Міністерство освіти і науки України
Національний університет «Одеська політехніка»
Інститут інформаційної безпеки, радіоелектроніки та телекомунікацій
Кафедра кібербезпеки та програмного забезпечення

Гонтар Максим Валерійович
студент групи РЗ-181

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

Розробка програмного додатку для виявлення підробок світлин

Спеціальність:
125 Кібербезпека

Спеціалізація, освітня програма:
Кібербезпека

Керівник:
Вікторія Вікторівна Зоріло,
к.т.н., ст.викл

Одеса - 2022

Міністерство освіти і науки України
Національний університет «Одеська політехніка»
Інститут інформаційної безпеки, радіоелектроніки та телекомунікацій
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
____ _____2022р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Гонтару Максиму Валерійовичу

1.Тема роботи: *Розробка програмного додатку для виявлення підробок світлин*

2. Керівник роботи: *Зоріла Вікторія Вікторіна*, *к.т.н., ст.викл*,

затверджені наказом ректора від „17” 05.2022р. №168-в .

3.Зміст роботи: *Аналіз сучасних методів виявлення підробок світлин, розробка методу виявлення інструменту «Латка» як підробка світлин, створення програмного забезпечення реалізований телеграм-бот виявлення інструменту «Латка».*

4. Перелік ілюстративного матеріалу: *рисунки програмного продукту, рисунки програмного коду, рисунки алгоритму користуванням продукту.*

5. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
Охорона праці	*доц. Ярова І.А.*		

Дата видачі завдання “_13__”____05_____2022р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання	Примітка
1	*Аналіз літератури за темою кваліфікаційної роботи*	*05-03-2022*	*виконано*
2	*Аналіз програмного забезпечення, що надає можливість створення телеграм-боту*	*18-03-2022*	*виконано*
3	*Аналіз методів виявлення фальсифікацій зображень*	*10-04-2022*	*виконано*
4	*Розробка алгоритму*	*25-04-2022*	*виконано*
5	*Імплементація алгоритму*	*13-05-2022*	*виконано*
6	*Підготовка пояснювальної записки.*	*17-05-2022*	*виконано*
7	*Підготовка презентації та доповіді*	*26-05-2022*	*виконано*
8	*Попередній захист*	*10-06-2022*	*виконано*
9	*Нормоконтроль, рецензування*	*17.06.2022*	*виконано*

Здобувач вищої освіти _____ *Гонтар М.В.*

Керівник роботи *Зоріло В.В.*

ЗАВДАННЯ

на розробку розділу «Охорона праці» у кваліфікаційній роботі бакалавра

студенту *Гонтару Максиму Валерійовичу*
(прізвище, ім'я, по батькові)

Інститут інформаційної безпеки, радіоелектроніки та телекомунікацій
(повне найменування інституту**)**

Кафедра кібербезпеки та програмного забезпечення
(повне найменування кафедри**)**

Дата отримання завдання 13.05.2022
Консультації 16.05.2022, 23.05.2022
Дата закінчення розділу 16.06.2022

Тема роботи *Розробка програмного додатку для виявлення підробок світлин*

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) (прізвище та ініціали)

« __ » _____2022 р.

Консультант з охорони праці

_____ Ярова І.А.
(підпис)

« __ » _____ 2022 р.

АНОТАЦІЯ

Кваліфікаційна робота на тему «Розробка програмного додатку для виявлення підробок світлин» на здобуття першого (бакалаврського) рівня вищої освіти за спеціальністю 125 – Кібербезпека, спеціалізація, освітня програма: Кібербезпека, містить 25 рисунків, 1 таблицю, 4 додатки, 10 літературних джерел за переліком посилань. Робота виконана на 50 сторінок загального тексту і 35 сторінок основного тексту.

Метою роботи є підвищення ефективності виявлення підробок світлин, виконаних засобами інструменту Adobe Photoshop «Латка».

У роботі був виконаний огляд методів виявлення підробок світлин та їх проблеми. Був обраний метод для реалізації виконання поставленої задачі кваліфікаційної роботи.

У результати проведеної роботи був розроблений алгоритм, який був реалізований у чат-боті Telegram.

ІНФОРМАЦІЯ, ІНФОРМАЦІЙНА БЕЗПЕКА, ЦЗ, МЕТОДИ ВИЯВЛЕННЯ ЦІЛІСТНОСТІ ЦИФРОВИХ ЗОБРАЖЕНЬ.

ABSTRACT

Qualification work on «Development of a software application for detecting counterfeit photographs» for the first (bachelor's) level of higher education in specialty 125 – Cybersecurity, specialization, educational program: Cybersecurity, contains 25 figures, 1 table, 4 appendices, 25 references. The work is performed on 50 pages of general text and 35 pages of main text.

The aim of this work is to increase the efficiency of detecting forgeries of photographs made with the help of the Adobe Photoshop tool "patch".

The paper reviews the methods of detecting counterfeit photographs and their problems. A method was chosen to implement the task of qualification work.

As a result of this work, an algorithm was developed and implemented in the Telegram chatbot.

INFORMATION, INFORMATION SECURITY, CA, METHODS OF DETECTING THE INTEGRITY OF DIGITAL IMAGES.

ЗМІСТ

ВСТУП

У сучасному світі світлини стали невід'ємною частиною кожної людини. Вже давно соціальні мережі, інтернет-магазини та інші соціальні методи взаємодії між людьми стали дуже розвинутими, і це супроводжується використанням великої кількістю світлин. Вони зустрічаються нам на кожному кроці. Кожен день мільйон користувачів продивляються фото друзів, рідних та близьких. Підтвердженням цьому стали соціальні мережі Instagram, Facebook та інші.

Реальність така, що фотокамера є у кожної людини. Це через те, що з кожним роком виробники телефонів випускають моделі, у яких камери не відстають від професійних пристроїв. Ми, самі того не помічаючи, фіксуємо різні моменти нашого життя, будь то значуща подія тощо. Це призводить до того, що у вашому розпорядженні опиняється тисячі фотографій.

Оскільки в сучасному інтенсивно розвиненому світі існує багато загальнодоступного програмного забезпечення, яке дозволяє створювати та редагувати цифрові зображення (ЦЗ), існує кілька способів підробити їх за допомогою змін, які рідко бачить людське око.

Частіше за все фотографії, які зроблені власноруч, не влаштовують користувача, і він намагається методом фотомонтажу виправити недоліки і поділитися нею в соціальних мережах, це ніяк не шкодить суспільству. Але дуже шкода, що в наш час, коли люди ставлять свої інтереси вище за інші, це призводить до дезінформації, яка супроводжується підробленими зображеннями.

Раніше для приховання оригінальності зображення використовували копіювання об'єкту на зображенні, але зараз почали використовувати інструмент «Латка». Цей інструмент виділяється тим, що при переносі об'єкту на зображенні цей інструмент підбирає колірну палітру фону (на місці, куди ви перенесли об'єкт) та підлаштовує кольори так, що зображення після виконання операції стає не невідмінним від оригіналу, але на зображенні залишаються області, які не змінилися після використання інструменту.

На сьогоднішній день існують методи, які є достатньо ефективними для пошуку фальсифікацій зображення, але різниця в тому, що всі методи розраховані на пошук копійованих об’єктів, які були виконані методом інструменту «Штамп» (переміщує виділений об’єкт і не виконує ніяких операцій). Також один із методів розрахований на пошук масштабування об’єкту на зображені.

Мета роботи: підвищення ефективності виявлення підробок світлин, виконаних засобами інструменту Adobe Photoshop «Латка».

Для досягнення мети необхідно вирішити наступні задачі.

1. Огляд методів виявлення підробок світлин.

2. Вибір методу для реалізації.

3. Апробація розробленого додатку та телеграм-бота в реальних умовах.

Практична цінність роботи полягає у розробці механізму виявлення копіювання об’єкту методом використання інструменту «Латка» шляхом створення телеграм-боту. Використання даного боту допоможе у сфері купівлі автомобілів, товарів, у сфері кібербезпеки та ін. З його допомогою можна виявити підробку фотографій і визначити порушення цілісності цифрових зображень.

1 АНАЛІЗ СУЧАСНИХ МЕТОДІВ ВИЯВЛЕННЯ ПОРУШЕНЬ ЦІЛІСНОСТІ ЦИФРОВИХ ЗОБРАЖЕНЬ

1.1 Значимість цифрових зображень та їх формати

Цифрові зображення грають дуже велику роль на теперішній день, їх використовують у всіх сферах життя: будь то робота, особисте життя, бізнес, навчання. В деяких сферах цифрові зображення відіграють велику роль та є допоміжною річчю в особових справах.

Існує багато просунутого програмного забезпечення, завдяки якому користувач може редагувати цифрові зображення. Це дає змогу вибрати один із багатьох варіантів фальсифікації зображення, в результаті чого недосвідченим людям не вдасться дізнатися, була фальсифікація чи ні.

Враховуюче те, що у сучасному світі багато правопорушників намагаються приховати свої неправомірні дії шляхом підробки цифрових зображень, які можуть виступати у ролі доказів, – це питання стає актуальним у вирішенні проблеми виявлення фальсифікацій зображень.

Перед тим як перейти до огляду існуючих методів, ознайомимося з теорією, що таке цифрове зображення, та які існують формати зображень.

Цифрове зображення – масив даних, отриманий шляхом дискретизації (аналого-цифрового перетворення) оригіналу. Після кодування за допомогою певного алгоритму і запису на носій, цей масив даних стає файлом. У сучасному процесі поліграфічного виробництва всі ілюстрації й елементи оформлення представлені цифровими зображеннями різних типів. Цифрові зображення за способом дискретизації оригіналу поділяються на растрові, векторні та змішаного типу.

На сьогоднішній день існує дуже багато форматів зображення (BMP, JPEG, GIF, PNG, TIFF, AI, CDR, SVG, WMF), але ми роздивимося тільки ті, які зустрічаються нам частіше всього [1]. Формат BMP (Bitmap Picture) – це формат для зберігання зображень у растровому вигляді, який був розроблений компанією Microsoft. Хоч і стискання можливе, частіш за все формат BMP використовують

без нього. Формат BMP можливо використовувати у великій кількості програм, оскільки він інтегрований в ОС Windows. Файли формату BMP можуть мати розширення .bmp, .dib і .rle. Формат GIF (Graphics Interchange Format) – широко використовуваний формат графічних зображень, здатний зберігати стиснені дані до 256 кольорів без втрати якості. Компанія CompuServe розробила апаратно-незалежний формат GIF (GIF87a) у 1987 році для передачі растрових зображень через Інтернет. У 1989 році формат був модифікований (GIF89a), який реалізував підтримку прозорості та анімації. Формат PNG (Portable Network Graphics) – також популярний, він зберігається у растровому форматі графічну інформацію, та стискається без втрат. Формат PNG створений для заміни формату GIF. Цей формат дуже популярний і широко використовується в інтернеті. Формат JPEG (Joint Photographic Experts Group) – один з популярних графічних форматів, застосовуваний для зберігання фотографій і подібних їм зображень. Файли, які містять дані JPEG, зазвичай мають розширення .jpeg, .jfif, .jpg, JPG або JPE. Проте .jpg є найпопулярнішим на всіх платформах. Широко використовується формат JPEG. Стиснення засноване на усередненні суміжних кольорів пікселів (інформація про яскравість не усереднена) і придушенні високочастотних складових у просторовому спектрі фрагментів зображення.

В наш час телефони фотографують в форматі JPEG, професійне обладнання частіш за все фотографую у форматі BMP, щоб цифрове зображення в результаті вийшло детальне та без стискання.

Стиснення зображень – це використання алгоритмів для стиснення даних у зображення, які зберігаються в цифровій формі. Стиснення зменшує розмір зображення, що зменшує час передачі зображення по мережі та економить місце для зберігання.

1.2 Методи фальсифікації та виявлення фальсифікації зображення

У даний час існує досить багато способів підробки світлин [2]. Одними з таких є:

- ретушування (re-touching) – завдяки цій операції в цифрове зображення вноситься ряд змін, які ми можемо назвати розмиттям (blur) [3] або змазуванням (smudge);
- посилення (enhancing) – після обробки, на цифровому зображені змінюється різкість (sharpen) [4], колір (color), контраст (contrast adjustment);
- композиція (compositing) – завдяки композиції можна об'єднувати між собою два або більше світлин і тим самим створити нове зображення.

До глобального редагування цифрових зображень можна віднести:

- копіювання об'єкту – перенесення певної області цифрового зображення в іншу частину цього зображення;
- масштабування об'єкту – збільшення чи зменшення виділеного об'єкту на цифровому зображені.

Методи фальсифікації ретушування, посилення загалом використовуються у власних цілях, що призводить до створення фейків у інтернеті. Ретушування використовують для замазування об'єктів на цифровому зображені з цілю виділити конкретний об'єкт. Посилення дуже поширене в світі цифрових зображень. Його використовують для зміни кольору об'єкту, зміни різкості що в результаті на світлині об'єкти стають чіткіші.

Прикладом існуючих методів фальсифікації зображення може бути метод експертизи [5]. Завдяки експертизі можна знайти наявність слідів ретуші. Зрозуміти це можна завдяки відмінності тонів в околиці контуру об'єкта. Такі ознаки вказують на порушення цілісності інформації на світлині.

Самим поширеним методом фальсифікації світлини є застосування функції клонування [6]. Оскільки у кожної людини є телефон, загалом на який можна встановити ПО (простий у використані) для зміни цілісності цифрових зображень, або підробка цифрового зображення в Adobe Photoshop інструментом штамп. Це призводить до створення недостовірної інформації, підробки документів, шахрайства тощо. Ця проблема дуже поширена в наш час, їй присвячено багато робіт щодо виявлення клонування об'єкту на зображені.

Оскільки існує багато методів зміни цифрового зображення, що відрізняються за своєю суттю, існує й багато способів виявлення цілісності світлини. Оскільки з часом методи фальсифікації покращуються, то це вимагає постійного вдосконалення методів пошуку цілісності цифрових зображень.

Через те що в сучасному світі з кожним днем фальсифікація стає все більш поширеною. Одним із методів захисту цифрового зображення від несанкціонованих змін є будування у світлину водяного знаку.

Наведемо приклад виявлення клонування об’єкту на цифровому зображені методом дослідження метрик [7].

Для виявлення клонованих ділянок у задачах фальсифікацій пропонується наступна методика.

Нехай є зображення розміру I розміром $M \times N$. Побудуємо розбиття зображення на блоки $D = \{d_1, d_2, \dots, d_1\}$ розміру $p \times p$, що не перетинаються, і на блоки $C = \{C_1, C_2, \dots, c_s\}$, що перетинаються, розміру $p \times p$ де $p < M$ і $p < N$. Необхідно знайти для кожного блоку $d_n i = 1, \dots, l$ такий блок $c_j, j = 1, \dots s$, тобто $\text{Metrica}(d_i, C_j) = \delta$ де $\delta -$ деяка числова константа. Якщо такий блок знайдений, наприклад d_i та c_j, ці блоки позначаються як дубльовані. Об'єднання дубльованих блоків $\cup d_k, k \le l$, $\cup c_f, f \le s$ визначає можливі клоновані частини зображення I.

Розглянемо кілька найбільш відомих метрик для визначення подібності блоків у зображенні I, таких як MSE, коефіцієнт кореляції, відстань Мінковського та HS. Для обчислення метрик перетворимо зображення з кольорової моделі RGB на YUV

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.274 & -0.322 \\ 0.211 & -0.522 & 0.311 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix}.,$$

Рисунок 1.1 – Модель YUV

Модель YUV (див. рис. 1.1) заснована на розкладанні зображення на три складові: Y – найзначніша складова визначальна яскравість точки зображення; U,

V – дві колірні складові (їх називають кольоророзностями, т.к. U = G - R, V = G - B).

При роботі з метриками використовували лише складову Y. Найбільш простою метрикою є середньоквадратична помилка MSE (mean squared error) (рис. 1.2):

$$MSE = \frac{1}{p \times p} \sum_{rt} \left(d_{rt} - c_{rt}\right)^2,$$

Рисунок 1.2 – Середньоквадратична помилка MSE

де d_{rt}, c_{rt} – значення яскравості кожного пікселя блоку.

Коефіцієнт кореляції вказує на силу зв'язку між досліджуваними об'єктами. Розглянемо коефіцієнт кореляції Пірсона:

$$R = \frac{\sum_{rt} \left(d_{rt} - \overline{d_{rt}}\right)\left(c_{rt} - \overline{c_{rt}}\right)}{\sqrt{\sum_{rt} \left(d_{rt} - \overline{d_{rt}}\right)^2 \sum_{rt} \left(c_{rt} - \overline{c_{rt}}\right)^2}},$$

Рисунок 1.3 – Коефіцієнт кореляції Пірсона

де $\overline{d_{rt}}, \overline{c_{rt}}$ – середнє значення яскравості блоку.

Ступнева відстань Мінковського, є узагальненим виразом Евклідової відстані (рис. 1.4):

$$M = \left(\sum_{rt} \left|d_{rt} - c_{rt}\right|^p \right)^{\frac{1}{p}},$$

Рисунок 1.4 – Евклідова відстань

де p – довільне ціле число.

Метрика Histogram Similarity (HS) обчислюється за результатами побудови гістограми блоку зображення. Гістограма – це графік розподілу півтонів зображення, в якому по горизонтальній осі представлена яскравістю, а по вертикалі – число пікселів блоку зображення з цим значенням яскравості.

$$HS = \sum_{y=0}^{255} \left| f_{d_i}(y) - f_{c_j}(y) \right|,$$

Рисунок 1.5 – Метрика Histogram Similarity

де $f_{d_i}(y)$, $f_{c_j}(y)$ – значення гістограми для блоків d_i та c_j відповідно.

Для досліджень візьмемо зображення, отримані цифровою камерою. Додані клоновані ділянки за допомогою програми Adobe Photoshop CS. Результати фальсифікації не піддавалися стиску (рис. 1.6). Використовувалося розбиття зображення на блоки розміру 8x8 пікселів.

Рисунок 1.6 – Вихідні зображення (ліворуч) та фальсифіковані (праворуч)

До кожного фальсифікованого зображення як міра подоби блоків застосували розглянуті вище метрики. Позначаємо блоки як дубльовані, якщо значення метрик MSE (2), відстань Мінковського (4), HS (5), дорівнює нулю, а значення коефіцієнта кореляції (3) дорівнює 1. Знайдені блоки, що дублюються, поміщаються в підсумкове зображення у ті позиції, де їх було виявлено. Результати застосування метрик, виявленні клонованих ділянок представлені на рисунку 1.6. З наведених результатів можна помітити, що всі розглянуті метрики визначають практично точно клоновані ділянки. Для більш точної межі клонованих ділянок необхідно використовувати розбиття на блоки менших розмірів, що у свою черга може призвести до появи лжеблоків. Для прискорення роботи методу доцільно розбивати зображення на блоки великих розмірів, що у свою чергу може послабити точність виявлення меж клонованих ділянок, а також може призвести до не виявлення таких.

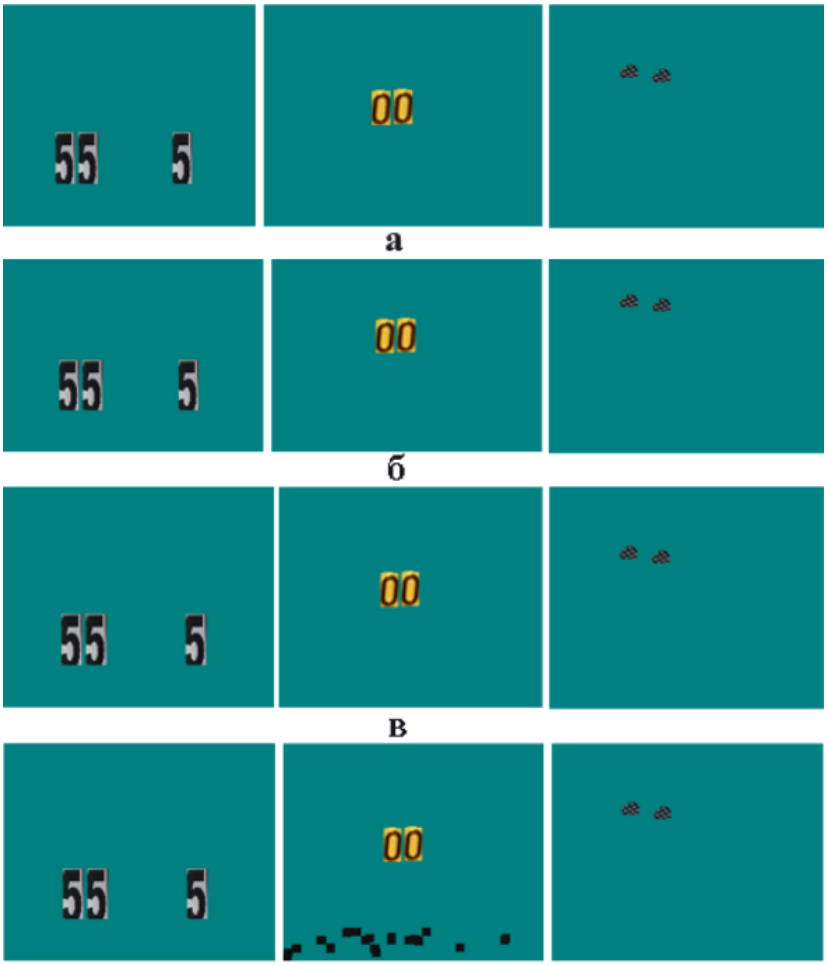

Рисунок 1.7 – Результати роботи метрик: а – MSE; б – коефіцієнта кореляції;

в – відстані Мінковського; г – Histogram Similarity

Наступним етапом наших досліджень є з'ясування, які значення набувають метрики поруч із виявленими дублюючими ділянками. Це дозволить експериментальним шляхом виявити поріг, який дозволить додатково обробляти ті блоки, значення метрик яких відповідають встановленим пороговим значенням. Під обробкою у цій ситуації розуміємо розподіл блоку на під блоки менших розмірів. Розглянемо, як розподіляються значення метрик для другого фальсифікованого зображення поруч із знайденими дублюючими блоками (див. рис. 1.7). У випадку метрик MSE, відстань Мінковського та HS фіксується мінімальне значення метрики для поточного блоку. Для коефіцієнта кореляції – максимальне значення.

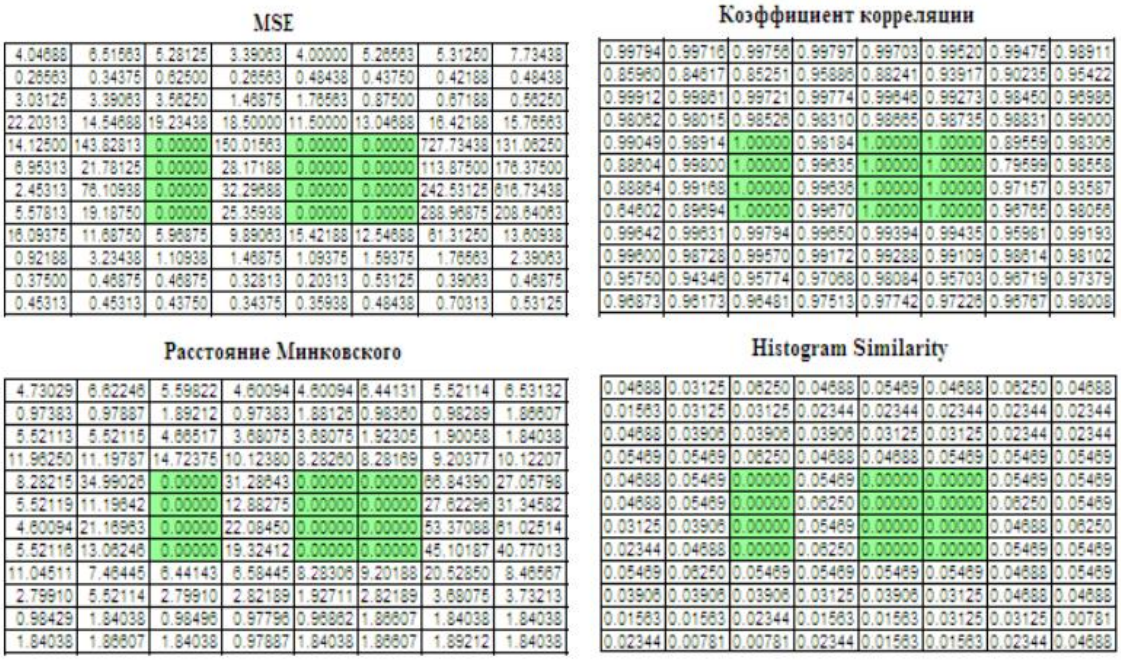

Рисунок 1.8 – Розподіл значень метрик біля дублюючих ділянок
(Дубльовані ділянки позначені зафарбованим фоном)

Для отримання більш точної межі ділянок, що дублюються, слідує розглянути блоки, що оточують знайдені або знаходяться між ними. Ці блоки можуть містити частини фальсифікацій, які можна буде виявити, розбивши їх на дрібніші. Для автоматизації процесу уточнення меж ділянок фальшування,

необхідно підібрати таке граничне значення, поява якого було б малоймовірно в інших блоки зображення, що не містять частин дублюючих ділянок. за отриманим результатам (див. рис. 1.8) можна побачити, що метрики MSE, відстань Мінковського та HS не підходять для такого уточнення.

В розділі був проведений аналіз сучасних методів виявлення порушень цілісності цифрових зображень, та метод виявлення фальсифікації світлин після того як була використана операція копіювання об’єкту. Вирішено задачу 1 з поставлених в меті роботи.

2 РОЗРОБКА МЕТОДУ ВИЯВЛЕННЯ ІНСТРУМЕНТУ «ЛАТКА» ЯК ПІДРОБКА СВІТЛИН

2.1 Опис використання інструменту «Латка»

На сьогоднішній день існує багато ПЗ, завдяки якому можна редагувати цифрові зображення, такі як Adobe Lightroom, Affinity Photo, PhotoLab тощо. Але найпопулярнішим із великого вибору програм є Adobe Photoshop, який надає можливість використовувати гнучкі налаштування, завдяки яким редагувати фото можна як вам заманеться. Це не лише популярне а і тяжке в освоєнні програмне забезпечення. Ненавченому користувачеві, який вперше зайде у фотошоп, буде незрозуміло, яке налаштування за що відповідає.

В Adobe Photoshop ви можете змінити фото настільки, наскільки вистачить фантазії. Всі ті налаштування, які ми роздивлялися, можна виконати в даній програмі. Але ми подивимся два методи підробки світлин, які по суті однакові, но виконують на зображенні різні дії. Це копіювання певної області інструментом «Штамп» та «Латка».

Використання інструменту «Штамп» дуже просте у використанні, для цього потрібно на лівій панелі інструментів обрати "Штамп" (рис.2.1).

Рисунок 2.1 – Інструмент «Штамп»

Затискаємо клавішу Alt і клацаємо лівою кнопкою миші по області, яку хочемо скопіювати (рис.2.2).

Рисунок 2.2 – Вибір області для копіювання

Відпускаємо клавішу Alt і наводимо курсор на те місце куди хочем перенести скопійований об’єкт (рис. 2.3).

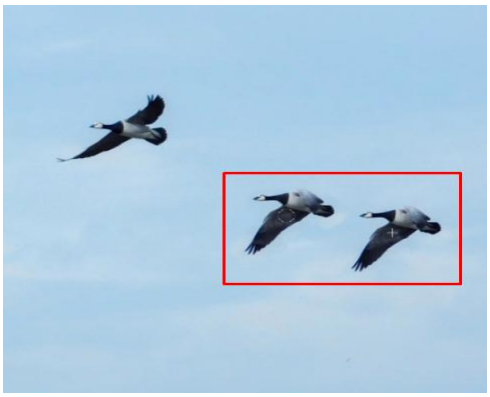

Рисунок 2.3 – Копіювання об’єкту

Ось так працює інструмент «Штамп». Завдяки цьому можна приховувати недоліки на зображені або робити більшу кількість об’єктів.

Далі ми розглянемо копіювання методом використання «Латки». Він знаходиться також на лівій панелі інструментів (рис. 2.4).

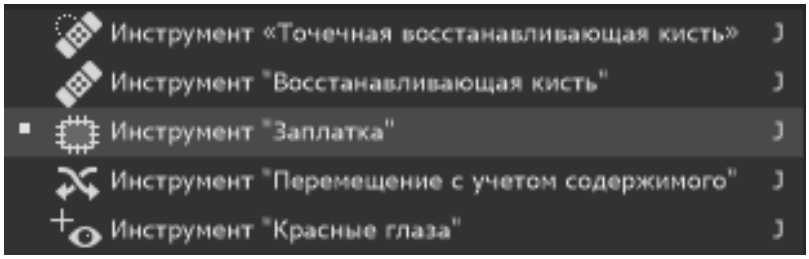

Рисунок 2.4 – Вибір інструмента «Латка»

Щоб скопіювати певну область потрібно її виділити, нажимаємо клавішу M та виділяємо (рис. 2.5).

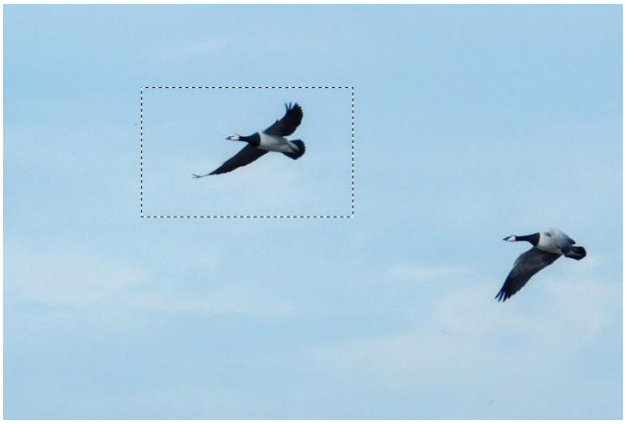

Рисунок 2.5 – Виділення об’єкту

Обираємо інструмент, наводимся курсором на виділений об’єкт та перетягуємо її на потрібну область (рис.2.6)

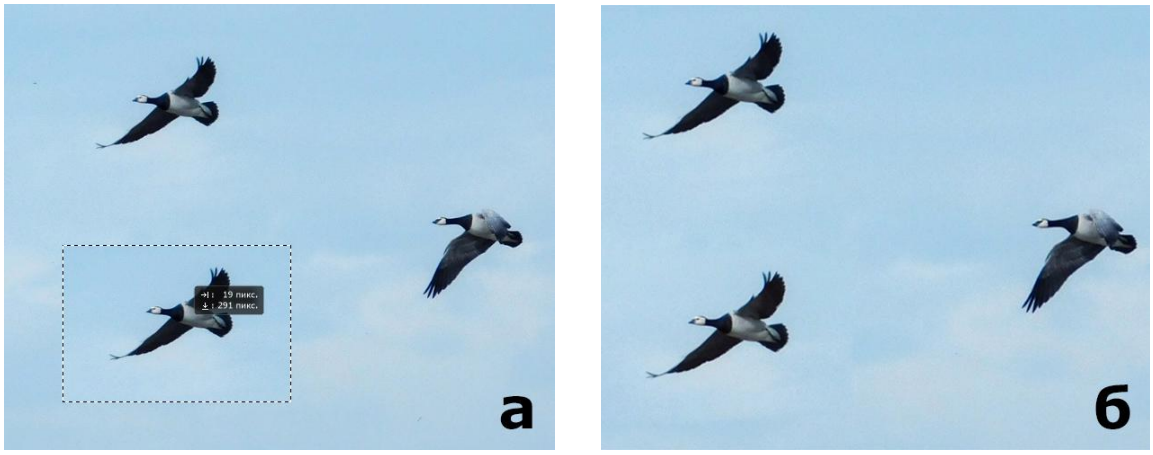

Рисунок 2.6 – Копіювання об’єкту інструментом «Латка»

Основна відмінність представлених способів копіювання у тому, що при використанні «штампу» об’єкт копіюється без змін, А при використані «Латки» перенесений об’єкт синтезує сусідній вміст для його органічного накладання на оточуючий вміст, тим сам на об’єкті змінюються значення RGB.

2.2 Створення алгоритму для фіксації використання інструмента «Латка» на цифровому зображенні.

Застосовуючи методи «Штамп» і «Латка» здійснено пошук однакових блоків в яких були задані деякі числові константи.

Якщо використовувати «Штамп», то копійований об’єкт без додаткових змін зберігає свої значення,. Тому метод добре впорається з пошуком однакових значень що дасть нам гарний результат.

Але при використанні інструменту «Латка», як вже було сказано, блоки міняють числові константи. На зображенні (рис.2.7) представлений один і той же блок 3x3, але після використання методу «Латка» міняються значення компоненти RGB, тому метод метрик нам не підходить.

Сама суть розбиття зображення на перетинаючі блоки популярна серед методів виявлення фальсифікацій [8], тому в цій роботі ми будемо теж її використовувати.

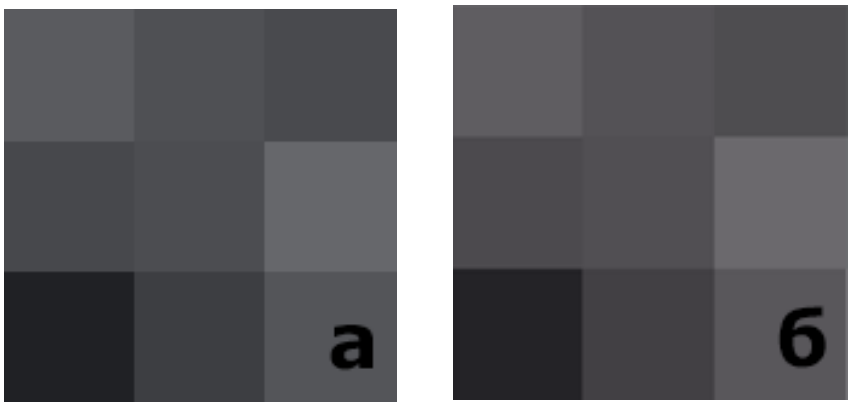

Рисунок 2.7 – Зміна значень компонентів зображення після використання інструменту «Латки».

Головним питанням є те, щоб цим алгоритмом можна було користуватися без завантаження додатковго ПЗ, тому прийшли до висновку що інтерфейсом даного алгоритмом буде виступати телеграм-бот [9].

Для створення телеграм боту нам потрібно зайти у Телеграм та в пошуку знайти бота під ім’ям @BotFather. Він створений розробниками Телеграму та доступний для всіх користувачів. Сенсом даного боту є те, що в ньому при натисканні однієї кнопки ви створюєте свого бота, надаєте йому ім’я та опис. Після цього @BotFather відправляє вам токен Bot API, який є інтерфейсом на основі HTTP, створений для розробників, які займаються створенням телеграм ботів.

Токен API є унікальним ідентифікатором програми, що запитує доступ до вашої служби. Ваша служба генеруватиме маркер API для програми, яка буде використовуватися при запиті вашого сервісу. Потім можна порівняти маркер, який вони надають, з тим, який ви зберігаєте для аутентифікації. Бот створено, тепер розробимо алгоритм який буде виконувати даний бот.

При переносі об’єкту інструментом «Латка» відбуваються зміна значень, але залишаються блоки які не змінюються при переносі. Томи ми прийшли до висновку, що краще для перевірки використовувати блоки розміром 3х3.

Алгоритм буде починатися з того, що ми завантажуємо в бота зображення файлом (тому що при звичайній відправці в телеграмі, зображення стискається). Починається розбиття зображення на перетинаючі блоки 3х3, відкидаючи верхній

та нижній ряд пікселів. Теж саме робимо з правим та лівим стовпчиком, щоб була можливість построїти рівну матрицю. Для того щоб виконати дану функцію був реалізований наступний код:

```
def get_im_matrix(patch):
    matrix_image = []
    img = Image.open(patch)
    rgb_im = img.convert('RGB')
    img_width, img_height = get_im_size(img)
    for height in range(img_height):
        line = []
        for width in range(img_width):
            r, g, b = rgb_im.getpixel((width, height))
            line.append([r, g, b])
        matrix_image.append(line)
    return matrix_image
```

Щоб почати перевірку блоків ми маємо обрати значення, яке буде порівнюватися між блоками. Цим параметром буде виступати червона компонента пікселю. Для того щоб порівняти між собою блоки с компонентною потрібно з цих блоків построїти матрицю. Данну функцію виконую наступна частина коду:

```
image_matrix = get_im_matrix(patch)
    matrix_height, matrix_width =
matrix_size(image_matrix)
    matrix_for_mm = generate_new_matrix(matrix_height-
around*2, matrix_width-around*2)
    for height in range(around,
len(matrix_for_mm)+around):
        for width in range(around, len(matrix_for_mm[-
1])+around):
            mm_matrix = []
```

```
                for i in range(-1 * (step-1), step):
                    local_matrix = []
                    for j in range(-1 * (step-1), step):

local_matrix.append(image_matrix[height+i][width+j][step-
2])
                    mm_matrix.append(local_matrix)
                matrix_for_mm[height-around][width-
around].append(mm_matrix)
```

Після перевірки блоків потрібно якимось чином показати місця схожих блоків. Для цього ми створюємо пусту матрицю по розмірам зображення для подальшого використання її в цілях відображення однакових блоків. Для створення матриці був розроблений наступний програмний код:

```
def generate_new_matrix(matrix_height, matrix_width):
    new_matrix = []
    for width in range(0, matrix_height):
        new_matrix.append([])
        for height in range(0, matrix_width):
            new_matrix[-1].append([])
    return new_matrix
```

Ми створили пусту матрицю. Тепер для того, щоб відзначати копіюючи місця, розробимо наступний алгоритм: якщо були знайдені однакові блоки, то ми зафарбовуємо його червоним кольором, якщо збіги не знайдені - зафарбовуємо чорни кольором. Щоб це виконувалось в автоматичному режимі, розробляємо наступний код:

```
         matrix_for_im =
generate_new_matrix(len(matrix_for_mm),len(matrix_for_mm[-
1]))
        for height in range(0, len(matrix_for_mm)):
            print(f'{height}/{len(matrix_for_mm)-1}')
```

```
            for width in range(0, len(matrix_for_mm[-1])):
                counter =
super_puper_array.count(matrix_for_mm[height][width])
                counter = counter if counter > porog else
0
                counter = 255 if counter*koef > 255 else
counter*koef

matrix_for_im[height][width].append(counter)
                matrix_for_im[height][width].append(0)
                matrix_for_im[height][width].append(0)
```

Після проведеної роботи користувачу відправляється підсумковий варіант, який можна аналізувати і робити висновки.

Тепер перевіримо працездатність даного алгоритму. На зображені, яке ми зменшимо в розмірах для швидшої роботи алгоритму, в форматі з втратами (JPEG) (рис. 2.8) ми інструментом «Латка» виконаємо копіювання числа "0" (рис. 2.9) та методом виявлення покажемо результат (рис. 2.10).

Рисунок 2.8 – Оригінальне зображення

Рисунок 2.9 – Приклад копіювання об’єкту

Рисунок 2.10 – Результат перевірки виявлення копіювання

Як ми бачимо на результаті (рис. 2.10), є чіткі області в місцях, де був копійований «0», але виникають питання по червоній області, в якій ніяких змін ми не робили . Тому давайте перевіримо на зображені в форматі без втрат (BMP).

По результату (рис. 2.11) можна зробити висновок, що на зображені без втрат є чіткі виділення копійованого об’єкту, і майже немає хибних тривог. Якщо проаналізувати зображення (рис. 2.9), можна зробити висновок: область, на якій є хибна тривога при стисненні зображення, створює між собою ідентичні блоки, тому алгоритм помічає цю область як копійовану.

Рисунок 2.11 – Результат перевірки виявлення копіювання на зображенні без втрат

Для аналізу інформації про хибні тривоги був проведений експеримент з використанням 50 цифрових зображень, в яких було виконане копіювання об’єкту інструментом «Латка», та збережено у форматі (JPEG). З тими ж цифровими зображеннями проведені ті самі маніпуляції, але збережено у форматі без втрат (BMP).

Розглянемо 2 випадки:

- перший випадок – це відсоток похибок на місцях де не було копіювання, тобто місця де можуть попастися блоки які з’явилися при стиску зображення. До таких випадків відносяться області схожого кольору, такі випадки виникають у місцях, на яких не виконувалося ніяких маніпуляцій;

- другий випадок – відсоток похибок на місцях, де було копіювання, тобто виділені блоки, які з’явилися на місці копіювання, але не мають схожих виділень. Результат представлений в таблиці 2.1.

Таблиця 2.1 – Результат експерименту

Формат	Відсоток похибок на місцях, де не було копіювання	Відсоток похибок на місцях, де було копіювання
JPEG	100%	18%
BMP	14%	8%

Як видно з таблиці 2.1, за відсотком похибок на місцях, де не було копіювання на цифровому зображенні формату JPEG, помилки зустрічаються на всіх зображеннях. Це зумовлено тим, що при стисканні на зображенні на конкретних місцях з’являються схожі блоки між собою. На світлинах з форматом BMP похибок майже немає. З цього ми робимо висновок, що похибки, які з’являються на стислих зображеннях, знаходяться в місцях великого скупчення однакового тону.

У даному розділі був розглянутий метод виявлення фальсифікації зображення, на якому було виконано копіювання об’єкту інструментом «Латка». Аналіз методу показав, що виявити латку на світлині вдалося. На основі отриманих результатів розроблено метод виявлення результатів масштабування як підробки цифрового зображення з точки зору використання фото підробки у форматі з втратами та без. В даному розділі була досягнута 2 мета, яка була поставлена у бакалаврській роботі.

3 СТВОРЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ РЕАЛІЗОВАНИЙ ТЕЛЕГРАМ-БОТОМ ВИЯВЛЕННЯ ІНСТРУМЕНТУ «ЛАТКА»

3.1 Середовище програмування

Алгоритм, який ми розробили в другому розділі, був реалізований в програмному середовищі Python 3.10.4. [10] Середовище Python на сьогоднішній день дає використовувати велику кількість бібліотек, завдяки яким можна вирішити безліч конкретних типів задач. Бібліотека – це набір модулів та функцій, які полегшують деякі специфічні операції у використанні конкретної мови програмування.

Python – це інтерпретована високорівнева об'єктно-орієнтована мова програмування з динамічною семантикою. Його високорівневі вбудовані структури даних у поєднанні з динамічним типізацією та динамічним зв’язуванням роблять його дуже привабливим для швидкої розробки додатків і для використання в якості мови сценаріїв або як мови підключення для підключення існуючих компонентів.

Простий і легкий для вивчення синтаксис Python наголошує на зручності читання і, отже, знижує вартість обслуговування програми. Python підтримує модулі та пакети, що сприяє модульності програм та повторному використанню коду. Інтерпретатор Python та велика стандартна бібліотека доступні у вихідному або бінарному вигляді безкоштовно для всіх основних платформ і можуть вільно поширюватися.

Програмісти часто закохуються в Python через більш високу продуктивність, яку він пропонує. Оскільки етапу компіляції немає, цикл редагування-тестування-налагодження дуже швидкий. Налагоджувати програми Python легко: помилка або неправильне введення ніколи не призведе до помилки сегментації. Натомість, коли інтерпретатор виявляє помилку, він створює виняток. Коли програма не перехоплює виняток, інтерпретатор друкує трасування стека.

Налагоджувач вихідного коду дозволяє перевіряти локальні та глобальні змінні, обчислювати довільні вирази, встановлювати точки зупину, виконувати код рядок за рядком тощо. Налагоджувач написаний на самому Python, що демонструє можливості інтроспекції Python. З іншого боку, найшвидший спосіб налагодити програму та додати кілька операторів друку до вихідного коду.

Багато відомих нам компаній та організацій використовують Python: Spotify та Amazon використовують Python для аналізу даних та створення рекомендацій. Walt Disney використовує Python як скриптову мову для анімації. YouTube та Instagram повністю написані на Python. Якщо цього недостатньо, є ще NASA: їхня система автоматизації процесів WAS теж створювалася засобами Python.

Особливість даної мови програмування полягає в тому, що Python має безліч бібліотек, за допомогою яких досягти своїх цілей у програмуванні можна набагато швидше. Наприклад, Pygame дозволяє написати ігри та мультимедійні програми. Для роботи із bigdata використовується бібліотека Pandas. Django використовується для серверної частини для розробки різних програм.

Переваги Python.

1. Швидкість розвитку. У порівнянні з іншими популярними мовами (Java, C), програми Python вимагають набагато менше коду. Це значно прискорює розробку, дозволяючи створювати складне програмне забезпечення швидше, ніж інші QoS.

2. Логічна граматика. Логічний синтаксис мови спрощує читання та розуміння коду, що робить його досить легким для вивчення. Python по праву вважається однією з найкращих мов програмування для початківців.

3. Масштабованість. Програми та програми, написані на Python, легко розширювати та розширювати завдяки можливості налаштувати їхню логіку високого рівня.

4. Універсальність. Python – це інтерпретована мова, яка використовується для кодування майже на всіх сучасних платформах. Він не потребує компіляції, а його код можна записати як звичайні текстові документи.

Одним з важливих факторів бурхливої популяризації Python вважається численна спільнота розробників та ентузіастів цієї мови. Його розвиток ведеться на базі регулярно оновлюваної та чітко регламентованої документації PEP (пропозицій щодо розвитку Python)

3.2 Бібліотеки Python, які були використані в алгоритмі

Як було з'ясовано, в Python є дуже багато бібліотек, які можна використовувати для будь-яких задач. Для роботи з цифровими зображеннями будемо використовувати бібліотеку Pillow, математичні функції з матрицями будемо використовувати бібліотеку NumPy.

Pillow – це зручний форк PIL, створений Alex Clark і Contributors. PIL – це бібліотека зображень Python, створена Фредріком Лундом та його учасниками. Станом на 2019 рік розробка Pillow підтримується Tidelift.

Бібліотека зображень Python додає можливості обробки зображень до вашого інтерпретатора Python.

Ця бібліотека забезпечує велику підтримку форматів файлів, ефективне внутрішнє уявлення та досить потужні можливості обробки зображень.

Основна бібліотека зображень призначена для швидкого доступу до даних, що зберігаються у кількох основних форматах пікселів. Він повинен забезпечити міцну основу загального інструменту обробки зображень.

NumPy – це open-source модуль для python, який надає загальні математичні та числові операції у вигляді пре-компільованих, швидких функцій. Вони поєднуються у високорівневі пакети. Вони забезпечують функціонал, який можна порівняти із функціоналом MATLAB. NumPy (Numeric Python) надає базові методи для маніпуляції з великими масивами та матрицями. SciPy (Scientific Python) розширює функціонал numpy величезною колекцією корисних алгоритмів, таких як мінімізація, перетворення Фур'є, регресія та інші прикладні математичні техніки.

3.3 Взаємодія алгоритму з Telegram

Щоб наш алгоритм міг працювати через бот телеграм було обрано бібліотеку telebot. Telebot – це бібліотека завдяки якій можливо взаємодіяти з API Telegram через ідентифікатор телеграм-боту.

Програма побудована на взаємодії користувача Telegram з чат-ботом. Чат-бот – це автоматизований багатофункціональний помічник, який може показувати інформацію передплатникам та збирати інформацію на запит згідно із заздалегідь підготовленими сценаріями.

Для взаємодії з чат-ботом потрібно вказати параметр в код, таким є token, який можна отримати у офіційного чат-боту телеграм який відповідає за створення нових ботів. За створення чат-боту відповідає @BotFather, якого можна знайти в пошуку телеграм.

3.4 Алгоритм роботи користувача з чат-ботом

Для зручного використання алгоритму по пошуку фальсифікацій світлин, було прийнято рішення розробити Телеграм-бот. Це зумовлено тим, що Телеграмом на сьогоднішній день користуються майже всі, так як він надійний в питаннях безпеки та конфіденційності та простий у використанні.

Першим кроком ми заходимо у Телеграм та в пошуку вбиваємо назву боту "@Check_on_photoshop_bot" , далі переходимо в чат з ботом,який вказаний на рисунку 3.1

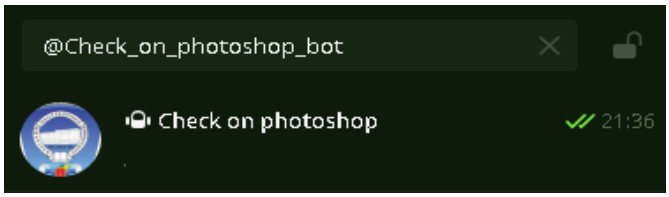

Рисунок 3.1 – Чат-бот який нам потрібен

Щоб почати роботу з ботом, потрібно у полі введення повідомлення написати команду /start. Після цього бот відправить вам повідомлення з вимогами які потрібно виконати перед тим як почати роботу з чат-ботом (рис. 3.2).

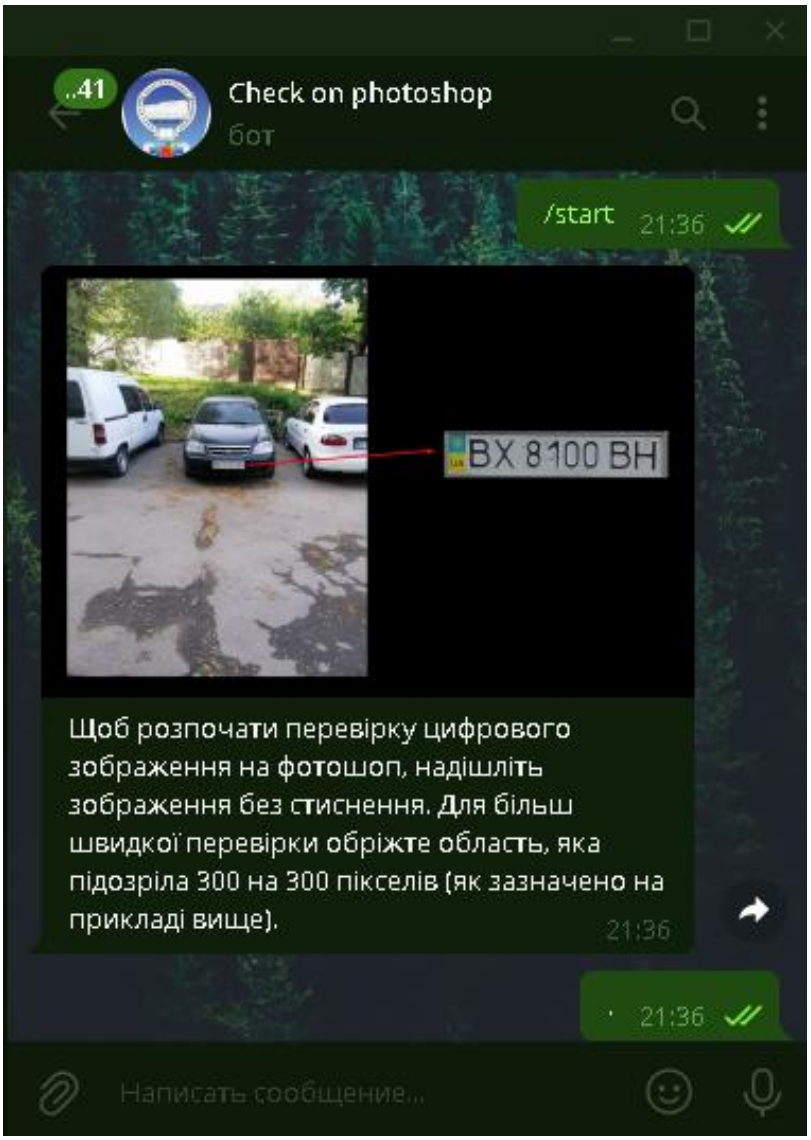

Рисунок 3.2 – Вимоги, які потрібно виконати перед тим як почати роботу з чат-ботом

Наступним кроком буде відправка зображення способом «Відправити без стискання». Це зроблено для того, щоб Телеграм не зіпсував якість цифрового зображення, і не привело до збою роботи алгоритму. Після відправки зображення чат-боту він надсилає відповідні повідомлення, у якому ми бачимо, що пошук фальсифікації почався (рис.3.3).

Після виконаної роботи бот відправляє вам оброблений варіант зображення, яке можливо аналізувати та робити висновки щодо цілісності зображення, яке ми надсилали (рис. 3.4).

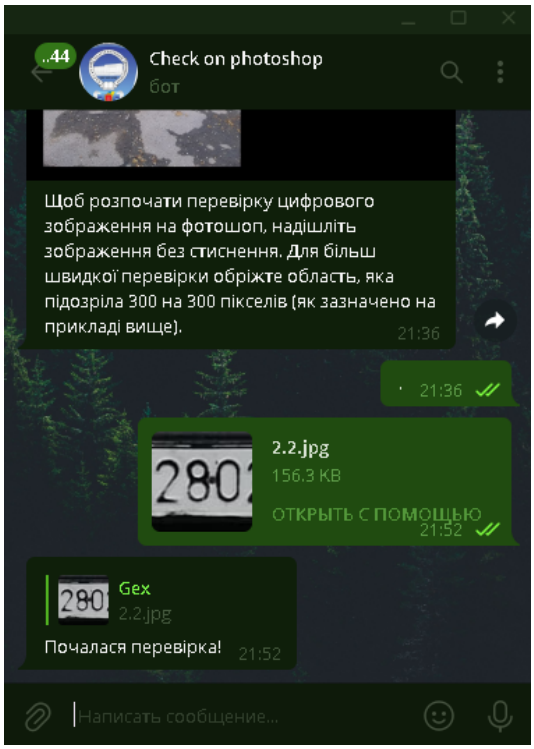

Рисунок 3.3 – Початок роботи алгоритму

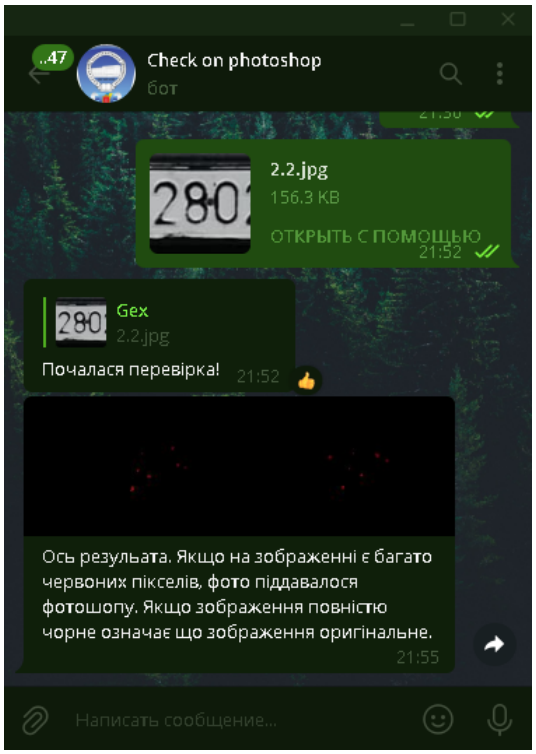

Рисунок 3.4 – Оброблений варіант зображення з виявленою фальсифікацією

Для покращення та спрощення роботи з ботом здійснено програмування боту на автоматичну відповідь користувачу. За відправлення чат-боту сповіщення

«Щоб розпочати перевірку цифрового зображення на фотошоп, відішліть зображення без стиснення. Для більш швидкої перевірки обріжте область, яка підозріла 300 на 300 пікселів (як зазначено на прикладі вище).», та надання користувачу приклад обрізання зображення був розроблений наступний код:

```
@bot.message_handler(commands=["start", "help"])
    def start(message):
        bot.send_photo(message.chat.id,
open(f'G:/MAKS/Uchoba/Diplom/bot/files/download/22.jpg',
'rb'),caption=messages_text.start_message)
```

```
start_message = 'Щоб розпочати перевірку цифрового зображення на фотошоп, відішліть зображення без стиснення. Для більш швидкої перевірки обріжте область, яка підозріла 300 на 300 пікселів (як зазначено на прикладі вище).'
```

Для надходження сповіщення «Почалася перевірка!», початку роботи алгоритму по виявленню фальсифікації зображення, яке надіслав користувач, та відсилання чат-бота підсумковий результат після перевірки, була написана наступна частина коду:

```
        with open(patch, 'wb') as download_photo:
            download_photo.write(downloaded_file)
        bot.reply_to(message,
text=messages_text.getfile_message)
        calculator(patch, message.document.file_name)
        bot.send_photo(message.chat.id,
open(f'{upload_dir}/{message.document.file_name}',   'rb'),
caption=messages_text.final_message)
```

```
getfile_message = 'Почалася перевірка!'
```

```
final_message = 'Ось результат. Якщо на зображенні є багато червоних пікселів, фото піддавалося фотошопу. Якщо зображення повністю чорне означає що зображення оригінальне.'
```

В головному коді є чотири основні модулі, які відповідають за роботу чат-боту та самого алгоритму. Цими модулями є bot.py який відповідає за взаємодію алгоритму з чат-ботом (Додаток А), за розрахунки відповідає модуль calculator.py (Додаток Б). За запуск боту та за відправки ним сповіщань відповідають модулі main.py (Додаток В) tamessages_text.py (Додаток Г).

В даному розділі ми розглянули покроково яким чином користувачу почати перевірку через чат-бот, тим самим ми вирішили 3 задачу, яка була поставлена в меті роботи.

4 ОХОРОНА ПРАЦІ

Одним з найважливіших чинників в життєдіяльності людини є її здоров'я та безпека. В любій сфері людина піддається під шкідливі фактори, які негативно впливають на стан людини її здоров'я. Тому основною частиною є охорона праці, завдання якої полягає у захисті людини від негативних факторів у будь-якій сфері. Завдання охорони праці полягає у поліпшені умов праці, доведені цих умов до ідеалу, та зниження ризиків захворювань та травм на робочих місцях.

В наш час істотно збільшується кількість спеціалістів, які працюють комп'ютерною технікою, тому давайте визначимо, що може впливати на людину під час роботи за комп'ютером.

Програмісти в основному працюють в офісних приміщеннях. В приміщеннях, які обкладенні комп'ютерами та іншими електронними пристроями, виникає ряд факторів, які негативно впливають на людину.

Небезпечні фактори на робочому місці:

- підвищений рівень шуму на робочому місці;
- недостатня освітленість робочої зони;
- підвищений рівень електромагнітних випромінювань;
- підвищена чи знижена вологість повітря;
- підвищена чи знижена температура повітря робочої зони.

Згідно документів ДСН 43.3.6 037-99 рівень шуму, який допустимий на робочому місці людини яка тривалий час працює за комп'ютером, становить не більше 50 дБА. Якщо на робочому місці відсутні прилади другу, кондиціонери та інші прилади які видають звук, при таких умовах фактичне зашумлення робочого місця має дорівнювати до 45дБА.

Для того щоб дотримуватися норм потрібно.

1. Зменшити кількість приладів друку та копіювання до мінімума, та виділити місце віддалене від працівників.

2. Пристрої для дотримання мікроклімату в нормі, розташувати в 5м від початку робочих місць.

3. Перейти на новітні офісні персональні комп’ютери, які будуть менше створювати шум.

4. Прилади, які використовують для життєдіяльності (кавоварки, мікрохвильові пічі, холодильники тощо), розташовувати в окремих приміщеннях (кімнатах відпочинку, кухнях тощо).

5. Робочі місця відокремити перегородкою.

Шум це один із основних факторів, але немало важливим фактором є положення тіла при сидінні під час роботи за комп’ютером. Під час роботи за комп’ютером ваше робоче місце має бути обладнане кріслом, яке буде тримати у правильному напрямку ваше тіло. Якщо не дотримуватися правил які вказують на правильне положення тіла під час сидіння за робочим місцем, то це може привести до того що ваше фізичне здоров’я постраждає.

Ще одним із основних факторів на робочому місці є освітлення. Якщо недотримуватися норм, то це може призвести до погіршення зору. У зв’язку з тим, що робота проходить у приміщенні, освітлення має бути приближеним до сонячного світла завдяки штучним джерелам світла.

Частіше за все на робочу місці освітлення представляє собою комбінацію природного та штучного освітлення. У вечірній період часу природне освітлення у приміщенні становить 400лк. Такий показник не є нормою що до документу ДБН В.2.5-28-2006. Нормою освітлення при роботі з комп’ютером або документами становить не менше ніж 500лк. Тому в вечірній період робоче місце має бути обладнане штучними джерелами освітлення.

На зорову систему люди впливає не лише освітлення, а і положення екрану відносно очей людини, ці норми описані у пунктах 4.14-4.15 ДСанПіН 3.3.2.007-98. Згідно документа, нормою у відстані очей людини до екрану має складати не менше 600-700мм. Окрім відстані важливим пунктом є забезпечення зручного зорового спостереження у вертикальній площині під кутом 30 градусів.

Рівень напруженості електричного поля, під впливом якого може потрапити людина яка працю за комп’ютером становить 15кВ/м протягом восьмигодинного робочого дня. У документі ДСанПІН 3.3.2.007-98, норма напруження

електричного поля має дорівнювати 15кВ/м. Для іонізуючих електромагнітних випромінювання нормативне значення визначено як 7,74 * 10-12 А/кг, а фактичне значення 7 * 10-12 А/кг.

Параметри мікроклімату теж мають відповідати нормам які указані у документі ДСанПіН 3.3.2.007-98. Температура повітря на робочому місці нормативне значення має дорівнювати 22-24°С тим часом як фактичне значення має дорівнювати 22°С. Не менш важливим чинником є вологість повітря. Нормою вологості повітря у приміщені де працюють з комп'ютером становить 40-60%. Швидкість руху повітря в приміщені має становити як указано у документі - 0,1 м/с.

Професійна діяльність людей, що працюють у сфері інформатики, на робочих місцях, розташованих у спеціально обладнаних кабінетах, може супроводжуватися великим шумом. Щоб усунути цю проблему, використовуйте такі заходи як зменшення кількості обладнання, яке спричиняє надмірний шум, правильне розташування, контроль якості обладнання (якщо надмірний шум пов'язаний з несправною роботою обладнання), використання звукоізоляційного обладнання та звуко поглиначів. Рівень шуму також може залежати від розташування будівлі, де розташований офіс. Тому важливо, щоб місце розташування трималося на значній відстані від місць, де багато шуму. Прикладом такого місця є аеропорт чи будівництво, місце скупчення великої кількості народу.

Робоча поза люди під час роботи за комп'ютером це дуже важливий фактор який впливає на фізичний стан людини. Окрім норм яких потрібно дотримуватися щодо положення тіла при роботі за комп'ютером які вказані у документі ДСанПіН 3.3.2.007-98, також можна скористатися комплексом вправ для хребта, рук, та для поліпшення кровообігу. Комплекс вправ також вказаний у документі ДСанПіН 3.3.2.007-98.

Для забезпечення безпечних умов роботи програміста його робоче місце повинно мати достатнє освітлення. Щоб отримати достатнє освітлення, необхідно використовувати сумісне освітлення. Природного освітлення може бути

достатньо при денному освітленні, але крім цього дуже важливо додаткове штучне освітлення. Крім присутності природнього освітлення, важливо також правильне розташування та розподілення штучних джерел світла у приміщені. Комбінація цих пристроїв повинна забезпечувати задовільний освітлений показник у будь-який робочий час доби.

Для дотримання норм вологості повітря у приміщені слід використовувати зволожувачі повітря. Робота цього пристрою повинна сприяти тому, щоб вологість не знижувалася і не піднімалася щодо параметрів які є нормою і дорівнюють 40-60%. Низькі показники вологості при тривалому впливі на людину може привести до пошкодження дихальних шляхів. Якщо у приміщені високий рівень вологості, то слід використовувати якісну систему вентиляції.

Нормований рівень температури в приміщенні має бути досягнений шляхом встановлення системи опалення. Температура в офісі по нормам які вказані у документі ДСанПіН 3.3.2.007-98 має становити 22-24°С.

Пожежа є типовим ризиком для будь-якого виробничого процесу або іншої діяльності людини, пов’язаної з роботою в приміщенні. Пожежі можуть завдати величезних економічних і матеріальних збитків і навіть смерті. Тому пожежна безпека є особливо важливою в галузі охорони праці.

Якщо річ заходить про питання пожежної безпеки, в першу чергу слід зазначити наявність вогнегасників, їх кількість та тип на тому чи іншому виробничому процесі. Тип вогнегасників залежить від наявності речовин, матеріалів та пристроїв які використовуються у робочих приміщеннях. В той же час кількість залежить від площі приміщення та його типу.

При наявності первинних речовин у приміщені поділяються на класи, що є об’єктами пожежної небезпеки визначає клас будівлі. У документі НАПБ Б.03.002-2007 зазначені категорії приміщень та будівель.

Категорія В (Пожежонебезпечна). Горючі гази, легкозаймисті, горючі і важко горючі рідини, а також речовини та матеріали, які здатні при взаємодії з водою, киснем повітря або один з одним вибухати і горіти або тільки горіти; горючий пил і волокна, тверді горючі та важко горючі речовини і матеріали, за

умови, що приміщення, в яких вони знаходяться (обертаються), не відносяться до категорій А, Б

Після вивчення документа НАПБ 105-03 ми прийшли до висновку, що для нашої діяльності підходить категорія В. Приміщення с цією категорією потребує наявність таких засобів пожежної безпеки, як евакуаційні виходи, плани евакуації, системи сповіщення та первинні засоби пожежогасіння.

Оскільки наша будівля, в якій знаходиться робоче місце потрапляє до категорії В, можемо зробити такі розрахунки, розмір робочого місця дорівнює 50 кв.м, таке місце повинно бути оснащене двома вуглекислотними вогнегасниками, кожен з яких повинен містити 6 кг вогнегасної речовини.

Ми обираємо такий вогнегасних тому що ми прямцюємо в приміщені у якому знаходиться багато електричних приладів.

Вогнегасники вуглекислотні – це вогнегасники, які належать до газової категорії, оскільки вогнегасним засобом у них є рідкий вуглекислий газ в зарядному балоні. Перебуваючи під власним надлишковим тиском від 5,7 до 15 МПа, він, коли його випускають назовні, гасить осередки займання.

Цей тип вогнегасників найпоширеніший в сучасному світі, він простий у використанні та не вимагає спеціальних навичок.

В підсумку проведеної роботи яка була спрямована на виявлення небезпечних і шкідливих факторів, які можуть з’явитися на робочому місці програміста ми можемо виділити 5 шкідливих факторів.

1. Підвищений рівень шуму на робочому місці.
2. Недостатня освітленість робочої зони.
3. Підвищений рівень електромагнітних випромінювань.
4. Підвищена чи знижена вологість повітря.
5. Підвищена чи знижена температура повітря робочої зони.

При дотримані всіх правил та норм, на робочу місці не будуть виникати ніякі шкідливі фактори, та це буде впливати на продуктивну роботу без пошкоджень фізичного стану програміста.

Оскільки робочі міста програмістів оснащені ПК, то при їх використані можливе виникнення наступних ситуацій.

1. Короткі замикання.
2. Перевантаження.
3. Підвищення перехідних опорів в електричних контактах.
4. Перенапруження.
5. Виникнення струмів витоку.

При виникненні ситуацій котрі перечисленні зверху відбувається різке виділення теплової енергії, яка може призвести до виникнення пожежі.

ВИСНОВКИ

В результати виконання кваліфікаційної роботи було виконанні наступні задачі.

Проаналізували сучасні методи підробок світлин та проблеми виявлення фальсифікації цифрових зображень взагалі та методи виявлення копіювання об’єктів на зображені. В результаті з’ясували, що методів достатньо, але вони всі спрямовані на конкретний метод копіювання і з цього виходить, що задача виявлення фальсифікацій цифрових зображень не є до кінця вирішеною. Зокрема, відомі методи не вирішують проблему фальсифікації зображення повністю, а саме – виявлення копіювання об’єкту методом урахування змісту («Латка»);

Визначено параметри, завдяки яким буде здійснюватися пошук після розбиття зображення на блоки 3x3. Цим параметром виступає червона компонента пікселю, аналіз яких дозволить знайти однакові блоки, тим самим дізнатися, чи було здійснення копіювання на цифровому зображені;

Розроблено та реалізовано чат-ботом у Telegram метод виявлення копіювання об’єкту, яке було здійснено інструментом «Латка» в умовах збереження світлини у форматі з втратами та без.

ПЕРЕЛІК ПОСИЛАНЬ

1. Графічні формати. URL: https://uk.wikipedia.org/wiki/Графічні_формати.

2. Зорило В.В., Лебедєва О.Ю. Комплексний метод виявлення та локалізації областей клонування в цифрових зображеннях. *Праці Одеського політехнічного університету*. Одеса, 2015. № 1(45). С. 101-105.

3. Зоріло В.В., Карпова О.А. Алгоритм виявлення обробки цифрового зображення фільтром «Motion blur». *Інформатика та математичні методи моделювання*. 2019.Т. 9, №1-2. С. 49-58.

4. Зоріло В.В., Кіосєва О.І. Модифікація алгоритму виявлення штучного підвищення різкості цифрового зображення. *Інформатика та математичні методи в моделюванні*. 2018. Т.8. №2. С. 114-120.

5. Григоренко С.Н. Сравнительный анализ эффективности метода выявления результатов клонирования в условиях постобработки изображения. *Інформатика та математичні методи моделювання.* 2016. Т. 6, №2. С. 193-199.

6. Зорило В.В. Виявлення клонування як фальсифікація цифрового зображення. *Вісник Національного технічного університету «ХПІ». Тематичний випуск «Системний аналіз, управління та інформаційні технології».* Харків, 2011. № 35. С. 31-38.

7. Лебедєва О.Ю., Лебедєв Ю.Ф. Дослідження метрик використовуваних при виявленні клонування ділянок зображень в задачах виявлення фальсифікації. *Вісник Національного технічного університету «ХПІ».* Харків, 2011. № 35. С. 25-31.

8. Бобок І.І. Метод відокремлення клону від прообразу в цифровому зображенні в умовах відсутності відмінностей при їх постобробці. *Інформатика та математичні методи моделювання.* 2017. Т. 6 , №4, с. 276-284

9. Telegram для навчання: створюємо канали та ботів. URL: https://naurok.com.ua/post/telegram-dlya-navchannya-stvoryuemo-kanali-ta-botiv

10. Офіційний сайт Python. URL: https://www.python.org/

Додаток А. Програмний код bot.py

```
import telebot
import messages_text
from calculator import calculator
upload_dir = './files/upload'
download_dir = './files/download'
token = '5324732342:AAHUmzrPg24OGYyhRsvPYB-fYCC9dM2aZJ0'
def bot_start():
    bot = telebot.TeleBot(token)
    @bot.message_handler(commands=["start", "help"])
    def start(message):
        bot.send_photo(message.chat.id,
open(f'G:/MAKS/Uchoba/Diplom/bot/files/download/22.jpg',
'rb'),caption=messages_text.start_message)
    @bot.message_handler(content_types=["document"])
    def message_come(message):
        downloaded_file =
bot.download_file(bot.get_file(message.document.file_id).file_path)
        patch = f'{download_dir}/{message.document.file_name}'
        with open(patch, 'wb') as download_photo:
            download_photo.write(downloaded_file)
        bot.reply_to(message,
text=messages_text.getfile_message)
        calculator(patch, message.document.file_name)
        bot.send_photo(message.chat.id,
open(f'{upload_dir}/{message.document.file_name}', 'rb'),
caption=messages_text.final_message)

    bot.polling(True)
```

Додаток Б. Програмний код calculator.py

```
from PIL import Image
import numpy as np
import os
step = 2
koef = 64
porog = 2
around = 3

def get_im_size(image):
    return image.size

def matrix_size(matrix):
    return len(matrix), len(matrix[-1])

def get_im_matrix(patch):
    matrix_image = []
    img = Image.open(patch)
    rgb_im = img.convert('RGB')
    img_width, img_height = get_im_size(img)
    for height in range(img_height):
        line = []
        for width in range(img_width):
            r, g, b = rgb_im.getpixel((width, height))
            line.append([r, g, b])
        matrix_image.append(line)
    return matrix_image

def generate_new_matrix(matrix_height, matrix_width):
#генерирует новою пустую матрицу
    new_matrix = []
    for width in range(0, matrix_height):
        new_matrix.append([])
        for height in range(0, matrix_width):
```

```
                new_matrix[-1].append([])
        return new_matrix

    def array_to_im_array(data):
        return np.squeeze(np.asarray(data))

    def calculator(patch, name):
        image_matrix = get_im_matrix(patch)
        matrix_height, matrix_width = matrix_size(image_matrix)
        matrix_for_mm = generate_new_matrix(matrix_height-around*2,
matrix_width-around*2)
        for height in range(around, len(matrix_for_mm)+around):
            for width in range(around, len(matrix_for_mm[-
1])+around):
                mm_matrix = []
                for i in range(-1 * (step-1), step):
                    local_matrix = []
                    for j in range(-1 * (step-1), step):

local_matrix.append(image_matrix[height+i][width+j][step-2])
                    mm_matrix.append(local_matrix)
                matrix_for_mm[height-around][width-
around].append(mm_matrix)
        super_puper_array = []
        for line in matrix_for_mm:
            for elem in line:
                super_puper_array.append(elem)
        print(super_puper_array)
        matrix_for_im =
generate_new_matrix(len(matrix_for_mm),len(matrix_for_mm[-1]))
        for height in range(0, len(matrix_for_mm)):
            print(f'{height}/{len(matrix_for_mm)-1}')
            for width in range(0, len(matrix_for_mm[-1])):
                counter =
super_puper_array.count(matrix_for_mm[height][width])
```

```
                counter = counter if counter > porog else 0
                counter = 255 if counter*koef > 255 else
counter*koef
                matrix_for_im[height][width].append(counter)
                matrix_for_im[height][width].append(0)
                matrix_for_im[height][width].append(0)
        im = Image.fromarray(array_to_im_array(matrix_for_im),
mode='RGB')
```

Додаток В. Програмний код main.py

```
import bot

bot.bot_start()
```

Додаток Г. Програмний код messages_text.py

```
    start_message = 'Щоб розпочати перевірку цифрового зображення на фотошоп, надішліть зображення без стиснення. Для більш швидкої перевірки обріжте область, яка підозріла 300 на 300 пікселів (як зазначено на прикладі вище).'
    getfile_message = 'Почалася перевірка!'
    final_message = 'Ось результат. Якщо на зображенні є багато червоних пікселів, фото піддавалося фотошопу. Якщо зображення повністю чорне означає що зображення оригінальне.'
```