Міністерство освіти і науки України
Національний університет «Одеська політехніка»
Україно-польський інститут
Кафедра кібербезпеки та програмного забезпечення

Гулич Володимир Володимирович,
студент групи ЛРУ-181

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

Розробка програмного забезпечення для виявлення масштабування графічного контенту

Спеціальність:
125 Кібербезпека

Спеціалізація, освітня програма:
Управління кібербезпекою

Керівник:
Зоріло Вікторія Вікторівна,
ст. викл.

Одеса – 2022

Міністерство освіти і науки України

Національний університет «Одеська політехніка»

Україно-польський інститут

Кафедра кібербезпеки та програмного забезпечення

Рівень вищої освіти перший (бакалаврський)

Спеціальність 125 – Кібербезпека

Освітня програма – Кібербезпека

ЗАТВЕРДЖУЮ

Завідувач кафедри КБПЗ

_____

д.т.н.,проф. А.А.Кобозєва

___ _____202_р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

Гуличу Владиславу Володимировичу

1.Тема роботи: *Розробка програмного забезпечення для виявлення масштабування графічного контенту*, керівник роботи *Зоріло Вікторія Вікторівна, ст. викл.,* затверджені наказом ректора від „17” 05 2022 р. №168-в

2.Зміст роботи*: аналіз сучасного стану досліджень, проведення експерименту, реалізація програмного забезпечення, охорона праці.*

3. Перелік ілюстративного матеріалу: *оригінальні та фальсифіковані зображення, результати роботи методів, тестування програмного забезпечення.*

4. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
Охорона праці	Ярова І. А. к.т.н., доцент	Завдання видав	Завдання рийняв

5. Дата видачі завдання "_____"_____20___ р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання	Примітка
1	*Аналіз джерел з теми випускної кваліфікаційної роботи*	*20.03.2022*	*виконано*
2	*Обробка отриманих даних*	*26-03-2022*	*виконано*
3	*Експеримент з алгоритмом*	*10-04-2022*	*виконано*
4	*Аналіз і вибір засобу реалізації програмного забезпечення*	*15-04-2022*	*виконано*
5	*Розробка програмного забезпечення*	*15-05-2022*	*виконано*
6	*Підготовка тексту роботи*	*28-05-2022*	*виконано*
7	*Підготовка презентації та доповіді*	*29-05-2022*	*виконано*
8	*Попередній захист*	*03-06-2022*	*виконано*
9	*Нормоконтроль, рецензування*	*17-06-2022*	*виконано*

Здобувач вищої освіти _____ *Гулич В.В.*

Керівник роботи _____ *Зорило В.В.*

ЗАВДАННЯ

на розробку розділу «Охорона праці» у кваліфікаційній роботі бакалавра

студенту Гуличу Владиславу Володимировичу, група ЛРУ-181

Українo-польський інститут

Кафедра кібербезпеки та програмного забезпечення

Дата отримання завдання 11.05.2022

Консультації 19.05.2022, 26.05.2022

Дата закінчення розділу 15.06.2022

Тема роботи *Розробка програмного забезпечення для виявлення масштабування графічного контенту*

Зміст розділу:

1. Аналіз умов праці і вибір основних заходів виробничої безпеки
2. Аналіз пожежної безпеки і вибір заходів і засобів пожежної безпеки

Керівник дипломної роботи

_____ Зоріло В.В.

(підпис)

« __ » _____2022 р.

Консультант з охорони праці

_____ Ярова І.А.

(підпис)

« __ » _____ 2022 р.

АНОТАЦІЯ

Кваліфікаційна робота на тему: «Розробка програмного забезпечення для виявлення масштабування графічного контенту» на здобуття бакалаврського рівня вищої освіти за спеціальністю 125 Кібербезпека спеціалізація «Управління кібербезпекою». Дана робота включає 21 Рисунок та 3 таблиці, 1 додаток та 17 джерел з переліку посилань. Кваліфікаційна робота виконана на 46 сторінках загального тексту та 43 сторінках основного тексту.

Мета даної роботи полягає в проведенні експерименту з методом аналізу рівня помилок для виявлення масштабування з від'ємним коефіцієнтом окремих частин зображення та створенні боту який буде містити в собі реалізацію цього методу. У роботі було проведено аналіз можливих фальсифікацій зображення та методів їх виявлення. У результаті аналізу виявлено що окремо виділеного методу не існує та вирішено провести експеримент з існуючим методом аналізу рівня помилок, який інтегрований в бота в месенджері Telegram.

У результаті роботи було створено бота в месенджері Telegram, в який було інтегровано метод аналізу рівня помилок. Після чого експериментально підтверджено можливість використання методу для пошуку масштабованих частин зображення. Тож поставлені завдання для кваліфікаційної роботи виконано.

ЦИФРОВЕ ЗОБРАЖЕННЯ, МАСШТАБУВАННЯ ЗОБРАЖЕННЯ, ЗМЕНШЕННЯ ОБ'ЄКТУ,ФАЛЬСИФІКАЦІЯ ЗОБРАЖЕНННЯ, ЦИФРОВА КРИМІНАЛІСТИКА.

ANNOTATION

This qualification work on the topic: "Development of software for graphic content scaling detection" for a bachelor's degree in higher education in the specialty 125 Cybersecurity specialization «Cybersecurity Management». This work includes 21 figures and 3 tables in the main text, 1 appendix and 17 sources from the list of links. Qualification work performed on 46 general text pages and 43 pages of the main text.

The purpose of this work is to conduct an experiment with the method of error level analysis to detect scaling with a negative coefficient of individual parts of the image and create a bot that will include the implementation of this method. The analysis of possible image falsifications and methods of their detection was carried out in the work. As a result of the analysis, it was found that a separate method does not exist and it was decided to conduct an experiment with the existing method of error level analysis, which is integrated into the bot in the Telegram messenger.

As a result, a bot was created in the Telegram messenger, which integrated the method of error level analysis. Then the possibility of using the method to find scalable parts of the image was experimentally confirmed. Therefore, the set tasks for the qualification work have been fulfilled.

DIGITAL IMAGE, IMAGE SCALING, OBJECT REDUCTION, IMAGE FALSIFICATION, DIGITAL CRIMINAL SCIENCE.

ЗМІСТ

ВСТУП

Сучасний світ дедалі більше стає залежним від зображень. Оскільки передача та засвоєння інформації відбувається значно легше за допомогою зображень. Тому одним з головних питань постає оригінальність цього зображення, адже існує безліч способів фальсифікувати зображення, наприклад клонування областей зображення, ретушування та накладання фільтрів, накладання сторонніх об’єктів на зображення та масштабування областей на зображенні.

Тому мета даної роботи полягає в розробці програмного забезпечення для виявлення масштабування зображення. Оскільки деякі методи вдається адаптувати до нових завдань, то було вирішено провести експеримент обраного методу та інтегрувати його в бот месенджеру Telegram. Основними завданнями кваліфікаційної роботи стали:

1) Аналіз сучасного стану досліджень в області виявлення фальсифікацій зображень.
2) Проведення експерименту з обраним методом аналізу рівня помилок.
3) Реалізація алгоритму та інтегрування з ботом.

Результатом даної кваліфікаційної роботи є бот в месенджері Telegram, який виконує такі функції як:

- Прийом зображень для перевірки;
- Обробка зображення інтегрованим алгоритмом;
- Видача кінцевого результату користувачеві.

1 АНАЛІЗ СУЧАСНОГО СТАНУ ДОСЛІДЖЕНЬ В ОБЛАСТІ ВИЯВЛЕННЯ ФАЛЬСИФІКАЦІЙ ЗОБРАЖЕННЯ

1.1 Методи виявлення клонування областей на цифровому зображенні

Коли ми говоримо про клоновані області на цифровому зображенні зазвичай маються на увазі певні ідентичні об’єкти, які можуть повторюватися. Але це не зовсім так, оскільки методом клонування областей на зображенні можна видалити певні області, які не потрібні користувачу на зображенні. Суть в тому, що даний метод дозволяє точно скопіювати ділянку зображення та закрасити його пензликом в новому місці. Цей інструмент в основному використовують для видалення небажаних об’єктів на зображенні чи пом’якшені тіней чи текстур, але також даний метод має негативні наслідки, в вигляді фальсифікації оригінального зображення, що може призвести до негативних наслідків та втрати авторських прав на зображення, оскільки воно вже інше.

Саме тому було створено методи для того, щоб виявляти фальсифіковані, тобто клоновані області на цифрових зображеннях. Наприклад в 2019 А. К. Чакраверті було представлено метод гібридного підходу до пошуку клонованих об’єктів при копіюванні та переміщені підробленого зображення[1]. В цій роботі було запропоновано новий гібридний підхід до пошуку клонованого об’єкту на зображенні. Оскільки при копіюванні-переміщенні зображення підробляються шляхом створення клонів об’єкту всередині одного зображення, то було об’єднано два сучасних методи, які називаються методом художнього прийому “adhoc” та методом аналізу основних компонентів (principle component analysis (PCA)), який засновано на методі масштабно-інваріантного перетворення ознак (scale invariant feature transform (SIFT)). В двоетапному процесі було оброблено зображення об’єкту за допомогою запропонованого методу динамічного розміру блоку, заснованого на модифікації локальної контрастності та адаптивної корекції гістограми з обмеженням контрасту до бажаного PSNR та рівня сірого, а після інтегрування adhoc та PCA на основі SIFT для розробки гібридного підходу. На рисунку 1.1, який наведено нижче можемо бачити результат роботи методу.

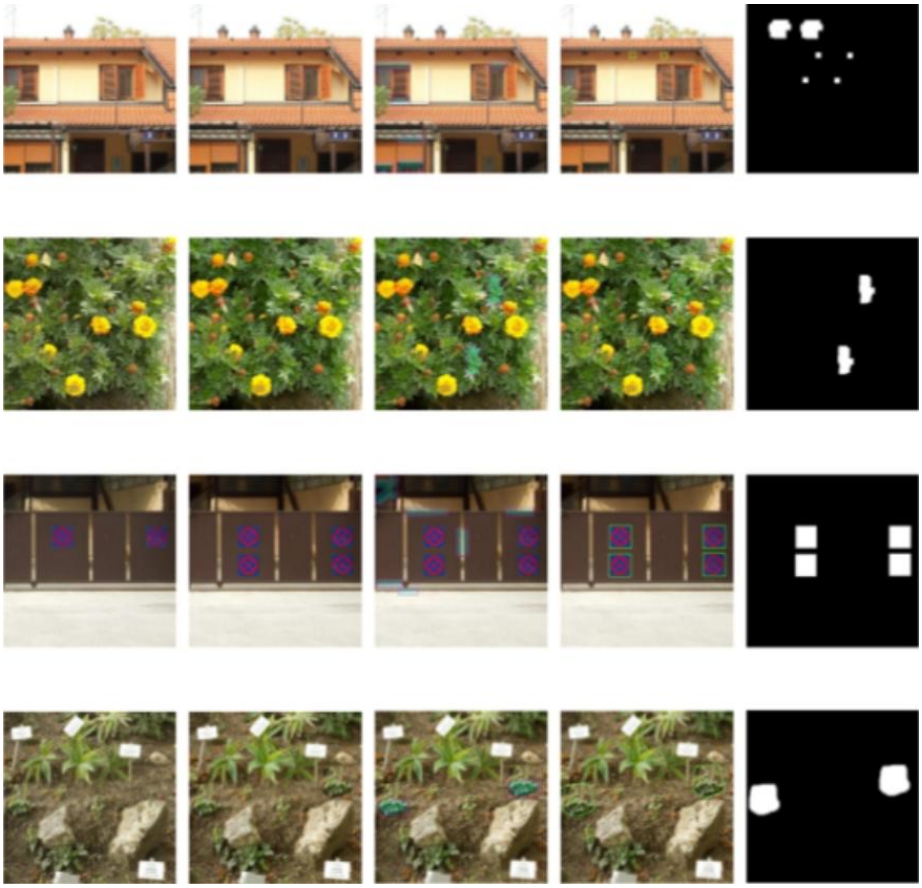

Рисунок 1.1 – результати роботи методу гібридного підходу

Як ми бачимо, результати методу досить точні, що свідчить про те, що метод дійсно має великий потенціал у виявленні фальсифікованих зображень, а саме його клонованих частин.

Також для виявлення копіювання областей був представлений алгоритм Е.А. Армас Вега під назвою «Метод виявлення підробок методом копіювання-переміщення, що заснований на характеристиках блоків дискретного косинусного перетворення» [2]. Цей метод базований на методі Фрідріха. Алгоритм дій полягає в наступному:

- Перетворити зображення в відтінки сірого;
- Розділити зображення на невеликі перекриваючі блоки розміром $n \times n$;

- Вирахувати DCT-перетворення кожного блоку, відсортувати коефіцієнти зигзагоподібним чином та обрізати список щоб від містив k перших елементів;
- Провести лексикографічне сортування обрізаних списків коефіцієнтів і для кожного списку вирахувати міру схожості між його ближчими сусідами. Якщо схожість нижче порогу, то блоки вважаються ідентичними;
- Для кожної пари визначається міра переносу. Якщо кількість векторів в заданому напрямку перевищує передбачену кількість, то кожен блок розглядається як частина фальсифікації копіювання-переміщення.

На рисунку 1.2 наведено результати пошуку фальсифікації методом копіювання-переміщення за допомогою представленого методу.

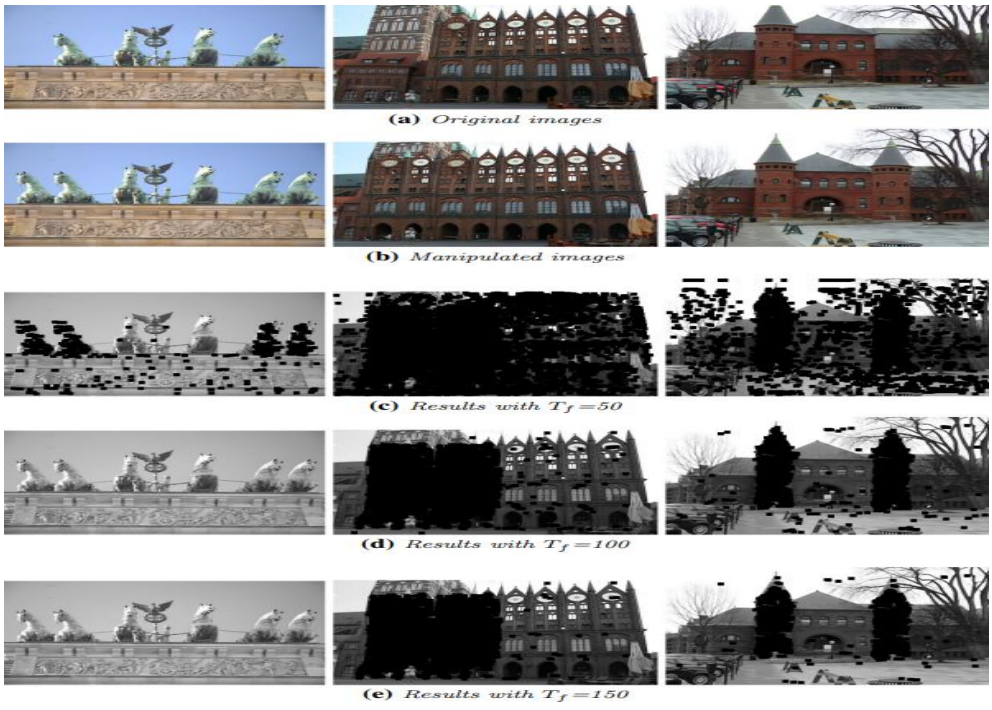

Рисунок 1.2 – результати методу заснованого на характеристиках блоків дискретного косинусного перетворення

Отож метод, що представлено Е.А. Армасом Вега підходить для пошуку клонованих об’єктів на цифровому зображенні. Також по заявлені автору його

метод має середню точність 97,88% в порівняні з методом на який він опирається – 64,53%.

1.2 Методи виявлення ретуші та накладених фільтрів на зображенні

Ретушування зображень в сучасних реаліях життя знайшло своє місце майже в кожному куточку де ці зображення використовуються. Наприклад на обкладинках журналів можна нерідко бачити неймовірно красиві та бездоганні фото, які в своїй більшості є результатами ретушування, що створює ідеалізоване та нереалістичне представлення щодо зображень. Тому виявлення такого роду фальсифікації зображення стає дедалі необхіднішим, для того, щоб інформувати користувачів про те наскільки зображення було відхилене від реальності, а фоторедакторам дати можливість відстежувати наскільки вони перебільшили зі змінами. Таким чином було запропоновано метрику сприйняття ретуші зображень [3], яка показує наскільки зображення лиця та тіла було схилене до спотворення. Ця метрика була розроблена Е. Кі та Х. Фарідом., вона кількісно визначає вплив геометричних і фотометричних модифікацій на сприйняття шляхом розповсюджених методів ретуші зображення. Масштаби маніпуляції з зображеннями кількісно оцінюються за допомогою восьми зведених статистичних даних, які вилучені з певних моделей. Величини геометричних модифікацій виявляються через дані середніх значень і стандартних відхилень величини руху, що були розраховані окремо для обличчя та тіла. Величина фотометричої модифікації кількісно визначається середнім та стандартним відхиленням простору протяжності фільтрів згладжування або ж підвищення різкості та середнє та стандартне відхилення показника схожості SSIM. Автори метрики виявили, що ці показники можна використовувати для автоматичної оцінки ступеню ретуші зображення. На рисунку 1.3 приведено результати роботи запропонованої метрики.

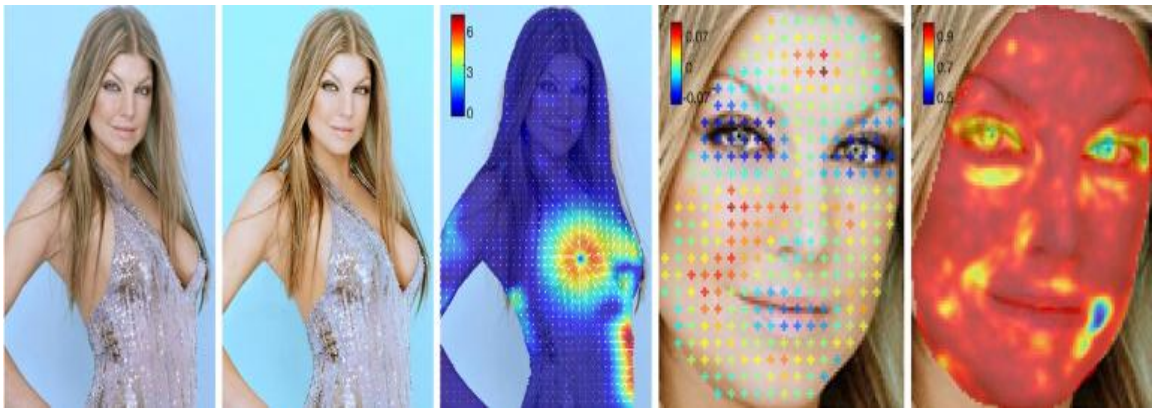

Рисунок 1.3 – результати роботи метрики сприйняття ретуші зображення

Також завдяки сучасним сервісам для розміщення своїх зображень, користувачі мають змогу накласти певні фільтри на своє зображення, таким чином змінюючи його що в багатьох випадках унеможливлюють його відновлення при втраті оригіналу. Ще однією проблемою при накладанні фільтрів є те, що після цього значно знижується точність сучасних систем розпізнавання. Для вирішення цієї проблеми була представлена робота від С. Б'янко та інших [4]. В цій роботі вони описали метод, який базується на нейронній мережі, що буде автоматично видаляти фотографічні фільтри. В якості вхідних даним метод приймає кольорове зображення, яке може бути оброблене фотографічним фільтром і в якості вихідних даних формує також кольорове зображення, але з відтворенням «природніх» кольорів. Особлива увага приділяється тому, що при внесенні вхідних даних, ніякої інформації про те, який фільтр використовується не надається. Як завіряють автори нейронної мережі її точність сягає 98,9% та точність розпізнавання фільтрів сягає 97%. Нейронна мережа видає результат майже без втрат, так як автори проекту мінімізували середньоквадратичну помилку між бажаним та вихідними даними, оскільки деталі зберігаються за рахунок локальних перетворень. Тому дана розробка значно спрощує процес виявлення фотофільтрів на зображенні, та дає змогу побачити майже оригінальне зображення, а на рисунку 1.4 наведено результати роботи нейронної мережі.

Рисунок 1.4 – результат роботи нейронної мережі С. Б’янко та ін. (зліва-направо оригінальне зображення, зображення з фотофільтром, вихідні результати)

1.3 Метод виявлення доданих фрагментів на зображення

В 2010 році П. Рінгвуд створив сервіс в вигляді веб-сайту під назвою “errorlevelanalysis.com”. Але в 2012 році авто зачинив сайт. Після чого Hacker Factor відтворили веб-сайт Рінгвуда [5]. На сайті було створено алгоритм, який оцінює потенційний рівень помилок зображення в форматі JPEG, тобто вимірюється кількість змін під час повторного зберігання. Оскільки при редагуванні зображення змінені частини будуть мати потенційний рівень помилок, який виділяється від іншої частини зображення, то даний алгоритм буде вказувати на редагування у вигляді потенційно високого рівня помилок. На рисунку 1.5 показано результат роботи алгоритму аналізу рівня помилок, на зображення був доданий напис.

Рисунок 1.5 – результат роботи алгоритму аналізу рівня помилок

1.4 Методи виявлення масштабування областей зображення

Використання методу масштабування областей на зображенні може унеможливити розпізнавання певних об’єктів на зображенні чи завадити повноцінному представленню того, що саме зображено чи як далеко від інших об’єктів. Тому в 2014 році Трифоновою К.О. було запропоновано метод локалізації та ідентифікації контекстно-залежного масштабування в цифровому зображенні [6]. Автором було розроблено алгоритм дій для подальшого знаходження масштабованих областей зображення. А саме:

- Побудувати матрицю зображення;
- Розбити отриману матрицю на блоки 8×8;
- Побудувати для матриці зображення матрицю нульових сингулярних чисел в основі яких лежить метод визначення максимальної кількості нульових сингулярних чисел;
- Виявлення областей: виділяються області в яких:

  а) більшість елементів мають нульове значення;

  б) більшість елементів мають ненульове значення;
- Ідентифікація:

  а) області нульових значень – результат копіювання та вставки в із зображення;

  б) області ненульових значень – прикордонний контур, що свідчить про дію контекстно-залежного масштабування;

Алгоритм було протестовано на 200 зображеннях та отримано статистику помилок першого та другого роду, що склали 3,5% та 7,2% відповідно. Даний алгоритм був представлений для виявлення масштабування з позитивним коефіцієнтом, тобто більшим 1. Але алгоритмів для виявлення масштабування з коефіцієнтом менше 0 не було знайдено. Але все ж таки деякі алгоритми можна адаптувати для пошуку областей на зображенні, які були масштабовані, а саме зменшені.

2 АЛГОРИТМ ERROR LEVEL ANALYSIS ТА РЕЗУЛЬТАТИ ЕКСПЕРИМЕНТІВ

2.1. Алгоритм Error lever analysis (ELA)

Даний алгоритм, про який було розказано в першому розділі представляє собою інструмент для оцінки потенційного рівня помилок в форматі JPEG. Алгоритм вимірює кількість змін під час повторного зберігання в форматі JPEG. Коли цифрове зображення редагується, то частини які були редаговані будуть мати потенційний рівень помилок, який відрізняється від іншої частини зображення. Стики, малювання та суттєві правки на зображенні будуть зазвичай відрізнятися від основної частини зображення, що свідчить про суттєво відмінність в потенційному рівні помилок. Оскільки цей алгоритм працює з форматом JPEG та його властивостями варто відзначити алгоритм його стискання[7]. Якщо коротко, то:

- зображення перетворюють с формату RGB в YCbCr;
- після цього канали Cb та Cr проріджують, тобто блоку пікселів привласнюють середнє значення;
- згодом значення каналів розбивають на блоки 8×8;
- кожен блок піддається дискретно-косинусному перетворенню (ДКП), що є різновидністю дискретного перетворення Фур'є. Отримується матриця коефіцієнтів 8×8. Коефіцієнт з нульовим номером – так званий DC-коефіцієнт, який є найголовнішим і є усередненим значенням всіх значень. Інші 63 коефіцієнти – AC-коефіцієнти.
- отримані коефіцієнти квантуються, тобто кожен множиться на коефіцієнт матриці квантування (зазвичай однієї матриці не існує. Кожен кодувальник використовує свою матрицю квантування);
- після чого квантовані коефіцієнти кодуються кодами Хаффмана.

Саме на цьому принципі і ґрунтується алгоритм аналізу рівня помилок. Отже, якщо на зображенні видимі принципіальні різні артефакти стискання, то це свідчить про додання елементів на зображення. Іншими словами, якщо на

зображенні є елементи неоднорідності артефактів стискання, то зображення редагували, наприклад в фоторедакторах.

Масштабування об’єктів зображення за допомогою програмного продукту компанії Adobe, а саме програмного забезпечення Adobe Photoshop відбувається за допомогою ресамплінгу або ж предискретизацією. Ресамплінг[8] (агнл. – resampling) – зміна об’єму даних зображення при зміні його розмірів в пікселях. При зменшенні об’єкту він буде втрачати частину інформації, наприклад через викидання певних рядків цього об’єкту для подальшого зменшення розмірів це називається даунсамплінг (англ. –downsampling). Через це яскравість та контрастність об’єкту масштабування буде змінюватися згідно обраному алгоритму в налаштуваннях продукту. Тому було вирішено провести експеримент з використанням методу аналізу рівня помилок.

2.2 Підготовка до експерименту

Оскільки алгоритм працює із зображеннями в форматі JPEG, то потрібно зібрати базу зображень в даному форматі. База зображень була зібрана з відкритих джерел, а саме відкритого телеграм-каналу «Збройні Сили України. Війна з окупантами»[9]. Було обрано 100 зображень, та завантажено для їх подальшої обробки. На рисунку 2.1 показано базу зображень.

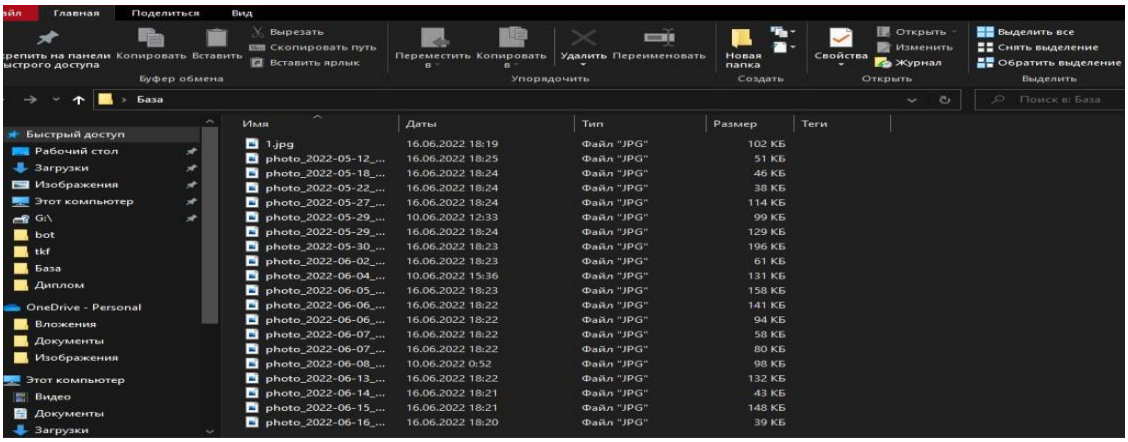

Рисунок 2.1 – база завантажених зображень

Всі зображення мають різний розмір, роздільну здатність та всі збережені без втрат. Після цього всі зображення редагуються за допомогою програмного застосунку Adobe Photoshop 2022. Алгоритм редагування зображень такий:

1. Завантажуємо зображення в додаток «Файл-Відкрити-Обираємо потрібне зображення». Обираємо профіль кольорів «Без змін»;
2. За допомогою інструменту «Виділення об’єктів» виділяємо потрібний об’єкт на зображенні;
3. Копіюємо об’єкт на новий шар «Права клавіша миші на об’єкт-Скопіювати на новий шар»;
4. Робимо невидимим новий шар іконка ока в панелі шарів (зазвичай правий нижній куток);
5. Обираємо оригінальний шар натисканням лівою кнопкою на нього;
6. Знову обираємо об’єкт;
7. Виконуємо заливку виділеної області «Права кнопка миші-Виконати заливку». Параметри заливки: зміст – з урахуванням змісту, режим накладання – нормальний.
8. На панелі шарів обираємо попередньо перенесений туди шар з об’єктом оригінального зображення. Ліва кнопка миші по потрібному шару. Та робимо його видимим. Ліва кнопка миші по іконці ока.
9. Виконуємо масштабування об’єкту за допомогою трансформації. Комбінація клавіш Ctrl+T. Трансформуємо до потрібного вигляду.
10. Зберігаємо отримане зображення. Файл-Зберегти копію…-Обираємо тип файл JPEG-Вводимо назву-Зберегти. (Параметри JPEG: якість – найкраща. Допустимі втрати 1% - менше виставити неможливо). Та натискаємо «ОК».

Пункт під номером 7 є важливим пунктом, оскільки попередню область потрібно зробити подібною до фону на оригінальному зображенні, адже без цього масштабований об’єкт буде знаходитися на фоні оригінального об’єкту. Саме для цього існує інструмент «заливка». Параметри заливки можна обрати в налаштуваннях самої заливки, але в даному випадку виконується заливка з урахуванням змісту. Суть в тому, що дана заливка виконує заповнення обраної

частини зображення таким змістом, який схожий з суміжними ділянками зображення. Для отримання найкращих результатів обрана область повинна захоплювати область яку ми хочемо відтворити на фінальному зображенні.

2.2. Результати експерименту

Після збору бази зображень та їх фальсифікації за допомогою застосунку та описаного методу фальсифікації починаємо перевіряти їх за допомогою алгоритму аналізу рівня помилок. Нижче приклад оригінального та фальсифікованого зображення.

Рисунок 2.2. – оригінальне та фальсифіковане зображення

На рисунку 2.2 представлено оригінальне зображення (зліва) та фальсифіковане з масштабованим об’єктом (праворуч). На фальсифікованому зображенні було зменшено чоловіка в правій частині зображення. А на рисунку 2.3 представлено результати роботи алгоритму з цими зображеннями.

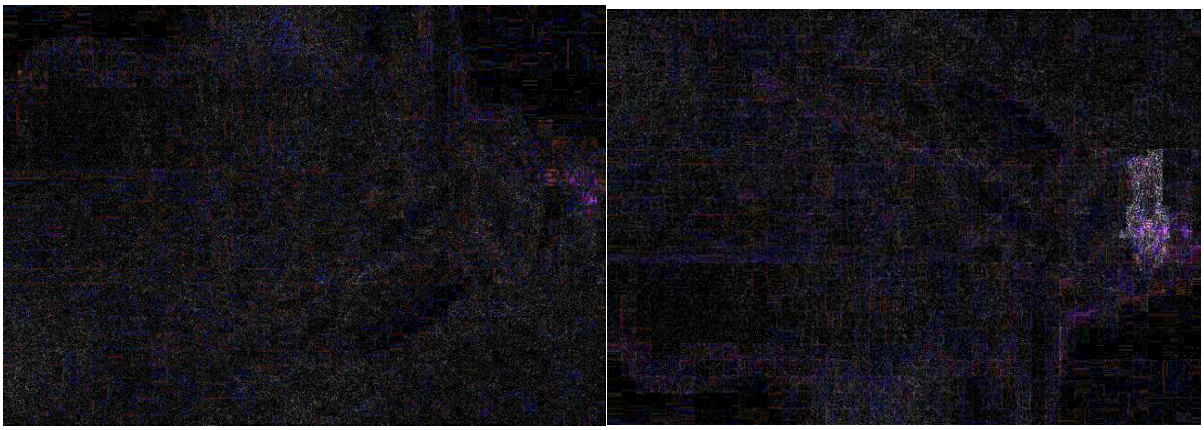

Рисунок 2.3. – результат роботи алгоритму

Як бачимо з прикладу роботи алгоритму на фальсифікованому зображенні явно виділена область масштабування. Тому алгоритм відпрацював на відмінно. Нижче наведена таблиця, в якій будуть відображені результати роботи алгоритму у відсотковому відношенні.

Таблиця 2.1 – результати роботи алгоритму пошуку масштабованого об'єкту

	Артефакти присутні	Артефакти відсутні
Оригінальне зображення	7%	93%
Зображення з масштабованою областю	98%	2%

Аналізуючи таблицю можна вважати, що при обробці оригінального зображення їх більшість не має жодних артефактів, що свідчить про те, що зображення не піддавалося редагуванню. Артефакти присутні на 7% оригінальних зображень, що можна трактувати як похибку при зберіганні чи зменшенню всього зображення за допомогою сторонніх програм. Оскільки це може посилити контрастні контури, роблячи їх набагато яскравішими на виході[10]. На зображеннях з масштабованою областю алгоритм ідентифікував її на 98% зображень. Але на 2% зображень алгоритм не виділив області, які були

масштабовані. Це сталося через те, що на оригінальному зображенні контури після зменшення його роздільної здатності мають більш високий рівень помилок ніж масштабована область.

Також було проведено експеримент з масштабованими та переміщеними об’єктами на зображенні. Масштабування об’єкту відбувається за таким самим алгоритмом, а для його переміщення потрібно:

- Вибрати шар з об’єктом;
- Зробити шар видимим та натиснути на об’єкт;
- після натискання на об’єкт, він виділиться рамкою, що свідчить про те, що можна проводити його редагування. Натисканням та утриманням лівої кнопки миші на об’єкті переміщуємо його на будь-яку частину зображення.

На рисунку 2.4 наведено оригінальне та фальсифіковане зображення. Фальсифікація була створена за допомогою масштабування та переміщення об’єкту.

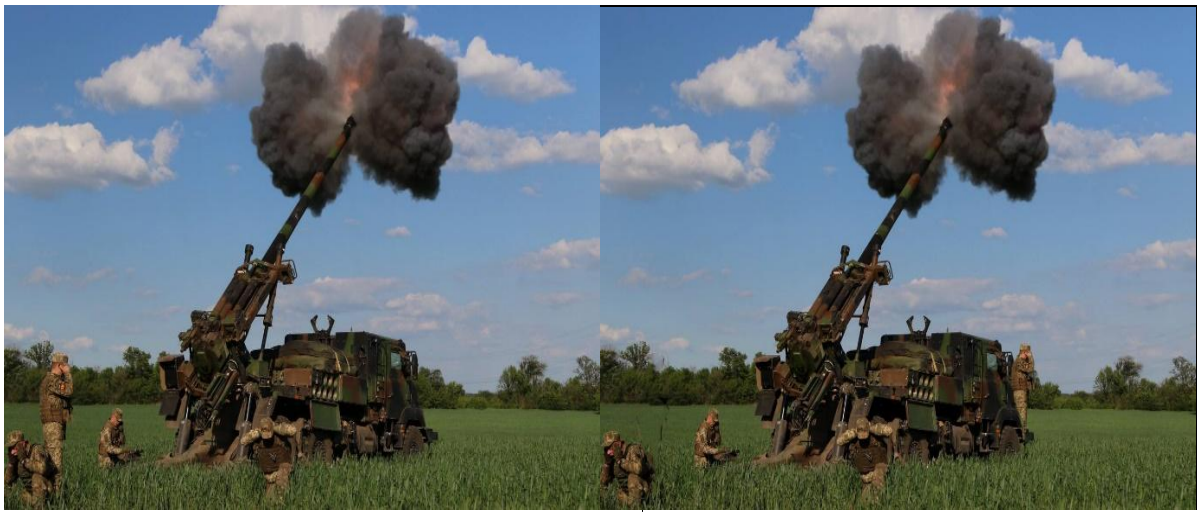

Рисунок 2.4. – оригінальне та фальсифіковане зображення

З рисунку 2.4 видно, що на фальсифікованому зображенні було зменшено чоловіка, який стоїть в лівій частині зображення, але його було переміщено з лівої в праву частину зображення, наперед автомобіля. Оскільки фон, де знаходився чоловік був залитий заливкою з урахуванням змісту, то можна заявити, що дана область буде відмічена алгоритмом як модифікована область, оскільки програма

Photoshop при роботі з будь-якими об’єктами автоматично підвищує контрастність, що робить їх яскравішими, але це не видно людському оку. Так відбудеться і зі зменшеним чоловіком, оскільки його перемістили на область з однією контрастністю, а він сам має висококонтрастний контур через маніпуляції ПЗ. Тому перейдемо до перегляду результату роботи алгоритму. Приклад роботи алгоритму з такими фальсифікаціями представлено на рисунку 2.5.

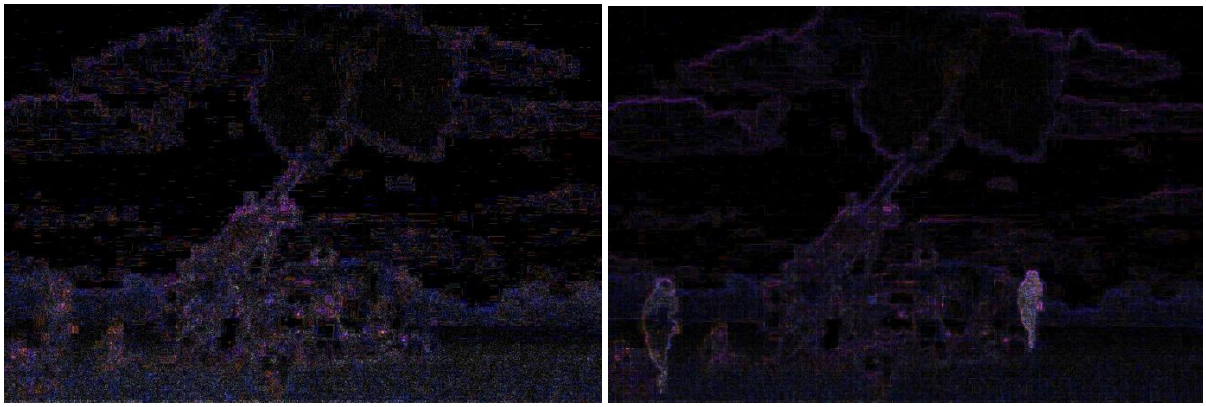

Рисунок 2.5. – результат роботи алгоритму з фальсифікацією масштабування та переміщення

На рисунку 2.5 ми бачимо результат роботи алгоритму. Зліва оригінальне зображення на ньому присутні однотонні контури зображення без видимих артефактів, що свідчить про те, що зображення є оригінальним. Зображення з правого боку має два артефакти зліва видно контур залитої області, а з правого боку наявний об’єкт, який був масштабований та переміщений. Виходячи з цього, говоримо про те, що алгоритм відпрацював та знайшов зменшений та переміщений об’єкт. Для проведення цього експерименту було створено базу з 10 зображень. В таблиці 2.2 сформовано звідну таблицю результатів роботи алгоритму з фальсифікації у вигляді масштабування та переміщення об’єкту.

Таблиця 2.2 – результати роботи алгоритму при пошуку масштабовано-переміщеного об’єкту

Зображення	Артефакти відсутні	Артефакти присутні
Оригінальне	100%	0%
З масштабованим та переміщеним об’єктом	0%	100%

Проаналізувавши таблицю робимо висновок, що алгоритм точно і ефективно знаходить масштабовані об’єкти які були переміщені, та область, де знаходився цей об’єкт на оригінальному зображенні.

Третьою частиною експерименту стало виявлення клонування частин зображення. Клонування об’єкту відбувається за рахунок інструменту «Штамп», який дає змогу копіювати зміст з вихідної ділянки та використовувати його на інших ділянках зображення. Послідовність дій для клонування об’єкту за допомогою даного інструменту:

- відкрити зображення за допомогою Файл-відкрити-обрати зображення-ОК;
- на боковій панелі інструментів обрати інструмент «Штамп»;
- на зображенні затиснувши клавішу Alt, натиснути лівою кнопкою миші на область яку хочемо копіювати;
- переводимо курсор в потрібну область та натискаємо ліву кнопку миші, цим самим додаючи скопійовану область на зображення.

На рисунку 2.6 зображено оригінальне зображення та зображення з клонованою областю за допомогою інструменту «штамп».

Рисунок 2.6. – оригінальне та фальсфіковане зображення

Тепер перевіримо роботу алгоритму на цих зображеннях та перевіримо наявність артефактів, тим самим будемо бачити чи фіксує алгоритм такі види фальсифікацій, як клонування. Роботу алгоритму з даними зображеннями буде представлено на рисунку 2.7.

Рисунок 2.7. – результати роботи алгоритму на оригінальному (зліва) та фальсифікованому а саме з клонуванням (праворуч)

Розглядаючи отримані результати можна зробити висновок, що алгоритм не може виявити клонування об’єктів за допомогою інструменту «штамп» оскільки на обох зображеннях присутні монотонні кольори. В таблиці 2.3 буде наведено порівняльну статистику роботи алгоритму при пошуку клонованих областей. Експеримент було проведено на 10 зображеннях.

Таблиця 2.3 – результати роботи алгоритму пошуку клонованих областей зображення

Зображення	Артефакти відсутні	Артефакти присутні
Оригінальне	80%	20%
З фальсифікацією клонування	80%	20%

Аналіз таблиці та результати експерименту вказують на те, що даний алгоритм не може виявити фальсифікацію клонування об’єктів. Присутність артефактів на 20% зображень зумовлено тим, що вони були збережені за допомогою фоторедактору, що свідчить про те, що могла проводитися обробка кольорів чи інші маніпуляції. Даний висновок можна зробити згадавши роботу фоторедакторів, що при будь-яких маніпуляціях з зображеннями вони

автоматично підвищують контрастність контурів незалежно від бажання користувача.

Отож підсумок даного розділу полягає в тому, що алгоритм аналізу рівня помилок дійсно може допомогти при аналізі зображень, які могли бути фальсифіковані методами масштабування та масштабування переміщення об’єкту на зображенні. Саме тому було вирішено інтегрувати даний алгоритм в телеграм бота. Інтегрування та реалізація алгоритму описані в наступному розділі.

3 РЕАЛІЗАЦІЯ АЛГОРИТМУ ЗА ДОПОМОГОЮ БОТУ В МЕСЕНДЖЕРІ TELEGRAM

3.1. Python та Telegram

Було прийнято рішення створити бот в месенджері Telegram за допомогою мови програмування Python. Також алгоритм було реалізовано на даній мові. На це рішення вплинуло декілька факторів, одним з яких стало спрощення дій для використання даного алгоритму, оскільки користувачеві не потрібно буде встановлювати нове програмне забезпечення на свій пристрій, що буде економити місце на ньому та його ресурси. Бот – програма, яка автоматично чи за певної команди виконує різні дії. Задача бота полягає в тому, щоб спростити організацію вільного часу, оскільки не потрібно бути покидати екосистему меседжеру та автоматизувати повторні дії. Починати роботу з ними досить просто, варто знайти бота, який нас цікавить в пошуку та вступити в переписку. В переписці будуть надані всі інструкції як його використовувати.

Реалізацію алгоритму та боту було вирішено створити також за допомогою мови програмування Python. Оскільки Python – найбільш популярна мова програмування, яка швидко розвивається. Також вона досить зрозуміла, без технічного підґрунтя, універсальна та доволі легка в освоєнні. Варто виділити основні плюси даної мови програмування, а саме:

- простота – синтаксис зрозумілий і схожий на англійську мову. Частини коду це блоки, правильний порядок яких дасть змогу створити бажану роботу;
- готові рішення – в Python вже є готові бібліотеки, які створені сторонніми розробниками, так звані фреймворки. Вже зараз існують сотні готових бібліотек для вирішення певних завдань. Тому не потрібно писати десятки рядків коду, а можна використати бібліотеку;
- розвинена спільнота – оскільки програмування вважається складним напрямком, то доводиться багато чого опановувати самостійно. Але наразі у спільноті Python є досить багато кваліфікованих спеціалістів, які можуть допомогти з різними питаннями на спеціалізованих форумах;

- універсальність – оскільки дану мову використовують в багатьох напрямках такі як боти, бази даних, доповнена реальність, нейронні мережі чи ігри, то можна сказати що при гарних знаннях роботі з мовою програмування можна переходити з однієї сфери використання в іншу. Також стане легше вивчати інші мови, оскільки алгоритми вже відомі, то залишиться тільки вивчати синтаксис;
- затребуваність – Python-розробники дуже затребувані на ринку праці, оскільки вакансій більше ніж кандидатів, то замовники чи роботодавці будуть боротися за сильних кандидатів;

Також причиною обрання мови програмування Python стала наявність вже готових бібліотек для роботи з зображеннями. А саме це такі бібліотеки як PIL/Pillow та NumPy. PIL/Pillow – безкоштовна бібліотека для відкриття, роботи та збереження різних форматів зображень. Бібліотека містить в собі базовий функціонал для обробки зображень, що включають точкові операції, фільтри з наборами вбудованих ядер згортки та перетворень кольорових просторів. NumPy – одна з основних бібліотек з підтримкою масивів. Таким чином зображення представляється масивом NumPy, який містить пікселі точок даних. Тому при виконанні різних операцій ми відразу можемо змінювати піксельні значення зображення.

3.2 Створення боту

Бот для даного алгоритму був створений засобами Telegram та Python. За допомогою вже існуючого боту від команди розробників месенджеру, створюємо нашого бота, вводимо його нікнейм та отримуємо посилання на створеного бота та токен авторизації, за допомогою якого можна звертатися до нього зі сторонніх застосунків, що зображено на рисунку 3.1.

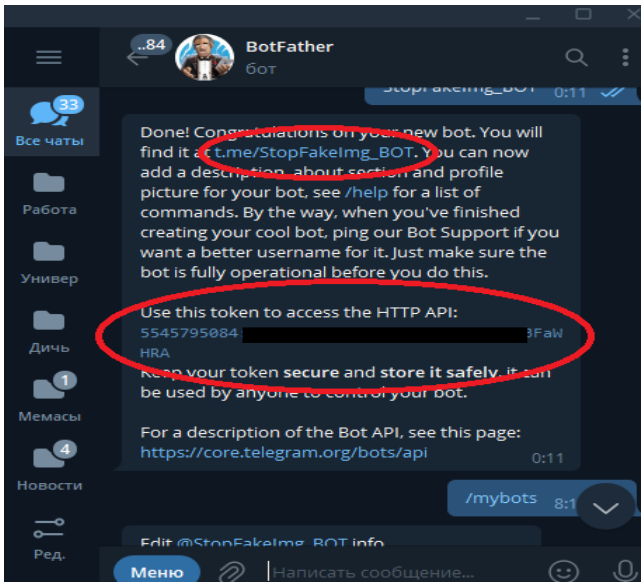

Рисунок 3.1 – відповідь на реєстрацію боту.

Після реєстрації боту за допомогою мови програмування Python отримуємо до нього доступ через токен авторизації та прописуємо боту алгоритм дій. Для початку вводимо команду, яка буде вітати користувача при натисканні кнопки /start та допомоги /help:

```
"""Доступ до бота за допомогою токену"""
bot = telebot.TeleBot(BOT.TOKEN)
@bot.message_handler(commands=["start", "help"])
def start(message):
        """Виведення повідомлення при старті"""
        bot.send_message(message.chat.id,
BOT.start_message)
```

Після цього для роботи з зображеннями прописуємо боту алгоритм дій при відправці йому зображення:

```
@bot.message_handler(content_types=["document"])
def message_come(message):
        """Якщо зображення відправлене документом"""
        downloaded_file                                    =
bot.download_file(bot.get_file(message.document.file_id).fi
le_path)
        patch                                              =
f'{DIRS.download_dir}/{message.document.file_name}.jpg'
        with open(patch, 'wb') as download_photo:
            download_photo.write(downloaded_file)
```

```
        ela(f'{message.document.file_name}.jpg')
        bot.send_photo(message.chat.id,
open(f'{DIRS.upload_dir}/{message.document.file_name}.jpg',
'rb'))
@bot.message_handler(content_types=["photo"])
def message_come(message):
        """Якщо зображення відправлене фотографією"""
        downloaded_file                                    =
bot.download_file(bot.get_file(message.photo[0].file_id).fi
le_path)
        patch                                              =
f'{DIRS.download_dir}/{message.photo[0].file_id}.jpg'
        with open(patch, 'wb') as download_photo:
            download_photo.write(downloaded_file)
        ela(f'{message.photo[0].file_id}.jpg')
        bot.send_photo(message.chat.id,
open(f'{DIRS.upload_dir}/{message.photo[0].file_id}.jpg',
'rb'))
```

Цим самим бот при отриманні зображення зберігає його в директорію на пристрої на якому він запущений та ініціює запуск алгоритму для аналізу зображення.

3.3. Реалізація алгоритму аналізу рівня помилок

Оскільки метод аналізу рівня помилок базується на порівнянні двох зображень, тобто вхідного в форматі .JPEG та нового, яке було збережене з фіксованими втратами. Які задаються вручну. Тому було реалізовано функцію збереження вхідного зображення з втратами в тимчасову директорію:

```
def _reopen(im: Image, photo_name: str) -> Image:
    """Перезбереження зображення з втратами в тимчасову
директорію"""
    path_to_temp_photo = \
        f'{DIRS.temp_dir}{"".join(photo_name.split(".")[:-
1])}{photo.tag}'
    im.save(
        path_to_temp_photo,
        'JPEG',
        quality=photo.qulity
    )
```

```
    return Image.open(path_to_temp_photo)
```

Після виконання зберігання зображення з втратами, за допомогою модулю ImageChops з бібліотеки PIL з використанням функції різниці (Difference)[11] отримуємо абсолютне (по модулю) попіксельну різницю між двома зображеннями: вхідним та збереженим з втратами:

```
def ela(photo_name: str) -> None:
    im = Image.open(f'{DIRS.download_dir}{photo_name}')
    temp_im = _reopen(im, photo_name)
    ela_im = ImageChops.difference(im, temp_im)
    ela_im = _elcalc(ela_im)
    _save_result(ela_im, photo_name)
```

Вирахувавши абсолютну попіксельну різницю та сформувавши з отриманої матриці нове зображення. Звернемо увагу на те, що мінімальна різниця буде сягати близьких до 0 чисел, оскільки при зберіганні в форматі з втратами значення пікселів можуть бути зменшені, а не збільшуватися. Розраховуємо рівні помилок за допомогою такої функції:

```
def _elcalc(ela_im: Image) -> Image:
    """Розрахунок рівня посилок та формування нового зображення """
    extrema = ela_im.getextrema()
    max_diff = max([ex[1] for ex in extrema])
    scale = 255.0/max_diff
    return ImageEnhance.Brightness(ela_im).enhance(scale)
```

Ця функція спочатку отримує значення максимального та мінімального значення пікселів в кожному рядку завдяки функції getextrema[12]. Після чого серед мінімальних значень знаходиться найбільше значення, яке використовується у виведенні коефіцієнту. Наступний крок відбувається за допомогою модулю ImageEnhance та його функції Brightness[13], що регулює яскравість отриманого зображення з нашим виведеним коефіцієнтом. Відрегулювавши яскравість зберігаємо зображення до директорії створеною функцією:

```
def _save_result(result: Image, photo_name: str) -> None:
    """Зберігання результату"""
    path_to_result = f'{DIRS.upload_dir}{photo_name}'
    result.save(path_to_result)
```

3.4. Відзначення властивостей боту і його тестування

Створивши бот в месенджері Telegram варто протестувати його можливості та відзначити позитивні та негативні властивості. Для початку відзначимо його позитивні властивості, серед яких можна виділити:

- економія місця – відбувається за рахунок того, що не потрібно встановлювати додаткове програмне забезпечення для перевірки зображення на фальсифікацію. Також економія місця відбувається за рахунок того, що не потрібно завантажувати перевірене зображення на свій пристрій, його можна переглянути в самому листуванні з ботом;
- економія ресурсів пристрою – оскільки бот виконує операції на пристрої, який є його хостом, то і всі операції та обчислення відбуваються на ньому, що економить ресурси пристрою користувача;
- економія часу – користувачу не потрібно покидати екосистему месенджеру щоб перевірити зображення, можна просто перейти до листування з ботом та перевірити підозріле зображення;
- економія сил розробника – месенджер існує як на мобільних та і на стаціонарних платформах з будь-якою операційною системою, що дає змогу розробнику не перейматися за різні аспекти цих систем, а кожен користувач матиме доступ до боту;

Але при використанні боту є і негативні сторони такі як:

- потреба в підключенні до мережі Інтернет – оскільки відправка та прийом повідомлень відбувається за допомогою глобальної мережі Інтернет;
- можливий збій на стороні хосту – призводить до відключення боту та неможливісті скористатися ним;
- відсутність спеціальних сповіщень – оскільки в месенджері бот не відрізняється від реальних адресатів, то неможливо виділити від кого саме прийшло повідомлення;

- відсутність списку ботів – Telegram не має масштабного списку ботів, за допомогою якого можна знайти потрібного. Ця задача цілком і повністю лежить на плечах користувача, якому потрібно знати назву боту;
- розмір кнопок – якщо в боті використовуються кнопки для управління, то розробнику варто звернути увагу на текст, який буде відображатися на ній, оскільки кнопка не збільшується, то великі тексти можуть бути скорочені до розміру кнопки та мати вигляд «Текст…», що зробить не зрозумілим роботу з ботом;
- розташування елементів управління – зазвичай всі додатки розробляються по певним макетам, до яких звикли користувачі. При використанні боту всі органи управління будуть знаходитися внизу екрану, до чого користувачу потрібно звикати;
- відсутність перезапису – більшість додатків використовують функцію перезапису останнього результату, оскільки бот не може роботи цього, то деяка конфіденційна інформація (зображення) може залишитися в листуванні з ботом;
- скролінг (гортання) – через проблему відсутності перезапису користувачеві потрібно постійно скролити (гортати) листування з ботом для пошуку певної інформації (зображення);

На мою думку хоч негативних сторін більше аніж позитивних, але вони нівелюються важливістю позитивних. Перейдемо до тестування боту:

1. Для початку бота потрібно знайти в самому месенджері це можна зробити за допомогою спеціального посилання, яке ми отримали від бота розробників Telegram (рис. 3.2) чи ввівши його нікнейм в поле пошуку (рис. 3.3).

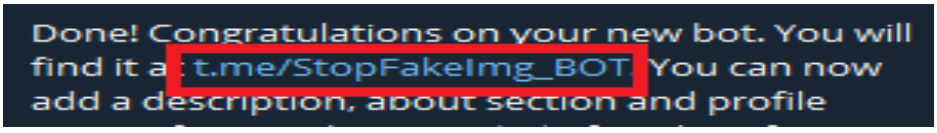

Рисунок 3.2 – посилання на бота при його реєстрації

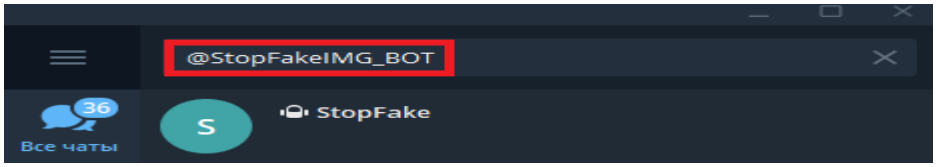

Рис. 3.3. – пошук боту за його нікнеймом

2. Переходимо в листування з ботом та бачимо відсутність листування і єдину кнопку «Запустити» (Рисунок 3.4). Натискаємо її та отримуємо відповідь від боту(3.5).

Рисунок 3.4 – перехід в листування з ботом та його первинний інтерфейс

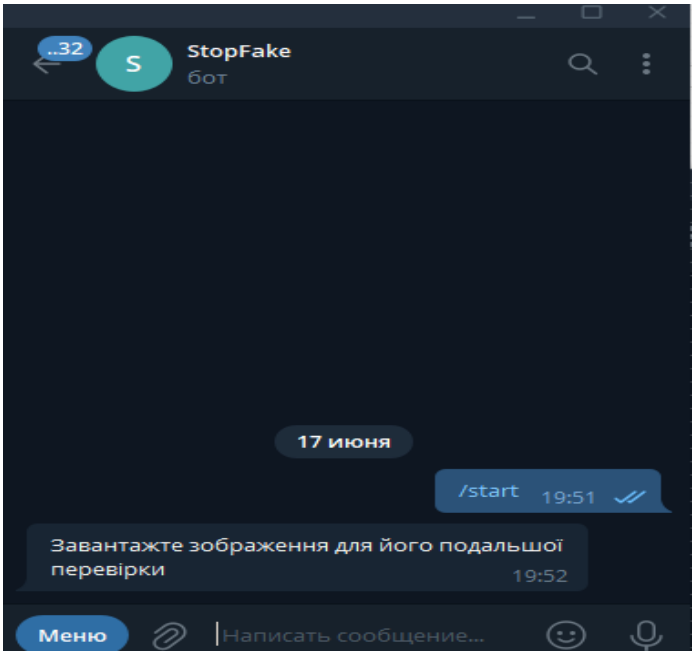

Рисунок 3.5 – відповідь боту на запуск

3. Для завантаження зображення можна скористатися скріпкою в нижній чистині екрану, та обрати потрібне зображення (рис. 3.6) чи перемістити його в вікно листування (рис. 3.7) після чого підтвердити відправку. Для вибору відправки зображення перевага надається відправці файлом без стискання, аніж відправці картинкою зі стисканням, оскільки стискання може привести до появи нових помилок на зображенні, що призведе до ускладнення розпізнавання.

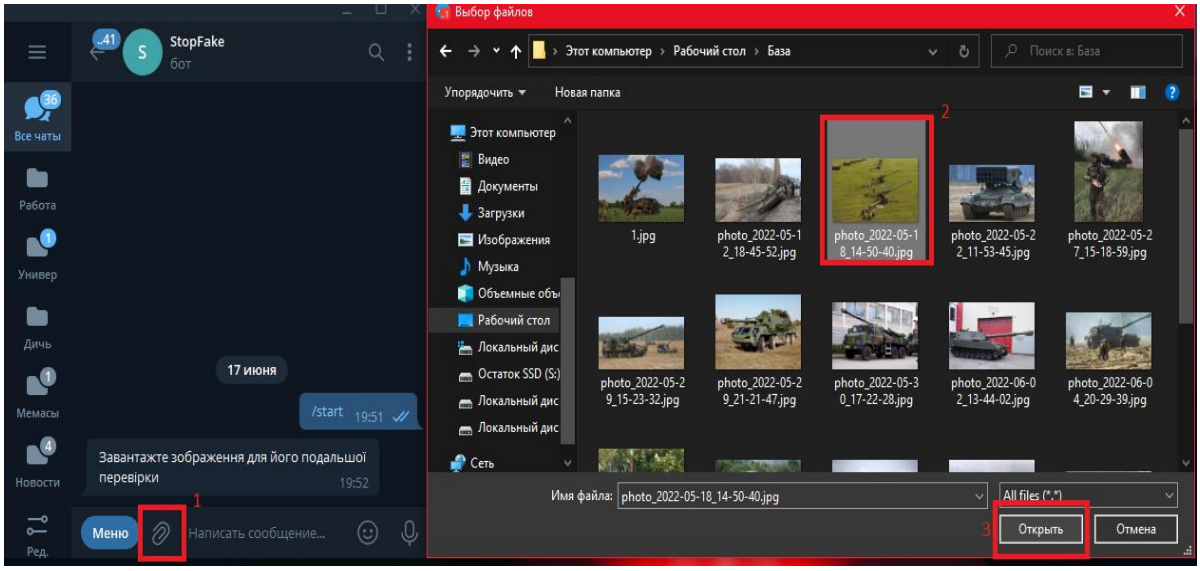

Рисунок 3.6 – Відправка зображення за допомогою скріпки

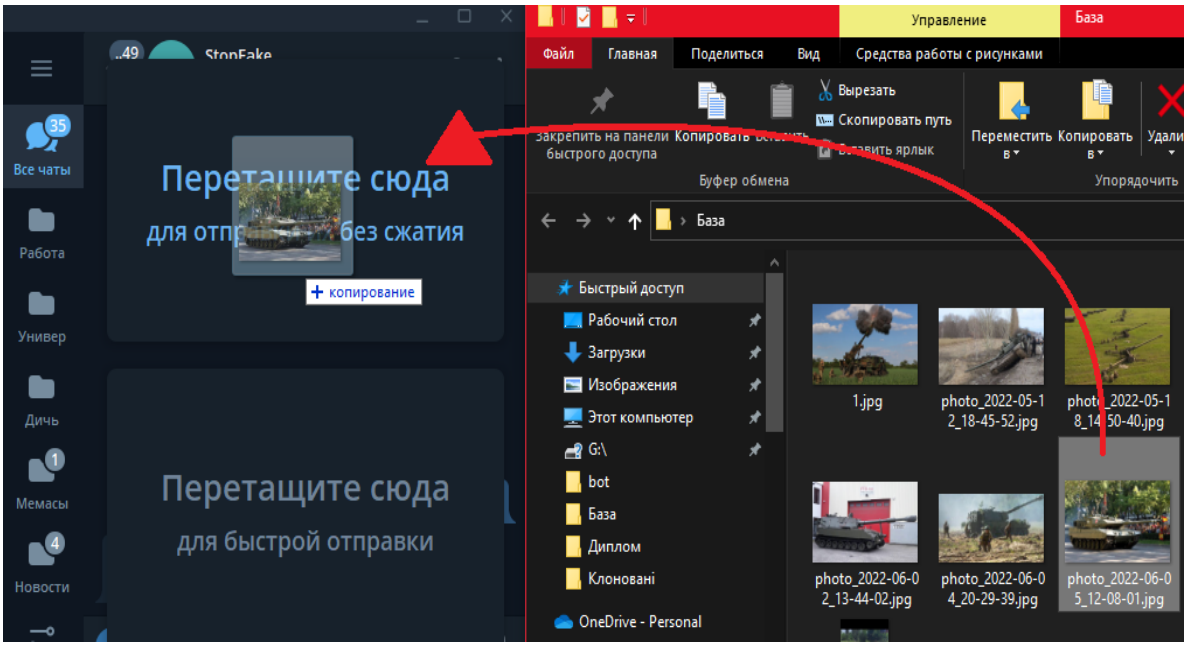

Рисунок 3.7 – відправка зображення шляхом його перетягування

4. Відразу після отримання зображення бот починає опрацювати його та видає результат в листуванні (рис. 3.8) в якості фотографії, яку можна одразу переглянути.

Рисунок 3.8 – повідомлення від боту з опрацьованим зображенням

Таким самим чином завантажуємо зображення з масштабованим об’єктом і отримуємо результат на рисунку 3.9.

Рисунок 3.9 – повідомлення від боту з результатом опрацювання зображення з масштабованим об’єктом (для наглядного представлення збільшено яскравість)

Таким чином застосування боту в месенджері Telegram дає змогу перевіряти зображення на наявність фальсифікації після інтегрування в бот алгоритму аналізу рівня помилок, що дозволяє виконати цю перевірку.

4 ОХОРОНА ПРАЦІ

Оскільки більшість свого часу людина проводить на робочому місці, то стає очевидним потреба в захисті свого життя від шкідливих та небезпечних факторів, які можуть бути присутніми на робочому місці. Для допущення працівника на робоче місце варто провести його аналіз, незалежно від важкості та тривалості його роботи на цьому місці.

Саме тому існує ДСТУ 12.0.003-74[14], за яким визначають шкідливі та небезпечні фактори робочого місця. Для аналізу було обрано робоче місце програміста. Основними шкідливими та небезпечними факторами для робочого місця можуть бути:

- Підвищена запиленість чи загазованість повітря;
- Підвищена чи знижена температура повітря;
- Підвищений рівень шуму;
- Підвищена чи знижена вологість повітря;
- Підвищене значення напруги в електричному ланцюжку;
- Недостатня освітленість, рівень природного світла;
- Великий рівень віддзеркалення світла;

Основним компонентом комфортної роботи є мікроклімат робочого місця. Тому варто розібрати поняття мікроклімату – клімат внутрішнього середовища робочого приміщення. Визначається температурою, вологістю та швидкістю руху повітря, що діє на організм працюючого. Температура робочого місця повинна бути в межах 24-25℃ влітку та 20-24℃ взимку при відносній вологості повітря 50%. Саме тому на робочому місці є кондиціонер, який регулює температуру повітря влітку та тепла підлога, яка регулює температуру взимку. Оскільки при використанні приладів, регулюючих температуру вологість повітря може змінюватися то на робочому місці присутній освіжувач повітря. Ці предмети дозволяють виконувати роботу в будь-яку пору року з нормальними показниками мікроклімату згідно ДСанПіН 3.3.2.007-98[15].

Підвищена запиленість робочого місця небезпечна тим, що під час виконання завдань на робочому місці робітник може вдихати частки пилу. Але оскільки по закінченню робочого дню приміщення прибирається клінінговими компаніями, що включає в себе вологе прибирання, то накопиченню пилу вдається запобігати.

Освітлення робочого місця може бути:

- природнім, що відбувається за рахунок освітлення деякої частини площини за рахунок потрапляння світлових променів
- штучним – освітлення, яке відбувається за рахунок встановлення в приміщенні різних джерел світла. Воно поділяється на загальне, місцеве та комбіноване;
- суміщене – використання як природнього так і штучного освітлення.

На даному робочу місці використовується суміщене освітлення. Оскільки під час роботи в денний час освітленість робочого місця сягає 500-600 лк., то даний параметр задовольняє норми. Але у вечірній час освітленість робочого місця може сягати нижче 400лк. Саме тоді використовується штучне комбіноване освітлення. Загальне штучне освітлення рівномірно освітлює приміщення робочих зон, а місцеве концентрує освітленість саме на робоче місце. Для штучного освітлення використовуються такі лампи:

- для загального – світлодіодна лампа: 1000 lm, 4200 K. Встановлюється над кожним робочим місцем та між ними;
- для місцевого – світлодіодна лампа: 1000 lm, 4200 K. Встановлюється безпосередньо на кожному робочому місці в вигляді настільної лампи.

Було обрано саме світлодіодні лампи, оскільки на відміну від ламп розжарювання вони не мають негативної дії нагрівання, що призводить до підвищення температури в приміщенні та не чутливі до перебої електромережі.

Для комфортної роботи варто врахувати коефіцієнти відбиття світла. Оскільки при високому рівні відбиття поверхонь це може засліпити, або створювати некомфортні умови для роботи. Коефіцієнти відбиття поверхонь:

- стіни – бетон, пофарбований фарбою сірого кольору з коефіцієнтом відбиття 0,5;
- стеля – гіпсокартон, пофарбований білою матовою фарбою з коефіцієнтом відбиття 0,6;
- підлога – паркет темно сірого кольору з коефіцієнтом відбиття 0,35;

Ці параметри забезпечують комфортні умови праці, оскільки жодна з поверхонь не відбиває велику кількість світла та не засліплює програміста. Але в разі виникнення відблисків на екрані, то існує вірогідність того, що сам екран є поверхнею, яка відбиває світлові промені. Тому варто розташувати його так, щоб уникнути цього.

Підвищений рівень шуму виникає внаслідок використання великої кількості офісного приладдя, наприклад принтери, кондиціонери, сканери. Також шум надходить ззовні приміщення, оскільки офіс знаходиться поряд з вулицею, де дозволено рух автомобілів. Саме тому було вжито заходи для зменшення рівня шуму:

- Використання звукопоглинаючого матеріалу на стінах будівлі ззовні у вигляді мінеральної вати та гіпсокартону;
- Використання прорезинених підкладок для офісної техніки, оскільки в них присутні рухомі матеріали, що провокують вібрацію та виходячи з цього утворюють деякий рівень шуму;
- Винесення офісної техніки в окрему частину робочого приміщення та використання перегородки для неї;

Завдяки заходам зі зниження шуму вдалося знизити його до 47 Дб, що відповідає ДСН 3.3.6.037-99[16].

Робота програміста це постійний контакт з електрообладнанням, що може призвести до електричного ураження в разі перепадів напруги, оголених кабелів, відсутності заземлення та підвищеної напруги в електричній мережі. Саме тому на підприємстві використовуються такі засоби безпеки:

- Розетки з кришками, які унеможливлюють потрапляння в них сторонніх предметів;

- Спеціальні короби для кабелів – передбачають контакт кабелю з працівниками;
- Джерела безперебійного живлення – дозволяє захистити техніку від проблем з електричною мережею;
- Встановлено захисне заземлення;

Важливим параметром комфортної роботи є можливість відпочинку. Саме тому було впроваджено технічні перерви, на яких працівникам надається можливість розім'ятися чи випити кави.

Для зменшення навантаження на очі програміста було вжито низку заходів:

– зниження яскравості екрану на 10%;

– встановлення висококонтрастної теми, що зменшить кількість кольорів, які навантажують очі програміста;

– використання захисної плівки на екрані, яка створена для зменшення кількості світла синього спектру;

– створене нагадування про розминку очей;

Пожежа – це неконтрольоване горіння поза спеціальною зоною, яке поширюється в часі і просторі з інтенсивним виділенням тепла і диму, що є дуже небезпечним для здоров'я людини та навколишнього середовища. Для попередження пожеж фахівцями розроблено велику кількість правил, інструкцій, наказів, вказівок та нормативних документів, дотримання яких поліпшить пожежну безпеку на об'єкті.

Для співробітників залишається важливим виконання елементарних правил пожежної безпеки, адже безвідповідальне ставлення до них може спричинити пожежу. В основному пожежі в офісних приміщеннях, які належать до класу «В», відбуваються через такі порушення правил пожежної безпеки:

- куріння в невідведених для цього місцях;
- порушення правил користування електроприладами;
- необережне поводження з відкритим полум'ям;
- перевантаження перехідників чи подовжувачів;

Робоче приладдя – один з головних факторів виникнення пожежі. Оскільки їх поломка призводить до виникнення горіння, то у разі виявлення несправності варто відключити їх від електромережі та віддати в ремонт до спеціалістів.

У разі виникнення пожежі в приміщенні встановлена пожежна сигналізація, яка реагує на дим, який виникає внаслідок горіння, це забезпечить швидке реагування на появу пожежі.

Враховуючи кількість електрообладнання гасіння пожежі варто проводити за допомогою вуглекислих вогнегасників. Принцип їх роботи полягає в тому, щоб витіснити окисник – повітря з місця займання та зменшити його температуру до -70℃ та припинити процес горіння.

У разі поширення пожежі до рівня, коли неможливо зупинити власноруч потрібно негайно покинути приміщення. Саме тому для таких подій створюються плани евакуації. Вони допомагають спрямовувати співробітників в безпечні місця. План евакуації має містити два виходи, оскільки під час пожежі один з ним може виявитися недоступним. Також для підвищення рівня обізнаності співробітник потрібно проводити наради чи тренінги про дії та заходи при пожежі. Оскільки це підвищить продуктивність співробітників під час пожежі та буде запобігати поширенню панічних настроїв. Під час тренінгів провидити планові евакуації, вказувати на правильне поводження з електрообладнанням та користуванням вогнегасниками.

Саме тому потрібно навести список, для профілактики пожежі в офісних приміщеннях[17]:

- вивчити інструкції з пожежної безпеки;
- вивчити шляхи евакуації при пожежі;
- не загороджувати шляхи евакуації;
- розбиратися в знаках безпеки;
- знати розташування систем пожежного сповіщення;
- знати, де розташовані місця для паління;
- знати про заборону роботи з несправною офісною технікою.

Оскільки більшість людей та працівників інформуються про план дій при пожежі, але іноді працівники не можуть їх виконати через паніку. Тому основними заборонами при пожежі є:

- ігнорування сигналів пожежної сигналізації;
- паніка;
- користування ліфтом;
- пересування в повний зріст при високій задимленості;
- переоцінювати свої сили;
- зволікати з викликом пожежних;
- відкривати чи розбивати вікна;
- намагатися загасити офісні пристрої водою;
- ховатися у віддалених приміщеннях, шафах.

ВИСНОВКИ

В кваліфікаційній роботі виконані всі завдання з переліку тих, які були поставлені до виконання.

Для виконання першого завдання було проведено роботу над аналізом сучасного стану проблеми фальсифікації зображення. Було отримано результати, які вказують що робота для виявлення фальсифікації зображень є основною в цифровій криміналістиці. Саме тому існують методи, які можуть виявляти різного роду фальсифікації, але не було знайдено методу для виявлення фальсифікації за допомогою масштабування з від'ємним коефіцієнтом. Саме тому було необхідно знайти рішення даної проблеми.

Наступною задачею стало проведення експерименту з методом аналізу рівня помилок, оскільки саме він представляє собою рішення проблеми виявлення масштабування з від'ємним коефіцієнтом. При проведенні експерименту з даним методом було використано не лише оригінальні та масштабовані зображення, але й зображення з масштабуванням та переміщенням і з клонуванням.

Після проведення експерименту стала очевидна його результативність, тому було вирішено реалізувати його на мові програмування Python та інтегрувати його в бот месенджера Telegram. Що відповідає завданню з переліку поставлених завдань для кваліфікаційної роботи. Для виконання завдання було реалізовано метод аналізу рівня помилок на мові програмування Python та інтегровано в бота месенджеру Telegram після чого роботу бота було протестовано. В результаті виконання проведеної роботи було отримано бота в месенджері Telegram, який відповідає поставленим завданням, необхідними для його запуску.

ПЕРЕЛІК ПОСИЛАНЬ

1. Чакраверті А. К., Дір В. Гібридний підхід до пошуку клонованих об’єктів у копіюванні підроблених зображень. URL: https://www.inderscienceonline.com/doi/abs/10.1504/IJFSE.2019.104705
2. Армас Вега Е.А.. Методика виявлення підробки копіювання та переміщення на основі функцій блоків дискретного косинусного перетворення. URL: https://www.researchgate.net/publication/344783000_Copy-move_forgery_detection_technique_based_on_discrete_cosine_transform_blocks_features
3. Кі Е., Фарід Х.. Метрика сприйняття для ретушування фотографій. URL: https://www.pnas.org/doi/10.1073/pnas.1110747108#abstract
4. Б’янко С. Видалення художнього фотофільтра за допомогою згорткових нейронних мереж. URL: https://www.researchgate.net/publication/322032456_Artistic_photo_filter_removal_using_convolutional_neural_networks
5. Що таке фотофорензика? URL: https://fotoforensic.com/faq.php#What%20is%20FotoForensics
6. Трифонова К.О., Кілін О. Є.. Метод локалізації та ідентифікації контекстно-залежного масштабування в цифровому зображенні. URL: http://www.tkea.com.ua/siet/archive/2014-t1/117.pdf
7. Декодування JPEG. URL: https://habr.com/ru/post/102521/
8. Ресамплінг. URL: https://aleksius.com/143-resampling#:~:text=resampling%20(ещё%20называют%20передискретизация)%20–,размеров%20в%20пикселах%20или%20разрешения.
9. Збройні сили України. Війна з окупантами. URL: https://t.me/zsuwar
10. Як відрізнити фейкове зображення за допомогою криміналістичних інструментів. URL: https://teletype.in/@royal_bank/0-xYFhgjT
11. Модуль «операції з каналами». URL: https://pillow.readthedocs.io/en/stable/reference/ImageChops.html

12. Методи getbands() та getextrema(). URL: https://www.geeksforgeeks.org/python-pil-getbands-and-getextrema-method/#:~:text=getextrema()%20method%20%E2%80%93,each%20band%20in%20the%20image.&text=image%20path%5D)-,Parameters%3A,path%20can%20also%20be%20mentioned.

13. Модуль ImageEnhance. URL: https://pillow.readthedocs.io/en/stable/reference/ImageEnhance.html#module-PIL.ImageEnhance

14. ДСТУ 12.0.003-74*. ССБТ. Небезпечні і шкідливі фактори. Класифікація. [Чинний від 1999.01.01].

15. ДСанПіН 3.3.2.007-98. Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин. [Чинний від 1998.12.10].

16. ДСН 3.3.6.037-99. Санітарні норми виробничого шуму, ультразвуку та інфразвуку. [Чинний від 1999.01.12]. Київ, 1999. 10с.

17. Що потрібно та чого не можна роботи при пожежі в офісі: чек-лист. Профілактика виникнення пожежі в офісі. URL: https://neohr.ru/korporativnaya-kultura/article_post/chto-nuzhno-i-chego-nelzya-delat-pri-pozhare-v-ofise-chek-list

Додаток А. Лістинг програмного коду

bot.py:

```
import telebot
from config import BOT, DIRS
from image import ela
def bot_main():
    bot = telebot.TeleBot(BOT.TOKEN)
    @bot.message_handler(commands=["start", "help"])
    def start(message):
        """Виведення повідомлення при старті чи натисканні «допомога»"""
        bot.send_message(message.chat.id, BOT.start_message)
    @bot.message_handler(content_types=["document"])
    def message_come(message):
        """Якщо відправили документом"""
        downloaded_file = bot.download_file(bot.get_file(message.document.file_id).file_path)
        patch = f'{DIRS.download_dir}/{message.document.file_name}.jpg'
        with open(patch, 'wb') as download_photo:
            download_photo.write(downloaded_file)
        ela(f'{message.document.file_name}.jpg')
        bot.send_photo(message.chat.id, open(f'{DIRS.upload_dir}/{message.document.file_name}.jpg', 'rb'))
    @bot.message_handler(content_types=["photo"])
    def message_come(message):
        """Якщо відправили фотографією"""
        downloaded_file = bot.download_file(bot.get_file(message.photo[0].file_id).file_path)
        patch = f'{DIRS.download_dir}/{message.photo[0].file_id}.jpg'
        with open(patch, 'wb') as download_photo:
            download_photo.write(downloaded_file)
        ela(f'{message.photo[0].file_id}.jpg')
        bot.send_photo(message.chat.id, open(f'{DIRS.upload_dir}/{message.photo[0].file_id}.jpg', 'rb'))
    bot.polling(True)
bot_main()
```

config.py:

```
# dirs
class DIRS:
    download_dir = './photos/download/'
    upload_dir = './photos/upload/'
    temp_dir = './photos/temp/'
# quality for save
class photo:
    qulity = 90
    tag = '_alg.jpeg'
# bot
```

```
class BOT:
    TOKEN = '5545795084:AAGNaUz6CPtzKoiZE2IjlwkREFqOBFa****'
    start_message = 'Для активації боту відправте йому повідомлення
з фото'
```

image.py:

```
from __future__ import print_function
from config import DIRS, photo
from PIL import Image, ImageChops, ImageEnhance

def _reopen(im: Image, photo_name: str) -> Image:
    path_to_temp_photo = \
        f'{DIRS.temp_dir}{"".join(photo_name.split(".")[:-
1])}{photo.tag}'
    im.save(
        path_to_temp_photo,
        'JPEG',
        quality=photo.qulity
    )
    return Image.open(path_to_temp_photo)
def _elcalc(alg_im: Image) -> Image:
    extrema = alg_im.getextrema()
    max_diff = max([ex[1] for ex in extrema])
    scale = 255.0/max_diff
    return ImageEnhance.Brightness(alg_im).enhance(scale)

def _save_result(result: Image, photo_name: str) -> None:
    path_to_result = f'{DIRS.upload_dir}{photo_name}'
    result.save(path_to_result)
def alg(photo_name: str) -> None:
    im = Image.open(f'{DIRS.download_dir}{photo_name}')
    temp_im = _reopen(im, photo_name)
    alg_im = ImageChops.difference(im, temp_im)
    alg_im = _alcalc(alg_im)
    _save_result(alg_im, photo_name)
if __name__ == '__main__':
    alg('1.jpg')
```

main.py:

```
 if __name__ == '__main__':
    pass
```