маніпуляцій із заміною облич, в контексті появи нових відповідних наборів даних. Одним з перших публічно доступних наборів даних, згідно [6], з оригінальними та фейковими відео з маніпуляціями із заміни обличчя є UADFV датасет [7]. У цьому наборі даних присутні 49 оригінальних відео, що послужили основою для створення такої ж кількості Deep Fake роликів. В середньому, довжина кожного відео складає близько 11.14 секунди, при цьому стандартна роздільна здатність становить 294×500 пікселів. Цей набір даних вперше використовувався у роботі [8]. Розвинутий там підхід був заснований на виявленні моргання очима у відеороликах, що є фізіологічною ознакою, яка слабо представлена у синтетичних фейкових відео. Цей метод було випробувано авторами на стандартних наборах даних для детекції моргання очима. Також він продемонстрував перспективні результати при виявленні відеороликів, створених за допомогою технології DeepFake.

Наступним кроком в області виявлення маніпуляцій із заміною ідентичності була робота Коршунова та Марселя [9]. У цьому дослідженні було представлено базу даних DeepfakeTIMIT з 620 фейковими відео, де було здійснено заміну облич за допомогою технології, заснованої на генеративно-суперницьких мережах (GAN)[1]. Для створення бази було відібрано 16 пар осіб із схожими обличчями з відкритої бази даних VidTIMIT. Для кожного із 32 учасників були розроблені дві моделі: модель низької якості (LQ) із розміром зображення 64×64 та модель високої якості (HQ) із розміром 128×128. Враховуючи наявність 10 відеороликів на кожного учасника в базі даних VidTIMIT, було створено 320 відеороликів для кожної версії, загалом утворивши 620 відео із заміненими обличчями. У роботі було показано, що алгоритми розпізнавання обличчя на основі VGG та Facenet не підходять для виявлення маніпуляцій з обличчями. Також було оцінено кілька базових алгоритмів виявлення заміни облич, і показано, що підхід на основі синхронізації руху губ не дає змоги виявити невідповідності між рухом губ і аудіо супроводженням. В той же час автори показали, що підхід, заснований на мірах якості зображення з використанням SVM класифікатора, може виявляти фейкові відео з EER 3,3% та 8,9% для LQ та HQ сценаріїв відповідно.

У роботі [14] було розроблено автоматизований бенчмарк для виявлення маніпуляцій з обличчям. Розроблений бенчмарк базується на DeepFakes[1], Face2Face [15], FaceSwap[2] та NeuralTextures [16] засобах маніпуляції з обличчям при різних розмірах та рівнях компресії. Бенчмарк є публічно доступним. Він містить прихований тестовий набір даних, а також базу даних з понад 1.8 мільйона маніпульованих зображень. Цей датасет отримав назву FaceForensics++ та є більшим на порядок, ніж попередні публічно доступні подібні датасети. На основі цих даних автори провели аналіз 7 детекторів фальсифікацій [17]-[22]. Під час бенчмарку найкращі результати отримала модель XceptionNet [17], яка є традиційною CNN на основі роздільних конволюцій з резидуальними

[1] https://github.com/deepfakes/faceswap
[2] https://github.com/ MarekKowalski/FaceSwap/

з'єднаннями. Модель було навчено на наборі даних ImageNet та адаптовано шляхом заміни останнього повністю з'єднаного шару на два виходи.

2.2 Другий етап розвитку

У 2019 році за підтримки Google було створено датасет DeepFakeDetection (Google DFD) [31]. Цей набір даних включає 363 реальні відео (28 акторів у 16 різних сценах). Також він включає 3068 фальшивих відео, створених, як і у попередньому випадку, на основі реалізації DeepFake FaceSwap. Пізніше цей датасет було включено до бази даних FaceForensics++.

У роботі [24] було представлено новий великомасштабний датасет фейкових відео, CelebDF, який містить 5 639 високоякісних фейкових відео знаменитостей, створених за допомогою вдосконаленого процесу синтезу. На основі представленого набору даних автори провели співставлення 9 методів виявлення фейкового контенту, які були доступні публічно або отримані безпосередньо від авторів [7], [14], [22], [25]-[30]. Під час проведення співставлення найкращим чином себе показали підходи Xception [14] та DSP-FWA [30]. Xception відповідає методу виявлення фейків на основі моделі XceptionNet [17], яка навчена на датасеті FaceForensics++. Використовувалися три варіанти Xception, а саме: Xception-raw, Xception-c23 та Xception-c40. Xception-raw навчені на необроблених відео, тоді як Xception-c23 та Xception-c40 навчені на відео в форматі H.264 з середнім (23) та високим (40) ступенем компресії відповідно. DSP-FWA є вдосконаленим методом на основі FWA, до складу якого входить модуль просторового пірамідального пуллінгу (SPP) [30], щоб краще обробляти варіації у роздільних здатностях оригінальних цільових облич. У роботі показано, що продуктивність розглянутих методів знижується разом із зростанням ступеня компресії. Зокрема, продуктивність FWA та DSP-FWA значно погіршується на рекомпресованому відео, тоді як продуктивність Xception-c23 та Xception-c40 не відчуває значного впливу. Це очікувано, оскільки останні методи навчались на компресованих H.264 відео, тому вони більш стійкі в цій ситуації.

2.3 Сучасний етап розвитку

Facebook у співпраці з Microsoft, Amazon, MIT та іншими, запустили наприкінці 2019 року новий виклик під назвою Deepfake Detection Challenge (DFDC) [31]. Спочатку вони випустили демонстраційний набір даних, що складається з 1,131 реальних відео (66 акторів) та 4,119 фальшивих відео. Пізніше був випущений повний набір даних DFDC [32], який містить понад 100 000 відеокліпів з участю 3 426 оплачених акторів (470 ГБ контенту). Ці відеокліпи були створені за допомогою Deepfake, GAN та нелінійних методів навчання.

Так у роботі [33] було представлено великий тестувальний набір даних для виявлення фальсифікацій обличчя. Перша версія цього набору, DeeperForensics-1.0, є одним з найбільших на сьогодні набором даних для виявлення фальсифікацій обличчя, що складається з 60 000 відео та 17,6 мільйонів кадрів, що у 10 разів перевищує розмір існуючих наборів даних такого типу. Фейкові відео генерувалися за допомогою нової системи обміну обличчями[3]. Крім того, в роботі було проведено всебічне дослідження, яке оцінює п'ять базових методів виявлення [17], [34]-[37].

Зазначимо, що з моменту появи перших баз даних у цьому списку до появи останніх, відбулися значні візуальні поліпшення їх наповнення та суттєво було збільшено реалізм фейкових відео. В результаті, бази даних із маніпуляціями з заміни ідентичності часто поділяються на декілька генерацій (див. наприклад [6]). При цьому UADFV, DeepfakeTIMIT та FaceForensics++ відносять до першої генерації, CelebDF, Google DFD, DFDC Preview – до другої, а DeeperForensics і DFDC – до третьої (див. рис. 2.1).

Останнім часом з’явилось ще декілька наборів даних для виявлення маніпуляцій з обличчям [38]. У роботі [39] було запропоновано набір даних WildDeepfake для виявлення глибоких фейків. WildDeepfake складається з 1 180 099 високоякісних зображень 7 314 послідовностей облич, витягнутих з 707

[3] https://github.com/EndlessSora/DeeperForensics-1.0

відео (як глибоких фейків, так і реальних), зібраних з веб-ресурсів, тому містить більше різноманітних сцен, облич і дій. Також у цьому дослідженні було запропоновано дві мережі виявлення глибоких фейків на основі уваги (ADDNets).

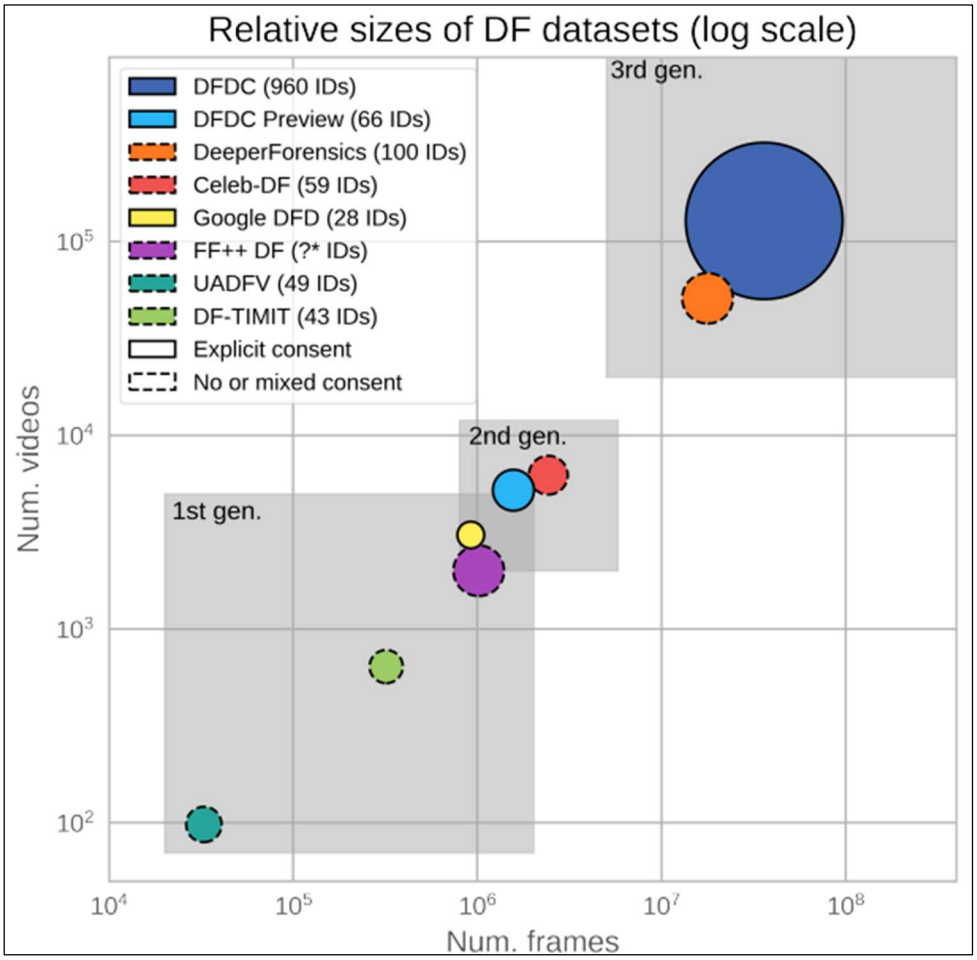


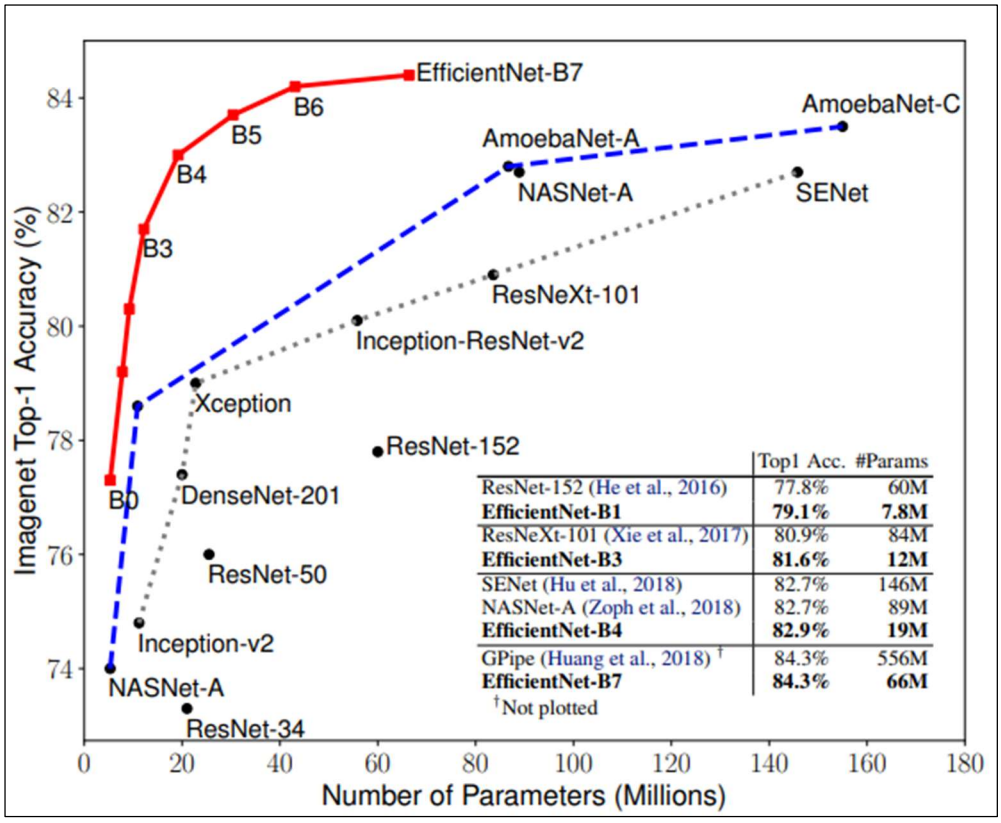
Рисунок 2.1 – Генерації наборів даних з маніпуляціями типу заміни ідентичності (за даними [32]). Осі показано в логарифмічному масштабі. Розміри кіл візуалізують кількість фейків, наявних у наборі даних.

У [40] було представлено корейський набір даних для виявлення глибоких фейків (KoDF). Він містить 175,776 фальшивих відеокліпів та 62,166 реальних відеокліпів з 403 суб'єктів. Фейкові відео було створено за допомогою шести різних моделей синтезу. Іншим цікавим набором даних є ForgeryNet [41].

3 ОПИС ПРИЙНЯТИХ ПРОЕКТНИХ РІШЕНЬ

3.1 Вибір архітектури моделей виявлення фейкового контенту

Ми обрали три моделі згорткових нейронних мереж для проведення досліджень EfficientNet [47], XceptionNet [14] та ResNet [37]. EfficientNet являє собою сімейство глибоких нейронних мереж, які відрізняються за розмірами та складністю архітектури (див. рис. 3.1). Найпопулярнішими є моделі від EfficientNet-B0 до EfficientNet-B7, де число в кінці назви вказує на рівень масштабування моделі. Наприклад, EfficientNet-B0 має найменшу кількість параметрів, тоді як EfficientNet-B7 є найбільшою та найскладнішою моделлю з великою кількістю параметрів.



	Top1 Acc.	#Params
ResNet-152 (He et al., 2016)	77.8%	60M
EfficientNet-B1	**79.1%**	**7.8M**
ResNeXt-101 (Xie et al., 2017)	80.9%	84M
EfficientNet-B3	**81.6%**	**12M**
SENet (Hu et al., 2018)	82.7%	146M
NASNet-A (Zoph et al., 2018)	82.7%	89M
EfficientNet-B4	**82.9%**	**19M**
GPipe (Huang et al., 2018) †	84.3%	556M
EfficientNet-B7	**84.3%**	**66M**

†Not plotted

Рисунок 3.1 – Залежність розміру моделей від точності (за даними [47])

Кожна з моделей EfficientNet має відповідність між глибиною мережі, шириною та роздільною здатністю (див. рис. 3.2). Зазвичай, більші моделі мають більше шарів, більшу ширину та вищу роздільну здатність, що дозволяє їм вирішувати більш складні завдання, але при цьому вони вимагають значного

обсягу обчислювальних ресурсів для навчання та використання. Крім того, EfficientNet також включає в себе варіації моделей, такі як EfficientNet-Lite, які оптимізовані для використання на мобільних пристроях з обмеженими ресурсами та інші моделі, які можуть бути використані для конкретних завдань та обсягів даних. Ми проводили тренування з трьома розмірами архітектури B4, B5 та B7.

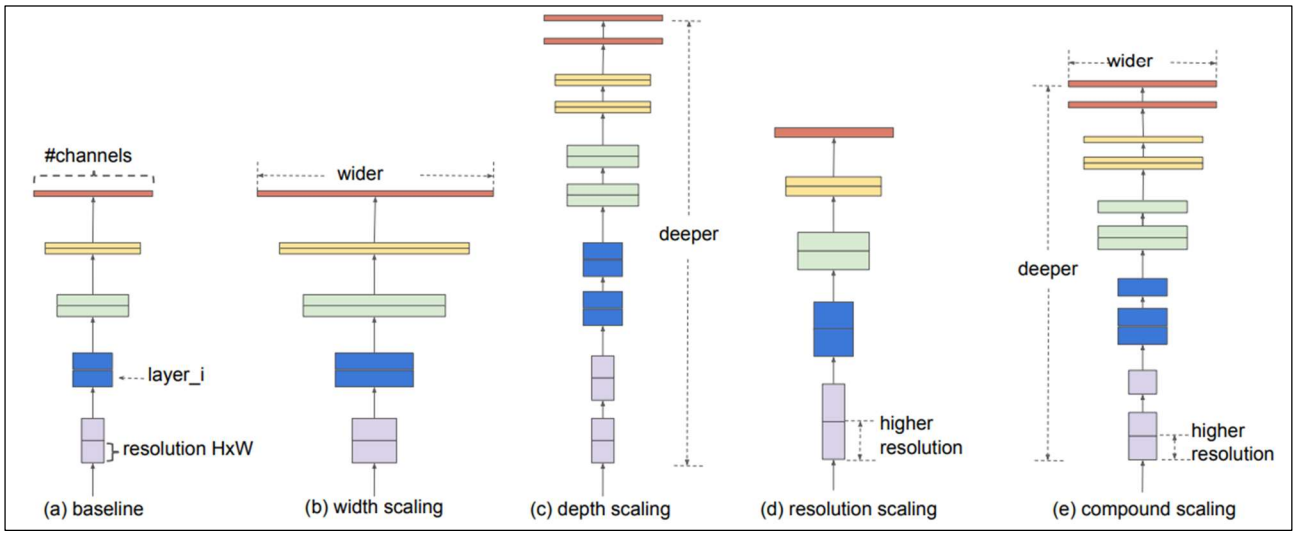


Рисунок 3.2 – Масштабування архітектури EfficientNet: (a) базова мережа; (b)-(d) звичайне масштабування, яке збільшує ширину, глибину або роздільну здатність відповідно; (e) – комплексний метод масштабування, який рівномірно масштабує всі три виміри з фіксованим співвідношенням (за даними [47])

Другою з обраних архітектур є XCeptionNet (див. рис. 3.3). Ця архітектура являє собою вдосконалення концепції Inception, запропонованої в Google InceptionV3. Основна ідея XCeptionNet полягає в застосуванні групи крос-канальних і групових згорток, що робить мережу ефективнішою та легшою для навчання.

Особливість XCeptionNet полягає у використанні групових згорток для зменшення кількості параметрів та обчислювальної складності, а також в розділенні каналів, що дозволяє отримувати незалежні згортки для кожного вхідного каналу. Ці особливості роблять XCeptionNet ефективним використанням обчислювальних ресурсів та дозволяють досягати високої точності на різних завданнях у сфері комп'ютерного зору.

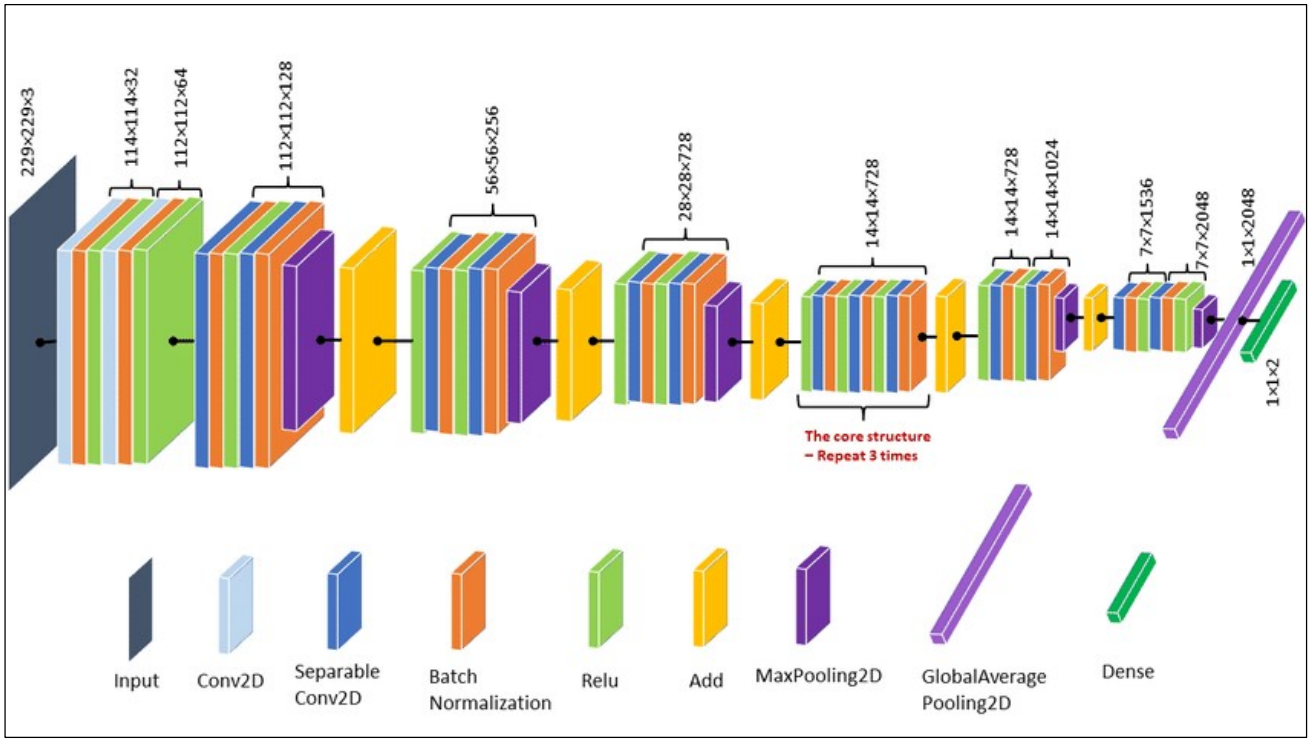


Рисунок 3.3 – Архітектура моделі XCeptionNet (за даними [48])

Остання обрана для досліжнення архітектура – ResNet (Residual Neural Network) (див. рис. 3.4). Ця архітектура вирізняється використанням з'єднань, що пропускають дані, або так званих "залишкових блоків". Ця архітектура є однією з найбільш впливових моделей у галузі комп'ютерного зору. Основна ідея ResNet полягає в тому, щоб додавати вхідні дані до виходу кожного блоку замість заміни їх новими обчисленнями, що дозволяє зменшити проблему зникнення градієнту та сприяє збереженню інформації під час навчання глибоких мереж.

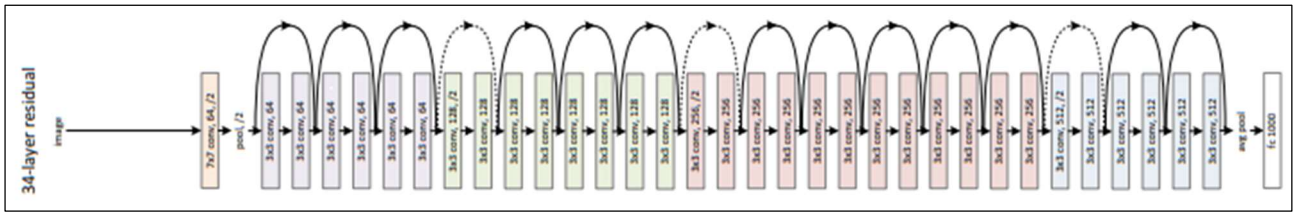


Рисунок 3.4 – Архітектура моделі ResNet (за даними [37])

ResNet використовує також "скорочені з'єднання", які дозволяють пропускати деякі шари мережі, забезпечуючи більш ефективне та швидке навчання. Ця архітектура дозволяє тренувати глибокі нейронні мережі з

десятками, навіть сотнями, шарів, забезпечуючи високу точність на різноманітних завданнях комп'ютерного зору, включаючи класифікацію зображень, виявлення об'єктів та семантичну сегментацію.

3.2 Метрики якості навчання моделей

В якості метрик, що обрані для визначення якості тренування моделей використовувались такі величини: BCE, AUC та AP.

BCE (Binary Cross-Entropy) - це метрика для вимірювання точності класифікаційних моделей. Для отримання значення BCE використовується формула 3.1:

$$BCE = -\frac{1}{n}\sum_{i=1}^{n}\{y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)\} \tag{3.1}$$

де n – кількість зразків у тестовому наборі даних,

y_i – фактичний клас зразка i,

$\hat{y}_i$ – передбачена ймовірність того, що зразок i належить до класу 1.

AUC (Area Under the Curve) – це метрика, що використовується для оцінки якості моделей класифікації, зокрема їхньої здатності розрізняти між позитивними та негативними класами. AUC вимірює площу під кривою ROC (Receiver Operating Characteristic), яка відображає відношення між чутливістю (True Positive Rate) та специфічністю (False Positive Rate) моделі при різних порогах відсічення.

Крива ROC представляє собою лінію, яка показує, як змінюється чутливість моделі в залежності від специфічності при різних значеннях порогу відсічення. Чим ближче AUC до 1, тим краще модель розділяє класи, тобто чим більше площа під кривою ROC, тим краще її робота. AUC може приймати значення в діапазоні від 0 до 1, де значення ближче до 1 вказує на кращу якість моделі, а значення навколо 0.5 вказує на випадковий класифікатор.

AP (Average Precision) – це показник, що узагальнює криву влучності-

повноти (precision-recall curve) як середнє значення влучності, досягнуте на кожному порозі, зі збільшенням повноти від попереднього порогу, яке використовується як вага (формула 3.2):

$$AP = -\frac{1}{n}\sum_{i=1}^{n}(R_i - R_{i-1})P_i, \tag{3.2}$$

де P_i та R_i – це точність та повнота на i-ому порозі.

4 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

4.1 Опис засобів програмної реалізації

При побудові моделі виявлення маніпуляцій з заміною ідентичності, використовувались наступні засоби:

Мова програмування. Для програмної реалізації використовувалась мова Python. Ця мова програмування у наші часи є найпотужнішим інструментом розробки моделей машинного навчання через свою гнучкість, читабельність та широку підтримку бібліотек.

Інструменти для обробки даних та детектори облич. Бібліотека OpenCV використовувалася для файлових операцій із зображеннями та відео. Фреймворки Pandas та NumPy застосовувалися для аналізу та виконання операцій з даними. Особливо корисні для обробки та перетворення даних перед тренуванням моделі. Детектор облич MTCNN використовувався для виявлення облич та визначення їх характеристик, таких як положення очей, носа, рота та контурів обличчя.

Засоби аугментації даних. Для збільшення різноманітності навчальних даних шляхом аугментації вихідних зображень використовувалась бібліотека Albumentations.

Фреймворки машинного навчання. Для машинного навчання використовувався фреймворк PyTorch. Цей фреймворк є стандартом для побудови глибоких нейронних мереж, забезпечуючи потужні можливості для тренування та оцінки моделей. Для попередньої обробки даних та оцінки моделей використовувався пакет Scikit-learn.

Інструменти візуалізації. Для візуалізації даних використовувалась бібліотека Matplotlib.

Набори даних для навчання та тестування. Для навчання та тестування моделей використовувався набор даних DFDC, що включає як справжні, так і фейкові відео.

Система версіонування. В якості системи версіонування використовувався Git.

Обчислювальні ресурси. Для тренування моделей використовувався ПК на базі на процесора Intel Core i7 13700K з 16 ядрами (24 логічних процесора), 64 Gb оперативної пам’яті, двома NVMe SSD накопичувачами обсягом 2 Тб та GPU Nvidia RTX 3090 24 Gb.

Використання цих інструментів та технологій допомогло створити ефективну та достатньо точну модель для виявлення маніпуляцій з заміною ідентичності.

4.2 Підготовка даних для навчання моделі

Етап підготовки даних є критично важливим при побудові моделі виявлення маніпуляцій з заміни ідентичності. Підготовка даних має значний вплив на ефективність, точність та надійність кінцевої моделі. В якості основного набору даних ми будемо спиратися на DFDC [31]. Для підготовки даних ми будемо дотримуватися схеми роботи [42] і використовувати підхід на основі аналізу окремих кадрів з відеофайлів. Однією з головних переваг цього методу є його обмежена часова та обчислювальна спроможність, особливо при обробці відео високої роздільної здатності або великої тривалості. Серед недоліків зазначимо, що аналіз окремих кадрів може упустити контекстуальні або часові взаємозв'язки, які стають помітними лише при розгляді відео в цілому. Також цей підхід не дає можливості виявляти фейки у аудіо супроводженні, які хоч і існують у DFDC, але не враховувались фейками при проведенні конкурсу [31].

Для отримання окремих кадрів з відеофайлів використовувався клас MovieDataset, що реалізує абстрактний клас Dataset з пакету torch (див. рис. 4.1). Для захвату окремих відеокадрів використовуються можливості бібліотеки OpenCV, яка є дуже популярним інструментом у галузі обробки зображень та комп'ютерного зору через свою потужність, широкий функціонал та підтримку спільноти.

Отримані на першому етапі кадри аналізувались за допомогою детектора облич з метою знаходження обмежувальних рамок. Після аналізу низки детекторів облич (див. наприклад, [43], [44]) в якості детектора було обрано

MTCNN [45].

```
class MoviesDataset(Dataset):

    def __init__(self, movies) -> None:
        super().__init__()
        self.movies = movies

    def __len__(self) -> int:
        return len(self.movies)

    def __getitem__(self, index: int):
        movie = self.movies[index]
        movie_frames = OrderedDict()

        movie_capture = cv2.VideoCapture(movie)
        frames_count = int(movie_capture.get(cv2.CAP_PROP_FRAME_COUNT))

        for frame_index in range(frames_count):
            movie_capture.grab()
            success, image = movie_capture.retrieve()
            if not success:
                continue
            image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
            image = Image.fromarray(image)
            image = image.resize(size=[s // 2 for s in image.size])
            movie_frames[frame_index] = image
        return movie, list(movie_frames.keys()), list(movie_frames.values())
```

Рисунок 4.1 – Клас MovieDataset, що реалізує абстрактний клас Dataset з пакету torch

На рисунку 4.2 наведено основний метод процедури пошуку облич та генерації обмежувальних рамок. Наступним кроком за допомогою знайдених обмежувальних рамок було вирізано і сформовано набір облич, які в подальшому будуть використовуватись при навчанні CNN.

Після підготовки набору облич для забезпечення можливостей з доповнення тренувального датасету (додавання аугментованих зображень) ми знаходили особливі точки обличчя або орієнтири (landmarks). Процедура знаходження особливих точок обличчя наведена на рисунку 4.3.

```python
def find_face_boxes(movies, data_dir):
    face_detector = MtcnnFaceDetector(device=CUDA_DEVICE)
    movies_dataset = MoviesDataset(movies)
    loader = DataLoader(movies_dataset, num_workers=cpu_count() - 2,
                        batch_size=1, collate_fn=lambda x: x, shuffle=False)
    for item in tqdm(loader):
        result = {}
        movie, indices, frames = item[0]
        batches = [frames[i:i + face_detector._batch_size] for i in range(0, len(frames), face_detector._batch_size)]
        for j, frames in enumerate(batches):
            face_boxes = face_detector._detect_faces(frames)
            result.update({int(j * face_detector._batch_size) + i : bboxes for i, bboxes in zip(indices, face_boxes)})
        id = os.path.splitext(os.path.basename(movie))[0]
        face_boxes_dir = os.path.join(data_dir, FACE_BOXES_DIR)
        os.makedirs(face_boxes_dir, exist_ok=True)
        with open(os.path.join(face_boxes_dir, "{}.json".format(id)), "w") as f:
            json.dump(result, f)
```

Рисунок 4.2 – Основний метод процедури генерації обмежувальних рамок обличь

```python
def save_landmarks(movie_path, data_dir):
    movie_path = movie_path[:-4]
    movie_dir = os.path.join(data_dir, FACE_IMAGES_DIR, movie_path)
    face_landmarks_dir = os.path.join(data_dir, FACE_LANDMARKS_DIR, movie_path)
    faces_with_landmarks_dir = os.path.join(data_dir, FACES_WITH_LANDMARKS_DIR, movie_path)
    os.makedirs(face_landmarks_dir, exist_ok=True)
    for frame in range(320):
        if frame % 10 != 0:
            continue
        for actor in range(2):
            image_id = "{}_{}.png".format( *args: frame, actor)
            landmarks_id = "{}_{}".format( *args: frame, actor)
            original_movie_path = os.path.join(movie_dir, image_id)
            landmark_path = os.path.join(face_landmarks_dir, landmarks_id)
            face_with_landmarks_path = os.path.join(faces_with_landmarks_dir, image_id)

            if os.path.exists(original_movie_path):
                try:
                    image_ori = cv2.imread(original_movie_path, cv2.IMREAD_COLOR)[...,::-1]
                    frame_img = Image.fromarray(image_ori)
                    face_boxes, conf, landmarks = detector.detect(frame_img, landmarks=True)
                    if landmarks is not None:
                        landmarks = np.around(landmarks[0]).astype(np.int16)
                        np.save(landmark_path, landmarks)
                        if (DEMO_MODE):
                            save_image_with_landmarks(frame_img, face_boxes, landmarks, face_with_landmarks_path)
                except Exception as e:
                    print(e)
                    pass
```

Рисунок 4.3 – Процедура генерації особливих точок обличчя

Приклади облич з обмежувальними рамками та особливими точками зображені на рисунку 4.4.

Оскільки DFDC датасет містить оригінали, з яких було створено фейкові відео, самі фейкові відео та інформацію щодо оригіналу кожного фейкового відео, це дало змогу створити різницеві SSIM маски [46], що містять 1 для змінених

пікселів і 0 в іншому випадку. Ці маски було використано при генерації аугментацій, які ми розглянемо нижче. Приклади SSIM масок зображені на рисунку 4.5.

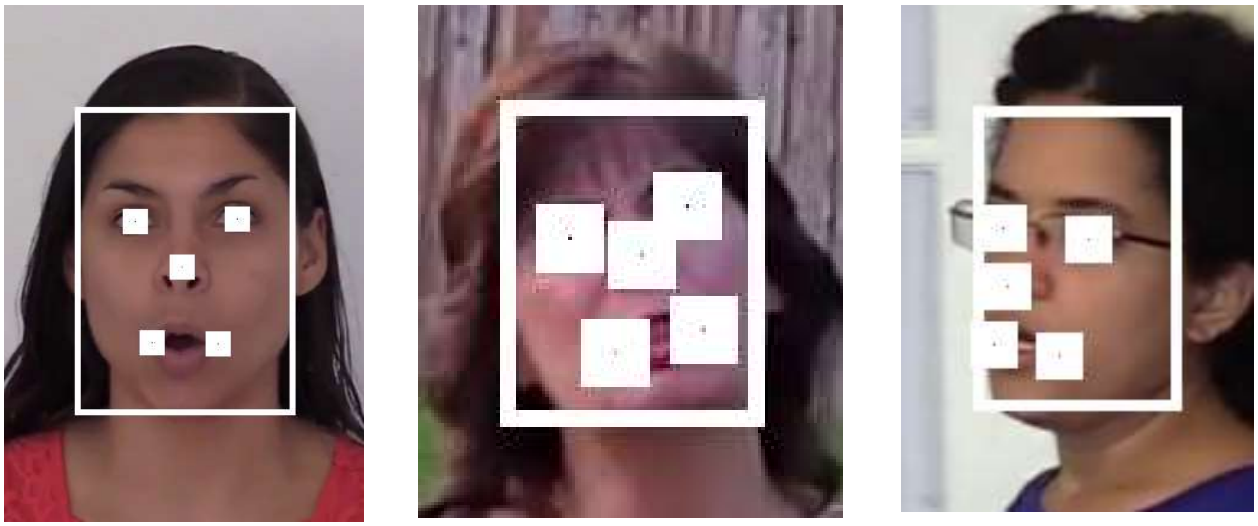


Рисунок 4.4 – Приклади облич з обмежувальними рамками та особливими точками

Рисунок 4.5 – Приклади різницевих SSIM масок

Завершальним кроком на стадії підготовки даних є побудова фолдів, тобто розбиття набору даних на розділи з метою застосування процедури крос-валідації. Крос-валідація дозволяє оцінити ефективність моделі шляхом розділення доступних даних на набір для тренування та набір для тестування.

Використання крос-валідації допомагає уникнути перенавчання (overfitting) та недонавчання (underfitting) моделі, а також дозволяє отримати більш об'єктивну

оцінку її ефективності.

4.3 Аугментація даних

На етапі навчання моделі з метою отримання кращих результатів набір даних зазвичай доповнюється за допомогою процедури аугментації. Аугментація даних (data augmentation) – це техніка, яка використовується для збільшення розміру навчального набору даних шляхом випадкового змінення зображень чи даних. Мета аугментації полягає в тому, щоб розширити різноманітність даних, які використовуються для навчання моделі, тим самим поліпшуючи її загальність і здатність до узагальнення.

В якості аугментацій при тренуванні моделі ми використовували переважно стандартні аугментації з бібліотеки Albumentations[4], такі як компресія зображення, накладання гаусового шуму та розмиття, горизонтальне відображення, змінення розміру, змінення яскравості, контрастності та насиченості, Fancy PCA, перетворення кольорового зображення в градації сірого та застосування афінних перетворень (переміщення, масштабування та обертання). Зазначенні перетворення здійснювались з різними імовірностями від 0.05 до 0.7.

Окрім аугментацій зазначених вище, ми використовували дещо спрощену версію підходу Face-Cutout[5], який було запропоновано [42]. Face-Cutout, використовує позиції орієнтирів на обличчі, які будо знайдено на етапі підготовки даних, для аугментації навчальних зображень. Позиції орієнтирів включають місцезнаходження очей, вух, носа, рота та лінії щелепи (див. рис. 4.4). Ці позиції використовуються для створення багатокутників за допомогою процедури Face-Cutout. Приклади таких багатокутників наведені на рис. 4.6.

Як вже зазначалось вище, на етапі підготовки даних було згенеровано різницеві SSIM маски. Face-Cutout алгоритм отримує на вхід зображення обличчя

[4] https://github.com/albumentations-team/albumentations
[5] https://www.kaggle.com/competitions/deepfake-detection-challenge/discussion/145721

та відповідну маску для генерації нового навчального зображення. При цьому маска використовується для зменшення вирізання зміненої частини обличчя задля того, щоб модель могла сфокусуватися на ділянці з маніпуляціями. Основні етапи процедури Face-Cutout можна побачити на рисунку 4.7.

(a) Original (b) Eyes (c) Mouth (d) Nose (e) Convex hull (f) Grid mask

Рисунок 4.6 – Приклад Face-Cutout аугментацій: (a) оригінальне обличчя без накладання аугментацій; (b), (c), (d) вирізання очей, рота, та носа відповідно; (e) випадкові вирізання опуклих оболонок з заповненням вирізання випадковими значеннями; (f) видалення секцій зображення.

```
136  def face_cutout(data_root, image, img_file, label, mask, real_movie):
137      landmark_path = os.path.join(data_root, FACE_LANDMARKS_DIR, real_movie, img_file[:-4] + ".npy")
138      if os.path.exists(landmark_path) and random.random() < LANDMARK_PROB:
139          landmarks = np.load(landmark_path)
140          image = remove_face_element(image, landmarks)
141      elif random.random() < CONVEX_HULL_PROB:
142          revome_convex_hull(image)
143      elif random.random() < GRID_MASK_PROB:
144          image, mask = remove_grid_mask(image, label, mask)
145      return image, mask
```

Рисунок 4.7 – Основні етапи процедури Face-Cutout

Згідно рисунку 4.7 процедура Face-Cutout складається з 3 можливих сценаріїв: remove_face_element, remove_convex_hull та remove_grid_mask. У першому сценарії на основі особливих точок може бути вирізано очі, рот чи ніс. У другому здійснюються випадкові вирізання опуклих оболонок. У третьому вирізаються секції зображення.

4.4 Основні етапи побудови та навчання моделей

Основні етапи процедури побудови та навчання моделей зображено на рисунку 4.8.

```
def main():
    args = parse_args()
    prepare_environment(args)

    start_time = time.time()
    cudnn.benchmark = True

    conf = load_config(args.config)
    model = construct_model(args, conf)
    loss_functions = construct_loss_functions(conf)
    optimizer, scheduler = construct_optimizer(conf['optimizer'], model)
    bce_best = 100
    start_epoch = 0
    batch_size = conf['optimizer']['batch_size']

    train_dataset = DFDCClassifierDataset(
        mode="train", oversample_real=not args.no_oversample, fold=args.fold,
        face_cutouts_enabled=args.face_cutouts_enabled, face_images_dir=args.face_images_dir, data_path=args.data_dir,
        label_smoothing=args.label_smoothing, folds_csv=args.folds_csv,
        transforms=create_train_transforms(conf["size"]), normalize=conf.get("normalize", None)
    )
    validation_dataset = DFDCClassifierDataset(
        mode="val", fold=args.fold, face_images_dir=args.face_images_dir, data_path=args.data_dir,
        folds_csv=args.folds_csv, transforms=create_val_transforms(conf["size"]), normalize=conf.get("normalize", None)
    )
    validation_data_loader = DataLoader(validation_dataset, batch_size=batch_size * 2, num_workers=args.workers,
                                        shuffle=False, pin_memory=False)
    os.makedirs(args.logdir, exist_ok=True)
    logs_dir = args.logdir + '/' + conf.get("prefix", args.prefix) + conf['model'] + "_" + str(args.fold)
    summary_writer = SummaryWriter(logs_dir)
    bce_best, start_epoch = resume_checkpoint(args, bce_best, model, start_epoch)
    current_epoch = start_epoch
    model, optimizer = init_model(model, optimizer, args, conf)

    snapshot_name = (
        "{}{}_{}_{}".format( *args: conf.get("prefix", args.prefix), conf['network'], conf['model'], args.fold))
```

```

    validation_dataset.reset( epoch: 1, args.seed)
    max_epochs = conf['optimizer']['schedule']['epochs']
    for epoch in range(start_epoch, max_epochs):
        train_dataset.reset(epoch, args.seed)
        train_sampler = construct_trian_sampler(args, epoch, train_dataset)
        freeze_epochs_if_required(args, epoch, model)

        train_data_loader = DataLoader(train_dataset, batch_size=batch_size, num_workers=args.workers,
                                       shuffle=train_sampler is None, sampler=train_sampler, pin_memory=False,
                                       drop_last=True)

        train_epoch(current_epoch, loss_functions, model, optimizer, scheduler, train_data_loader, summary_writer, conf,
                    args.local_rank, args.only_changed_frames)
        model = model.eval()

        if args.local_rank == 0:
            epoch_state = {'epoch': current_epoch + 1, 'state_dict': model.state_dict(), 'bce_best': bce_best,}
            torch.save(epoch_state, args.output_dir + '/' + snapshot_name + "_last")
            torch.save(epoch_state, args.output_dir + snapshot_name + "_{}".format(current_epoch))
            if (epoch + 1) % args.test_every == 0:
                bce_best = evaluate_val(
                    args, validation_data_loader, bce_best, model, snapshot_name=snapshot_name,
                    current_epoch=current_epoch, summary_writer=summary_writer, start_time=start_time
                )
        current_epoch += 1
```

Рисунок 4.8 – Основні етапи процедури побудови та навчання моделі

Як можна побачити з рисунку 4.8 процедура тренування моделі складається з декількох етапів.

Першим етапом є завантаження параметрів, побудова моделі та функції втрат. Модель реалізується за допомогою класу DFDCClassifier, який розширює базовий клас всіх нейронних мереж пакета torch Module (див. рис. 4.9).

```
43  class DFDCClassifier(nn.Module):
44      def __init__(self, model, dropout_rate=0.0) -> None:
45          super().__init__()
46          self.model = model_params[model]["init_op"]()
47          self.avg_pool = AdaptiveAvgPool2d((1, 1))
48          self.dropout = Dropout(dropout_rate)
49          self.fc = Linear(model_params[model]["features"], out_features: 1)
50
51      def forward(self, x):
52          x = self.model.forward_features(x)
53          x = self.avg_pool(x).flatten(1)
54          x = self.dropout(x)
55          x = self.fc(x)
56          return x
```

Рисунок 4.8 – Реалізація моделі згорткової нейронної мережі

Для побудови моделі класифікації фейкових облич, ми використовували попередньо навчені моделі класифікації зображень з пакету timm. Пакет timm (також відомий як "PyTorch Image Models") – це бібліотека, яка містить реалізації різних архітектур глибоких нейронних мереж для обробки зображень, побудовані на основі бібліотеки PyTorch. Цей пакет надає зручний і простий спосіб доступу до широкого спектру моделей, включаючи відомі архітектури, такі як ResNet, EfficientNet, ViT (Vision Transformer), DPN (Dual Path Network), MobileNet, і багато інших.

В якості останніх шарів нашої нейронної мережі використовувались шари усередненого пулінга, дропаута та повнозв’язний шар як це часто робиться у випадку навчання шляхом трансферу даних.

В якості функції втрат ми використовували BCE (див. розділ 3.2). В якості оптимайзера ми як правило використовували стохастичний градієнтний спуск

(SGD). Для планування швидкості навчання (learning rate scheduling) ми використовували PolyLR, або поліноміальне планування швидкості навчання. PolyLR є методом зменшення швидкості навчання, заснованим на поліноміальній функції. Ця стратегія поступово зменшує швидкість навчання відповідно до поліноміального розпаду, який зазвичай визначається наступною формулою 4.1:

$$LR(t) = LR_0 \left(1 - \frac{t}{T}\right)^p, \tag{4.1}$$

де LR_0 – початкова швидкість навчання,

t – поточна ітерація або епоха,

T – загальна кількість ітерацій або епох,

p – ступінь полінома.

В нашому випадку як правило використовувались наступні значення цих змінних $LR_0 = 0.01, T = 30, p = 0.9$.

Планування швидкості навчання дозволяє динамічно змінювати швидкість навчання протягом процесу навчання моделі, що може сприяти кращій збіжності та покращенню результатів.

На наступному кроці відбувається ініціалізація тренувального та валідаційного датасетів, які представлені класом DFDCClassifierDataset (див. рис. 4.9). Цей клас імплементує абстрактний клас Dataset пакету torch. Функціонування класу може здійснюватися у двох режимах навчальному та тестовому. В тестовому режимі окрім стандартних трансформацій зображень, додатково використовуються FC-трансформації описані у попередньому розділі.

Подальше навчання здійснюється за допомогою налаштування циклів навчання (епох), де модель тренується на навчальних даних та перевіряється на валідаційних даних.

Зазначимо, що після кожної епохи, зберігаються отримані ваги нейронної мережі, що дозволяє їх загрузку у майбутньому. Окрім цього ми використовуємо клас SummaryWriter бібліотеки tensorboard для зберігання швидкості навчання та

обчислення втрат на етапі тренування та значення BCE під час валідації. SummaryWriter дозволяє реєструвати дані, такі як скалярні значення, гістограми, зображення, текст та інше, які потім можна переглядати в TensorBoard — веб-інтерфейсі для візуалізації динамічних графіків тренувальних і тестових метрик, архітектур моделей, розподілів даних тощо.

```
class DFDCClassifierDataset(Dataset):

    def __init__(self, data_path=DATA_DIR, fold=0, label_smoothing=0.01, face_cutouts_enabled=True,
                 face_images_dir=FACE_IMAGES_DIR, folds_csv=FOLDS_FILE, normalize=NORMALIZATION_PARAMS,
                 mode="train", reduce_val=True, oversample_real=True, transforms=None):
        super().__init__()
        self.init_self(data_path, face_cutouts_enabled, face_images_dir, fold, folds_csv, label_smoothing, mode,
                       normalize, oversample_real, reduce_val, transforms)

    def __getitem__(self, index: int):
        while True:
            movie, img_file, label, ori_video, frame, fold = self.data[index]
            try:
                if self.mode == "train":
                    label = np.clip(label, self.label_smoothing, 1 - self.label_smoothing)
                img_path = os.path.join(self.data_root, self.face_images_dir, movie, img_file)
                image = cv2.imread(img_path, cv2.IMREAD_COLOR)
                image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
                mask = build_ssim_mask(self.data_root, image, img_file, ori_video, movie)
                if self.mode == "train" and self.face_cutouts_enabled:
                    image, mask = face_cutout(self.data_root, image, img_file, label, mask, ori_video)
                    if (DEMO_MODE):
                        movie_path = os.path.join(self.data_root, FC_IMAGE_DIR, movie)
                        if (not os.path.exists(movie_path)):
                            os.makedirs(movie_path)
                        fc_image_path = os.path.join(self.data_root, FC_IMAGE_DIR, movie, img_file[:-4] + "_fc.png")
                        cv2.imwrite(fc_image_path, image)
                valid_label = np.count_nonzero(mask[mask > 20]) > 32 or label < 0.5
                valid_label = 1 if valid_label else 0
                if self.transforms:
                    data = self.transforms(image=image, mask=mask)
                    image = data["image"]

                image = img_to_tensor(image, self.normalize)
                return {"image": image, "labels": np.array((label,)), "img_name": os.path.join(movie, img_file),
                        "valid": valid_label, "rotations": 0}
            except Exception as e:
                traceback.print_exc(file=sys.stdout)
                print("Broken image", os.path.join(self.data_root, self.face_images_dir, movie, img_file))
                index = random.randint( a: 0, len(self.data) - 1)
```

Рисунок 4.9 – Фрагмент коду класу DFDCClassifierDataset

В наступному розділі ми розглянемо експериментальні дослідження виконані за допомогою розглянутої програмної реалізації.

5 ОПИС ЕКСПЕРИМЕНТАЛЬНИХ ДОСЛІДЖЕНЬ

5.1 Трансферне навчання

Трансферне навчання (transfer learning) є методом машинного навчання, який полягає у використанні знань, отриманих під час розв'язання однієї задачі, для вирішення іншої схожої або пов'язаної задачі. Ця методика стала особливо популярною в глибинному навчанні, оскільки вона дозволяє значно прискорити час навчання моделей і покращити їхню продуктивність, особливо коли доступна кількість даних для нової задачі є обмеженою. Ми використовуємо цю методику.

В нашому дослідженні ми будемо застосовувати наступні попередньо навчені моделі з пакету timm:

- tf_efficientnet_b4;
- tf_efficientnet_b4_ns;
- tf_efficientnet_b5;
- tf_efficientnet_b5_ns;
- tf_efficientnet_b7_ns;
- xception;
- resnet50.

Використання попередньо навчених моделей надає чималі переваги, які можуть значно поліпшити ефективність і швидкість розробки проекту. Використання таких моделей дозволяє розробникам уникнути початку навчання з нуля, що заощаджує час і обчислювальні ресурси. Ці моделі, як правило, навчаються на великих та різноманітних наборах даних, що дозволяє їм вивчити багатий спектр особливостей. Таке навчання сприяє покращенню загальної продуктивності моделей, особливо коли вони адаптуються до специфічних задач за допомогою тонкого налаштування чи трансферного навчання.

Попередньо навчені моделі також знижують ризик перенавчання, оскільки базові особливості вже були натреновані на більш широких даних, що зменшує вплив випадкових особливостей конкретного набору даних. Це також допомагає моделям краще узагальнювати, тобто ліпше справлятися з новими, невідомими

раніше даними, що критично важливо для їх ефективного застосування в реальному світі. Таким чином, попередньо навчені моделі виявляються незамінними в арсеналі інструментів дослідників та розробників у сфері машинного навчання та штучного інтелекту.

Усі наші моделі були ініціалізовані заздалегідь навченими вагами ImageNet. Крім того, EfficientNet було додатково навчено з використанням методу Noisy Student.

Техніка Noisy Student є одним з методів самонавчання для покращення ефективності нейронних мереж. Вона була представлена в статті Google AI з 2019 року [49] і в основному застосовується в контексті навчання з використанням нерозмічених даних. Ідея полягає в тому, щоб нейронна мережа, яка вже навчена на певному наборі даних, допомагала навчатися на новому наборі даних, використовуючи методи додавання шуму та регуляризації (див. рисунок 5.1).

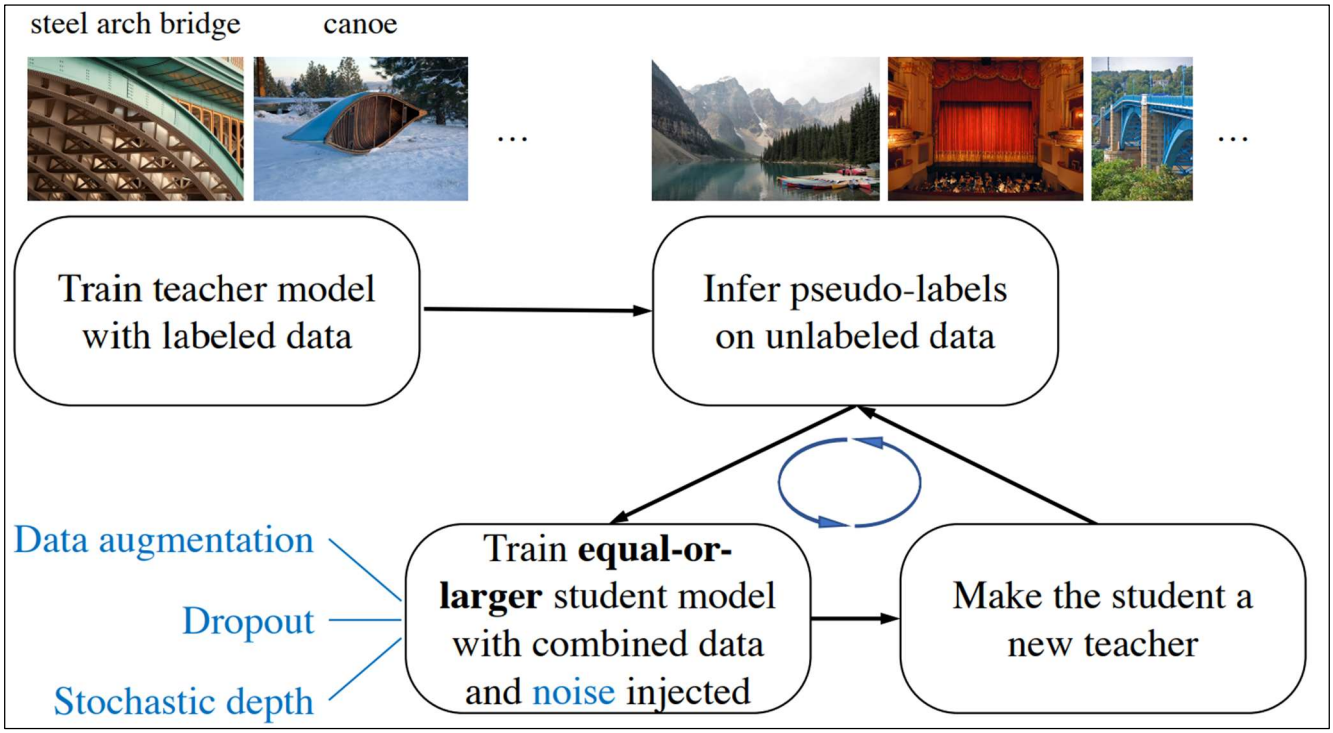


Рисунок 5.1 – Схема навчання моделі за технікою Noisy Student (за даними [49])

Основний принцип техніки Noisy Student полягає в тому, що модель, яка вже навчена на одному наборі даних, може бути використана для розмічання невідомих даних із нового набору даних. Потім ці згенеровані мітки можуть бути

використані для навчання нової моделі на цьому наборі даних. Під час цього процесу також використовуються техніки регуляризації, такі як дропаут або аугментація, для підвищення стійкості моделі до шуму. Таким чином, Noisy Student дозволяє ефективно використовувати нерозмічені дані для покращення якості моделі без необхідності в дорогому та часвитратному процесі ручної розмітки даних.

5.2 Проведення експериментальних досліджень

Результати проведених обчислень наведені в таблиці 1. Для кожної моделі наведено кількість параметрів моделі, час обчислень та метрики BCE, AUC та AP, які розглядались у попередньому розділі. Назви моделей у таблиці містять використану архітектуру нейронної мережі; NS у назві моделі означає, що модель додатково навчалась за допомогою методу Noisy Student, FC – означає, що при тренуванні мережі використовувалась спрощена версія алгоритму Face Cutout. Навчання проводилось на Intel Core i7 з 64 Gb та GeForce RTX 3090 24 Gb.

Таблиця 5.1 – Результати навчання моделей виявлення фейкового контенту з використанням набору даних DFDC

Model	Parameters (millions)	Train time (hours)	BCE	AUC	AP
B4+NS+FC	19	4,4	0,374	0,952	0,9949
B5	30	5,5	0,347	0,957	0,9953
B5+FC	30	5,7	0,337	0,952	0,9949
B5+NS+FC	30	5,8	0,302	0,956	0,9953
B7	43	9,4	0,338	0,952	0,9947
Xception	22,9	3,5	0,351	0,932	0,9921
ResNet50	25,6	3,3	0,322	0,936	0,9931

Згідно таблиці 5.1, моделі XCeption та ResNet50 показують трохи гірші результати (~93% AUC), ніж моделі на базі EfficientNet (~95% AUC). Досить несподіваним фактом виявилось те, що практично всі моделі на базі архітектури EfficientNet показали дуже близькі значення AUC (~95%) і різницю між ними можна лише спостерігати за значеннями BCE. При цьому ми також не

спостерігали суттєвих змін між результатами тренування з використанням Noisy Student та FaceCutout та без них. В той же час в [42] спостерігалось збільшення AUC на величину ~5% при використанні методу Face Cutout. При цьому значення AUC наприклад для моделі XCeptionNet були дещо нижчими (AUC 81.99%), ніж отримані у цій роботі (AUC 93.2%) при використанні датасету DFDC для навчання та застосуванні методу Face Cutout. При використанні комбінованого датасету при навчанні AUC в [42] дорівнювало 95.66%, що дещо перевищує отримане нами значення.

На рисунку 5.1 наведені ROC-криві для моделей B5, B5+FC, B5+NS+FC. Назви моделей містять використану архітектуру нейронної мережі; NS у назві моделі означає, що модель додатково навчалась за допомогою методу Noisy Student, FC – означає, що при тренуванні мережі використовувалась спрощена версія алгоритму Face Cutout.

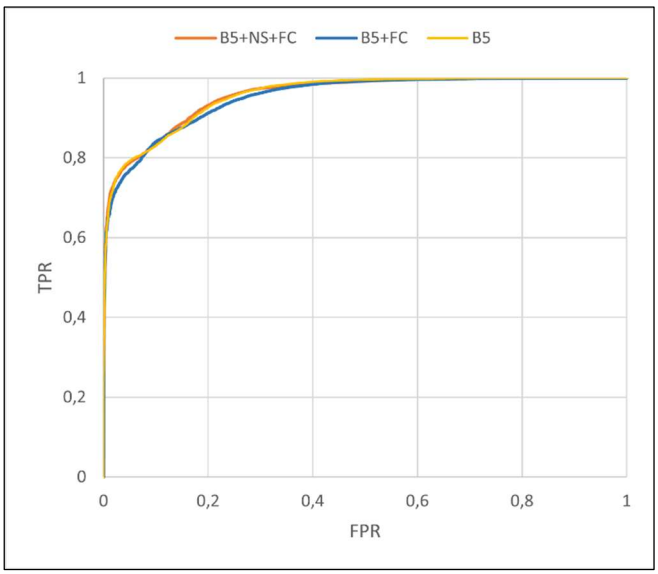


Рисунок 5.1 – ROC-криві для моделей B5, B5+FC, B5+NS+FC

З рисунку видно, що, як і зазначалось вище, ми не спостерігаємо суттєвих змін при використанні моделей, навчених за допомогою методик Noisy Student та Face Cutout, тоді як в [42] спостерігалось збільшення AUC на величину ~5%. Планується більш детально дослідити причини відсутності впливу цих методик у подальших дослідженнях.

ВИСНОВКИ

Під час виконання кваліфікаційної роботи було дано визначення поняття фейкового контенту та розглянуто його види. Основну увагу у даному аналітичному огляді було зосереджено на маніпуляціях з обличчям, а саме маніпуляціями із замінами ідентичності, тобто маніпуляціях, які включають заміну обличчя однієї людини на обличчя іншої у відео або на фото (deepfakes). Було відзначено, що останнім часом цей тип маніпуляцій став особливо популярним і відомим у зв'язку з появою ефективних інструментів та розвитком технологій машинного навчання. Було сформульовану задачу, яка вирішується буде вирішуватись у кваліфікаційній роботі.

Було проведено огляд існуючих на сьогоднішній день наборів даних із маніпуляціями з заміни ідентичності та обрано набір даних, який використовувався при навчанні моделей виявлення фейкового контенту. При цьому також ми визначили та здійснили етапи попередньої обробки даних. Було також проведено огляд існуючих моделей виявлення маніпуляцій з заміни ідентичності. Проаналізовано їх переваги, недоліки та досліджено результати, які вони надають при тестуванні на різних наборах даних.

Проведено навчання моделей згорткових нейронних мереж на основі архітектур EfficientNet, XCeptionNet та ResNet та отримано метрики якості проведеного навчання. Навчання проводилося за допомогою сучасного набору даних Deepfake Detection Challenge Dataset. Результати було порівняно з даними інших досліджень.

Таким чином, ми детально розглянули різноманітні підходи та алгоритми, які використовуються в сучасних дослідженнях і виявили, що ефективність виявлення маніпуляцій із заміною ідентичності неперервно покращується завдяки інноваційним технічним рішенням. Як і у багатьох інших областях, таких як, наприклад, розробка засобів ППО чи антивірусів, розробка та виявлення фейкового контенту знаходяться у постійній боротьбі і поява нових моделей

створення фейкового контенту приводить до появи нових моделей його виявлення.

Сьогодні технології машинного навчання залучені до цієї битви на обох боках. Тому нині дуже важливо приділяти увагу соціальним та етичним аспектам створення фейкового контенту за допомогою методів глибокого навчання та інших схожих технологій. Ці аспекти охоплюють широкий спектр питань від індивідуальних прав до колективної безпеки та довіри у суспільстві. Зазначимо, що розвиток цих технологій повинен йти рука об руку з обговоренням етичних та соціальних норм та правил, що регулюють використання та розповсюдження фейкового контенту.

Враховуючи швидкий технологічний розвиток, про який свідчить велика кількість досліджень у цьому напрямку, можна очікувати значного прогресу в цій області в найближчому майбутньому. Підвищення обізнаності про фейковий контент, розвиток нових алгоритмів та підходів його виявлення, а також більш глибоке розуміння соціальних аспектів цієї проблеми забезпечать більш ефективний захист сучасного суспільства від негативних впливів фейкового контенту.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Кіценко Ю. О., Смеляков К. С. Розробка моделі виявлення фейкового контенту на основі архітектури EfficientNet. *Матеріали 28-го міжнародного молодіжного форуму «Радіоелектроніка та молодь у XXI столітті»* : Зб. матеріалів форуму. Т. 6. Харків, 2024. С. 435–436.

2. Smelyakov K., Kitsenko Y., Chupryna A. Deepfake detection models based on machine learning technologies. *2024 IEEE open conference of electrical, electronic and information sciences (estream)*. 2024. P. 1–6. URL: https://doi.org/10.1109/eStream61684.2024.10542582.

3. Citron D. How DeepFake Undermine Truth and Threaten Democracy. URL: https://www.ted.com (дата звернення: 01.06.2024).

4. Cellan-Jones R. Deepfake Videos Double in Nine Months. URL: https://www.bbc.com/news/technology-49961089 (дата звернення: 01.06.2024).

5. BBC Bitesize. Deepfakes: What Are They and Why Would I Make One? URL: https://www.bbc.co.uk/bitesize/articles/zfkwcqt (дата звернення: 01.06.2024).

6. Deepfakes and beyond: a survey of face manipulation and fake detection / R. Tolosana et al. *Information fusion*. 2020. Vol. 64. P. 131–148.

7. Yang X., Li Y., Lyu S. Exposing deep fakes using inconsistent head poses. *ICASSP 2019-2019 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. 2019. P. 8261–8265.

8. Li Y., Chang M.-C., Lyu S. In ictu oculi: exposing ai created fake videos by detecting eye blinking. *2018 IEEE International workshop on information forensics and security (WIFS)*. 2018. P. 1–7.

9. Korshunov P., Marcel S. Deepfakes: a new threat to face recognition? Assessment and detection. *ArXiv preprint arxiv:1812.08685*. 2018.

10. Bregler C., Covell M., Slaney M. Video rewrite: driving visual speech with audio. *Seminal graphics papers: pushing the boundaries, volume 2*. 2023. P. 715–722.

11. Kingma D. P., Welling M. Auto-encoding variational bayes. *ArXiv preprint arxiv:1312.6114*. 2013.

12. Generative adversarial nets / I. Goodfellow et al. *Advances in neural information processing systems*. 2014. Vol. 27.

13. Karras T., Laine S., Aila T. A style-based generator architecture for generative adversarial networks. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019. P. 4401--4410.

14. Faceforensics++: learning to detect manipulated facial images / A. Rossler et al. *Proceedings of the IEEE/CVF international conference on computer vision*. 2019. P. 1–11.

15. Face2face: real-time face capture and reenactment of rgb videos / J. Thies et al. *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016. P. 2387--2395.

16. Thies J., Zollhofer M., Nießner M. Deferred neural rendering: image synthesis using neural textures. *Acm transactions on graphics (TOG)*. 2019. Vol. 38, no. 4.
P. 1–12.

17. Chollet F. Xception: deep learning with depthwise separable convolutions. *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017. P. 1251--1258.

18. Fridrich J., Kodovsky J. Rich models for steganalysis of digital images. *IEEE transactions on information forensics and security*. 2012. Vol. 7, no. 3. P. 868--882.

19. Cozzolino D., Poggi G., Verdoliva L. Recasting residual-based local descriptors as convolutional neural networks: an application to image forgery detection. *Proceedings of the 5th ACM workshop on information hiding and multimedia security*. 2017. P. 159--164.

20. Bayar B., Stamm M. C. A deep learning approach to universal image manipulation detection using a new convolutional layer. *Proceedings of the 4th ACM workshop on information hiding and multimedia security*. 2016. P. 5--10.

21. Distinguishing computer graphics from natural images using convolution neural networks / N. Rahmouni et al. *2017 IEEE workshop on information forensics and security (WIFS)*. 2017. P. 1--6.

22. Mesonet: a compact facial video forgery detection network / D. Afchar et al. *2018 IEEE international workshop on information forensics and security (WIFS)*. 2018. P. 1–7.

23. Google AI. Contributing Data to Deepfake Detection Research. URL: https://ai.googleblog.com/2019/09/contributing-data-to-deepfake-detection.html (дата звернення: 01.06.2024).

24. Celeb-df: a large-scale challenging dataset for deepfake forensics / Y. Li et al. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020. P. 3207–3216.

25. Two-stream neural networks for tampered face detection / P. Zhou et al. *2017 IEEE conference on computer vision and pattern recognition workshops (CVPRW)*. 2017. P. 1831–1839.

26. Li Y., Lyu S. Exposing deepfake videos by detecting face warping artifacts. *ArXiv preprint arxiv:1811.00656*. 2018.

27. Matern F., Riess C., Stamminger M. Exploiting visual artifacts to expose deepfakes and face manipulations. *2019 IEEE winter applications of computer vision workshops (WACVW)*. 2019. P. 83–92.

28. Multi-task learning for detecting and segmenting manipulated facial images and videos / H. H. Nguyen et al. *2019 IEEE 10th international conference on biometrics theory, applications and systems (BTAS)*. 2019. P. 1–8.

29. Nguyen H. H., Yamagishi J., Echizen I. Use of a capsule network to detect fake images and videos. *ArXiv preprint arxiv:1910.12467*. 2019.

30. Spatial pyramid pooling in deep convolutional networks for visual recognition / K. He et al. *IEEE transactions on pattern analysis and machine intelligence*. 2015. Vol. 37, no. 9. P. 1904–1916.

31. The deepfake detection challenge (dfdc) preview dataset / B. Dolhansky et al. *ArXiv preprint arxiv:1910.08854*. 2019.

32. The deepfake detection challenge (dfdc) dataset / B. Dolhansky et al. *ArXiv preprint arxiv:2006.07397*. 2020.

33. Deeperforensics-1.0: a large-scale dataset for real-world face forgery detection / L. Jiang et al. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020. P. 2889–2898.

34. Learning spatiotemporal features with 3d convolutional networks / D. Tran et al. *Proceedings of the IEEE international conference on computer vision*. 2015. P. 4489–4497.

35. Temporal segment networks: towards good practices for deep action recognition / L. Wang et al. *European conference on computer vision*. 2016. P. 20–36.

36. Carreira J., Zisserman A. Quo vadis, action recognition? A new model and the kinetics dataset. *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*. 2017.

37. Deep residual learning for image recognition / K. He et al. *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016. P. 770–778.

38. Sohan M. F., Solaiman M., Hasan M. A. A survey on deepfake video detection datasets. *Indonesian journal of electrical engineering and computer science*. 2023. Vol. 32, no. 2. P. 1168–1176.

39. Wilddeepfake: a challenging real-world dataset for deepfake detection / B. Zi et al. *Proceedings of the 28th ACM international conference on multimedia*. 2020. P. 2382–2390.

40. Kodf: a large-scale korean deepfake detection dataset / P. Kwon et al. *Proceedings of the IEEE/CVF international conference on computer vision*. 2021. P. 10744–10753.

41. Forgerynet: a versatile benchmark for comprehensive forgery analysis / Y. He et al. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021. P. 4360–4369

42. Towards solving the deepfake problem: an analysis on improving deepfake detection using dynamic face augmentation / S. Das et al. *Proceedings of the IEEE/CVF international conference on computer vision*. 2021. P. 3776–3785.

43. The neural network technologies effectiveness for face detection / K. Smelyakov et al. *2020 IEEE third international conference on data stream mining \& processing (DSMP)*. 2020. P. 201–205.

44. The neural network models effectiveness for face detection and face recognition / K. Smelyakov et al. *2021 IEEE open conference of electrical, electronic and information sciences (estream)*. 2021. P. 1–7.

45. Joint face detection and alignment using multitask cascaded convolutional networks / K. Zhang et al. *IEEE signal processing letters*. 2016. Vol. 23, no. 10. P. 1499–1503.

46. Image quality assessment: from error visibility to structural similarity / Z. Wang et al. *IEEE transactions on image processing*. 2004. Vol. 13, no. 4. P. 600–612.

47. Tan M., Le Q. Efficientnet: rethinking model scaling for convolutional neural networks. *International conference on machine learning*. 2019. P. 6105–6114.

48. Mehmood A. Efficient anomaly detection in crowd videos using pre-trained 2D convolutional neural networks. *IEEE access*. 2021. Vol. 9. P. 138283–138295.

49. Self-training with noisy student improves imagenet classification / Q. Xie et al. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020. P. 10687–10698.