УДК 004.056.5 **DOI:10.30837/rt.2023.4.215.02**

О.Д. ДОЛГАНЕНКО, О.В. СЄВЄРІНОВ, канд. техн. наук., Д.О. В'ЮХІН, В.П. КОЦЮБА, канд. техн. наук, А.В. КРЕПКО

АНАЛІЗ МЕТОДІВ ВИЯВЛЕННЯ АТАК БІОМЕТРИЧНОЇ АВТЕНТИФІКАЦІЇ ЗА ОБЛИЧЧЯМ У МОБІЛЬНИХ ПРИСТРОЯХ

Вступ

Актуальність теми біометричної автентифікації на мобільних пристроях викликана рядом факторів. По-перше, за останні кілька років спостерігається значний приріст кількості користувачів мобільних пристроїв, таких як смартфони і планшети. Користувачі все більше довіряють особисту інформацію своїм пристроям. Гарантування безпеки на цих пристроях, зокрема шляхом використання біометричної автентифікації, є актуальною проблемою [1]. Мобільні додатки фінансового призначення покладаються на вбудовані камери смартфону для біометричної автентифікації за обличчям або підтвердження особистості, порівнюючи вхідне зображення або відео із зображеннями з документів. Такі заходи залишають вразливість, особливо в час зростання популярності передової технології Deepfake, яка дозволяє зловмисникам видавати себе за інших людей із вражаючою точністю. Оскільки задокументовані випадки експлойтів Deepfake зростають, існує нагальна потреба в надійних заходах безпеки.

Метою цієї роботи є аналіз існуючих методів для виявлення Deepfake відео, зосереджуючись, зокрема, на алгоритмах, які можуть працювати локально на мобільних пристроях (без підключення до Інтернету), а також ефективних рішеннях на стороні сервера. Мета полягає в тому, щоб оцінити ці методи не лише за їхньою точністю розпізнавання підроблених відео, але й на сумісність із мобільними платформами та відповідність стандартам. Зрештою, це дослідження має на меті прокласти шлях до розробки універсального, точного та надійного інструменту (бібліотеки) для виявлення Deepfake підробок, який можна легко вбудувати в вихідний код будь-якого мобільного додатку, тим самим підвищуючи його рівень безпеки.

Введення до предметної області

Відповідно до статистики станом на 2022 р. 43 % респондентів не могли візуально ідентифікувати Deepfake відео, а решта 57 % могли впевнено їх розпізнати. Для того щоб проаналізувати як саме людина розпізнає підроблені відео, можна звернутися до офіційних джерел. Дослідження, проведене Міністерством внутрішньої безпеки Сполучених Штатів Америки, містить перелік особливостей, на які слід звернути увагу при перегляді потенційно підробленого відео:

1) при загально чіткому відео помітні нечіткості або розмитість на обличчі (або навпаки);
2) зміна тону шкіри біля краю обличчя;
3) подвоєння контуру підборіддя, брови або обличчя;
4) розмиття обличчя при перетину із руками чи іншими предметами;
5) ефект обрізаних фрагментів обличчя навколо рота, очей і шиї;
6) неприродні рухи м'язів обличчя, відсутність моргання;
7) неочікувана зміна фону та/або освітлення впродовж відео;

Інформування користувачів про методи виявлення Deepfake відео в новинах та медіа – справді важлива задача, але передбачається, що технології підробки відео- та аудіозаписів у найближчому майбутньому значно покращаться.

Відповідно до статистики [2], 32 % користувачів використовують біометричну автентифікацію за обличчям на персональних мобільних пристроях, а 28 % – у сферах банківської діяльності, фінансів та страхування. При цьому, в США біометрична автентифікація методом розпізнавання обличчя складає всього 16 % від загальної кількості користувачів, але не

дивлячись на це, вражає своєю ринковою вартістю в 5 мільярдів доларів, яка, за прогнозами, збільшиться до 13 мільярдів доларів у 2026 р.

Загалом, біометрична автентифікація має широкий спектр галузей, проте у цьому дослідженні особлива увага приділяється потенційним ризикам саме для методу розпізнавання обличчя на мобільних платформах, що пов'язані із зростаючою ефективністю технології deepfake.

Методи створення Deepfake

Застосування технології Deepfakes реалізоване переважно використанням глибоких нейронних мереж, технології заміни обличчя та Generative Adversarial Networks (GAN), автоматичні кодери. Створення Deepfake відео стає все більш доступним завдяки платформам, таким як Zao, DeepFace Lab та GitHub, що надають користувачам широкий набір інструментів для цього процесу.

Найпоширенішим способом створення Deepfake є використання процесу автоматичних кодерів. Цей процес складається з кодування (стиснення) і декодування (відновлення) зображення людини.

Процес кодування бере вхідне зображення та перетворює його на «приховане» стиснене представлення найважливіших характеристик цього зображення. Потім цей стиснутий масив можна передати в декодер.

Робота декодера полягає у відновленні вихідного зображення на основі вхідного стисненого представлення. Звичайно, відновити оригінальне зображення без втрати даних практично неможливо, але можна мінімізувати коливання помилок між вхідним і вихідним зображеннями.

Суть автоматичних кодерів може бути формалізована формулою

$$\arg\min A, BE[\Delta(x, (B * A)(x))]$$

Іще одним методом для створення deepfake є так звана GAN, що розшифровується як генеративна змагальна мережа, що є відносно новітнім підходом для некерованого та напівкерованого навчання. Це досягається шляхом неявного моделювання високовимірних розподілів даних. Такий підхід можна охарактеризувати тренуванням пари мереж, які змагаються одна з одною [3]. Тобто, GAN як генератор (G) створює підробки з метою створення реалістичних зображень. Експерт, відомий як дискримінатор (D) отримує підробки та справжні (автентичні) зображення і намагається розрізнити їх. Обидва тренуються одночасно та змагаються один з одним.

Навчання GAN включає знаходження параметрів дискримінатора (які максимізують його точність класифікації), а також знаходження параметрів генератора, які максимально плутають дискримінатор. Вартість навчання оцінюється за допомогою функції значення V(G, D), яка залежить як від генератора, так і від дискримінатора. Навчання передбачає вирішення наступної задачі:

$$D_{max}\, G_{min} V(G, D),$$

де $V(G, D) = \mathbb{E}_{p_{data}(x)} logD(x) + \mathbb{E}_{p_g(x)} \log(1 - D(x))$.

Під час навчання параметри однієї моделі оновлюються, а параметри іншої фіксуються. В найкращому сценарії дискримінатор навчений оптимально по відношенню до поточного стану генератора, після чого генератор знову оновлюється. Однак на практиці дискримінатор не може бути навчений до оптимального стану, а може бути навчений лише до невеликої кількості ітерацій, і генератор оновлюється одночасно з дискримінатором.

Методи розпізнавання deepfake, що базуються на машинному навчанні

Традиційні методи машинного навчання (ML), наприклад дерева рішень, підходять для виявлення Deepfake, оскільки їх легко зрозуміти та налаштувати. Вони роблять свій вибір логічним завдяки методу прийняття рішень на основі дерев. В алгоритмах дерев рішень для

виявлення Deepfake перший крок включає вибір атрибуту для перевірки, такі як значення пікселів або точність синхронізації губ, контури обличчя. Потім алгоритм визначає найбільш інформативну серед цих рис як кореневий вузол, при цьому використовуючи такі показники як приріст інформації або неоднорідність Джіні. Дані поділяються на підмножини на основі цієї кореневої функції, породжуючи нові гілки та зрештою досягаючи кінцевих вузлів для остаточної класифікації.

Однією з сильних сторін алгоритму є його адаптивність. Глибина дерева та критерії кінцевого вузла можуть бути точно налаштовані для покращення рівня виявлення, особливо для нових, більш просунутих підробок Deepfakes. Крім того, дерева рішень інтерпретуються однозначно. Чіткість та прозорів критеріїв в кожній точці прийняття рішень полегшує вдосконалення моделі, а також робить модель більш зрозумілою.

Мережі GAN, з іншого боку, відмінно справляються із генеруванням реалістичних облич, але можуть залишати деякі ледь помітні нечіткості. При застосуванні методів Deepfake часто змінюються (або поєднуються в одну) певні риси обличчя, для того щоб ще більше вести в оману глядачів. Також проводилися дослідження щодо розпізнавання підроблених відео на основі рухів голови та переміщення фрагментів обличчя. Хоча точність виявлення Deepfakes за допомогою машинного навчання є досить високою (наводиться точність до 98 %), результати сильно залежать від даних, які використовуються для навчання моделі та її проміжної перевірки в рамках епохи. У кількох дослідженнях також повідомляється, що використання непов'язаних наборів даних для тестування навченої мережі призводить до дуже поганої оцінки точності – близько 50 %.

Методи розпізнавання deepfake, що базуються на глибокому навчанні

Підходи, засновані на глибокому навчанні, домінують у сфері виявлення Deepfake, використовують різноманітні методи для виявлення штучних артефактів і фізіологічних невідповідностей у зображеннях і відео. У першій роботі Zhang використали симулятор GAN для виявлення артефактів Deepfake. Також задокументованими є дослідження, що базуються на аналізу ознак RGB, фізіологічних вимірюваннях, таких як ритм серцебиття.

Важливим параметром, що впливає на точність моделі, є її глибина [4]. У контрольованих умовах глибокі CNN виявилися більш ефективними, ніж поверхневі. У результаті, такі інноваційні підходи як механізми уваги, капсульні мережі і ансамблеве навчання досягли точності понад 99 %.

Незважаючи на високу продуктивність, проблема перенавчання залишається суттєвою. Для пом'якшення її впливу застосовуються такі методи, як оптичний потік і автокодери. Також слід відзначити використання методів попередньої обробки даних, аналіз частотної області та багатомодульні підходи для подальшої оптимізації моделі.

Аналізуючи існуючі дослідження, можна часто помітити додаткові алгоритми відокремлення облич від решти зображення для подальшого аналізу. Це підвищує точність, адже контекст зображення не має впливати на артефакти на обличчі.

Методи розпізнавання deepfake на основі статистичних вимірювань

Дослідження цього методу посилаються на унікальний «відбиток», відомий як PRNU, який можна інтерпретувати з цифрових фотографій. Цей ефект з'являється через мікродефекти на матрицях сучасних камер. За допомогою аналізу цього відбитка можна побудувати алгоритм виявлення Deepfake. Для цього відеокадри розділяються на групи, застосовуючи до кожного з них метод під назвою FSTV для вилучення відбитку, після чого аналізується кореляція цього показника із середнім значенням для кожного кадру.

Іще одним варіантом статистичних вимірювань є застосування алгоритму вилучення ознак із зображення використовуючи метод очікування-максимізації. На основі цього був запропонований статистичний фреймворк, який вимірював "розрив" між показниками розповсюдження справжніх відео та створених GAN. У результаті цей показник відстані вказує на легкість виявлення Deepfake. Тобто, чим більша результуюча відстань, тим легше ідентифі-

кувати підроблене відео. Також, завдяки цьому алгоритму, можна оцінювати ефективність інструменту для створення Deepfake.

Таким чином, проаналізувавши відомі дослідження, можна отримати статистику щодо відсотку застосування того чи іншого методу для розпізнавання Deepfake (рис. 1).

Рис. 1. Використання методів розпізнавання Deepfake у дослідженнях

Високий відсоток досліджень на основі одного методу може бути надійною ознакою його ефективності та доступності, однак ця статистика не гарантує, що це буде найкращий підхід для вирішення задачі на мобільній платформі.

Застосування на мобільних платформах

Як показав проведений аналіз, найбільш розповсюджений метод для виявлення Deepfake є використання глибокого навчання. Існує декілька варіантів щодо використання архітектури нейронної мережі на мобільному пристрої.

Найрозповсюдженіший метод для вирішення подібних задач базується на використанні веб-сервісу, у якому вся бізнес-логіка для виявлення Deepfake інкапсульована на самому сервері, доступ до якого здійснюється через інтерфейс HTTP REST API. При такому підході для ефективної роботи існує обмеження стабільного мережевого з'єднання. Окрім цього, користувач може зіткнутися із неочікуваними затримками, оскільки процес завантаження відео та результат виявлення маршрутизується через Інтернет. Однак у цього підходу є переваги: необмежений розмір нейронної моделі на сервері, а також можливість легко оновлювати та централізовано розширювати алгоритм виявлення відразу для всіх користувачів.

Існує і альтернативний метод, адже мобільні пристрої здатні розгортати моделі нейронної мережі локально [5]. Завдяки такому підходу вся обробка виконується на самому мобільному пристрої, що забезпечує незалежне від мережі виявлення Deepfake з низькою затримкою. Технічно для запуску нейронних мереж з мобільного пристрою існує кілька технологій. Бібліотеки PyTorch Mobile і TensorFlow Lite є лідерами у сфері машинного навчання і вони оптимізовані для застосування на мобільних платформах.

PyTorch Mobile, що є частиною екосистеми PyTorch, надає розробникам можливість легко перейти з стандартних PyTorch методів розробки до підтримки на мобільних платформах, зберігаючи при цьому схожість інтерфейсів. Отже, цикл розробки для тих, хто звик до PyTorch стає більш спрощеним і знайомим.

TensorFlow Lite (TFLite) – це спрощене реалізація технології TensorFlow, створена для мобільних пристроїв та IOT рішень. TFLite має ширший діапазон сумісності пристроїв, а також набору інструментів для перетворення, оптимізації та підвищення продуктивності моделі. Крім того, TFLite існує в мобільному просторі трохи довше, що надає йому більш зрілу екосистему та кращу підтримку спільноти.

Дослідження ефективності використання нейронної мережі на мобільному пристрої

Метою експерименту є дослідження можливості ефективного застосування локального розпізнавання Deepfake зображень на мобільному пристрої. До кроків експерименту належать навчання нейронної мережі на ПК для розпізнавання підроблених зображень, оцінка точності результуючої моделі, експортування її в зменшену модель для застосування на мобільному пристрої, порівняння точності вихідної та експортованої моделі, різницю в розмірі файлів, оформлення висновків.

За статистикою середній розмір додатку на сервісі Google Play не перевищує 60 МБ. Показник результуючого розміру файлів нейронної мережі є дуже важливим для дослідження, адже ефективність нейронної мережі може залежити від кількості шарів та параметрів, а це означає більший розмір вихідного файлу моделі. Налаштування експортування моделі може також вплинути на її розмір. Метою є мінімізувати вихідний розмір моделі, при цьому зберігши її точність.

Архітектура нейронної мережі

Для проведення експерименту було створено модель з використанням технології Tensor Flow. Розроблена нейромережа базується на моделі EfficientNetB0, яка є вдосконаленою згортковою нейронною мережею, що використовує комплексне масштабування для рівномірного масштабування ширини, глибини та роздільної здатності, досягаючи найсучаснішої точності зі значно меншою кількістю параметрів і обчислювальних витрат. На рис. 2 зображено її верхньорівневу архітектуру.

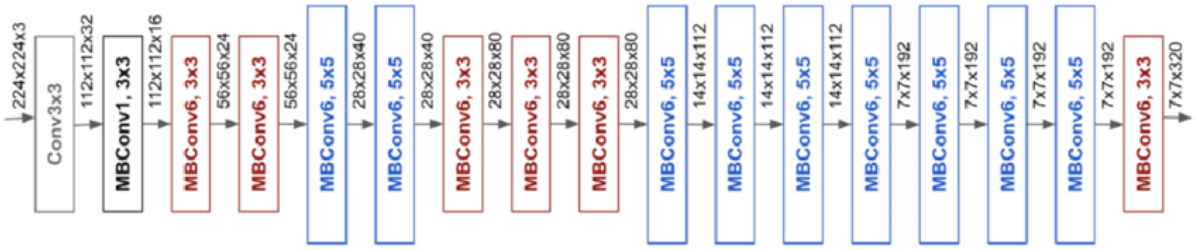

Рис. 2. Архітектура EfficientNetB0

Після EfficientNet додається рівень батч нормалізації для стабілізації і забезпечення вищого темпу навчання, водночас пом'якшуючи ефект перенавчання.

Після пакетної нормалізації в розробленій моделі використовується повністю пов'язаний (Dense) шар, що складається з 512 одиниць. Кожна одиниця, або нейрон, цього шару пов'язаний з усіма активаціями попереднього шару. Цей рівень призначений для вивчення складних шаблонів і комбінацій витягнутих ознак. Він використовує функцію активації «ReLU» (Rectified Linear Unit), яка вводить нелінійність, що дозволяє моделі вивчати складні функції.

Для того щоб ще більше підвищити здатність моделі до узагальнення, було введено шар Dropout після першого шару Dense. Таким чином, при кожному оновленні під час навчання, у частину одиниць введення встановлюється 0, що надалі допомагає запобігти ефекту перенавчання.

Далі йде ще один повністю зв'язаний щільний шар із 128 одиницями та активацією «ReLU», що збільшує глибину моделі та її здатність вивчати більш абстрактні уявлення.

На рис. 3 зображено повну схематичну архітектуру розробленої моделі.

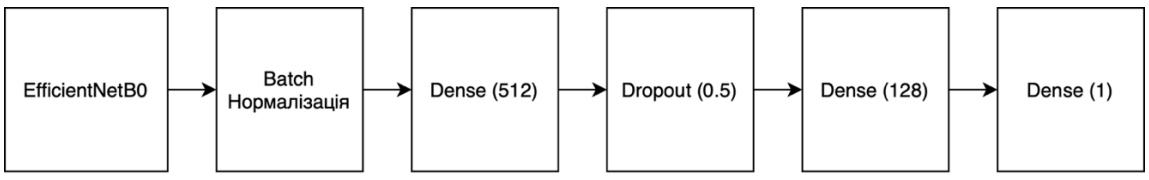

Рис. 3 Повна архітектура розробленої моделі

Архітектура моделі була описана мовою програмування Python з використанням бібліотеки "tensorflow.keras.models", " tensorflow.keras.layers" та інших.

Вхідні дані експерименту

Для навчання та перевірки роботи нейромережі було застосовано датасет FaceForensics++. Цей датасет включає 1000 оригінальних відео, довжиною 2-3 секунди, а також три окремих категорії підробок: FaceSwap, Face2Face та Deepfake. На рис. 4 зображено приклади вирізаних ділянок із обличчями з відео різних датасетів (оригінального та підробок).

Рис. 4. Приклад оригінального, Deepfakes, Face2Face та FaceSwap зображення

Відео всіх категорій було автоматично оброблено: з кожного окремого відео було обрано 2-3 випадкових фрейми, із яких було вирізано ділянки з обличчями. Таким чином, було отримано готовий датасет, який було поділено на дані для навчання, оцінювання та перевірки.

Повний перелік вхідних даних та налаштувань можна побачити в табл. 1. Можна побачити розміри датасетів для двох класів навчання (оригінальні та підроблені зображення). Слід зазначити, що датасет підроблених зображень складається з приблизно однакової кількості зображень отриманих, при цьому застосовуються три технології: Deepfakes, Face2Face та FaceSwap. Також слід додати, що зображення для оцінки та перевірки є унікальними по відношенню до тих, що використовуються для навчання.

Таблиця 1

Вхідні дані та налаштування навчання

Назва параметру	Значення
Оригінальних зображень для навчання	2945 зобр.
Підроблених зображень для навчання	6008 зобр.
Оригінальних зображень для оцінювання	150 зобр.
Підроблених зображень для оцінювання	150 зобр.
Оригінальних зображень для перевірки	82 зобр.
Підроблених зображень для перевірки	83 зобр.
Розмір пакету начання	34 зобр.
Кількість епох навчання (максимум)	20 епох
Допустима кількість епох із незмінним значенням втрати	5 епох

Слід додати, що розмір пакету навчання був підібраний після декількох попередніх спроб навчання. При розмірі пакету 34 було досягнуто найвищу точність моделі.

Вихідні дані навчання моделі

Після проведення навчання було отримано результати, що наведені в табл. 2.

Таблиця 2

Результати навчання моделі

Назва параметру	Значення
Кількість епох (результуюча)	11 епох
Кількість епох (ефективна)	6 епох
Втрата кроку навчання (step loss)	0.2027 од.
Втрата оцінювання	0.1436 од.
Точність оцінювання	0.9455 од.
Загальна точність (поріг 0.21)	95%
Розмір файлу моделі	57.9 МБ

З результатів навчання, що наведено в табл. 2, можна побачити, що результуюча кількість епох відрізняється від ефективної. Через те, що було встановлено параметр "терпіння незмінного значення втрати" у 5 епох, зупинка навчання фактично виникла після 6 епохи, а останні 5 епох було проігноровано.

Слід звернути увагу, що загальна точність моделі при встановленому порогу 0,21 (тобто, зображення із вихідним показником моделі менше за 0,21 вважаються підробленими) – 95 %, що є досить високим показником, враховуючи різноманітність датасету для навчання та перевірки.

Не менш важливим є розмір вихідного файлу моделі у 57,9 МБ. Порівнюючи це значення із середнім загальним розміром додатків на GooglePlay (60 МБ) можна дійти висновку, що модель такого розміру не може бути оптимальною для використання у додатку на мобільному пристрої.

Дослідження методів зменшення моделі

Наступним кроком експерименту є експортування моделі в сумісний з мобільними платформами формат. Експортування проводилося за допомогою скрипта мовою програмування Python із використанням бібліотеки "tensorflow.lite". Наведемо приклад коду, який перетворює файл проміжного етапу навчання (розширенням ".h5") у файл розширенням ".tflite", що є сумісним із мобільними пристроями:

```
epoch_result.save('/path/to/dir', save_format='tf')
loaded_model = tf.keras.models.load_model('/path/to/dir ', custom_objects=None,
compile=True)
converter = tf.lite.TFLiteConverter.from_keras_model(loaded_model)
tflite_model = converter.convert()
```

У результаті було отримано файл експортованої моделі розміром 18,9 МБ. Це приблизно в три рази менше за оригінальний файл. Необхідно перевірити точність роботи експортованої моделі на мобільному пристрої.

Для експериментальної перевірки точності роботи моделі було створено додаток під операційну систему Android мовою програмування Kotlin. Бібліотека, що застосовувалась для завантаження моделі, – "org.tensorflow:tensorflow-lite-support:0.2.0". Кроки підготовки зображень перед потраплянням до нейромережі та інші налаштування – ідентичні до тих, що застосовувалися на ПК. Окрім цього, було завантажено ідентичний набір підроблених та оригінальних зображень, які застосовувалися для перевірки моделі на ПК. Таким чином, середовища проведення експерименту максимально схожі та відрізняються тільки платформою виконання та файлом моделі.

У результаті було отримано точність моделі – 94,3 %. Бачимо зменшення показнику, по відношенню до 95 % на оригінальній моделі.

Основним методом зменшення розміру моделі при експорті є квантизація:

```
converter = tf.lite.TFLiteConverter.from_keras_model(loaded_model)
converter.target_spec.supported_types = [tf.float16]
```

При додаванні налаштування квантування float16, розмір моделі залишився незмінним – 18,9 МБ, при цьому точність залишилася 94,3 %.

Наступним кроком було прибрано квантування float16 та додано вбудований флаг tf.lite.Optimize.DEFAULT. Цей тип квантування під час перетворення статично квантує лише ваги моделі від числа з плаваючою комою до цілого, що забезпечує 8-бітну точність:

```
converter = tf.lite.TFLiteConverter.from_keras_model(loaded_model)
converter.optimizations = [tf.lite.Optimize.DEFAULT]
```

Застосувавши цей метод квантування, розмір моделі зменшився до 5,3 МБ, що є дуже гарним показником для застосування в мобільному додатку, порівняно з оригінальними 57,9 МБ. Однак точність при цьому зменшилася до 93,6 %.

Останнім кроком було поєднано обидва налаштування квантування з метою підвищити точність:

```
converter = tf.lite.TFLiteConverter.from_keras_model(loaded_model)
converter.optimizations = [tf.lite.Optimize.DEFAULT]
converter.target_spec.supported_types = [tf.float16]
```

Такий комбінований підхід досягнув розміру моделі в 9,5 МБ, при цьому точність повернулася до оригінальної – 94,3 %. Таким чином, було збережено точність, при цьому мінімізувавши розмір файлу. На рис. 5 можна побачити співвідношення моделі до вихідного розміру.

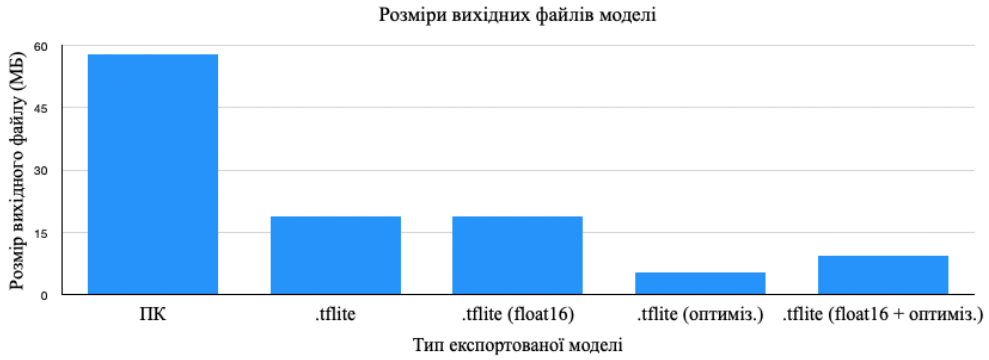

Рис. 5. Відношення типу вихідної моделі до розміру файлу

Схожим чином можна наглядно продемонструвати залежність точності моделі від її типу (рис. 6).

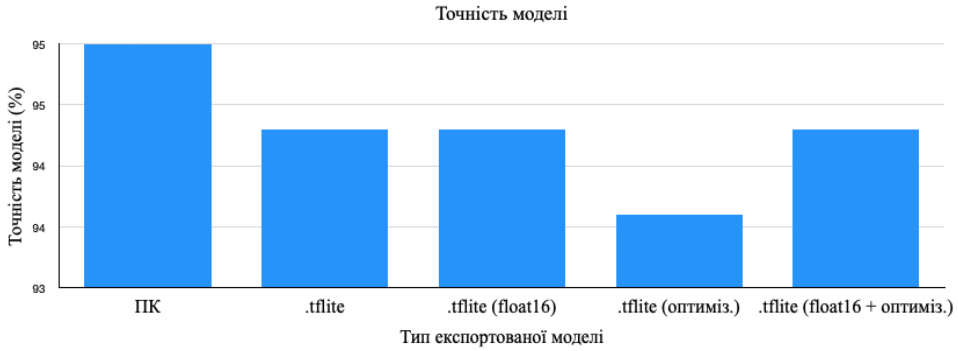

Рис. 6. Відношення типу вихідної моделі до її точності

Виходячи з цих даних, можна зробити висновок, що при застосуванні на мобільній платформі можливий варіант експортування моделі з мінімальною втратою її точності, при цьому застосовується квантування зменшення розміру вихідного файлу в 6-11 разів. Втрата

точності при цьому може погано вплинути на надійність розпізнавання deepfake, але максимальне зменшення розміру вихідного файлу надасть можливість створювати ансамблі з декількох моделей різної архітектури, що значно підвищить надійність системи.

Висновки

Проаналізовано основні методи створення Deepfake, до яких належать автокодери та мережі GAN, а також методи їх виявлення, такі як глибоке навчання, машинне навчання та методи, що базуються на статистичних вимірюваннях. Завдяки існуючим дослідженням було знайдено оптимальний підхід до створення моделі розпізнавання Deepfake.

Експериментально досліджено ефективність та доцільність локального розпізнавання Deepfake у системах біометричної автентифікації за обличчям на мобільних пристроях – отримано оптимістичні результати щодо точності та фізичного розміру файлу моделі. У результаті можна сказати, що застосування локальних методів розпізнавання Deepfake на мобільному пристрої є цілком реальним та надійним методом протидії атакам на системи біометричної автентифікації та загалом перевірки відображеного матеріалу на предмет підробки.

Список літератури:

1. Мироненко Є.В., Сєвєрінов О.В. Біометрична ідентифікація і автентифікація особи за геометрією обличчя. Харків : НТУ «ХПІ», 2020.

2. Passport-Photo-Biometric-Statistics [Електронний ресурс]. Режим доступу: https://passport-photo.online/blog/biometric-statistics/#gref

3. Creswell, Antonia; White, Tom; Dumoulin, Vincent; Arulkumaran, Kai; Sengupta, Biswa; Bharath, Anil A. (2018). Generative Adversarial Networks: An Overview // IEEE Signal Processing Magazine, 35(1), 53–65. DOI: 10.1109/MSP.2017.2765202

4. Hady A. Khalil;Shady A. Maged; (2021). Deepfakes Creation and Detection Using Deep Learning // International Mobile, Intelligent, and Ubiquitous Computing Conference. 2021. DOI: 10.1109/MIUCC52538.2021.9447642

5. Zainab Zahid, Ammar Haider, Nosheen Sabahat, Asim Tanwir. Vulnerabilities in Biometric Authentication of Smartphones // 2020 IEEE 23rd International Multitopic Conference (INMIC). 2020. DOI: 10.1109/inmic50486.2020.9318094

Надійшла до редколегії 10.09.2023

Відомості про авторів:

Долганенко Олександр Денисович – студент кафедри безпеки інформаційних технологій; Харківський національний університет радіоелектроніки; Україна; e-mail: sasha.dolganenko@gmail.com; ORCID: http://orcid.org/0000-0002-3996-6940

Сєвєрінов Олександр Васильович – канд. техн. наук, доцент, Харківський національний університет радіоелектроніки, професор кафедри безпеки інформаційних технологій, факультет комп'ютерної інженерії та управління, Україна; e-mail: oleksandr.sievierinov@nure.ua; ORCID: https://orcid.org/0000-0002-6327-6405

В'юхін Даніїл Олександрович – старший викладач кафедри безпеки інформаційних технологій, факультет комп'ютерної інженерії та управління; Харківський національний університет радіоелектроніки, Україна; e-mail: daniil.viukhin@nure.ua; ORCID: https://orcid.org/0009-0009-8442-9587

Коцюба Василь Петрович – канд. техн. наук, доцент, , викладач кафедри радіоелектронних систем пунктів управління Повітряних Сил,факультет автоматизованих систем управління та наземного забезпечення польотів авіації; Харківський національний університет Повітряних Сил імені Івана Кожедуба, Україна; e-mail: vasyl.kotsyuba@ gmail.com; ORCID: https://orcid.org/0000-0001-6336-8193

Крепко Алла Василівна – молодший науковий співробітник науково-дослідної лабораторії, факультет автоматизованих систем управління та наземного забезпечення польотів авіації; Харківський національний університет Повітряних Сил імені Івана Кожедуба Україна; e-mail: krepkoalla@ukr.net; ORCID: https://orcid.org/0009-0005-1659-4944