Вінницький національний технічний університет

Факультет менеджменту та інформаційної безпеки

Кафедра менеджменту та безпеки інформаційних систем

# МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

Удосконалення методу пошуку прихованої інформації в цифрових зображеннях на основі штучної нейронної мережі

Виконала: ст. 2-го курсу, групи УБ-20м
спеціальності 125– Кібербезпека
Освітня програма – Управління інформаційною безпекою
(шифр і назва напряму підготовки, спеціальності)

Шевченко М. В.
(прізвище та ініціали)

Керівник: Голова секції УБ кафедри МБІС
д.т.н., проф.
Яремчук Ю. Є.
(прізвище та ініціали)
«____» _____ 2021 р.

Опонент: к.т.н., доц., доцент каф. ОТ
Савицька Л. А.
(прізвище та ініціали)
«____» _____ 2021 р.

**Допущено до захисту**
Голова секції УБ кафедри МБІС
д.т.н., проф. Яремчук Ю.Є.
«____» _____ 2021 р.

Вінниця ВНТУ - 2021 рік

Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки
Кафедра менеджменту та безпеки інформаційних систем

Рівень вищої освіти ІІ-й (магістерський)
Галузь знань 12 – Інформаційні технології
Спеціальність 125 – Кібербезпека
Освітньо-професійна програма - Управління інформаційною безпекою

**ЗАТВЕРДЖУЮ**
**Голова секції УБ, кафедра МБІС**

**_____д.т.н., проф. Яремчук Ю.Є.**
“ 24 ” вересня 2021 р.

**З А В Д А Н Н Я**
**на магістерську кваліфікаційну роботу студенту**
Шевченко Марії Валентинівні
(прізвище, ім`я, по-батькові)

1. Тема роботи Удосконалення методу пошуку прихованої інформації в цифрових зображеннях на основі штучної нейронної мережі

Керівник роботи гол. сек. УБ каф. МБІС д.т.н., проф. Яремчук Юрій Євгенович
(прізвище, ім`я, по-батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від 24.09.2021 р. №277.

2. Строк подання студентом роботи 14.12.2021 р.

3. Вихідні дані до роботи: монографії та наукові статті по темі, існуюче програмне забезпечення по темі.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити). Для досягнення мети необхідно: В першому розділі дослідити та проаналізувати існуючі методи стеганографії та виявлення прихованих повідомлень у рамках статистичного та нейромережевого підходів у стегоаналізі, навести їхні переваги та недоліки. В другому розділі розробити удосконалений метод пошуку прихованої інформації в цифрових зображеннях на основі штучної згорткової нейронної мережі, сформувати алгоритм роботи програмного продукту. В третьому розділі розробити програмний засіб для реалізації удосконаленого методу.

5. Перелік графічного матеріалу (з точним зазначенням обов`язкових креслень): у першому розділі наведено 8 рис. та 3 табл.; у другому розділі наведено 5 рис. та 1 табл.; у третьому розділі наведено 10 рис. та 1 табл.

6. Консультанти розділів роботи

| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата | |
|---|---|---|---|
| | | завдання видав | завдання прийняв |
| Основна частина | Голова секції УБ кафедри МБІС д.т.н., проф. Яремчук Ю.Є. | | |
| | | | |
| Економічна частина | к.т.н., доц. Адлер Оксана Олександрівна | | |
| | | | |

7. Дата видачі завдання 24 вересня 2021 р.

## 1 КАЛЕНДАРНИЙ ПЛАН

| № | Назва та зміст етапу | Термін виконання | | Примітка |
|---|---|---|---|---|
| | | початок | закінчення | |
| 1 | Аналіз предметної області обраної теми | 27.09.2021 | 10.10.2021 | |
| 2 | Розробка алгоритму роботи | 11.10.2021 | 31.10.2021 | |
| 3 | Написання магістерської кваліфікаційної роботи на основі розробленої теми | 01.11.2021 | 24.11.2021 | |
| 4 | Передзахист магістерської кваліфікаційної роботи | 25.11.2021 | 26.11.2021 | |
| 5 | Виправлення, уточнення, корегування магістерської кваліфікаційної роботи | 27.11.2021 | 16.12.2021 | |
| 6 | Захист магістерської кваліфікаційної роботи | 20.12.2021 | 20.12.2021 | |

**Студент** Шевченко М.В.
( підпис ) (прізвище та ініціали)

**Керівник роботи** Яремчук Ю.Є.
( підпис ) (прізвище та ініціали)

# АНОТАЦІЯ

У роботі досліджено та проаналізовано існуючі методи стеганографії та виявлення прихованих повідомлень у рамках статистичного та нейромережевого підходів у стегоаналізі, наведено їхні переваги та недоліки.

Детально розглянуто та описано напрямки удосконалення методу пошуку прихованої інформації в цифрових зображеннях на основі штучної нейронної мережі, сформовано алгоритм роботи програмного продукту.

Обґрунтовано вибір програмного середовища та програмно реалізовано удосконалений метод.

Результатом роботи є підтвердження актуальності, та наукової цінності подальшого дослідження та вдосконалення методу, надалі він може бути розширений додаванням функціоналу проведення атак на виділення секретного повідомлення.

Ключові слова: стеганографія, стеганоаналіз, стеганоконтейнер, цифрові зображення, глибоке навчання, штучні нейронні мережі, згорткова нейронна мережа, карти ознак.

## ABSTRACT

The existing methods of steganography and detection of hidden messages within the statistical and neural network approaches in stegoanalysis are investigated and analyzed, their advantages and disadvantages are presented.

The directions of improvement of the method of search of the hidden information in digital images on the basis of an artificial neural network are considered and described in detail, the algorithm of work of a software product is formed.

The choice of software environment is substantiated and the improved method is implemented by software.

The result of the work is the confirmation of the relevance and scientific value of further research and improvement of the method, in the future it can be expanded by adding the functionality of attacks to highlight a secret message.



Keywords: steganography, steganoanalysis, steganocontainer, digital images, deep learning, artificial neural networks, convolutional neural network, feature maps.

**ЗМІСТ**

# ВСТУП

**Актуальність теми.**

Цифрова стеганографія – наука про приховану передачу інформації, яка часто здійснюється за рахунок вбудовування повідомлення, що передається в цифровий об'єкт, який не викликає підозри, шляхом незначної його модифікації. Найбільш популярним напрямом у стеганографії є розвиток методів вбудовування повідомлень у зображення. На противагу стеганографії розвивається стегоаналіз. Основна мета стегоаналізу – виявити присутність секретних повідомлень у цифрових контейнерах, таких як цифрові зображення, що надходять з відомого джерела.

У загальному випадку існуючі методи в основному складаються з двох етапів: виділення ознак та класифікація. На етапі виділення з кожного зображення отримується набір вручну підібраних ознак, щоб зафіксувати вплив операцій застосування. Успіх стегоаналізу загалом залежить від підбору ознак. Однак відсутність точних моделей природних зображень ускладнює цю роботу. Тому пропонуються різні евристичні методи. Зі збільшенням складності методів стеганографії враховуються складніші статистичні залежності між окремими елементами. В останні роки саме напрямок статистичного стегоаналізу розвивається найінтенсивніше.

На етапі класифікації статистичні методи засновані на використанні нейронних мереж, навчаються на основі підібраних ознак. Оскільки етап вилучення ознак базується на евристичних методах стеганоаналізу, то використання нейронних мереж в якості класифікатора потребує великої кількості підготовчих дій для збору ознак на яких буде навчатися нейронна мережа. Цей факт перекриває всі позитивні характеристики, які вносить використання нейронної мережі в якості класифікатора в системах стеганоаналізу.

Вирішенням проблеми може бути використання згорткових нейронних мереж. Це клас нейронних мереж, які використовуються для класифікації та пошуку об’єктів у цифрових зображеннях. Їх можна адаптувати для вирішення

задач стегоаналізу, оскільки за рахунок їх використання можна буде реалізувати об'єднання та автоматизацію етапів вилучення ознак та класифікації.

Це зумовлює актуальність завдання удосконалення методу пошуку прихованої інформації в цифрових зображеннях за рахунок згорткової нейронної мережі.

**Метою і задачею роботи є** удосконалення методу пошуку прихованої інформації в цифрових зображеннях за рахунок штучної нейронної мережі та реалізація удосконаленого методу.

Для досягнення цієї мети були поставлені та вирішені наступні завдання:

1) Дослідити та проаналізувати існуючі методи стеганографії та виявлення прихованих повідомлень у рамках статистичного та нейромережевого підходів у стегоаналізі.

2) Розробити архітектуру згорткової нейронної мережі для потреб стегоаналізу та архітектуру програмного засобу для аналізу зображень.

3) Розробити програмний продукт для удосконаленого методу.

4) Провести аналіз ефективності удосконаленого методу порівняно з існуючими методами статистичного стегоаналізу.

**Об’єкт дослідження** – застосування глибокого навчання в стегоаналізі.

**Предмет дослідження** – застосування згорткової нейронної мережі для стегоаналізу зображень.

**Наукова новизна**. Удосконалення методу пошуку прихованої інформації в цифрових зображеннях за рахунок використання розробленої архітектури згорткової нейронної мережі.

**Практична цінність одержаних результатів**.

Розроблено програмний продукт, який реалізує удосконалений, за рахунок штучної згорткової нейронної мережі, метод статистичного стегоаналізу пошуку прихованої інформації в цифрових зображеннях.

# 1 ЗАГАЛЬНИЙ АНАЛІЗ ІСНУЮЧИХ СТЕГАНОГРАФІЧНИХ МЕТОДІВ

## 1.1 Методи приховування інформації в контейнерах типу jpeg-файли

Більшість методів приховування використовують файли з надлишковою інформацією. Це, наприклад, BMP- і wav- файли. Однак найбільше розповсюдження отримали файли типу jpeg і mp3, які реалізовані шляхом стиснення вихідних даних із втратами. Основний принцип побудови таких файлів це видалення надлишкової інформації. Цей принцип використовує особливості зорового і звукового сприйняття людини. Однак, з точки зору приховування даних в таких файлах, видалення надлишкової інформації призводить до певних проблем [1].

Розглянемо алгоритм обробки даних JPEG.

JPEG заснований на схемі кодування, що базується на дискретних косинус-перетворення (DCT). Алгоритми, що базуються на DCT, стали основою різних методів стиснення. Ці алгоритми стиснення базуються не на пошуку однакових атрибутів пікселів, а на різниці між ними. Схема JPEG ефективна тільки при стисканні багатоградаційних зображень, в яких відмінності між ближніми пікселями, як правило, досить незначні [2]. Процес стиснення за схемою JPEG включає ряд етапів (рис. 1.1):

– перетворення зображення в оптимальний колірний простір;

– субдискретизація компонентів кольоровості усередненням груп пікселів;

– застосування дискретних косинус-перетворень для зменшення надмірності даних зображення;

– квантування кожного блоку коефіцієнтів DCT із застосуванням вагових функцій, оптимізованих з урахуванням візуального сприйняття людиною;

– кодування результуючих коефіцієнтів (даних зображення) застосуванням алгоритму Хаффмана для видалення надмірності інформації.

Розглянемо коротко особливості кожного з перерахованих етапів. Декодування JPEG здійснюється в зворотному порядку.


Рисунок 1.1 – Структура JPEG-перетворень

Крок 1. Переводимо зображення з колірного простору RGB в колірний простір яскравість/кольоровість, наприклад такого, як $YUV$ або $YcbCr$. При цьому досягається більш кращий ступінь стиснення. Компонента $Y$ являє собою інтенсивність, а $U$ і $V$ - кольоровість. Тому з'являється можливість архівувати масиви $Cr$ і $Cb$ компонент з великими втратами, а отже з великими ступенями стиснення. Перетворення кольорової моделі RGB в модель $YCbCr$ здійснюється з допомогою наступних співвідношень:

$$Y = 0{,}229R + 0{,}587G + 0{,}114B, \tag{1.1}$$

$$Cb = -0{,}1687R - 0{,}3313G + 0{,}5B + 128, \tag{1.2}$$

$$Cr = 0{,}5R - 0{,}4187G - 0{,}0813B + 128. \tag{1.3}$$

Крок 2. Розбиваємо вихідне зображення на матриці 8x8. Формуємо з кожної 3 робочі матриці ДКП – по 8 біт окремо для кожної компоненти. При великих ступенях стиснення цей крок може виконуватися трохи складніше. Зображення ділитися по компоненту $Y$, як і в першому випадку, а для компонент $Cr$ і $Cb$ матриці набираються через рядок і через стовпець. Тобто з вихідної матриці розміром 16x16 виходить тільки одна робоча матриця ДКП. При цьому, як неважко помітити, ми втрачаємо ¾ корисної інформації про колірні складові зображення і отримуємо одразу стиснення в 2 рази.

Крок 3. Формули прямого і зворотного дискретного косинусного перетворення представлені нижче. Дискретне косинусне перетворення перетворює матрицю пікселів розміром NxN в матрицю частотних коефіцієнтів відповідного розміру. Незважаючи на видиму складність, закодувати ці формули досить просто.

$$\text{ДКП}(i,j) = \\ = \frac{1}{\sqrt{2N}} C(i)C(j) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} pixel(x,y) cos\left[\frac{(2x+1)i\pi}{2N}\right] cos\left[\frac{(2y+1)j\pi}{2N}\right], \tag{1.4}$$

$$C(x) = \begin{cases} \frac{1}{\sqrt{2}}, x = 0 \\ 1, x > 0 \end{cases}. \tag{1.5}$$

Застосовуємо ДКП до кожної робочої матриці. При цьому ми отримуємо матрицю, в якої коефіцієнти в лівому верхньому кутку відповідають НЧ складової зображення, а в правому нижньому – ВЧ. Поняття частоти випливає з розгляду зображення як двовимірного сигналу. Плавна зміна кольору відповідає НЧ складової, а різкі скачки – ВЧ.

Шаг 4. Виробляємо квантування. В принципі це просто розподіл робочої матриці на матрицю квантування поелементно. Матриця для $Y$ компонент:

$$\begin{matrix} 3 & 5 & 7 & 9 & 11 & 13 & 15 & 17 \\ 5 & 7 & 11 & 13 & 15 & 17 & 19 & 21 \\ 7 & 11 & 13 & 15 & 17 & 19 & 21 & 23 \\ 9 & 11 & 13 & 15 & 17 & 19 & 21 & 23 \\ 11 & 13 & 15 & 17 & 19 & 21 & 23 & 25 \\ 13 & 15 & 17 & 19 & 21 & 23 & 25 & 27 \\ 15 & 17 & 19 & 21 & 23 & 25 & 27 & 29 \\ 17 & 19 & 21 & 23 & 25 & 27 & 29 & 31 \end{matrix} \tag{1.6}$$

Крок 5. Заключна стадія роботи кодера JPEG – це власне кодування. Воно включає три дії над округленою матрицею дискретного косинусного перетворення, для того, щоб підвищити ступінь стиснення. Перша дія – це заміна абсолютного значення коефіцієнта, розташованого в осередку (0,0) матриці, на відносне. Так як сусідні блоки зображення в значній мірі "схожі" один на одного,

то кодування чергового (0,0) елемента як різниці з попереднім дає менше значення. Коефіцієнти матриці дискретного косинусного перетворення обходяться зиґзаґом. Після чого нульові значення кодуються з використанням алгоритму кодування повторів (RLE), а потім результат обробляється за допомогою "кодування ентропії", тобто алгоритмів Хаффмана або арифметичного кодування, в залежності від реалізації [3].

"Кодування ентропії". Результатом роботи, спрощеної схеми кодування, є трійки такого вигляду: $< \text{Кількість нулів}, \text{Кількість Бітів}, \text{Коефіцієнт} >$

Тут:

Кількість нулів – кількість повторюваних нулів, що передують поточному (ненульовому) елементу матриці дискретного косинусного перетворення;

Кількість Бітів – кількість бітів, які слідують далі, що кодують значення коефіцієнта;

Коефіцієнт – значення ненульового елемента матриці дискретного косинусного перетворення.

Співвідношення між полями Кількість Бітів та Коефіцієнт наведено в таблиці 1.1.

Таблиця 1.1 – Співвідношення між Кількістю Бітів та Коефіцієнтами

| Кількість бітів | Коефіцієнт |
|---|---|
| 1 | [-1, 1] |
| 2 | [-3, -2], [2, 3] |
| 3 | [-7, -4], [4, 7] |
| 4 | [-15, -8], [8, 15] |
| 5 | [-31, -16], [16, 31] |
| 6 | [-63, -32], [32, 63] |
| 7 | [-127, -64], [64, 127] |
| 8 | [-255, -128], [128, 255] |
| 9 | [-511, -256], [256, 511] |
| 10 | [-1023, -512], [512, 1023] |

Таке кодування не настільки ефективно, як кодування Хаффмана, але на певні дані воно дає аналогічні результати. Після завершення цього етапу потік

даних JPEG готовий до передачі по комунікаційними каналам або інкапсуляціям в формат файлу зображення [4].

Розглянемо структуру JPEG.

Файл JPEG складається з маркерів, що зустрічаються при декодуванні файлів формату JPEG, які представлені в таблиці 1.2.

Таблиця 1.2 – Маркери JPEG-файла

| Тип маркера | Ідентифікатор | Значення стандарту | Визначення |
|---|---|---|---|
| SOF0 | C016 | Baseline DCT | Початок кадру, базовий метод |
| SOF1 | C016 | Extended sequential DCT | Початок кадру, розширений, послідовний метод |
| SOF2 | C216 | Progressive DCT | Початок кадру, прогресивний метод |
| DHT | C416 | Define Huffman table(s) | Визначення таблиць Хафмана |
| SOI | D816 | Start of image | Початок зображення |
| EOI | D916 | End of image | Кінець зображення |
| SOS | DA16 | Start of scan | Початок скану |
| DQT | DB16 | Define quantization table(s) | Визначення таблиць квантування |

У наведеній таблиці показані ті маркери, які є обов'язковими в будь-якому файлі формату JPEG і вимагають обов'язкової обробки.

Структура типового маркера: | ідентифікатор | довжина | дані маркера |

Ідентифікатор – два байта, які вказують на тип маркера.

Довжина – два байта, які містять довжину даних маркера в байтах (зворотний порядок) і власну довжину. Тобто якщо дані маркера мають довжину 10 байт, то значення довжини дорівнюватиме 10 (довжина даних маркера) + 2 (два байта зі значення довжини) = 12.

Дані маркера – набір байт, що вимагають обробки відповідно до типу маркера [5].

Стеганографічні алгоритми приховування інформації в частотній області.

Реальні зображення зовсім не є випадковим процесом з рівномірно розподіленими значеннями величин. Добре відомо, і це використовується в алгоритмах стиснення, що велика частина енергії зображень зосереджена в низькочастотній частині спектра. Звідси і потреба в здійсненні декомпозиції зображення на субсмуги. Стегоповідомлення додається до субсмуг зображення. Низькочастотні субсмуги містять переважну частину енергії зображення і, отже, носять шумовий характер. Високочастотні субсмуги найбільш схильні до впливу з боку різних алгоритмів обробки. Таким чином, для вкладення повідомлення найбільш підходящими кандидатами є середньочастотні субсмуги спектра зображення [6]. Типовий розподіл шуму та обробки по спектру частоти показано на рисунку 1.2.

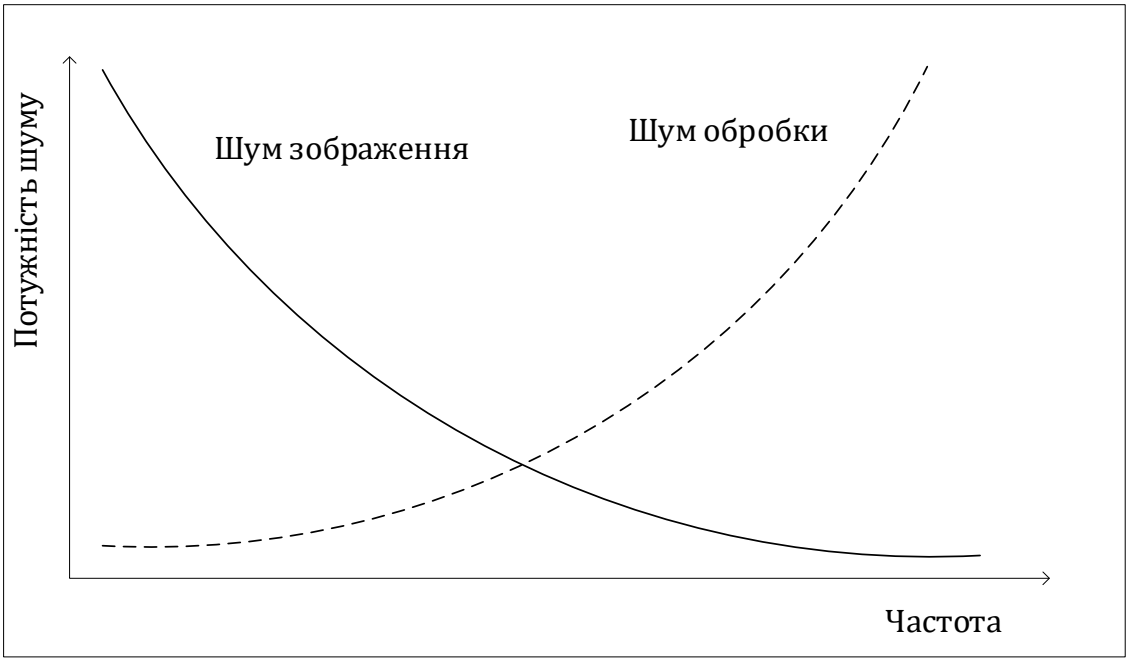



Рисунок 1.2 – Залежність шуму зображення та шуму обробки від частоти

Шум обробки з'являється в результаті квантування коефіцієнтів трансформанти. Його можна розглядати як зменшення кореляції між коефіцієнтами трансформанти вихідного зображення і квантовими коефіцієнтами. Наприклад, при високих ступенях стиснення може виникнути ситуація, коли будуть відкинуті цілі субсмуги. Тобто дисперсія шуму в цих субсмугах, нескінченна. У наявності зменшення кореляції між коефіцієнтами

субсмуги до квантування і після. Для отримання прийнятних результатів необхідно усереднити значення шуму обробки за багатьма зображеннями.

Перетворення можна впорядкувати по досягненим виграшами від кодування: одиночне, Адамара, Хаара, ДКП, Вейвлет, Карунена-Лоева (ПКЛ). Під виграшом від кодування розуміється ступінь перерозподілу дисперсій коефіцієнтів перетворення.

Найбільший виграш дає перетворення Карунена-Лоева (ПКЛ), найменший – розклад по базису одиничного імпульсу (тобто відсутність перетворення). Перетворення, що мають високі значення виграшу від кодування, такі як ДКП, вейвлет-перетворення, характеризуються різко нерівномірним розподілом дисперсій коефіцієнтів субсмуг. Високочастотні субсмуги не підходять для вкладення через великий шум обробки, а низькочастотні – через високий шум зображення. Тому доводиться обмежуватися середньочастотними смугами, в яких шум зображення приблизно дорівнює шуму обробки. Так, як таких смуг небагато, то пропускна здатність стегоканала невелика.

У разі застосування перетворення з більш низьким виграшом від кодування, наприклад, Адамара або Фур'є, є більше блоків, в котрих шум зображення приблизно дорівнює шуму обробки. Тому і пропускна здатність вище [7].

Отже, для підвищення пропускної здатності стеганографічного каналу краще застосовувати перетворення з меншими виграшами від кодування, які погано підходять для стиснення сигналів.

Ефективність застосування вейвлет-перетворення і ДКП для стиснення зображень пояснюється тим, що вони добре моделюють процес обробки зображення в СЧЗ, відокремлюють «значущі» деталі від «незначущих». Отже, їх доцільніше застосовувати в разі активного порушника, так як модифікація значущих коефіцієнтів може призвести до неприйнятного спотворення зображення. При застосуванні перетворення з низькими значеннями виграшу від кодування існує небезпека порушення вкладення, так як коефіцієнти перетворення менш чутливі до модифікацій.

Алгоритм Коча (Koch)

В даному алгоритмі в блок розміром 8x8 здійснюється вбудовування 1 біта ЦВЗ. Описано дві реалізації алгоритму: псевдовипадково можуть вибиратися два або три коефіцієнта ДКП [8]. Розглянемо варіацію алгоритму з двома вибраними коефіцієнтами. Вбудовування інформації здійснюється наступним чином: для передачі біта 0 домагаються того, щоб різниця абсолютних значень коефіцієнтів була б більше деякої позитивної величини, а для передачі біта 1 ця різниця робиться менше деякої негативної величини:

$$\left|c_b(j_{i,j}, k_{i,1})\right| - \left|c_b\left(j_{i,2}, k_{i,2}\right)\right| > s, \text{якщо } \varepsilon_i = 0, \tag{1.7}$$

$$\left|c_b(j_{i,j}, k_{i,1})\right| - \left|c_b\left(j_{i,2}, k_{i,2}\right)\right| > s, \text{якщо } \varepsilon_i = 0, \tag{1.8}$$

$$\left|c_b(j_{i,j}, k_{i,1})\right| - \left|c_b\left(j_{i,2}, k_{i,2}\right)\right| < -s, \text{якщо } \varepsilon_i = 1. \tag{1.9}$$

Алгоритм Кокса (Cox)

Цей алгоритм є робастним до багатьох операцій обробки сигналу. Виявлення вбудованого ЦВЗ в ньому виконується з використанням вихідного зображення. Впроваджувані дані представляють собою послідовність дійсних чисел з нульовим середнім і одиничною дисперсією. Для вкладення інформації використовуються кілька AC-коефіцієнтів ДКП всього зображення з найбільшою енергією. Є три способи вбудовування ЦВЗ в залежності з наступними виразами:

$$c_i' = c_i + as_i; c_i' = c_i(1 + as_i); c_i' = c_i e^{as_i} \tag{1.10}$$

Перший варіант може використовуватися в разі, коли енергія ЦВЗ порівнянна з енергією модифікованого коефіцієнта. В іншому випадку або ЦВЗ буде непрацездатним, або спотворення занадто великими. Тому так вбудовувати інформацію можна лише при незначному діапазоні зміни значень енергії коефіцієнтів.

При виявленні ЦВЗ виконуються зворотні операції: обчислюються ДКП вихідного і модифікованого зображень, знаходяться різниці між відповідними коефіцієнтами найбільшої величини [9].

Алгоритм Барні (Barni)

Цей алгоритм є удосконаленням алгоритму Кокса, і в ньому також виконується ДКП всього зображення. У ньому детектору вже не потрібно вихідного зображення, тобто схема сліпа. Для вбудовування ЦВЗ використовуються не найбільші АС-коефіцієнти, а середні за величиною. Як ЦВЗ виступає довільний рядок біт. Вибрані коефіцієнти модифікуються таким чином:

$$c_i' = c_i + as_i|c_i| \tag{1.11}$$

Далі виконується зворотне ДКП, і проводиться додатковий крок обробки: поточне й модифіковане зображення складаються з ваговими коефіцієнтами:

$$l''(x,y) = \beta(x,y)l'(x,y) + (1-\beta)l(x,y) \tag{1.12}$$

Тут $\beta \approx 1$ для текстурованих областей (в яких людське око мало чутливе до доданого шуму) і $\beta \approx 0$ в однорідних областях. Значення знаходиться не для кожного пікселя окремо, а для блоків фіксованого розміру, що не перекриваються.Наприклад, в якості доцільно використовувати нормалізовану дисперсію блоків. У детекторі ЦВЗ обчислюється кореляція між модифікованим зображенням і ЦВЗ [10].

$$\sum_{i=1}^{n} c_i^n s_i^n \tag{1.13}$$

Порівняння стійкості стеганографічних алгоритмів.

Під стійкістю стеганографічних алгоритмів розуміється ймовірність успішного відновлення прихованого повідомлення після впливу атак на контейнер. На основі аналізу розглянутих методів були отримані результати, які наведені в таблиці 1.3.

Таблиця 1.3 – Стійкість алгоритмів до різних впливів

| Алгоритми | Однозначність відновлення | Стійкість до фільтрації | Стійкість до геометричних перетворень | Стійкість до стискання | Стійкість до засобів статистичного аналізу |
|---|---|---|---|---|---|
| Коч | + | - | - | - | - |
| Кокс | + | - | + | + | - |
| Барні | - | + | - | + | + |

Розглянуті основні алгоритми приховування інформації, що використовують перетворення ДКП, не задовольняють всім основним критеріям стегосистем. Алгоритми Коча, Кокса і Барні порівнювалися за наступними критеріями: однозначність відновлення, стійкість до фільтрації, стійкість до геометричним перетворенням, стійкість до стиснення і стійкість до засобів статичного аналізу. Алгоритм Коча є стійким тільки до однозначного відновлення. Алгоритм Кокса стійкий до однозначного відновлення, геометричних перетворень і до стиснення. Алгоритм Барніустойчів до фільтрації, стиску і засобів статичного аналізу.

## 1.2 Огляд методів стеганоаналізу для графічних файлів

Безпека стеганосистем, які використовують той чи інший метод стеганографічного приховування, описується та оцінюється їх стійкістю. Оцінити стійкість стеганосистеми в теоретико-інформаційному сенсі [11] практично неможливо, тому вводиться поняття стеганографічної стійкості у практичному значенні.

Стеганографічна система називається стійкою в практичному сенсі, якщо немає стеганоаналітичного алгоритму, який міг би виявити наявність прихованої інформації [12].

Проведений аналіз існуючих методів стеганоаналізу показав, що в залежності від вихідних даних, що використовуються, їх можна розділити на дві основні групи:

1) Методи, призначені до роботи з конкретними заздалегідь відомими стеганографічними алгоритмами.

2) Методи, призначені для будь-яких алгоритмів стеганографії. Стеганоаналіз цими методами не вимагає знання використаного алгоритму стеганографії, алгоритму шифрування, стиснення, ключа і довжини повідомлення. Відомі методи цієї групи зазвичай побудовані на алгоритмах, які потребують попереднього навчання на серіях із заповнених та порожніх контейнерів.

Методи обох груп побудовані з урахуванням припущення про недоступність вихідного порожнього контейнера, використаного для впровадження інформації в досліджуваний стеганоконтейнер.

До першої групи відносять:

1) Сигнатурні методи;

2) Схемні методи аналізу.

Суть сигнатурних методів полягає в синтаксичному аналізі пред'явленої на вхід розпізнавального пристрою, послідовності термінальних символів, які визначають контейнер. У разі виявлення приналежності пред'явленої на вхід розпізнавального пристрою ланцюжка термінальних символів мові, що описує ту чи іншу стеганосистему, приймається рішення про її використання для приховування інформації. Як термінальні символи зазвичай беруть усі або частину стандартних символів ASCII – латинські літери, цифри та спеціальні символи.

До переваг цих методів відноситься можливість отримання результату, який однозначно характеризує застосовану для приховування даних стеганосистему.

Основним недоліком є невелике (менше 10%) число стеганопрограм, що залишають у контейнерах свої сигнатури [13].

Схемні методи застосовуються для перевірки гіпотез про наявність стеганографічного вкладення з апріорно відомою стеганосистемою.

Перевагою методів цього класу є відносно низька ймовірність виникнення помилок, а також той факт, що за позитивним результатом аналізу аналітик ідентифікує стеганосистему, що не залишає «слідів» (сигнатур) у контейнері, що дозволяє спробувати витягти приховану інформацію.

До другої групи входять наступні методи:

1) Візуальні методи:

– метод візуального аналізу;

– метод візуального аналізу бітових зрізів.

2) Статистичні методи:

– метод оцінки числа переходів значень молодших біт у сусідніх елементах зображення;

– метод оцінки частот появи $k$-бітових серій у потоці НЗБ елементів контейнера;

– метод аналізу розподілу пар значень з урахуванням критерію $\chi^2$;

– метод аналізу гістограм, побудованих за частотами елементів зображення;

– метод аналізу розподілу елементів зображення на площині;

– метод перевірки розподілу елементів на монотонність.

Візуальні методи базуються на здібності зорової системи людини аналізувати зорові образи та виявляти суттєві відмінності в зображеннях, які порівнюються.

Метод візуального аналізу є найпростішим способом аналізу графічних файлів, оскільки для цього досить просто подивитися на перехоплене зображення. Тим не менш, цей метод аналізу вже здатний встановити деякі обмеження на обсяг прихованих даних. Так, для повнокольорових реалістичних зображень у форматі BMP непомітними для ока будуть спотворення менше 3% [14], [15]. У випадку з JPEG візуально визначити наявність прихованої інформації неможливо. Якщо в результаті приховування у зображенні виникають незначні спотворення, їх можна пояснити застосуванням процедури стиснення.

Метод візуального аналізу бітових зрізів. Основна ідея методу полягає в порівнянні зображення загалом із зображеннями його бітових зрізів [16]. За допомогою програми зображення переглядають за шарами – бітовим зрізам.

Враховуючи те, що інтенсивність кожного кольору визначається рівно одним байтом, всього необхідно переглянути 8 таких зрізів. Для кожного із трьох кольорів перший зріз – це зображення, побудоване наймолодшими бітами, другий зріз – зображення, побудоване другим бітом і т. д. Отримане зображення бітового зрізу переглядають і порівнюють візуально з аналізованим зображенням.

Для методу візуального аналізу бітових зрізів велике значення має те, як саме здійснюється запис інформації, що приховується. Якщо вона записується в поспіль розміщені біти або рівномірно розподіляє біти повідомлення (на основі генератора псевдовипадкових чисел) по всьому зображенню, то факт прихування може бути встановлений з великою ймовірністю. Також візуально можна визначити наявність вбудованої інформації у разі запису повідомлення із заповненням. Оскільки імовірнісні характеристики повідомлення не збігаються з імовірнісними характеристиками молодших біт порожнього контейнера, то при перегляді бітового зрізу з вбудованими даними буде чітко видно межу між заповненою і не заповненою частиною. Щоб імовірнісні характеристики збігалися, під час запису інформації із заповненням, повідомлення необхідно зашифровувати [17].

У випадку з JPEG-файлами цей метод аналізу малопридатний, оскільки зміна будь-якого коефіцієнта перетворення призводить до зміни множини пікселів зображення. Впровадження повідомлення у молодші біти дискретних косинусних коефіцієнтів трохи змінить кожен із 256 пікселів, що візуально непомітно [18].

Статистичні методи базуються на понятті "природного" контейнера. Суть методів полягає в оцінюванні ймовірності існування стеганографічного вкладення з невідомою стеганосистемою на основі критерію оцінки близькості досліджуваного контейнера до "природного".

До переваг цієї групи методів відноситься необмежена сфера застосування, що досить суттєво як при перевірці гіпотези про наявність стеганографічного вкладення з невідомою стеганосистемою, так і при розробці схемних методів стеганоаналізу. Основним недоліком методів цього класу є саме припущення існування «природного» контейнера. Розглянемо низку статистичних методів, що застосовуються на практиці.

Метод оцінки числа переходів значень молодших біт у сусідніх елементах зображення. У методі використовується знання, що між молодшими сусідніми бітами елементів і між ними та іншими бітами природних контейнерів є

кореляційні зв'язки. При дослідженні файлів формату JPEG, як елементи аналізованої послідовності вибираються молодші біти сусідніх дискретних косинусних коефіцієнтів, відмінних від 0 та 1.

Залежність між бітами у відповідних розрядах елементів контейнера має марківський характер [19]. При цьому параметри залежності визначаються номером розряду. Під «переходом» розуміють перехід значення $i$-го елемента послідовності значення $i+1$ елемента послідовності $x$, $i = 1,2,\ldots,n-1$, $n$ – довжина послідовності. Так як послідовності є двійковими, то аналізується чотири види переходів: з 0 в 0, з 0 в 1, з 1 в 0 і з 1 в 1. За отриманими результатам будується гістограма. Для кожного розряду перший стовпець гістограми показує число переходів у потоці НЗБ з 0 до 0, другий стовпець – з 0 до 1, третій стовпець – з 1 до 0, четвертий стовпець – з 1 до 1.

Для порожнього контейнера та контейнера, що містить вбудовану інформацію, число переходів у потоці НЗБ буде різним. Розподіл НЗБ стеганоконтейнера має, зазвичай, випадковий характер. Відповідно кількість переходів у потоці НЗБ для всіх станів буде приблизно однаковим, що не властиво порожньому контейнеру (рис. 1.3 б) [20], [21].

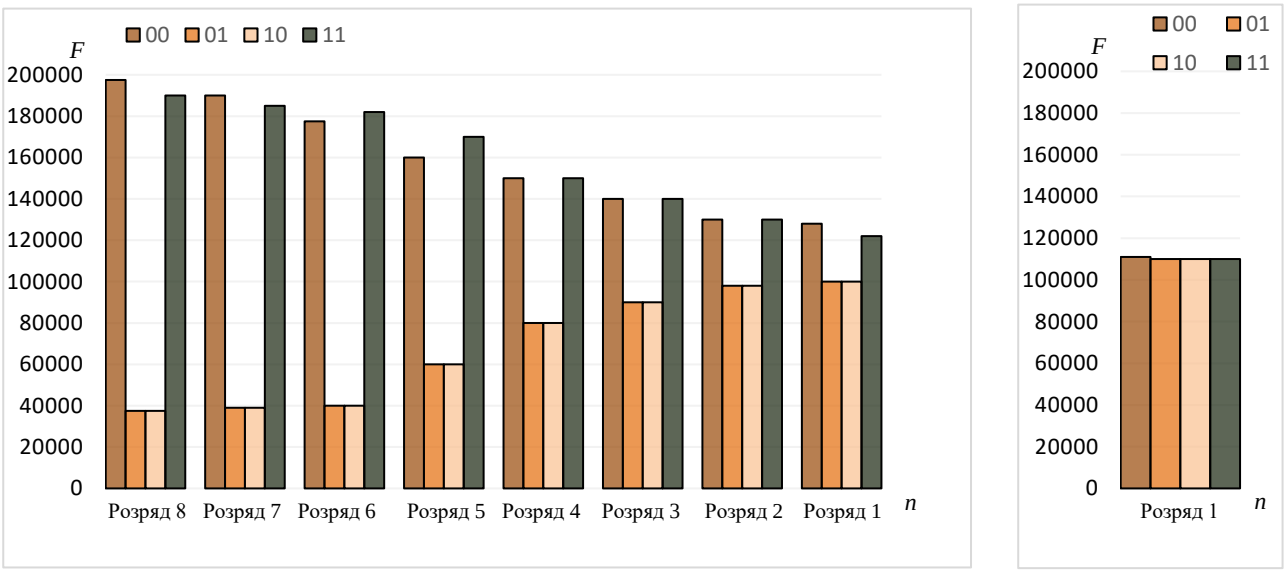



Рисунок 1.3 – Гістограма частот переходів бітових значень: а – порожнього контейнера, б – стеганоконтейнера

Метод оцінки частот появи $k$-бітових серій у потоці НЗБ елементів контейнера. Метод дозволяє оцінити рівномірність розподілу елементів у досліджуваній послідовності на основі аналізу частоти появи нулів та одиниць, і серій, що складаються з $k$ біт [22]. У бітовому поданні досліджуваної послідовності $x$ підраховується, скільки разів зустрічаються нулі та одиниці ($k = 1$), серії-двійки (00, 01, 10, 11: $k = 2$), серії-трійки (000, 001, 010, 011, 100, 101, 110,111: $k = 3$) і т.д. За підсумками результатів будується гістограма.

Для JPEG-зображень гістограма будується за значеннями частот появи бітових серій у потоці НЗБ дискретних косинусних коефіцієнтів, відмінних від –1, 0, 1.

Для незаповнених JPEG зображень не є характерним, щоб значення частот всіх компонентів були досить близько (рис. 1.4 а). У разі впровадження інформації значення частот зближуються (рис. 1.4 б). Цей факт використовується під час аналізу.

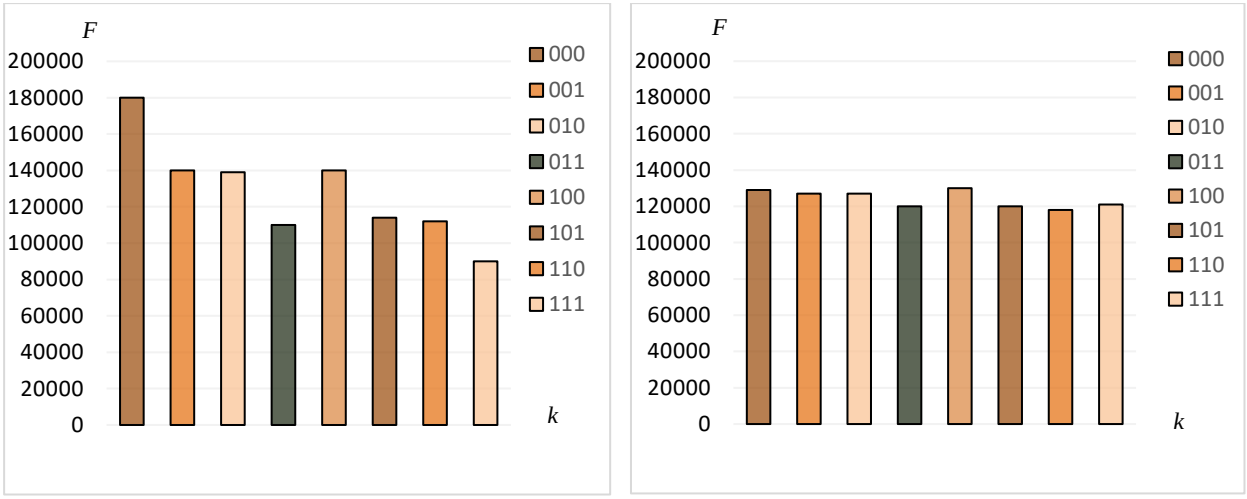



Рисунок 1.4 – Гістограма частот серії-трійки ($k = 3$) у потоці НЗБ: а – порожнього контейнера, б – стеганоконтейнера

Результати роботи методу залежать від стеганографічного перетворення, що використовується для вбудовування даних, що приховуються, а також від їх обсягу. Як правило, виявлення факту приховання можна здійснити при заповненні контейнера на 60% і вище.

Метод аналізу розподілу пар значень з урахуванням критерію $\chi^2$. У методі використовується аналіз гістограми, отриманої за елементами зображення та

оцінка розподілу пар значень цієї гістограми [23]. Для файлів JPEG пари значень квантуються коефіцієнтами дискретного косинусного перетворення, які відрізняються за молодшим бітом. Молодші біти зображень є випадковими. Частоти двох сусідніх елементів контейнера повинні бути досить далекі від значення частоти середнього арифметичного цих елементів. У «порожньому» зображенні ситуація, коли частоти елементів зі значеннями $2N$ та $2N + 1$ близькі за значенням, зустрічається досить рідко. При вбудовуванні інформації ці частоти зближуються або стають рівними. Ідея атаки хі-квадрат полягає у пошуку цих близьких значень і підрахунку ймовірності вбудовування на основі того, як близько розташовуються значення частот парних та непарних елементів аналізованого контейнера. Особливістю алгоритму є послідовний аналіз всього зображення і, відповідно, накопичення частот елементів.

Метод хі-квадрат є універсальним, оскільки підходить для аналізу зображень, створених різними програмами приховування. Проте результати роботи методу за критерієм хі-квадрат значною мірою залежить від способу приховування даних. При послідовному записі в НЗБ елементів контейнера метод забезпечує хороші результати (рис. 1.5), а при псевдовипадковому виборі молодших біт та розсіювання повідомлення по всій довжині контейнера метод не спрацьовує.

У роботі [24] автор запропонував досконалий "блоковий" варіант даного методу. Від класичного методу він відрізняється тим, що аналізоване зображення розбивається на блоки певного розміру, які можуть перетинатися, так і не перетинатися, і для кожного блоку розраховуються свої набори частот елементів і свої ймовірності приховування. Крім того, існує можливість вибору окремих областей зображення для їх подальшого аналізу. Такий підхід дозволяє виявляти наявність інформації, прихованої псевдовипадковим чином.

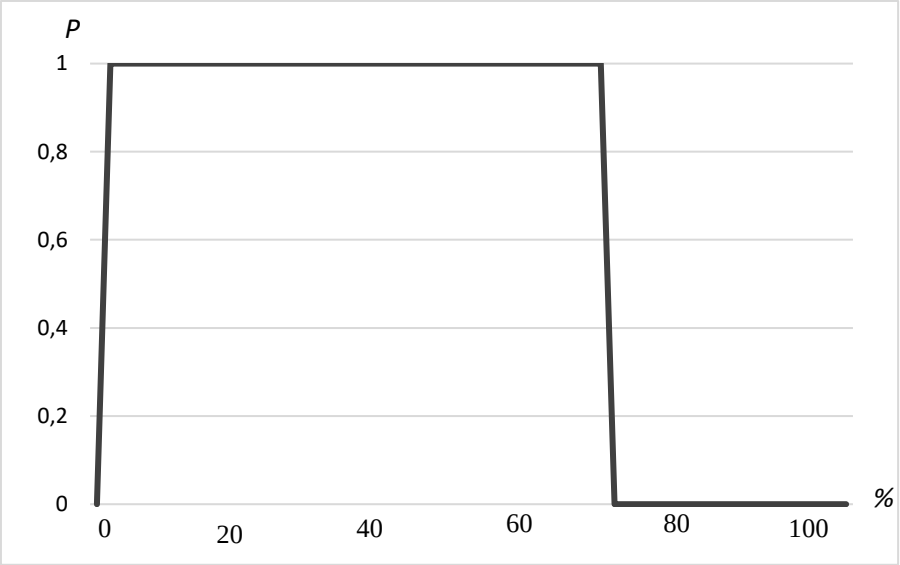



Рисунок 1.5 – Імовірність вбудовування за критерієм 2 при аналізі стеганоконтейнера, отриманого методом послідовної заміни

Метод аналізу гістограм, побудованих за частотами елементів зображення. Метод дозволяє оцінити рівномірність розподілу елементів аналізованого зображення, і навіть визначити частоту появи конкретного елемента.

Для зображень у форматі JPEG будується гістограма частот квантованих дискретних косинусних коефіцієнтів. Експериментально виявлено, що огинаюча гістограма порожнього зображення має більш гладкий характер (рис. 1.6 а) порівняно з гістограмами зображень, що містять стеганографічне вкладення (рис. 1.6 б). Звичайно, залежно від характеру та ступеня стиснення зображення, гістограми можуть змінюватися – у них можуть з'являтися стрибки та провали, але важливо те, що приховування інформації змінює загальний вигляд гістограм. Більшість стеганографічних програм, що працюють з JPEG, приховують дані в молодші біти дискретних коефіцієнтів, відмінних від 0 і 1. Як наслідок, частоти 0-х та 1-х DCT не змінюються, тоді як інші частоти або зменшуються, або збільшуються залежно від алгоритму вбудовування. При значних обсягах прихованої інформації гістограми часто набувають ступінчастого характеру, що нетипово для звичайних JPEG-зображень [25].

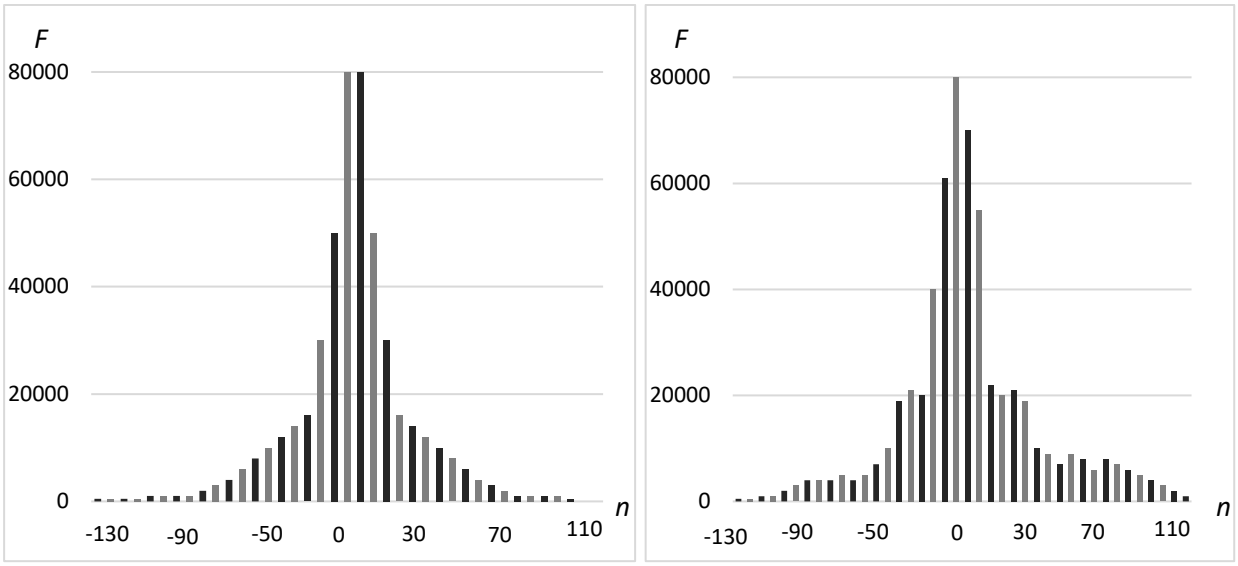



Рисунок 1.6 – Гістограма частот дискретних косинусних коефіцієнтів:

а – вихідного зображення, б – зображення, що містить приховану інформацію

Метод аналізу розподілу елементів зображення на площині. Метод призначений визначення залежностей між елементами досліджуваної послідовності.

На площину (поле) розміром $(2^R - 1) \times (2^R - 1)$, де $R$ – розрядність елемента послідовності, наносяться точки з координатами $(x_i, x_{i-1})$, $x_i$ – елементи досліджуваної послідовності $x$, $i = 1,2,\ldots,n-1$, $n$ – довжина послідовності. За отриманими даними проводиться аналіз.

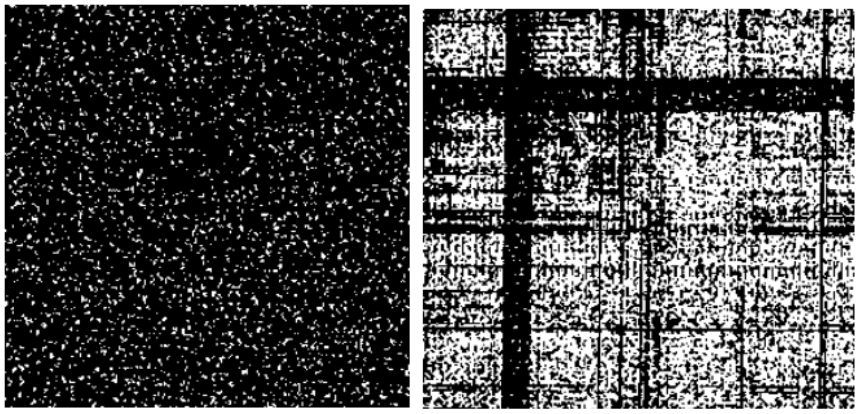


Рисунок 1.7 – Розподіл на площині елементів зображення, що містить приховану інформацію (праворуч) та вихідного зображення (ліворуч)

Якщо точки по всьому полю розташовані хаотично, то між елементами послідовності відсутні залежності, що характерно для контейнерів з

вбудованими даними (рис. 1.7 а). У разі незаповненого контейнера крапки на полі будуть розташовані нерівномірно чи утворювати «візерунки» (рис. 1.7 б) [15].

Метод перевірки розподілу елементів на монотонність.

Метод дозволяє оцінити рівномірність розподілу елементів зображення за результатами аналізу довжин ділянок незростання та неспадання елементів послідовності. Досліджувана послідовність $x$ графічно представляється у вигляді ділянок незростання, що не перетинаються та неспадаючих елементів послідовності, що слідують один за одним (рис. 1.8).

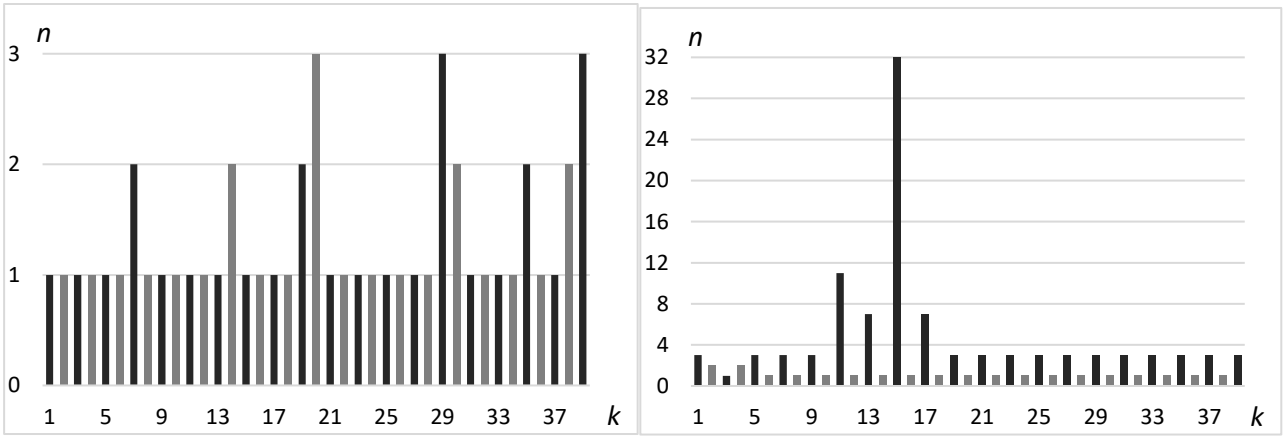



Рисунок 1.8 – Перевірка на монотонність: а – зображення, що містить приховану інформацію, б – вихідного зображення

Так як статистичні властивості стеганоконтейнера близькі до властивостей випадкової послідовності, то ймовірність появи ділянки незростання (неспадання) буде тим меншою, чим більша його довжина $n$ [15].

Статистичні методи не є засобом, що дозволяють зі 100% надійністю визначати наявність прихованої інформації. Вони дають можливість аналітику з певною ймовірністю судити про те, чи використовується стеганографія чи ні.

Крім наведених вище, існує ще низка методів, що ґрунтуються на різних математичних моделях процесу стеганографії [18], [19]. Аналіз показав, що в даний час існує певне відставання у розвитку методів стеганоаналізу від методів стеганоперетворення.

Таким чином, практична стійкість стеганосистем безпосередньо залежить від розвитку стеганоаналітичних методів.

### 1.3 Використання машинного навчання для стеганоаналізу

Як відповідь на появу алгоритмів Outguess і F5, так як ті ґрунтувалися на мінімізації внесеного використанням спотворення, 2002 року Сьюви Лью та Хані Фарід представили новий напрямок у стегоаналізу – спосіб машинного навчання [26]. Їх підхід полягає в побудові складних статистичних моделей для природних зображень та у знаходженні стійких закономірностей.

Машинне навчання – великий підрозділ штучного інтелекту, що вивчає методи побудови алгоритмів, здатних навчатися. Загальна постановка завдання звучить наступним чином. Є безліч об'єктів (ситуацій) і безліч можливих відповідей (відгуків, реакцій). Між об'єктами та відповідями існує залежність, яку необхідно виявити. Відома сукупність «прецедентів» – пара виду «об'єкт, відповідь», звана навчальною вибіркою. Ці дані використовуються для побудови алгоритму, здатного для будь-якого об'єкта видати досить точну відповідь.

Лью та його колега для класифікації використовували метод опорних векторів. Суть методу полягає в тому, що кожен об'єкт представляє собою радіус-вектор (точку) у багатовимірному лінійному просторі. Кожен об'єкт належить одному із двох класів. У процесі навчання обчислюється гіперплощина, яка поділяє об'єкти різних класів. Як подання зображення у лінійному просторі використовується вектор ознак, який обчислюється на підставі внутрішніх закономірностей. Наприклад, Фарід запропонував використовувати статистики розподілу груп пікселів, такі як математичне очікування, дисперсія, середньоквадратичне відхилення тощо. Пікселі групувалися по вертикалі, горизонталі та діагоналі. Також, пікселі піддавалися різним фільтрам та перетворенням. В результаті, було отримано 72-х розмірний вектор ознак [27].

Існує три різновиди методу опорних векторів: випадок лінійної роздільності, випадок нелінійної роздільності та метод із заміною ядра, які відрізняються ресурсоємністю навчання та точністю класифікації.

Нехай пара $(x_i, y_i), i = 1, \dots, N$ – об'єкт з навчальної вибірки, де $x_i$ – вектор ознак, $y_i$ приймає значення 1 для порожніх контейнерів та -1 для стего. У

разі лінійної роздільності будується гіперплощина, яка розмежовує позитивні та негативні екземпляри. Крапки, лежать на гіперплощині, задовольняють виразу:

$$w^t x_i + b = 0, \tag{1.14}$$

де $w$ – нормаль гіперплощини, $b/\|w\|$– відстань від початку координат до гіперплощини.

З безлічі допустимих гіперплощин вибирається та, у якої зазор між позитивними та негативними екземплярами максимальний. Якщо гіперплощина існує, тоді виконується умова:

$$w^t x_i + b \geq 1 \text{ при } y_i = 1 \text{ і } w^t x_i + b \text{ при } y_i = -1 \tag{1.15}$$

При цьому розмір зазору визначається як $2/\|w\|$. Таким чином, завдання зводиться до мінімізації $\|w\|^2$. Завдання оптимізації вирішується через лагранжіан:

$$L(w, b, a) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N} a_i (w^t x_i + b) y_i + \sum_{i=1}^{N} a_i \tag{1.16}$$

Отримані в результаті значення $w$ та $b$ можуть бути використані для класифікації нового екземпляра $z$, достатньо визначити в якому напівпросторі відносно гіперплощини лежить відповідний вектор ознаки. Якщо $w^t x_i + b \geq 0$, то екземпляр класифікується як стего. Інакше – як порожній контейнер.

У деяких випадках не існує гіперплощини, здатної повністю поділити екземпляри різних класів (рисунок 1.7, б). Деякі екземпляри можуть «лежати на неправильній стороні». У цьому випадку вводять м'які межі: $w^t x_i + b \geq 1 - \varepsilon_i$ при $y_i = 1$ і $w^t x_i + b \geq -1 + \varepsilon_i$ при $y_i = -1$. При вирішенні додається умова мінімізації сумарної помилки навчання, $\sum_{i=1} \varepsilon_i \rightarrow min$.

У деяких випадках ефективно розділити екземпляри нелінійною гіперплощиною. У цьому випадку застосовується універсальний метод опорних векторів із заміною ядра. Тут замість скалярного піднесення в степінь векторів використовується нелінійна функція ядра. В якості ядра прийнято вибирати радіальну базову функцію: $(x, y) = e^{-\gamma\|x-y\|}$, $\gamma > 0$.

Таким чином, вводиться додатковий оптимізаційний параметр, який знаходиться перебором на заданому відрізку. Використання заміни ядра

призводить до кращої ефективності поділу класів, але потребує великих ресурсомістких обчислень.

Ефективність класифікації залежить від вибору простору ознак. Дослідники запропонували кілька різних просторів ознак високої розмірності, наприклад, SPAM (розмірність 686) [29], калібрований простір ознак Певних (розмірність 548) [30].

Завдяки принципу відкритості за допомогою методу машинного навчання вдалося побудувати ефективні класифікатори для багатьох найпоширеніших стеганографічних систем: YASS [31], PQ [32], MOX [33] та інших. Тим самим було показано універсальність методу машинного навчання у стегоаналізу.

### 1.4 Висновки до Розділу 1 та постановка задачі

В першому підрозділі було розглянуто алгоритм обробки даних JPEG, описано структуру JPEG, здійснено детальний огляд алгоритмів приховування інформації для файлів формату JPEG, наведено порівняльну характеристику стійкості стеганографічних алгоритмів.

В другому підрозділі було здійснено огляд методів пошуку прихованої інформації в графічних зображеннях, наведено переваги та недоліки кожного методу.

В третьому підрозділі було розглянуто поняття «машинне навчання», описано підхід до класифікації заснований на методу опорних векторів.

Із проведеного аналізу випливає така мета роботи: удосконалити метод для аналізу зображень щодо прихованої інформації на основі спроектованої моделі згорткової нейронної мережі та розробити програмний засіб для його реалізації.

Для досягнення мети необхідно розв’язати такі задачі:

1) Розробити удосконалений метод пошуку прихованої інформації в цифрових зображеннях за рахунок нейронної мережі.

2) Розробити програмний засіб для реалізації удосконаленого методу.

# 2 УДОСКОНАЛЕННЯ МЕТОДУ СТЕГАНОАНАЛІЗУ ЦИФРОВИХ ЗОБРАЖЕНЬ ЗА РАХУНОК ШТУЧНОЇ НЕЙРОННОЇ МЕРЕЖІ

Клас методів машинного навчання, які базуються на побудові класифікаторів об'єктів для виявлення прихованих повідомлень, з використанням заданих навчальних вибірок, що містять заповнені та незаповнені контейнери в даний час розвивається досить інтенсивно і, у ряді випадків, показує дуже високу ефективність по відношенню до найбільш потайливих алгоритмів стеганографічно прихованої інформації. У більшості ситуацій суть цього підходу полягає у побудові класифікатора об'єктів-контейнерів для виявлення факту наявності стеганографічно прихованої інформації на основі реалізації процедури навчання за заданими наборами навчальних прикладів (навчальних вибірок). Відмінною особливістю такого підходу є, перш за все, його універсальність [34].

Методи та алгоритми машинного навчання, які їх реалізують можна розділити на дві великі групи:

– класичні «неглибокі» методи та алгоритми машинного навчання або методи поверхневого навчання (shallow methods);

– методи та алгоритми, засновані на застосуванні глибоких нейронних мереж (deep learning methods).

## 2.1 Аналіз можливості удосконалення методу пошуку прихованої інформації в цифрових зображеннях

При використанні методів глибокого навчання завдання полягає в навчанні бінарного класифікатора (класифікатора на два класи) для виявлення факту приховування даних (ЦВЗ, повідомлення) в аналізованому контейнері. При цьому в переважній більшості робіт розглядається завдання стегоаналізу контейнерів-зображень і згорткові нейронні мережі в різних модифікаціях і вдосконаленнях [35].

Для перевірки та порівняння алгоритмів стегоаналізу та систем ознак проводиться створення навчальних та тестових прикладів на основі найбільш потайливих алгоритмів вбудовування інформації. Для контейнерів-зображень у цій постановці інформація, що підлягає виявленню, зазвичай впроваджується за допомогою сучасних методів адаптивної стеганографії HUGO, S-UNIWARD і WOW [36]. Ці методи вважаються найбільш важкі для виявлення на даний момент і саме по них наведені кращі з відомих результатів виявлення факту наявності або відсутності стеганографічно прихованої інформації [37].

Однією з перших робіт у цьому напрямі є робота [38]. У роботі автори запропонували спеціалізовану архітектуру згорткової нейронної мережі, яку вони назвали CNN model called Gaussian-Neuron (GNCNN). Її особливістю було використання просторового фільтра високих частот з фіксованим ядром, спеціальних функцій активації у вигляді гаусіани, центрованої відносно нульового значення входу, та шарів субдискретизації з усередненням у межах вікна пулінгу (Average Pooling) замість часто використовуваного шару (Max Pooling). На вхід мережі подавалися чорно-білі зображення розміром 256×256, оброблені високочастотним фільтром розміром 5×5. Перший згортковий шар фільтрує вхід із ядром розміром 5×5. Другий згортковий шар приймає вихідні дані першого шару як вхідні дані і фільтрує його 16 ядрами розміру 5×5. Третій, четвертий і п'ятий згорткові шари застосовують згортки з 16 ядрами розміру 3×3 відповідно, а шостий згортковий шар – з 16 ядрами розміру 5×5. Активація у вигляді гаусіани застосовується до кожного вихідного сигналу, починаючи з другого по шостий згорткові шари. Кожен згортковий шар супроводжується пулінгом розміру 3×3 і з кроком 2, який працює з картою ознак у відповідному згортковому шарі, що призводить до підсумкового зниження розмірності видобутих ознак до 256.

Вилучені ознаки передаються в модуль класифікації, який складається з трьох повнозв'язних шарів. Вихід кожного нейрона у перших двох повнозв'язних шарах GNCNN активуються звичайною функцією Relu. Останній повнозв'язний шар має два нейрони, і його вихідний сигнал подається на вхід бінарного

класифікатора, що реалізується з використанням активації Softmax. Високочастотний стегошум, доданий до вихідного зображення, є дуже слабким сигналом, на який сильно впливає вміст зображення. Отже, за допомогою фільтрації верхніх частот можна посилити слабкий стегосигнал та зменшити вплив вихідного контенту.

Використання гаусівської активаційної функції забезпечує переважну реакцію згорткових шарів мережі на цей стегосигнал, значення якого локалізовані в межах нуля, і приглушення вхідних впливів, викликаних проходженням через фільтр окремих ділянок зображення.

У результаті, авторам вдалося отримати наступні результати стосовно алгоритмів адаптивної стеганографії HUGO, S-UNIWARD і WOW. Для BOSSBase у всіх трьох випадках впровадження стеганографічно прихованої інформації з різним корисним навантаженням GNCNN забезпечує набагато меншу помилку виявлення, ніж при використанні SVM з ядром Гауса і ознаковою системою SPAM. Порівняно з використанням набору SRM ансамблю класифікаторів, помилка виявилася приблизно на 2...5 % вищою залежно від рівня корисного навантаження. Експерименти на базі ImageNet показують, що GNCNN досягає помилки виявлення, близької до класифікатора з набором ознак SRM. У подальшій роботі цих авторів [39] розглядаються можливості вилучення та аналізу карт ознак для стегоаналізу, що формуються при навчанні глибоких мереж класу згорткових нейронних мереж.

У наступній за часом роботі [40] розглядається нова архітектура згорткової нейронної мережі, що містить всього два згорткових шари, причому перший шар тут виконує роль налаштовуваного фільтра передобробки, а другий має 64 канали і велику розмірність ядра, порівнянну з розмірністю оброблюваного зображення, виконуючи при цьому фактично функцію кількох повнозв'язних шарів. У шарах згортки використовуються функції активації у вигляді гіперболічного тангенса (Tanh). Завершує обробку шар класифікації на два класи з активацією Softmax. Отримані результати (аналогічно на чорно-білих зображеннях) показують суттєве підвищення якості виявлення стеганографічно

прихованої інформації, при якому значення точності класифікації досягає для HUGO, S-UNIWARD і WOW величин близько 80 ... 97% залежно від обсягу корисного навантаження. Однак висновки, зроблені в цій роботі, обмежені використанням одного і того ж ключа вбудовування стегоповідомлення в різні зображення [41]. У роботі Н. А. Нагорного [42] проведено перевірку роботи такої мережі і також показано переваги даної архітектури в порівнянні з GNCNN. При цьому запропоновано комбінований варіант двошарової мережі, в котрий додатково введений високочастотний просторовий фільтр передобробки, що використовується в GNCNN.

В останніх за часом публікаціях, присвячених використанню методології Deep Learning, використовуються найрізноманітніші за архітектурою глибокі мережі [43]. З цих мереж як найбільш ефективну слід виділити мережу Yedrouj-Net [44]. Її архітектура передбачає використання шести згорткових та трьох повнозв'язкових шарів. Крім того, у перших згорткових шарах використовуються нелінійні активації у вигляді функцій Abs(...) і Trunk(...) (лінійні функції з обмеженням по порозі знизу та зверху). Отримані з використанням такої мережі точності класифікації при аналізі даних з бази BOSSBase мають значення близько 72...86% і перевершують результати, які демонструються іншими архітектурами (мережі Xu-Net, Ye-Net [23]).

Порівняльні результати для представленої архітектури Yedrouj-Net стосовно мереж Xu-Net, Ye-Net, а також ансамблю класифікаторів у поєднанні з набором ознак SRM (EC+SRM) представлені в таблиці 2.1.

Таблиця 2.1 – Результати порівняння помилок виявлення для глибоких нейронних мереж при різних обсягах впровадження (корисного навантаження) у %

| Алгоритм стеганографічного приховування інформації що аналізується | WOW з різним параметром завантаження контейнера | | S-UNIWARD з різним параметром завантаження контейнера | |
|---|---|---|---|---|
| | pl = 0,2 | pl = 0,4 | pl = 0,2 | pl = 0,4 |
| EC+SRM | 63 | 74 | 63 | 75 |

Продовження таблиці 2.1

| | | | | |
|---|---|---|---|---|
| Yedrouj-Net | 72 | 86 | 63 | 77 |
| Xu-Net | 67 | 79 | 61 | 73 |
| Ye-Net | 67 | 77 | 60 | 69 |

Після проведення аналізу методів пошуку прихованої інформації в цифрових зображеннях за рахунок штучної нейронної мережі можемо зробити висновок, що суттєвим недоліком статистичних класифікаторів, відсутнім у методах на основі нейронних мереж, є їх вузька спеціалізація на строго визначених методах формування стегоконтейнерів.

В даний час найбільш значний обсяг досліджень та розробок у галузі стегоаналізу забезпечують дослідження в галузі навчання глибоких нейронних мереж для побудови класифікаторів цифрових контейнерів як універсального та потенційно ефективного підходу.

Для удосконалення методу пошуку прихованої інформації в цифрових зображеннях буде здійснюватися об’єднання та автоматизація етапів вилучення ознак та класифікації.

Аналітична робота в даному підрозділі здійснювалась для подальшого порівняння показників удосконаленого методу пошуку прихованої інформації в цифрових зображеннях за рахунок штучної нейронної мережі із уже розробленими методами.

## 2.2 Використання згорткової нейронної мережі, як інструмент удосконалення методу стеганоаналізу цифрових зображень

Стандартні методи пошуку стеганоконтейнеру засновуються на евристичних методах пошуку інформації в контейнері. Складність такого пошуку полягає в тому, що необхідно для кожного алгоритму приховування інформації розробляти набір ознак, що є трудомістким.

Підготовка набору ознак займає багато часу, а також потребує високої кваліфікації особи, яка здійснює розробку пакету ознак для конкретного

алгоритму. Тому в даному підрозділі запропоновано удосконалення пошуку за рахунок використання згорткової нейронної мережі.

Даний клас мереж глибинного навчання спроектований для задач обробки зображень, тому його можна модифікувати для виконання задач пошуку прихованої інформації в графічних контейнерах. Оскільки основною метою роботи згорткових шарів даної моделі нейронної мережі є пошук та виділення ознак їх можна адаптувати для виявлення ознак стеганоконтейнера, що автоматизує та пришвидшить їх пошук.

Як одна з найбільш представницьких моделей глибокого навчання, згорткові нейронні мережі являють собою ієрархічні нейронні мережі, яких навчальні фільтри та операції об'єднання застосовуються по черзі до необроблених вхідних зображень, приводячи до дедалі більше ієрархічним представленням складних об'єктів. Вони були застосовані до завданням розпізнавання [48] та класифікації [49] зображень та показали чудову продуктивність.

Згорткові нейронні мережі поєднує 3 архітектурні ідеї, як показано на рисунку 2.1: часткові області, спільні ваги та об'єднання [50].

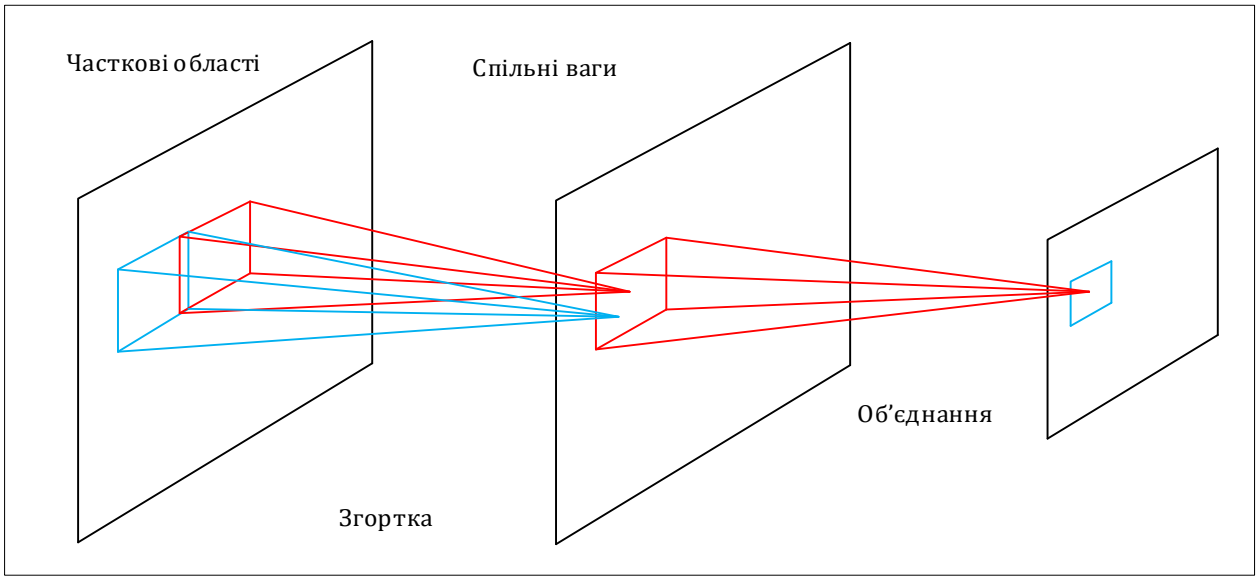



Рисунок. 2.1 – Один типовий шар у ЗНР з операцією згортки і об'єднання

Порівняно зі стандартними нейронними мережами прямого поширення з аналогічними розмірами шарів, ЗНР мають значно менше з'єднань та параметрів. Отже, їх набагато легше тренувати, в той час як теоретично найкраща продуктивність, швидше за все, буде лише трохи гірше. Типова ЗНР складається з двох видів шарів: згорткового шару, який зазвичай слідує за операцією об'єднання та шару класифікації. Вони різняться у тому, як реалізуються операції згортки та об'єднання і як навчаються мережі [51].

ЗНР приймається як основа для моделі стеганалізу.

По-перше, ЗНР можуть приймати необроблені дані як вхідні дані без необхідності в етапі виділення ознак У ЗНР процес виділення та класифікації ознак уніфіковано у межах єдиної структури. Це безпосередньо задовольняє цілі, що полягає у формуванні ознак, вбудовування з зображень замість того, щоб розглядати ЗНР як ще один класифікатор, побудований на існуючих методах стегоаналізу, таких як Spatial domain Rich Model(SRM).

По-друге, ЗНР є контрольованими моделями. На відміну від деяких завдань штучного інтелекту, марковані дані досить легко отримати у стеганалізі. Контейнери та стегоповідомлення можуть розглядатися як позитивні та негативні зразки відповідно.

По-третє, при використанні методу згорткового навчання можна навчати моделі відносно великомасштабними зображеннями (наприклад, 256×256 пікселів). У завдання штучного інтелекту, зображення можуть бути піддискретизовані до малих (наприклад, 32x32 пікселя), щоб зробити процес навчання швидше. Проте ця операція стирає шум стегоповідомлення, отже, робить неможливе виявлення. Тому обумовлена модель глибокого навчання, яка може працювати з великомасштабними зображеннями.

Незважаючи на успішне застосування до деяких завдань штучного інтелекту, діючі ЗНР не враховують статистичні властивості, які важливі для стегоаналізу. Щоб ефективно збирати корисну статистичну інформацію для стеганалізу, пропонується модель ЗНР з гаусовим нейроном, налаштовану по глибині модель ЗНР. Архітектура мережі представлена на рис. 2.2 (праворуч).

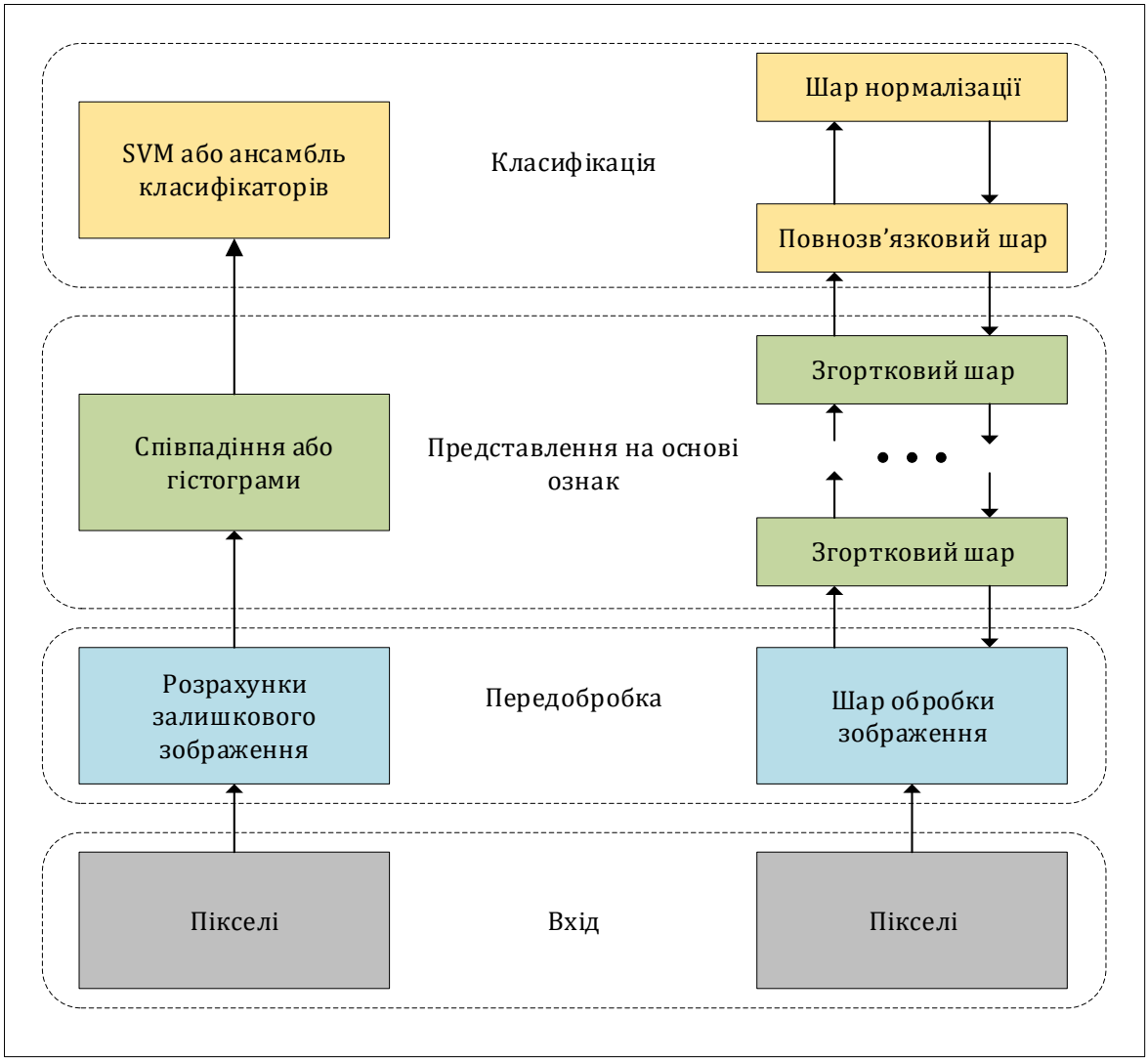



Рисунок 2.2 – ЗНР на основі Гауса (праворуч) та традиційна архітектури стегоаналізу на основі характеристик, складених вручну (ліворуч)

Ліва частина малюнку порівняння, показує порядок роботи традиційних схем стегоаналізу. Удосконалена модель (праворуч) вилучає пікселі на вході, і складається з трьох видів шарів:

– шар обробки зображення;

– кілька згорткових шарів для виділення ознак вбудовування;

– декілька повнозв'язкових шарів для класифікації.

На відміну від традиційних схем, виділення ознак виходять автоматично.

## 2.3 Алгоритм роботи удосконаленого методу заснованого на застосуванні глибоких нейронних мереж

Шар обробки зображення.

У цьому шарі виконується операція фільтрації з наперед визначеним фільтром верхніх частот, який фіксується під час тренування. Як правило, високочастотний стегошум, доданий до контейнера, є свого роду дуже слабким сигналом, на який сильно впливає вміст зображення. Отже, за допомогою фільтрації високих частот зміцнюється слабкий стегосигнал. Це може забезпечити гарну ініціалізацію для всієї мережі, та відповідно, домогтися гарної продуктивності порівняно із випадковою ініціалізацією.

В традиційних схемах стегоаналізу це поширена практика передобробки. Математично фільтрація може бути виражена як показано нижче.

$$R = K * I, \tag{2.1}$$

де $I$ – це зображення, $R$ - зображення після фільтрації верхніх частот (зазвичай зване залишковим зображенням), символ * означає операцію згортки, а $K$ – лінійний фільтр з кінцевою імпульсною характеристикою, який залежить від зсуву, для обчислення залишку. У цій моделі замість всього сімейства шумових залишків, які використовуються у сучасних наборах ознак, таких як SRM та PSRM, використовується лише один із відповідних фільтрів. Проте, буде доведено, що впізнане представлення функції можна порівняти з наборами просторових об'єктів великої розмірності.

У даній моделі використовується ядро згортки $kv$, показане нижче.

$$K_{kv} = \frac{1}{2}\begin{pmatrix} -1 & 2 & -2 & 2 & -1 \\ 2 & -6 & 8 & -6 & 2 \\ -2 & 8 & -12 & 8 & -2 \\ 2 & -6 & 8 & -6 & 2 \\ -1 & 2 & -2 & 2 & -1 \end{pmatrix}, \tag{2.2}$$

Згортковий шар.

Після застосування операцій фільтрації до стегоконтейнера у шарі обробки зображення, ієрархічно збирається разом відповідь цього сигналу з локального до глобального у згортковому шарі.

Вхід та вихід кожного згорткового шару – це набір масивів, який називається картою ознак. На виході, кожна карта ознак це конкретне уявлення ознак витягнуте зі всіх місць у вході. У згортковому шарі присутні три види операцій, згортка, нелінійна функція активації та об'єднання, зазвичай застосовуються послідовно, як показано нижче.

$$X_j^i = pool(f(\Sigma_i X_i^{l-1} * K_{ij}^l + b_j^l)), \quad (2.3)$$

де $f()$ означає нелінійну операцію, $pool()$ визначає об'єднання, $X_j^i$ це $j$-а карта ознак у шарі $l$, $X_i^{l-1}$ це $i$-а карта ознак у шарі $l-1$, $K_{ij}$ це ядро згортки, що навчається, що з'єднує $j$-у вихідну карту та $i$-у вхідну карту, $b_j^l$ це навчальний параметр зміщення для $j$-ї вихідної картки.

Для операції згортки, кожна вихідна карта ознак зазвичай поєднує у собі згортки з кількох вхідних карт ознак. Згорткова структура включає ідеї локальних регіонів та загальні ваги. З локальними регіонами, кожна низькорівнева ознака обчислюється лише з підмножини введення, такої як околиця пікселя в заданій позиції зображення. Такий екстрактор локальних ознак спільно використовує одні й ті самі параметри при застосуванні у різних сусідніх місцях введення, що еквівалентно згортці значень пікселя зображення з ядром, яка містить вагові параметри. Частка параметрів створює інваріантну щодо зсуву операцію, а також зменшує кількість вільних змінних, отже, збільшує продуктивність узагальнення мережі [52].

У стегоаналізі вважається, що для природних зображень отриманих фотокамерою наступна внутрішньокамерна обробка під час отримання зображення, така як колірна інтерполяція, шумозаглушення, корекція кольору та фільтрація, вводить складні залежності в шумову складову сусідніх пікселів. Але стеговий шум, викликаний стеганографічним використанням, буде порушувати ці залежності. Насправді, більшість методів стегоаналізу намагаються використовувати ці залежності виявлення присутності стегового шуму. Евристично передбачається, що через складні залежності хороша оцінка центрального пікселя може бути отримана з сусідніх пікселів, винятком оцінюваного пікселя. Потім, віднімаючи справжнє значення центрального

пікселя з оціненого можна отримати значення помилки передбачення, яке безпосередньо відображає, змінено чи ні піксель. В даний час успішні передбачувальні методи, які зазвичай являють собою деякі інваріантні до зсуву фільтри, розробляються вручну. В рамках пропонованої архітектури предиктори виробляються автоматично. Зважаючи на те, що операція фільтрації на рівні обробки зображень породила вихідний залишковий шум, операція згортки на кожному згортковому рівні полягає в ієрархічному захопленні залежностей між більшою околицею та забезпечення точності прогнозу.

Потім нелінійна активаційна функція застосовується поелементно до виходу операції згортки. Функція нелінійної активації має ефект обмеження амплітуди вихідного сигналу. Що ще важливіше, для багатошарових мереж це уможливлює вирішення деяких проблем, які неможливі з одношаровими. Важливим є вибір функції активації. Це визначається характером даних та передбачуваним розподілом цільових змінних. Зазвичай це сигмоїдальна функція, така як логістична сигмоїда або сигмоїдальна функція гіперболічного тангенсу. Для стегоаналізу використовується функція Гауса, яка може бути виражена як:

$$f(x) = \frac{-x^2}{e^{\sigma^2}}, \tag{2.4}$$

де $\sigma$ - параметр, що визначає ширину кривої. Нейрон із цієї функцією активації виробляє значну позитивну відповідь тільки тоді, коли вхідний сигнал потрапляє у невеликий інтервал навколо нуля. Максимальний відгук буде отримано у центрі нуля. Наскільки відомо, гаусова функція вперше використовується як функція активації в глибоких ЗНР, тому варто називати цей вид ЗНР гаусова ЗНР.

Як уже згадувалося, метою операції згортки є обчислення значень помилок прогнозування шляхом використання залежностей між сусідніми елементами. Мотивація до використання в основі гаусової нелінійної функції, полягає в тому, що вона краще відрізняє стего сигнал і сигнал контейнера від значень помилки передбачення порівняно із сигмоїдальними функціями. Рисунок 2.3 зображує простий приклад того, чому працює гаусова активація.

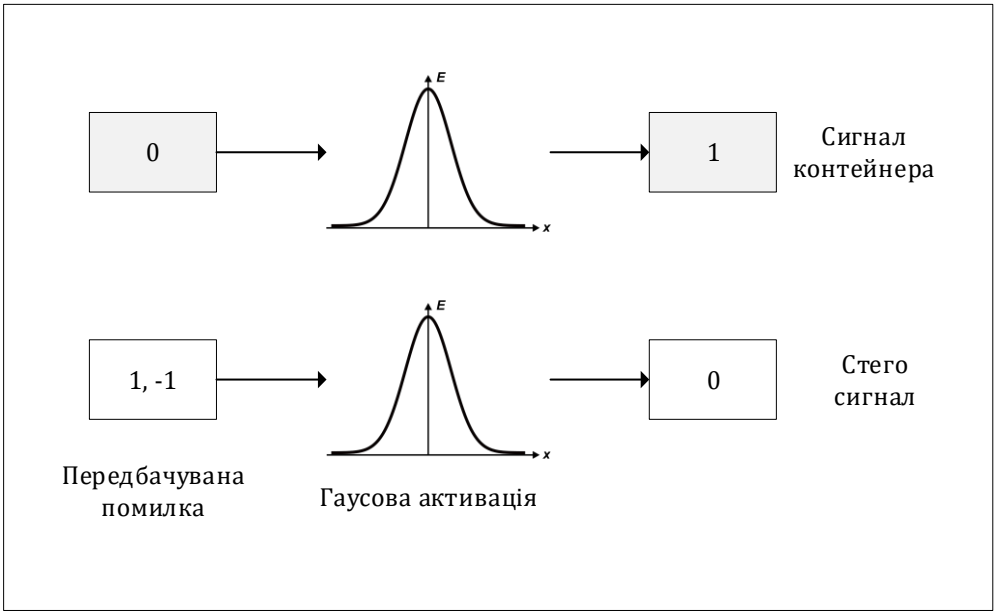



Рисунок 2.3 – Простий приклад, що ілюструє, чому активаційна функція Гауса працює для стегоаналізу у запропонованій моделі

Гауссівська активація може відрізняти сигнал стего і сигнал контейнера від значень помилки передбачення.

В ідеалі має бути три типи значень помилки прогнозування: 1, -1 і 0. Значення 1 та -1 означають, що піксель модифікований операцією вкладення, і вважається, що є сигнал стего. Значення 0 означає, що піксель не змінювався, і ми вважаємо, що це сигнал контейнера. При гаусівської активації значення помилки передбачення, що відповідають сигналу обгортки або стего сигналу, перетворюються на 0 або 1 відповідно. Отже, сигнал контейнера та стого сигнал розділені. На практиці важко отримати такі точні значення дискретного передбачення для кожного пікселя, щоб ухвалювати рішення «так» чи «ні». Фактично, значення помилки передбачення у структурі є безперервними, що відбиває силу стего сигналу у відповідних позиціях. З Гаусівською функцією активації значення, які далекі від нуля, які більш релевантні для сигналу стего, перетворюються приблизно до 0. Тим часом, значення навколо нуля, які більш релевантні сигналу контейнера, перетворюються приблизно до 1. Отже, використання гаусової функції активації дозволяє розробляти на основі мережі точні предиктори для стегоаналізу. Отримані в результаті активації дані потім передаються в об'єднуючу частину шару. Операція об'єднання призначена для

перетворення низькорівневого уявлення ознак у більш корисне, що зберігає важливу інформацію та відкидає непотрібні деталі [53]. Загалом для представлення ознак вищого рівня потрібна інформація від поступово більших вхідних регіонів. Операція об'єднання призводить до об'єднання інформації в межах набору невеликих локальних областей, одночасно скорочуючи час обчислень [54]. В операції об'єднання підсумкові результати сусідніх груп нейронів в одній і тій же карті просторових об'єктів підсумовуються.

Як правило, зазвичай існує два варіанти об'єднання: усереднене об'єднання та максимальне об'єднання. Перший варіант набуває середнього значення всередині області об'єднання:

$$pool(R_j) = \frac{1}{|R_j|}\Sigma_{i \in R_j} a_j, \tag{2.5}$$

У той час як операція максимального об'єднання вибирає максимальне значення:

$$pool(R_j) = \max a_i, \text{i} \in R_j, \tag{2.6}$$

де $R_j$ - область об'єднання в $j$-й карті ознак, $a_i$ - $i$ - й елемент в ній.

Максимальне об'єднання лише фіксує найсильнішу активацію у сфері об'єднання. Воно добре підходить до вигляду представлення ознаки, яка дуже розріджена (тобто має дуже низьку можливість бути активною) [55]. Однак для стеганалізу при такій операції може бути втрачена деяка корисна інформація.

Стего сигнал є свого роду дуже слабким сигналом, тому недостатньо для точного передбачення, просто опиратися на відповіді від окремих елементів. У цій моделі використовується усереднене об'єднання. При усередненому об'єднанні враховуються всі активації в області об'єднання, які повинні відкидати розподіли, спричинені окремими елементами. Завдяки об'єднанню всього сигналу в межах області об'єднання стего сигнал по всій області посилюється.

Об'єднана структура в кожному згортковому шарі зображена на рисунку 2.4.

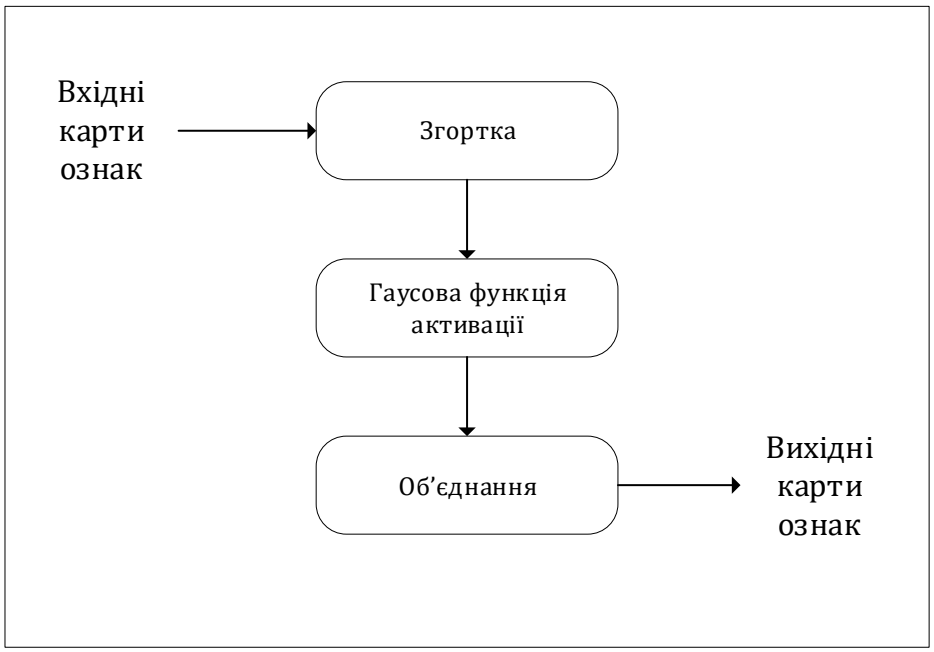



Рисунок 2.4 – Компоненти згорткового шару в моделі, включаючи згортку, гауссівську нелінійність та усереднене об'єднання

Кожен згортковий шар витягує об'єкти зі всіх карт попереднього шару. У міру того, як мережа стає все глибшою, складнішою і більшою високі залежності моделюються прогресивно за участю великих вхідних областей. Отже, цей сигнал агрегується ієрархічно від локального до світового. Ці відомі високорівневі ознаки дозволяють нейронам верхнього рівня передбачати чи змінена вхідна область чи ні.

Класифікаційний шар складається з кількох повнозв'язкових шарів.

Отримані ознаки передаються на ці шари. На верхньому шарі використовується нормована експоненційна функція активації для створення розподілу за всіма мітками класу. У даній моделі використовується двостороння нормована експоненційна функція, як показано нижче:

$$y_i = \frac{e^{x}i}{\Sigma_{j=1}^{2} e^{x}i}, \tag{2.7}$$

для $i = 1,2$ де $x_i$ – сумарний вхідний сигнал до нейрона $i$ у верхньому шарі, а $y_i$ - його вихід.

Щоб зменшити проблему перенавчання, слід застосовувати методику, звану «відсів», для регуляції пов'язаних шарів. У тренуванні з вибуттям, вихід кожного нейрона у відповідних шарах встановлюється рівним нулю із

ймовірністю 0,5. Цей метод покращує узагальнюючу здатність мережі та дає покращену продуктивність тесту.

Отже, алгоритм роботи удосконаленого методу пошуку прихованої інформації в графічних контейнерах заснованого на застосуванні штучних нейронних мереж, а саме згорткової нейронної мережі виглядатиме наступним чином (рис. 2.5).

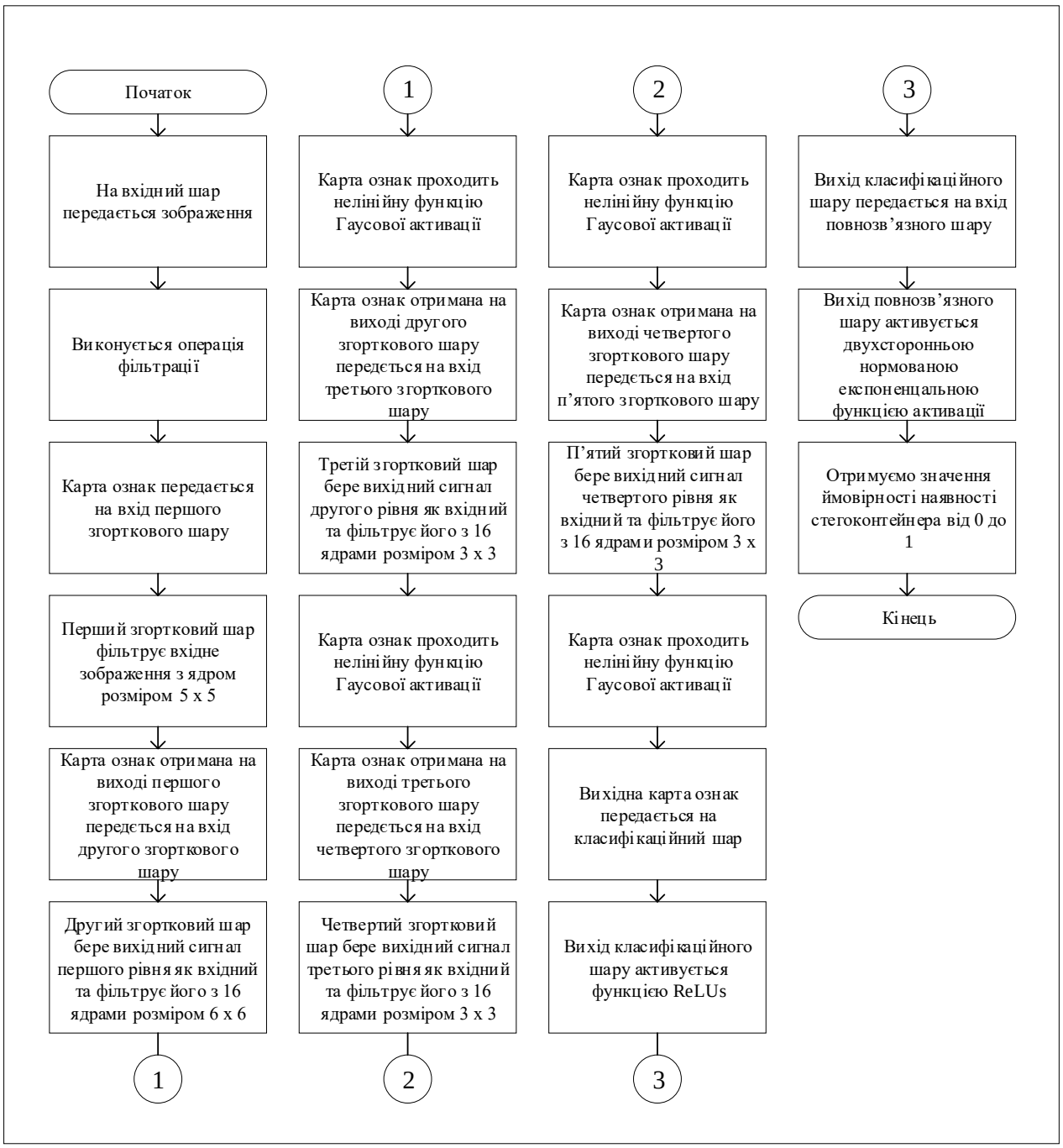



Рисунок 2.5 – Алгоритм роботи удосконаленого методу

Крок 1: На вхідний шар передається зображення.

Крок 2: Виконується операція фільтрації з наперед визначеним фільтром верхніх частот.

Крок 3: Отримана після фільтрації карта ознак передається на вхід першого згорткового шару.

Крок 4: Перший згортковий шар фільтрує вхідне зображення з ядром розміром $5 \times 5$.

Крок 5: Карта ознак отримана на виході першого згорткового шару передється на вхід другого згорткового шару.

Крок 6: Другий згортковий шар бере вихідний сигнал першого рівня як вхідний та фільтрує його з 16 ядрами розміром $5 \times 5$.

Крок 7: Карта ознак проходить нелінійну функцію Гаусової активації.

Крок 8: Карта ознак отримана на виході другого згорткового шару передється на вхід третього згорткового шару.

Крок 9: Третій згортковий шар бере вихідний сигнал другого рівня як вхідний та фільтрує його з 16 ядрами розміром $3 \times 3$.

Крок 10: Карта ознак проходить нелінійну функцію Гаусової активації.

Крок 11: Карта ознак отримана на виході третього згорткового шару передється на вхід четвертого згорткового шару.

Крок 12: Четвертий згортковий шар бере вихідний сигнал третього рівня як вхідний та фільтрує його з 16 ядрами розміром $3 \times 3$.

Крок 13: Карта ознак проходить нелінійну функцію Гаусової активації.

Крок 14: Карта ознак отримана на виході четвертого згорткового шару передється на вхід п’ятого згорткового шару.

Крок 15: П’ятий згортковий шар бере вихідний сигнал четвертого рівня як вхідний та фільтрує його з 16 ядрами розміром $3 \times 3$.

Крок 16: Карта ознак проходить нелінійну функцію Гаусової активації.

Крок 17: Вихідна карта ознак передається на класифікаційний шар.

Крок 18: Вихід класифікаційного шару активується функцією ReLUs.

Крок 19: Вихід класифікаційного шару передається на вхід повнозв’язного шару.

Крок 20: Вихід повнозв’язного шару активується двухсторонньою нормованою експоненцальною функцією активації.

Крок 21: Отримуємо значення ймовірності наявності стегоконтейнера від 0 до 1.

## 2.4 Висновки до Розділу 2

У другому розділі роботи було розглянуто напрямки удосконалення методу стеганоаналізу цифрових зображень за рахунок штучної нейронної мережі, що є основою для подальшої програмної реалізації. Проведено порівняльний аналіз існуючих робіт у даному напрямку та описано їх недоліки.

Наведено алгоритми роботи всіх модулів, зображено дані алгоритми схематично та обґрунтовано їх вибір.

Сформовано етапи створення згорткової нейронної мережі. Описано та обґрунтовано запропонований підхід до вирішення даної задачі з детальними ілюстраціями та прикладами розв’язання.

# 3 ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ СТЕГАНОАНАЛІЗУ ЦИФРОВИХ ЗОБРАЖЕНЬ ЗА УДОСКОНАЛЕНИМ МЕТОДОМ

На основі спроектованої моделі згорткової нейронної мережі був удосконалений та реалізований метод для аналізу зображень щодо прихованої інформації.

## 3.1 Вибір мови та засобів для розробки програмного засобу

Реалізація удосконаленого методу висуває деякі вимоги до мови програмування, а саме розширені можливості багатопоточності на GPU та низькорівнева робота з даними. З метою вибору були розглянуто дві, на сьогоднішній день, популярні мови реалізації нейронних мереж, C++ та Python. Для цих мов існує безліч бібліотек для реалізації глибинного навчання C++ – компільована, статично типізована мова програмування загального призначення.

Підтримує такі парадигми програмування, як процедурне програмування, об'єктно-орієнтоване програмування, узагальнене програмування. Мова має багату стандартну бібліотеку, яка включає в себе поширені контейнери та алгоритми, введення-виведення, регулярні висловлювання, підтримку багатопоточності та інші можливості. C++ поєднує властивості як високорівневих, і низькорівневих мов. У порівнянні з його попередником – мовою C, – найбільшу увагу приділено підтримці об'єктно-орієнтованого та узагальненого програмування.

C++ містить засоби розробки програм контрольованої ефективності для широкого спектра завдань, від низькорівневих утиліт та драйверів до складних програмних комплексів. Зокрема:

– Висока сумісність із мовою Сі: код на Сі може бути з мінімальними переробками скомпілюваний компілятором C++. Зовнішній інтерфейс є прозорим, так що бібліотеки можуть викликатися з C++ без додаткових витрат, і навіть при певних обмеженнях код на C++ може експортуватися зовні відрізняється від коду на Сі (конструкція extern "C").

– Як наслідок попереднього пункту – обчислювальна продуктивність. Мова спроектована так, щоб дати програмісту максимальний контроль над усіма аспектами структури та порядку виконання програми. Один із базових принципів C++ – «не платиш за те, що не використовуєш» - тобто жодна з мовних можливостей, що призводить до додаткових накладних витрат, не є обов'язковою для використання. Є можливість роботи з пам'яттю на низькому рівні.

– Підтримка різних стилів програмування: традиційне імперативне програмування (структурне, об'єктно-орієнтоване), узагальнене програмування, функціональне програмування, породжувальне метапрограмування.

– Автоматичний виклик деструкторів об'єктів у адекватному порядку (зворотному виклику конструкторів) спрощує та підвищує надійність управління пам'яттю та іншими ресурсами (відкритими файлами, мережевими з'єднаннями, з'єднаннями з базами даних тощо).

– Перевантаження операторів дозволяє коротко записувати вирази над користувацькими типами в природній алгебраїчній формі.

– Є можливість управління константністю об'єктів (модифікатори const, mutable, volatile). Використання константних об'єктів підвищує надійність і є підказкою для оптимізації. Перевантаження функцій-членів за ознакою константності дозволяє визначати вибір методу залежно від мети виклику (константний для читання, неконстантний для зміни).

– Шаблони C++ дають можливість побудови узагальнених контейнерів та алгоритмів для різних типів даних. Також шаблони дають можливість проводити обчислення на етапі компіляції.

– Можливість вбудовування предметно-орієнтованих мов програмування в основний код. Такий підхід використовує, наприклад бібліотека Boost.Spirit, що дозволяє задавати EBNF-граматику парсерів прямо у коді C++. Boost.Spirit реалізує рекурсивно-низхідний алгоритм, що накладає відповідні обмеження (такі як неприпустимість лівої рекурсії).

Бібліотеки C++ для роботи с нейронними мережами:

– OpenNN [56]

– FANN [57]

– ConvNet [58]

– ALGLIB [59]

Для реалізації багатопоточності у згорткових нейронних мережах на GPU C++ добре підходить бібліотека cuda-convnet2 [60].

Python – високорівнева мова програмування загального призначення, орієнтована на підвищення продуктивності розробника та читання коду. Синтаксис ядра Python мінімалістичний. Водночас стандартна бібліотека включає великий обсяг корисних функцій.

Python підтримує кілька парадигм програмування, у тому числі структурне, об'єктно-орієнтоване, функціональне, імперативне та аспектно-орієнтоване. Основні архітектурні риси – динамічна типізація, автоматичне керування пам'яттю, повна інтроспекція, механізм обробки винятків, підтримка багатопотокових обчислень та зручні високорівневі структури даних. Код у Python організовується у функції та класи, які можуть об'єднуватись у модулі. Вони у свою чергу можуть бути об'єднані в пакети.

Python – мова програмування, що активно розвивається, нові версії (з додаванням/зміною мовних властивостей) виходять приблизно раз у два з половиною роки. Внаслідок цього та деяких інших причин на Python відсутні стандарт ANSI, ISO або інші офіційні стандарти, їх роль виконує CPython.

Безперечною перевагою є те, що інтерпретатор Python реалізований практично на всіх платформах та операційних системах. Першою такою мовою був C, проте його типи даних на різних машинах могли займати різну кількість пам'яті, і це служило деяким перешкодою при написанні програми, яку можна перенести. У Python таких недоліків немає.

Наступна важлива риса – розширюваність мови, цьому надається велике значення і, як пише сам автор, мова була задумана саме як така, що розширюється. Це означає, що є можливість вдосконалення мови усіма зацікавленими програмістами. Інтерпретатор написаний на C та вихідний код доступний для будь-яких маніпуляцій. У разі потреби, можна вставити його у

свою програму та використовувати як вбудовану оболонку. Або ж, написавши на C свої доповнення до Python і скомпілювавши програму, отримати "розширений" інтерпретатор із новими можливостями.

Наступна перевага - наявність великої кількості модулів які підключаються до програми та забезпечують різні додаткові можливості. Такі модулі пишуться на C і самому Python і можуть бути розроблені всіма досить кваліфікованими програмістами. Як приклад можна навести такі модулі:

Numerical Python – розширені математичні можливості, такі як маніпуляції з цілими векторами та матрицями;

Tkinter – побудова додатків із використанням графічного інтерфейсу користувача (GUI) на основі широко поширеного на X-Windows Tk-інтерфейс;

OpenGL – використання широкої графічної бібліотеки моделювання дво- та тривимірних об'єктів Open Graphics Library фірми Silicon Graphics Inc. Цей стандарт підтримується, зокрема, у таких поширених операційних системах як Microsoft Windows 95 OSR 2, 98 та Windows NT 4.0.

Бібліотеки Python для роботи з нейронними мережами:

– PyBrain [61]

– Theano [62]

– Pylearn2 [63]

– Caffe [64]

Для організації багатопоточності в Python використовується стандартна бібліотека threading.py. Виклики функцій з неї використовуються у вище перерахованих бібліотек.

Із проведеного аналізу мов програмування було зроблено висновок, що для реалізації системи найкраще підходить мова Python. C++, враховуючи свої переваги, є інтерпретованою мовою, що накладає суттєві обмеження на швидкість роботи системи. Як основа для нейронної мережі було використано модель із бібліотеки PyBrain. Для реалізації багатопоточності на GPU використовувалася бібліотека cudaconvnet2. Для організації графічного інтерфейсу було використано бібліотеку Gtk +3 [65].

Для програмної реалізації методу буде використано крос-платформний редактор (Linux, Mac OS, Windows) Visual Studio Code.

### 3.2 Архітектура програмної реалізації удосконаленого методу

Виходячи з удосконаленої моделі нейронної мережі та поставленого завдання, була розроблена автоматизована архітектура системи, яка зображена на рисунку 3.1.

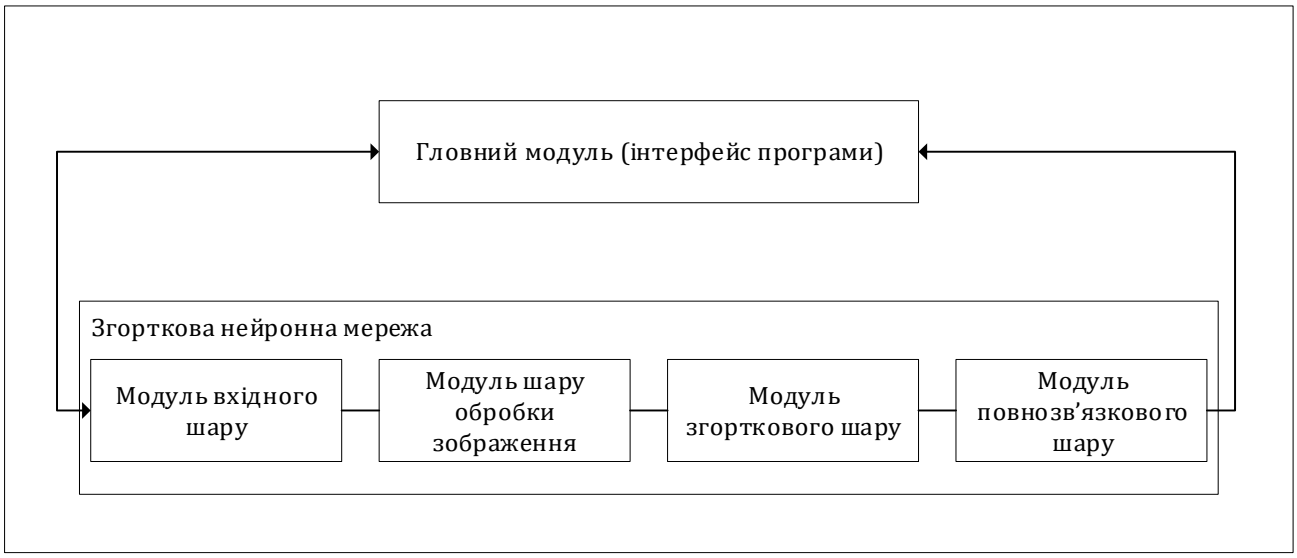



Рисунок. 3.1 – Архітектура системи

Користувач вказує зображення-контейнер, що перевіряється, або масив зображень у вікні головного модуля. Далі зображення передається модулю вхідного шару нейронної мережі. Потім передається модулю шару обробки зображення, де піддається операції фільтрації. Наступним у роботу вступає модуль, який описує функції згорткового шару. У ньому до карт ознак послідовно застосовуються операції згортки та гаусової нелінійності. Після, у модулі класифікаційного шару виводиться рішення про класифікацію, у вигляді числового значення у проміжку між 1 і 0. Рішення виводиться у вікно головного модуля або якщо це масив зображень у файл, який зберігається в домашній директорії користувача.

ГЗНМ навчалася шляхом мінімізації $-log\, y_t$, де $t \in \{1,2\}$ позначає цільовий клас, використовуючи алгоритм зворотного розповсюдження. Поширення помилок та адаптація до ваги у згорткових та повністю пов'язаних

шарах слідують стандартній процедурі. Всі параметри в рівнях вилучення ознак та рівнях класифікації оптимізуються спільно.

Реалізація параметрів згорткової нейронної мережі зображена на рисунку 3.2.

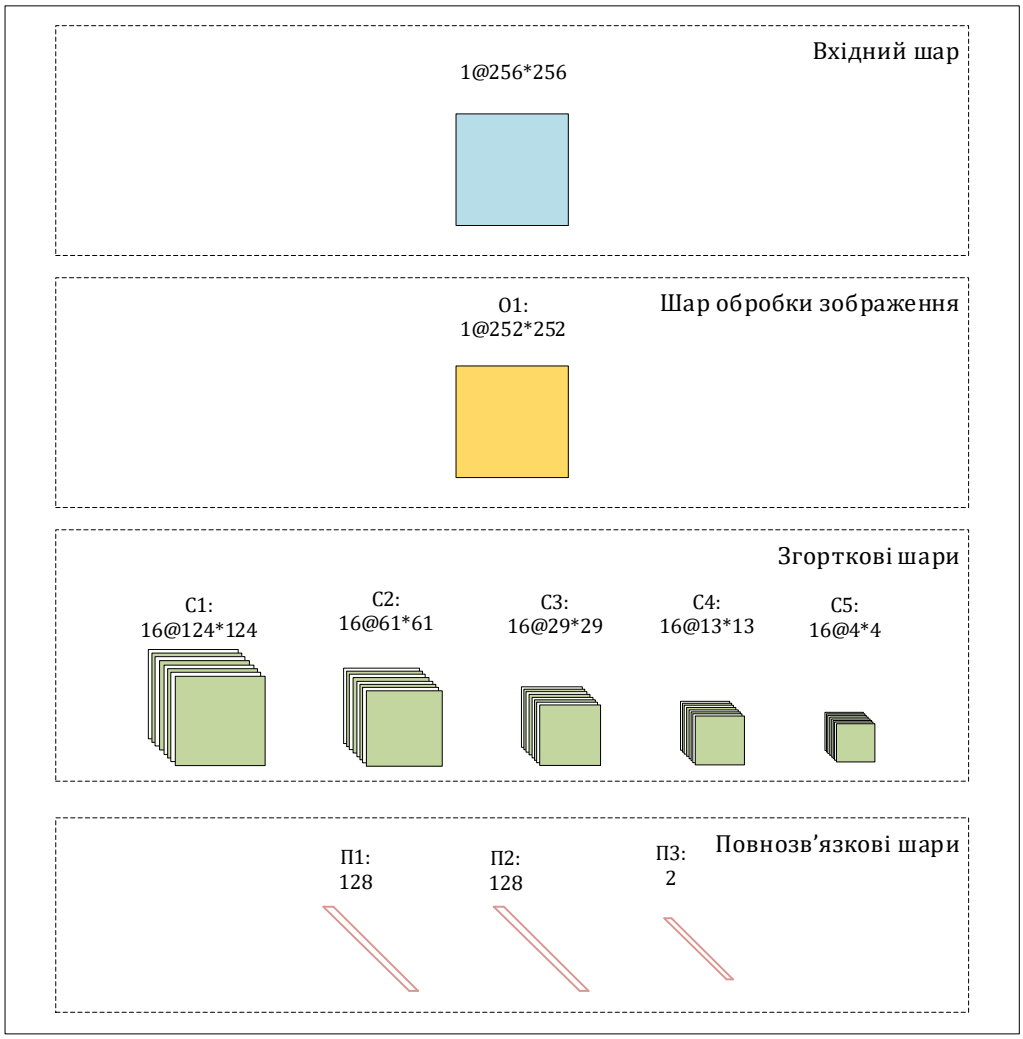



Рисунок. 3.2 – Параметри згорткової нейронної мережі реалізованої в даній системі. Форма "a @ b * b" означає кількість карт характеристик a і роздільна здатність b * b відповідного шару

Перший згортковий шар фільтрує вхідне зображення з ядром розміром $5 \times 5$. Другий згортковий шар бере вихідний сигнал першого рівня як вхідний та фільтрує його з 16 ядрами розміром $5 \times 5$. Третій, четвертий і п'ятий згорткові шари застосовують згортки з 16 ядрами розміром $3 \times 3$ відповідно. Гаусова нелінійна функція застосовується до всіх виходів другого-п'ятого згорткового шару. Тим часом, за кожним з другого-п'ятого згорткових рівнів слідує операція

усередненого об'єднання з розміром вікна $3 \times 3$ та кроком величиною 2, який працює на кожній з карт ознак у відповідному згортковому шарі та призводить до такої ж кількості карт просторових об'єктів зі зменшеним просторовим розширенням. Нарешті, вилучені 256 ознак передаються в модуль класифікації, що складається з трьох повнозв'язкових верств. Перші два пов'язані шари мають по 128 нейронів. Вихід кожного нейрона в перших двох пов'язаних шарах ГЗНМ активується функцією Rectified Linear Units (ReLUs) $f(x) = max(0, x)$ [66]. Останній повнозв'язний шар має два нейрони, і його вихід подається на двосторонню нормовану експоненційну функцію активації.

Розглянемо програмну реалізацію головних модулів згорткової нейронної мережі.

Основним і найважливішим модулем є модуль згорткового шару нейронної мережі:

```
def convolution(image, filt, bias, s=1):

 (n_f, n_c_f, f, _) = filt.shape
   n_c, in_dim, _ = image.shape
   out_dim = int((in_dim - f)/s)+1
   assert n_c == n_c_f,
   out = np.zeros((n_f,out_dim,out_dim))

for curr_f in range(n_f):
  curr_y = out_y = 0
      while curr_y + f <= in_dim:
 curr_x = out_x = 0
         while curr_x + f <= in_dim:
out[curr_f, out_y, out_x] = np.sum(filt[curr_f] * image[:,curr_y:curr_y+f, curr_x:curr_x+f]) +
bias[curr_f]
            curr_x += s
            out_x += 1
         curr_y += s
         out_y += 1
   return out
```

З точки зору програмування згортковий шар має вигляд функції, яка в якості параметрів приймає:

1) image – вхідне зображення

2) filt – фільтр ядра

3) bias – зміщення

4) s – крок згортки

На верхньому рівні знаходиться цикл for, який застосовує фільтри до вихідного зображення. В рамках кожної ітерації використовується два цикли while, які переміщують фільтр уздовж зображення (горизонтально та вертикально).

На кожному етапі отримується поелементне множення пікселів фільтра та фрагмента зображення (оператор *). Результуюча матриця за допомогою підсумовування перетворюється в число, до якого додається зміщення (bias).

Реалізація модулю повнозв’язкового шару виглядатиме наступним чином:

```
(nf2, dim2, _) = pooled.shape
fc = pooled.reshape((nf2 * dim2 * dim2, 1))
```

Використовуючи бібліотеку NumPy для програмування повнозв'язного шару достатньо використати метод reshape. Даний метод надає нову форму матриці не змінюючи її даних.

Лістинг класифікаційного шару має наступний вигляд:

```
z = w3.dot(fc) + b3
z[z<=0]=0
return np.argmax(prods)
```

До матриці вагів w3 застосовується функція бібліотеки NumPy dot, яка здійснює добуток двох матриць, матриці w3 та матриці fc. Матриця fc це матриця ознак отримана з останнього згорткового шару нейронної мережі. До добутку додається матриця зміщення b3. Наступним кроком застосовується нелінійна функція активації ReLU. Синтаксис мови програмування Python дозволяє звернутися до всіх елементів масиву за вказаним фільтром $z <= 0$ та присвоїти їм значення 0. Таким чином реалізується нелінійна функція активації значення ознак, які менше 0 перетворюються на нуль, а ознаки, які більше нуля не змінюються.

Останнім кроком до матриці отриманої із класифікаційного шару застосовується функція argmax для отримання ймовірності.

Загальний вигляд нейронної мережі виглядає наступним чином:

```
def predict(image, f1, f2,  f3, f4,  f5, w1, w2, b1, conv_s = 1):

   conv1 = convolution(image, f1, b1, conv_s)
   conv1[conv1<=0] = 0

   conv2 = convolution(conv1, f2, b2, conv_s)
   conv2[conv2<=0] = 0

   conv3 = convolution(conv2, f2, b2, conv_s)
   conv3[conv3<=0] = 0

   conv4 = convolution(conv3, f2, b2, conv_s)
   conv4[conv4<=0] = 0

   conv5 = convolution(conv4, f2, b2, conv_s)
   conv5[conv5<=0] = 0

   pooled = maxpool(conv5, pool_f, pool_s)
   (nf2, dim2, _) = pooled.shape

   fc = pooled.reshape((nf2 * dim2 * dim2, 1))

   z = w3.dot(fc) + b3
   z[z<=0] = 0

   out = w4.dot(z) + b4
   probs = softmax(out)

   return np.argmax(probs), np.max(probs)
```

На вхід передаються наступні параметри:

1) image – зображення;
2) f1 – f5 – фільтри ядра для кожного згорткового шару
3) w1, w2 – ваги для кожного з класифікаційних шарів
4) b1 – нормальне зміщення
5) s – крок згортки

Параметри f1 – f5 та w1, w2 отримуються після повного навчання нейронної мережі. В лістингу наведеному вище повністю реалізований розроблений алгоритм згорткової нейронної мережі описаний в розділі 2.3 даної магістерської кваліфікаційної роботи.

Навчання реалізованої згорткової нейронної мережі здійснювалось за допомогою популярного алгоритму оптимізації Adam [70]. Це популярний алгоритм у сфері глибокого навчання, оскільки він швидко досягає хороших результатів.

Розглянемо наступні параметри конфігурації алгоритму:

– альфа – Також називають швидкістю навчання або розміром кроку. Пропорція, в якій ваги оновлюються (наприклад, 0,001). Більші значення (наприклад, 0,3) призводять до швидшого початкового навчання перед оновленням швидкості. Менші значення (наприклад, 1.0E-5) уповільнюють навчання відразу під час навчання;

– бета1 – Оцінка експоненційної швидкості розпаду для першого моменту (наприклад, 0,9);

– бета2 – Експоненційна швидкість розпаду для оцінок другого моменту (наприклад, 0,999). Це значення має бути близько 1,0 для проблем із градієнтом (наприклад, проблеми НЛП та комп'ютерного зору);

– епсілон – Це дуже мале число, щоб запобігти будь-якому поділу на нуль у реалізації.

Лістинг реалізованого алгоритму виглядає наступним чином:

```
def adamGD(batch, num_classes, lr, dim, n_c, beta1, beta2, params, cost):

  [f1, f2, w3, w4, b1, b2, b3, b4] = params

  X = batch[:,0:-1]
  X = X.reshape(len(batch), n_c, dim, dim)
  Y = batch[:,-1]

  cost_ = 0
  batch_size = len(batch)

  df1 = np.zeros(f1.shape)
  df2 = np.zeros(f2.shape)
  dw3 = np.zeros(w3.shape)
  dw4 = np.zeros(w4.shape)
  db1 = np.zeros(b1.shape)
  db2 = np.zeros(b2.shape)
  db3 = np.zeros(b3.shape)
  db4 = np.zeros(b4.shape)
```

```
v1 = np.zeros(f1.shape)
v2 = np.zeros(f2.shape)
v3 = np.zeros(w3.shape)
v4 = np.zeros(w4.shape)
bv1 = np.zeros(b1.shape)
bv2 = np.zeros(b2.shape)
bv3 = np.zeros(b3.shape)
bv4 = np.zeros(b4.shape)

s1 = np.zeros(f1.shape)
s2 = np.zeros(f2.shape)
s3 = np.zeros(w3.shape)
s4 = np.zeros(w4.shape)
bs1 = np.zeros(b1.shape)
bs2 = np.zeros(b2.shape)
bs3 = np.zeros(b3.shape)
bs4 = np.zeros(b4.shape)

for i in range(batch_size):

    x = X[i]
    y = np.eye(num_classes)[int(Y[i])].reshape(num_classes, 1)

    grads, loss = conv(x, y, params, 1, 2, 2)
    [df1_, df2_, dw3_, dw4_, db1_, db2_, db3_, db4_] = grads

    df1+=df1_
    db1+=db1_
    df2+=df2_
    db2+=db2_
    dw3+=dw3_
    db3+=db3_
    dw4+=dw4_
    db4+=db4_

    cost_+= loss

v1 = beta1*v1 + (1-beta1)*df1/batch_size
s1 = beta2*s1 + (1-beta2)*(df1/batch_size)**2
f1 -= lr * v1/np.sqrt(s1+1e-7)

bv1 = beta1*bv1 + (1-beta1)*db1/batch_size
bs1 = beta2*bs1 + (1-beta2)*(db1/batch_size)**2
b1 -= lr * bv1/np.sqrt(bs1+1e-7)

v2 = beta1*v2 + (1-beta1)*df2/batch_size
s2 = beta2*s2 + (1-beta2)*(df2/batch_size)**2
f2 -= lr * v2/np.sqrt(s2+1e-7)

bv2 = beta1*bv2 + (1-beta1) * db2/batch_size
```

```
bs2 = beta2*bs2 + (1-beta2)*(db2/batch_size)**2
b2 -= lr * bv2/np.sqrt(bs2+1e-7)

v3 = beta1*v3 + (1-beta1) * dw3/batch_size
s3 = beta2*s3 + (1-beta2)*(dw3/batch_size)**2
w3 -= lr * v3/np.sqrt(s3+1e-7)

bv3 = beta1*bv3 + (1-beta1) * db3/batch_size
bs3 = beta2*bs3 + (1-beta2)*(db3/batch_size)**2
b3 -= lr * bv3/np.sqrt(bs3+1e-7)

v4 = beta1*v4 + (1-beta1) * dw4/batch_size
s4 = beta2*s4 + (1-beta2)*(dw4/batch_size)**2
w4 -= lr * v4 / np.sqrt(s4+1e-7)

bv4 = beta1*bv4 + (1-beta1)*db4/batch_size
bs4 = beta2*bs4 + (1-beta2)*(db4/batch_size)**2
b4 -= lr * bv4 / np.sqrt(bs4+1e-7)

cost_ = cost_/batch_size
cost.append(cost_)

params = [f1, f2, w3, w4, b1, b2, b3, b4]
```

Лістинг навчання нейронної мережі зображено нижче:

```
def train(num_classes = 10, lr = 0.01, beta1 = 0.95, beta2 = 0.99, img_dim = 28, img_depth
= 1, f = 5, num_filt1 = 8, num_filt2 = 8, batch_size = 32, num_epochs = 2, save_path =
'params.pkl'):

    m =50000
    X = extract_data('train-images-idx3-ubyte.gz', m, img_dim)
    y_dash = extract_labels('train-labels-idx1-ubyte.gz', m).reshape(m,1)
    X-= int(np.mean(X))
    X/= int(np.std(X))
    train_data = np.hstack((X,y_dash))

    np.random.shuffle(train_data)

    f1, f2, w3, w4 = (num_filt1 ,img_depth,f,f), (num_filt2 ,num_filt1,f,f), (128,800), (10, 128)
    f1 = initializeFilter(f1)
    f2 = initializeFilter(f2)
    w3 = initializeWeight(w3)
    w4 = initializeWeight(w4)

    b1 = np.zeros((f1.shape[0],1))
    b2 = np.zeros((f2.shape[0],1))
    b3 = np.zeros((w3.shape[0],1))
    b4 = np.zeros((w4.shape[0],1))

    params = [f1, f2, w3, w4, b1, b2, b3, b4]
```

```
    cost = []

    print("LR:"+str(lr)+", Batch Size:"+str(batch_size))

    for epoch in range(num_epochs):
        np.random.shuffle(train_data)
        batches = [train_data[k:k + batch_size] for k in range(0, train_data.shape[0],
batch_size)]

        t = tqdm(batches)
        for x,batch in enumerate(t):
            params, cost = adamGD(batch, num_classes, lr, img_dim, img_depth, beta1, beta2,
params, cost)
            t.set_description("Cost: %.2f" % (cost[-1]))


    with open(save_path, 'wb') as file:
        pickle.dump(params, file)

    return cost
```

Навчання проводилось до моменту, коли відсоток помилки виявлення стегоконтейнера не відповідав бажаному. За бажаний відсоток, в рамках даної роботи, а також з врахуванням технічних характеристик середовища у якому проводилось навчання згорткової нейронної мережі, було прийнято значення помилки виявлення стегоконтейнера менше 35%.

### 3.3 Огляд розробленого програмного продукту та аналіз результатів

Робота із удосконаленим методом відбувається у графічному режимі. Головне вікно програми зображено на рисунку 3.3. Реалізована програма містить 5 кнопок за допомогою яких можна здійснити наступні дії:

- відкрити файл зображення;
- відкрити масив зображень;
- почати сканування;
- зупинити сканування;
- завантажити результати.


Рисунок 3.3 – Головне вікно

Залежно від поставленого завдання користувач може передати системі на сканування один передбачуваний контейнер та масив контейнерів. Для цього потрібно активувати кнопку «Відкрити файл зображення» або «Відкрити масив зображень» відповідно. Після даної дії з’являється вікно вибору потрібного файлу/файлів із локальної директорії.

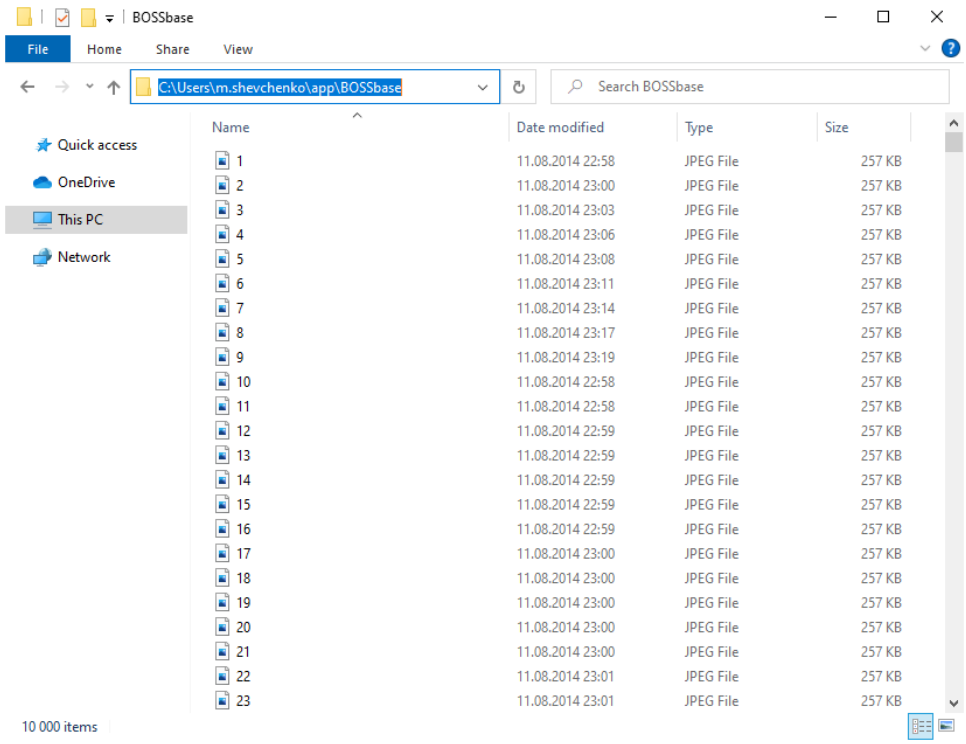



Рисунок 3.4 – Діалогове вікно вибору файлу/файлів.

Після вибору файлу на головному екрані з’явиться повідомлення про успішне завантаження файлу. Для того щоб почати сканування потрібно натиснути на кнопку «Почати сканування». Якщо потрібно зупинити сканування, то здійснюється активація кнопки «Зупинити сканування». Після сканування файлу на головному екрані з’явиться інформація про результати проведеного аналізу.

Система класифікує файли відповідно до того наявна прихована інформація в зображенні чи ні. Після визначення виводить результат:

– Позитивно;

– Негативно.

Якщо це один порожній контейнер, буде виведено наступне повідомлення «Negative» (рис. 3.5).

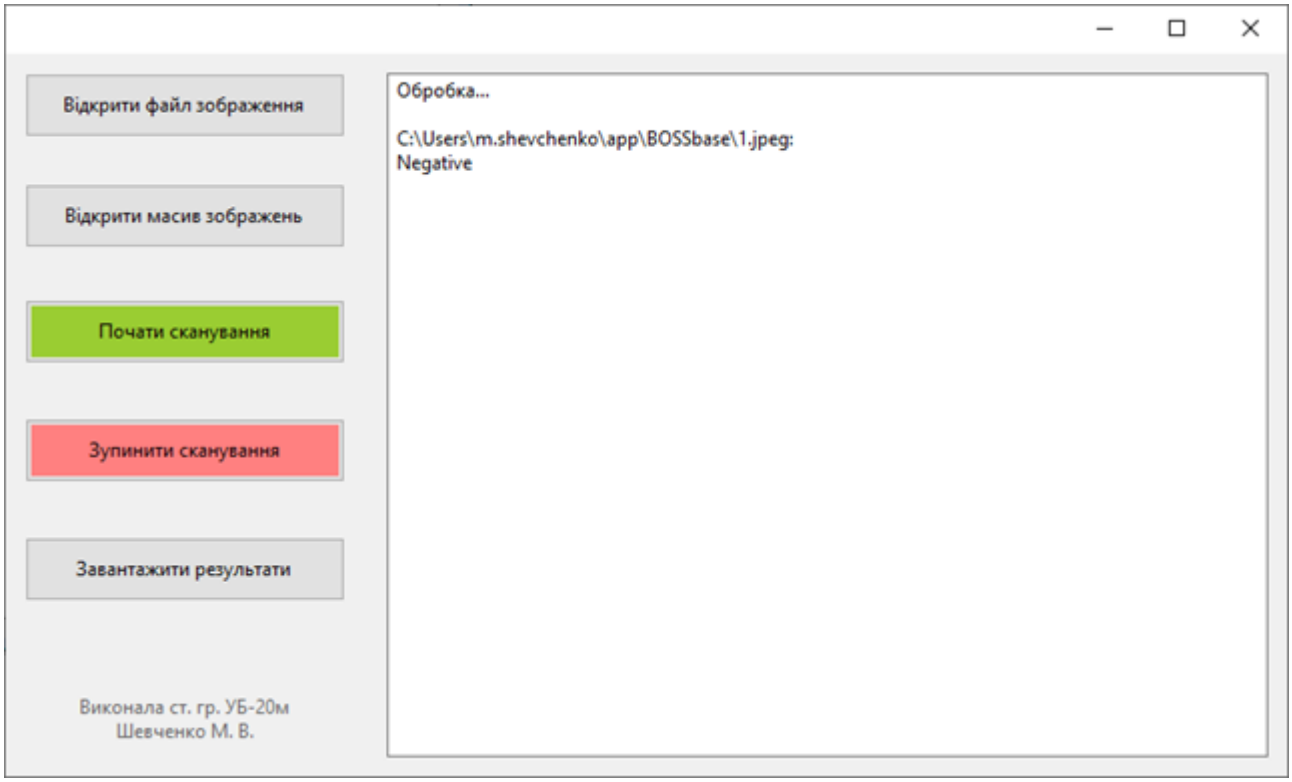



Рисунок 3.5 – Результат сканування порожнього контейнера

Якщо система класифікує зображення як стегоконтейнер, то буде виведено повідомлення «Positive» (рис. 3.6).

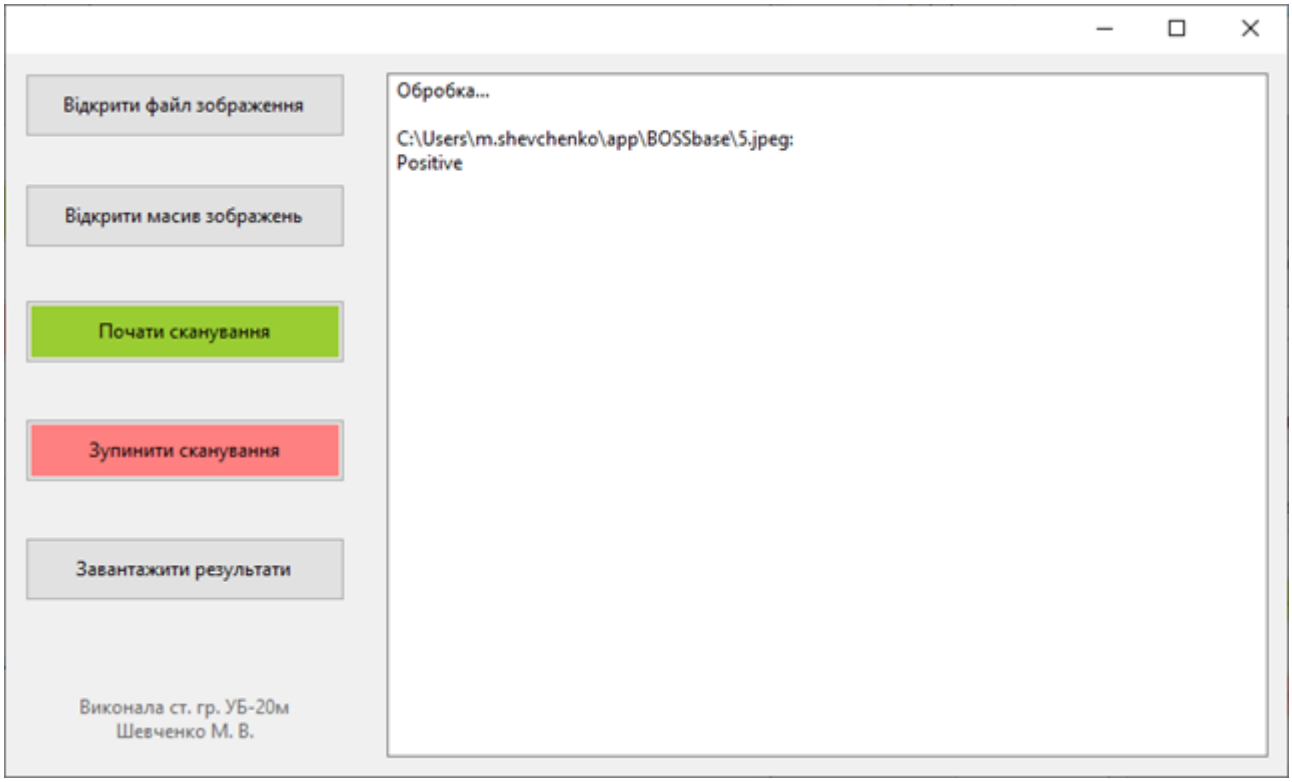



Рисунок 3.6 – Виявлено стегоконтейнер

Якщо необхідно перевірити декілька файлів одразу, користувач виділяє необхідні для перевірки файли та відкриває їх (рис. 3.7).

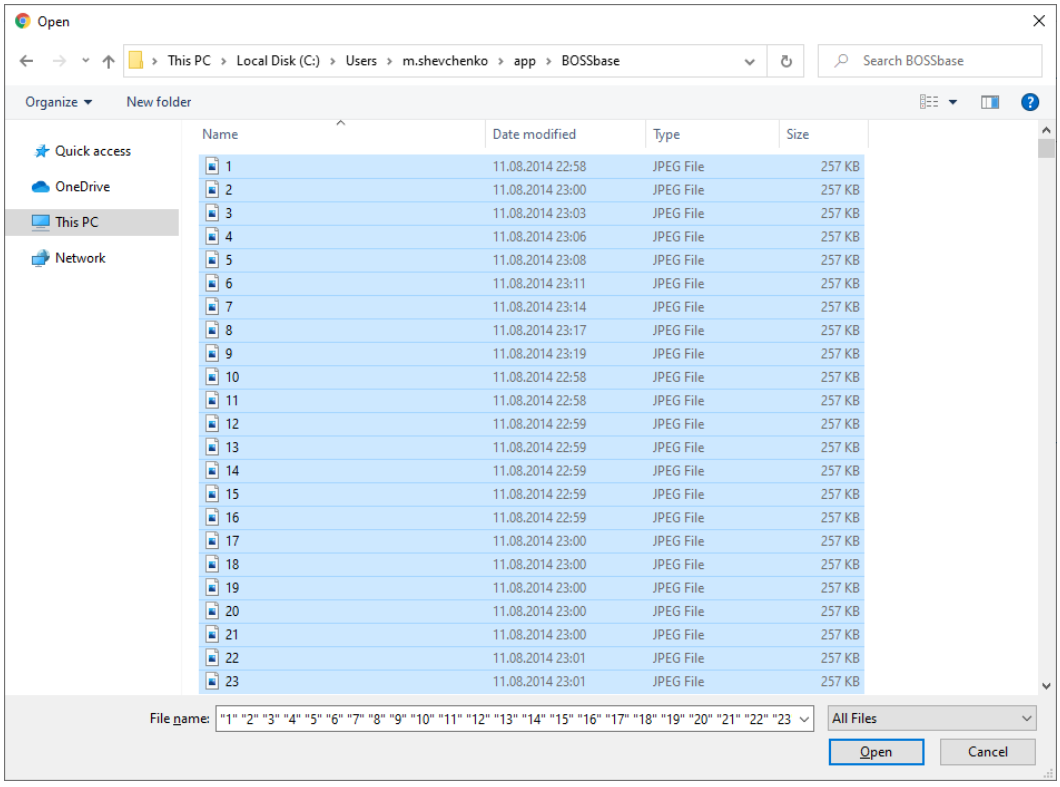



Рисунок 3.7 – Вибір кількох зображень

Далі система послідовно їх аналізує і виводить результат аналізу (рис. 3.8).

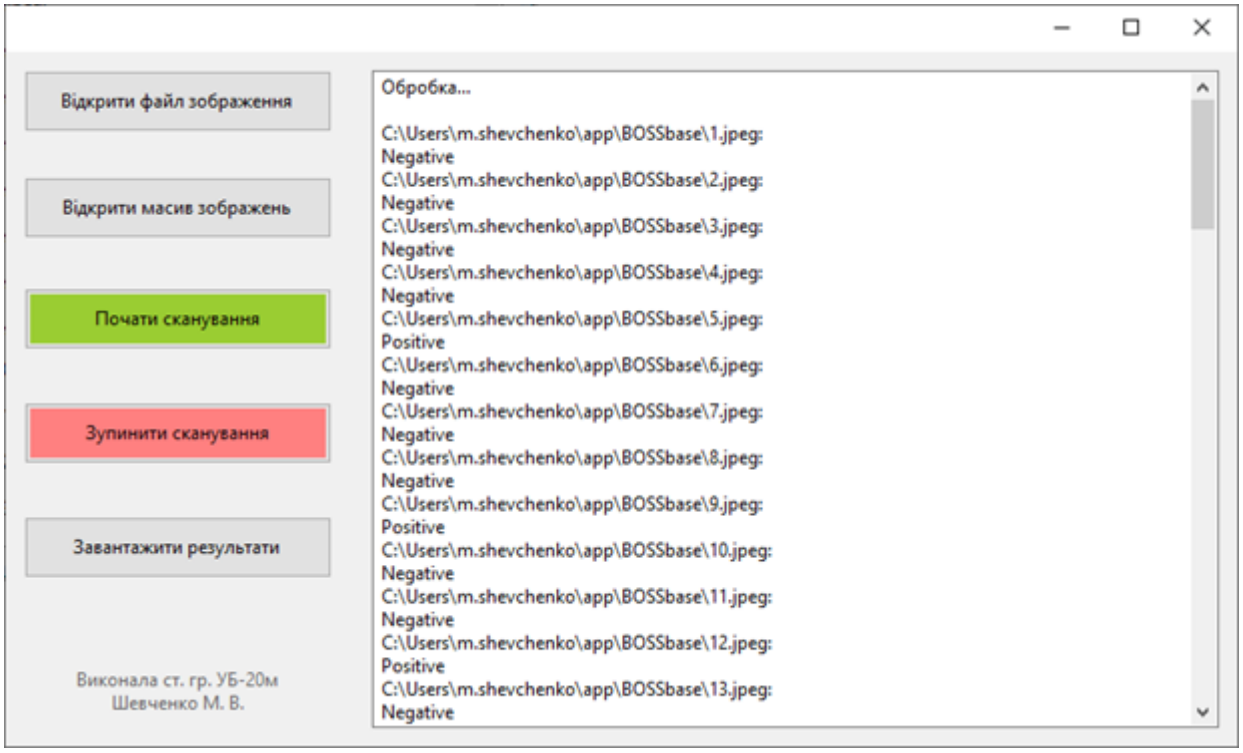



Рисунок 3.8 – Вибір кількох зображень

Якщо здійснювався аналіз масиву зображень, то є можливість експортувати результати аналізу у форматі .txt. Після активації кнопки «Завантажити результати» здійснюється експорт результатів сканування. На головному екрані з’явиться інформація про успішне завантаження (рис. 3.9).


Рисунок 3.9 – Перевірка масиву зображень

Файл із даними зберігається в локальну директорію exportFiles (рис. 3.10).

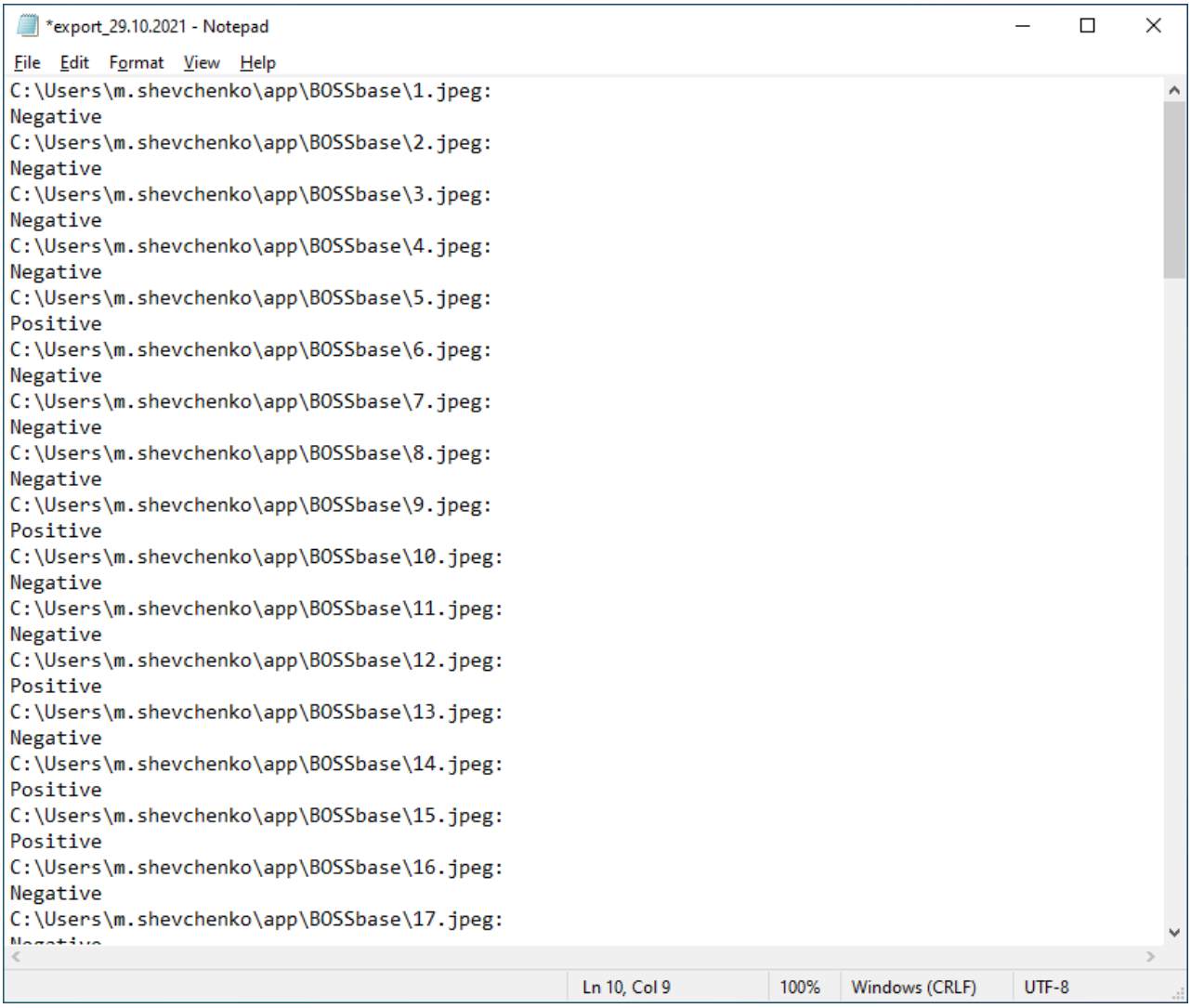



Рисунок 3.10 – Зміст звіту про перевірку масиву зображень

Мережа проходила навчання на сервері, всередині віртуальної машини, гіпервізорі QUEMU/KVM[57] з прямим прокиданням процесора Intel Xeon E5-2650 2.0 ГГц та графічного прискорювача Tesla K40 12G драйверами virtio. Середній час тренування становив близько 2 днів. При використанні більш сучасних графічних прискорювачів час може значно зменшитися.

Експерименти проводились на стандартизованій базі даних BOSSbase. Ця база даних містить 10 000 зображень у відтінках сірого, отримані сімома цифровими камерами в необробленому формат і потім стиснуті до розміру 512 × 512 пікселів. В експериментах додатково було змінено розмір зображень до

256×256 пікселів. Ця зміна розміру була необхідно лише через обмеження обчислювальних можливостей.

Щоб оцінити ефективність удосконаленого методу, експерименти були проведені на трьох сучасних стеганографічних алгоритми: HUGO, WOW, і S-UNIWARD. Продуктивність усіх методів оцінювалася на трьох корисних навантаженнях: 0,3, 0,4 та 0,5 біт на піксель (bpp). Помилки виявлення представлені у таблиці 3.1.

Таблиця 3.1 – Результати порівняння помилок виявлення для глибоких нейронних мереж при різних обсягах впровадження (корисного навантаження) у %

| bpp | HUGO | | | WOW | | | S-UNIWARD | | |
|---|---|---|---|---|---|---|---|---|---|
| | ЗНР | SRM | SPAM | ЗНР | SRM | SPAM | ЗНР | SRM | SPAM |
| 0.3 | 33.8% | 29.6% | 42.9% | 34.3% | 31.2% | 42.2% | 35.9% | 31.5% | 40.0% |
| 0.4 | 28.9% | 25.2% | 39.1% | 29.3% | 25.7% | 38.2% | 30.9% | 26.3% | 35.1% |
| 0.5 | 25.7% | 21.4% | 35.7% | 24.8% | 22.1% | 34.9% | 26.3% | 21.4% | 30.6% |

Результати порівнювалися з двома вручну підібраними наборами ознак: низькорозмірним набором SPAM [67], реалізованим за допомогою SVM на основі гаусової радіальної базисної функції, та високодозвільним набором SRM [68], одним із сучасних наборів ознак, реалізованим з допомогою ансамблю класифікаторів [69].

Удосконалений розроблений метод досягає набагато нижчої помилки виявлення, ніж набір SPAM. В порівнянні з набором SRM, реалізованим за допомогою ансамблю класифікаторів, помилка виявлення приблизно на 2-5% вище, залежно від корисного навантаження. Усі результати цих експериментів підтверджують, що автоматичне визначення ознак є ефективним в удосконаленому методі. І воно краще дозволяє протидіяти складним методам вбудовування.

### 3.4 Висновки до Розділу 3

Отже, у даному розділі представлена робота, що мала на меті удосконалення методу пошуку прихованої інформації в цифрових зображеннях на основі штучної нейронної мережі.

У першій частині розділу обґрунтовано обрану мову програмування та описано застосовані технології.

В другій частині розділу було розроблено автоматизовану архітектуру системи, реалізовано параметри згорткової нейронної мережі, описано структуру удосконаленого методу.

У третій частина розділу здійснено огляд програмного продукту та представлено аналіз результатів.

Таким чином, даний розділ описує та пояснює програмну реалізацію удосконаленого методу та обґрунтовує основні його особливості у практичному застосуванні.

# 4 ЕКОНОМІЧНА ДОЦІЛЬНІСТЬ СТВОРЕННЯ ПЗ ПОШУКУ ПРИХОВАНОЇ ІНФОРМАЦІЇ НА ОСНОВІ ЗНР

## 4.1 Проведення наукового аудиту науково-дослідної роботи

Суб'єктами виконання науково-дослідних робіт і розробок є наукові організації, науково-дослідні центри при закладах вищої освіти, науково-дослідні, проектно-конструкторські організації, експериментальні підприємства, а також науково-виробничі об'єднання.

Метою фундаментальних і частково пошукових досліджень не є одержання продукту, виробу або послуги, що можуть стати товаром і оформитися у вигляді певного комерційного інвестиційного проекту. Однак на їхній основі здійснюється генерація ідей, які можуть трансформуватися в проекти науково-дослідних і дослідно-конструкторських робіт. Тому пошукові роботи можуть мати деяку комерційну цінність.

Наукова і науково-технічна результативність НДР не може бути оцінена з використанням методу дисконтування грошових потоків, за винятком випадків, коли дослідження мають вартісні характеристики результату НДР як наукової інформації, тому у деяких випадках результати дослідження можуть мати вартісні характеристики результату науково-дослідної роботи як наукової інформації, яку купує замовник.Тобто у такому випадку може виникнути фактична ефективність науково-дослідної роботи.

Для наукових і пошукових науково-дослідних робіт зазвичай здійснюють оцінювання наукового ефекту.

Основними ознаками наукового ефекту науково-дослідної роботи є новизна роботи, рівень її теоретичного опрацювання, перспективність, рівень розповсюдження результатів, можливість реалізації. Науковий ефект НДР можна охарактеризувати двома показниками: ступенем наукової новизни та рівнем теоретичного опрацювання.

Значення показників ступеня новизни і рівня теоретичного опрацювання науково-дослідної роботи в балах наведено в табл. 4.1 та 4.2.

Таблиця 4.1 – Показники ступеня новизни науково-дослідної роботи

| Ступінь новизни | Характеристика ступеня новизни | Значення показник аступеня новизни, бали |
|---|---|---|
| *1* | *2* | *3* |
| Принципово нова | Робота якісно нова за постановкою задачі і ґрунтується на застосуванні оригінальних методів дослідження. Результати дослідження відкривають новий напрям в цій галузі науки і техніки. Отримано принципово нові факти, закономірності; розроблено нову теорію. Створено принципово новий пристрій, спосіб, метод | 60…100 |
| Нова | Отримано нову інформацію, яка суттєво зменшує невизначеність наявних значень (по-новому або вперше пояснено відомі факти, закономірності, впроваджено нові поняття, розкрито структуру змісту). Проведено суттєве вдосконалення, доповнення і уточнення раніше досягнутих результатів | 40…60 |
| Відносно нова | Робота має елементи новизни в постановці задачі і методах дослідження. Результати дослідження систематизують і узагальнюють наявну інформацію, визначають шляхи подальших досліджень; вперше знайдено зв’язок (або знайдено новий зв’язок) між явищами. В принципі, відомі положення поширено на велику кількість об’єктів, в результаті чого знайдено ефективне рішення. Розроблено більш прості способи для досягнення відомих результатів. Проведено часткову раціональну модифікацію (з ознаками новизни) | 10…40 |
| Традиційна | Робота виконана за традиційною методикою. Результати дослідження мають інформаційний характер. Підтверджено або поставлено під сумнів відомі факти та твердження, які потребують перевірки. Знайдено новий варіант рішення, який не дає суттєвих переваг порівняно з існуючим | 2…10 |
| Не нова | Отримано результат, який раніше зафіксований в інформаційному полі та не був відомий авторам | 1…2 |

Таблиця 4.2 – Показники рівня теоретичного опрацювання науково-дослідної роботи

| Характеристика рівня теоретичного опрацювання | Значення показника рівня теоретичного опрацювання, бали |
|---|---|
| 1 | 2 |
| Відкриття закону, розробка теорії | 80…100 |

Продовження таблиці 4.2

| 1 | 2 |
|---|---|
| Глибоке опрацювання проблеми: багатоаспектний аналіз зв'язків, взаємозалежності між фактами з наявністю пояснень, наукової систематизації з побудовою евристичної моделі або комплексного прогнозу | 60…80 |
| Розробка способу (алгоритму, програми), пристрою, отримання нової речовини | 20…60 |
| Елементарний аналіз зв'язків між фактами та наявною гіпоте- зою, класифікація, практичні рекомендації для окремого ви- падку тощо | 6…20 |
| Опис окремих елементарних фактів, викладення досвіду, результатів спостережень, вимірювань тощо | 1…5 |

Показник, який характеризує науковий ефект, визначається за формулою:

$$E_{нау} = 0{,}6 \cdot k_{нов} + 0{,}4 \cdot k_{теор},,,, \quad (4.1)$$

де $k_{нов}$, $k_{теор}$ – показники ступенів новизни та рівня теоретичного опрацювання науково-дослідницької роботи, бали;

0,6 та 0,4 – питома вага (значимість) показників ступеня новизни та рівня теоретичного опрацювання науково-дослідної роботи

$$E_{нау} = 0{,}6 \cdot 60 + 0{,}4 \cdot 55 = 58,,,, \quad (4.2)$$

Визначення характеристики показника $E_{нау}$ проводиться на основі висновків експертів, виходячи з граничних значень, які наведено в табл. 2.3.

Таблиця 4.3 – Граничні значення показника наукового ефекту

| Досягнутий рівень показника | Кількість балів |
|---|---|
| Високий | 70…100 |
| Середній | 50…69 |
| Достатній | 15…49 |
| Низький (помилкові дослідження) | 1…14 |

Відповідно до таблиці 4.3 значення показника наукового ефекту має середній рівень і становить 58 балів, адже для розробки приладу було проведено суттєве вдосконалення, доповнення і уточнення раніше досягнутих результатів.

## 4.2 Проведення комерційного та технологічного аудиту

Метою проведення комерційного і технологічного аудиту є оцінювання науково-технічного рівня та рівня комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності, тобто під час виконання магістерської кваліфікаційної роботи.

Для проведення комерційного та технологічного аудиту залучаємо 3-х незалежних експертів, які є провідними викладачами випускової кафедри.

Оцінювання науково-технічного рівня розробки та її комерційного потенціалу здійснімо із застосуванням п’ятибальної системи оцінювання за 12-ма критеріями, наведеними в табл. 4.4.

Таблиця 4.4 – Рекомендовані критерії оцінювання науково-технічного рівня і комерційного потенціалу розробки та бальна оцінка

| Критерії оцінювання та бали (за 5-ти бальною шкалою) | | | | | |
|---|---|---|---|---|---|
| № | 0 | 1 | 2 | 3 | 4 |
| 1 | 2 | 3 | 4 | 5 | 6 |
| Технічна здійсненність концепції | | | | | |
| 1 | Достовірність концепції не підтверджена | Концепція підтверджена експертними висновками | Концепція підтверджена розрахунками | Концепція перевірена на практиці | Перевірено роботоздатність продукту в реальних умовах |
| Ринкові переваги (недоліки): | | | | | |
| 2 | Багато аналогів на малому ринку | Мало аналогів на малому ринку | Кілька аналогів на великому ринку | Один аналог на великому ринку | Продукт не має аналогів на великому ринку |
| 3 | Ціна продукту значно вища за ціни аналогів | Ціна продукту дещо вища за ціни аналогів | Ціна продукту приблизно дорівнює цінам аналогів | Ціна продукту дещо нижче за ціни аналогів | Ціна продукту значно нижче за ціни аналогів |

Продовження таблиці 4.4

| 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|
| 4 | Технічні та споживчі властивості продукту значно гірші, ніж в аналогів | Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів | Технічні та споживчі властивості продукту на рівні аналогів | Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів | Технічні та споживчі властивості продукту значно кращі, ніж в аналогів |
| 5 | Експлуатаційні витрати значно вищі, ніж в аналогів | Експлуатаційні витрати дещо вищі, ніж в аналогів | Експлуатаційні витрати на рівні експлуатаційних витрат аналогів | Експлуатаційні витрати трохи нижчі, ніж в аналогів | Експлуатаційні витрати значно нижчі, ніж в аналогів |
| Ринкові перспективи | | | | | |
| 6 | Ринок малий і не має позитивної Динаміки | Ринок малий, але має позитивну динаміку | Середній ринок з позитивною динамікою | Великий стабільний ринок | Великий ринок з позитивною динамікою |
| 7 | Активна конкуренція великих компаній на ринку | Активна Конкуренція | Помірна конкуренція | Незначна конкуренція | Конкурентів немає |
| 8 | Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї | Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців | Необхідне незначне навчання фахівців та збільшення їх штату | Необхідне незначне навчання фахівців | Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї |
| 9 | Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні | Потрібні незначні фінансові ресурси. Джерела фінансування відсутні | Потрібні значні фінансові ресурси. Джерела фінансування є | Потрібні незначні фінансові ресурси. Джерела фінансування є | Не потребує додаткового фінансування |
| 10 | Необхідна розробка нових матеріалів | Потрібні матеріали, що використовуються у військово-промисловому комплексі | Потрібні дорогі матеріали | Потрібні досяжні та дешеві матеріали | Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві |
| 11 | Термін реалізації ідеї більший за 10 років | Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років | Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років | Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років | Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років |
| 12 | Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту | Необхідно отримання великої к-сті дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу | Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу | Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту | Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту |

Результати оцінювання науково-технічного рівня та комерційного потенціалу науково-технічної розробки потрібно зведемо до таблиці 4.5.

Таблиця 4.5 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки

| Критерії | Експерт | | |
|---|---|---|---|
| | 1 | 2 | 3 |
| | Бали: | | |
| 1. Технічна здійсненність концепції | 5 | 3 | 4 |
| 2. Ринкові переваги (наявність аналогів) | 4 | 3 | 3 |
| 3. Ринкові переваги (ціна продукту) | 3 | 2 | 4 |
| 4. Ринкові переваги (технічні властивості) | 4 | 4 | 2 |
| 5. Ринкові переваги (експлуатаційні витрати) | 5 | 5 | 3 |
| 6. Ринкові перспективи (розмір ринку) | 3 | 3 | 2 |
| 7. Ринкові перспективи (конкуренція) | 4 | 5 | 2 |
| 8. Практична здійсненність (наявність фахівців) | 3 | 3 | 2 |
| 9. Практична здійсненність (наявність фінансів) | 3 | 4 | 3 |
| 10. Практична здійсненність (необхідність нових матеріалів) | 4 | 2 | 4 |
| 11. Практична здійсненність (термін реалізації) | 4 | 2 | 4 |
| 12. Практична здійсненність (розробка документів) | 3 | 3 | 4 |
| Сума балів | *СБ₁=45* | *СБ₂=39* | *СБ₃=37* |
| Середньоарифметична сума балів *СБ$_c$* | $СБ_c = \frac{\sum_{i=1}^{3} СБ_1}{3}=40$ | | |

За результатами розрахунків, наведених в таблиці 4.5, зробимо висновок щодо науково-технічного рівня і рівня комерційного потенціалу розробки. При цьому використаємо рекомендації, наведені в табл. 4.6.

Таблиця 4.6 – Науково-технічні рівні та комерційні потенціали розробки

| Середньоарифметична сума балів СБ, розрахована на основі висновків експертів | Науково-технічний рівень та комерційний потенціал розробки |
|---|---|
| 41…48 | Високий |
| 31…40 | Вищий середнього |
| 21…30 | Середній |
| 11…20 | Нижчий середнього |
| 0…10 | Низький |

Згідно проведених досліджень рівень комерційного потенціалу розробки становить 40 балів, що, згідно таблиці 4.6, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки вище

середнього. Дане програмне забезпечення удосконалює метод пошуку прихованої інформації в цифрових зображеннях на основі штучної нейронної мережі. Однією з практичних функцій на відмінну від вже існуючих аналогів є удосконалення методу пошуку прихованої інформації в цифрових зображеннях за рахунок використання розробленої архітектури згорткової нейронної мережі.

### 4.3 Розрахунок витрат на здійснення науково-дослідної роботи

Витрати, пов’язані з проведенням науково-дослідної, дослідно-конструкторської, конструкторсько-технологічної роботи, створенням дослідного зразка і здійсненням виробничих випробувань, під час планування, обліку і калькулювання собівартості науково-дослідної роботи групуються за такими статтями:

– витрати на оплату праці;

– відрахування на соціальні заходи;

– паливо та енергія для науково-виробничих цілей;

– витрати на службові відрядження;

– спецустаткування для наукових (експериментальних) робіт;

– програмне забезпечення для наукових (експериментальних) робіт;

– витрати на роботи, які виконують сторонні підприємства, установи і організації;

– інші витрати;

– накладні (загальновиробничі) витрати.

До статті «Витрати на оплату праці» належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно-технічним працівникам, конструкторам, технологам, креслярам, копіювальникам, лаборантам, робітникам, студентам, аспірантам та іншим працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифними ставками згідно з чинними в організаціях системами оплати праці, також будь-

які види грошових і матеріальних доплат, які належать до елемента «Витрати на оплату праці».

Основна заробітна плата дослідників

Витрати на основну заробітну плату дослідників (Зо) розраховують відповідно до посадових окладів працівників, за формулою:

$$З_о = \sum_{i=1}^{k} \frac{М_{ni} \cdot t_i}{Т_р}, \quad (4.3)$$

де $k$ – кількість посад дослідників, залучених до процесу досліджень;

$M_{ni}$ – місячний посадовий оклад конкретного дослідника, грн;

$t_i$ – кількість днів роботи конкретного дослідника, дн.;

$T_p$ – середня кількість робочих днів в місяці, $T_p$=*21…23* дні.

Проведені розрахунки зведемо до таблиці 4.7.

Таблиця 4.7 – Витрати на заробітну плату дослідників

| Найменування посади | Місячний посадовий оклад, грн | Оплата за робочий день, грн | Кількість днів роботи | Витрати на заробітну плату, грн |
|---|---|---|---|---|
| Керівник відділу | 10 200 | 443 | 30 | 13 290 |
| Науковий працівник | 8 800 | 382 | 30 | 11 460 |
| Аспірант | 6 300 | 274 | 30 | 8 220 |
| Всього | | | | 32 970 |

Витрати на основну заробітну плату робітників ($З_р$) за відповідними найменуваннями робіт розраховують за формулою:

$$З_р = \sum_{i=1} C_i \cdot t_i, \quad (4.4)$$

де $C_i$ – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год;

$t_i$ – час роботи робітника на виконання певної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду Сі можна визначити за формулою:

$$С_і = \frac{М_м \cdot К_і \cdot К_с}{Т_р \cdot Т_{зм}}, \quad (4.5)$$

де $M_M$ – розмір прожиткового мінімуму працездатної особи або мінімальної місячної заробітної плати (залежно від діючого законодавства), грн;

$K_i$ – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду;

*Кс* – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих об'єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати.

$T_p$ – середня кількість робочих днів в місяці, приблизно *Тр* = 21…23 дні;

$t_{зм}$ – тривалість зміни, год.

$$\text{С}_\text{п} = \frac{6500 \cdot 1{,}8 \cdot 2}{22 \cdot 8} = 153{,}21 \text{ грн} \quad (4.6)$$

Таблиця 4.8 – Величина витрат на основну заробітну плату робітників

| Найменування робіт | Тривалість роботи, год | Розряд роботи | Тарифний коефіцієнт | Погодинна тарифна ставка, грн | Величина оплати на робітника грн |
|---|---|---|---|---|---|
| Front end розробник | 176 | 6 | 2 | 133 | 23 400 |
| Back end розробник | 176 | 6 | 2 | 133 | 23 400 |
| Системний аналітик | 176 | 5 | 1,7 | 113 | 19 890 |
| Qa тестувальник | 176 | 4 | 1,5 | 100 | 17 550 |
| Всього | | | | | 84240 |

Додаткова заробітна плата розраховується як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$\text{З}_\text{дод} = \text{З}_\text{о} + \text{З}_\text{р} \cdot \frac{\text{Н}_\text{дод}}{100\%}, \quad (4.7)$$

де $H_{дод}$ – норма нарахування додаткової заробітної плати.

$$\text{З}_\text{дод} = (84240 + 32970) \cdot 0{,}1 = 11\,721 \quad (4.8)$$

До статті «Відрахування на соціальні заходи» належать відрахування внеску на загальнообов'язкове державне соціальне страхування та для здійснення заходів щодо соціального захисту населення (ЄСВ – єдиний соціальний внесок).

Нарахування на заробітну плату дослідників та робітників розраховується як 22% від суми основної та додаткової заробітної плати дослідників і робітників за формуло:

$$З_н = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%}, \quad (4.9)$$

де $H_{зп}$ – норма нарахування на заробітну плату.

$$З_{дод} = (84\,240 + 32\,970 + 11\,721) \cdot 0{,}22 = 28\,364{,}82 \text{ грн.} \quad (4.10)$$

Спецустаткування для наукових (експериментальних) робіт

Вартість спецустаткування визначається за прейскурантом гуртових цін або за даними базових підприємств за відпускними і договірними цінами.

До балансової вартості устаткування окрім прейскурантної вартості входять витрати на його транспортування і монтаж, тому ці витрати беруться додатково в розмірі 10…12% від вартості устаткування.

Балансову вартість спецустаткування розраховують за формулою:

$$B_{спец} = \sum_{i=1}^{k} Ц_і \cdot С_{пр.і} \cdot К_і \quad (4.11)$$

де $Ц_i$ – ціна придбання одиниці спецустаткування даного виду, марки, грн;

$C_{пр}$ – кількість одиниць устаткування відповідного найменування, які придбані для проведення досліджень, шт.

$K_i$ – коефіцієнт, що враховує доставку, монтаж, налагодження устаткування тощо, ($K_i$ =1,10 ….1,12)

$k$ = кількість найменувань устаткування

Таблиця 4.9 – Витрати на придбання спецустаткування по кожному виду

| Найменування устаткування | Кількість, шт | Ціна за одиницю, грн | Вартість, грн |
|---|---|---|---|
| Комп'ютер | 4 | 27 000 | 108 000 |
| Принтер | 1 | 2 200 | 2 000 |
| Факс | 1 | 3 500 | 3 500 |
| Всього | | | 113 500 |

Витрати на придбання спецустаткування становитимуть:

$$В_{спец} = 113\,500 \cdot 1{,}1 = 124\,850 \text{ грн.} \quad (4.12)$$

де $Ц_{інпрг}$ – ціна придбання одиниці програмного засобу цього виду, грн;

$С_{прг}$ - кількість одиниць програмного забезпечення відповідного найменування, які придбані для проведення досліджень, шт.;

$К_і$ – коефіцієнт, що враховує інсталяцію, налагодження програмного засобу тощо, ($К_і$ = 1,10…1,12);

$k$ – кількість найменувань програмних засобів. Отримані результати зведемо до таблиці 4.10

Таблиця 4.10 – Витрати на придбання програмних засобів по кожному виду

| Найменування програмного засобу | Кількість, шт | Ціна за одиницю, грн | Вартість, грн |
|---|---|---|---|
| Операційна система Windows | 4 | 3 800 | 15 200 |
| Microsoft Visual Studio 2022 Professional | 4 | 3 300 | 13 200 |
| Всього | | | 28 400 |

До балансової вартості програмного забезпечення входять витрати на його інсталяцію, тому ці витрати беруться додатково в розмірі 10…12% від вартості програмного забезпечення.

Витрати на придбання програмних засобів становитимуть:

$$В_{прог} = 28\,400 \cdot 1{,}1 = 31\,240 \text{ грн.} \quad (4.13)$$

Амортизація обладнання, комп’ютерів та приміщень А, які використовувалися під час (чи для) виконання даного етапу роботи розраховуються за формулою:

$$А = \frac{Ц}{Тв} \cdot \frac{tk}{12} \quad (4.14)$$

де *Ц* – загальна балансова вартість всього обладнання, комп'ютерів, приміщень тощо, що використовувались для виконання даного етапу роботи, грн.;

$T_в$ – термін корисного використання, роки;

$t_k$ – термін використання обладнання, приміщень тощо, місяці.

Розрахунки амортизаційних витрат для даного програмного забезпечення зведені в табл. 4.11.

Таблиця 4.11 – Розрахунок амортизаційних відрахувань

| Найменування програмного забезпечення | Кількість | Балансова вартість, грн. | Термін корисного використання, роки | Термін використання, міс. | Величина амортизаційних відрахувань, грн. |
|---|---|---|---|---|---|
| Комп'ютер | 4 | 27 000 | 3 | 2 | 6 000 |
| Принтер | 1 | 2 200 | 3 | 2 | 122 |
| Факс | 1 | 3 500 | 3 | 2 | 194 |
| Операційна система Windows | 4 | 3 800 | 1 | 2 | 2 533 |
| Microsoft Visual Studio 2022 Professional | 4 | 3 300 | 1 | 2 | 2 200 |
| | Всього: | | | | 11 049 |

Витрати на силову електроенергію ($B_e$) розраховують за формулою:

$$В_е = В \cdot П \cdot Ф \cdot К_П, \quad (4.15)$$

де *В* – вартість 1 кВт-години електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії);

*П* – встановлена потужність обладнання, кВт;

*Ф* – фактична кількість годин роботи обладнання, годин.

$К_п$ – коефіцієнт, що враховує використання потужності, $К_{вп і}$ <1;

$$В_е = 2{,}3 \cdot 0{,}9 \cdot 8 \cdot 0{,}7 = 11{,}59 \text{ грн.}, \quad (4.16)$$

Проведені розрахунки занесемо до таблиці 4.12.

Таблиця 4.12 – Витрати на електроенергію

| Найменування операції | Найменування обладнання | Встановлена потужність, кВт | Тривалість операції, год | Кількість днів | Сума, грн |
|---|---|---|---|---|---|
| Front end програмування | Комп’ютер | 0,9 | 8 | 30 | 347,7 |
| Back end програмування | Комп’ютер | 0,9 | 8 | 30 | 347,7 |
| Системний аналіз | Комп’ютер | 0,9 | 8 | 30 | 347,7 |
| Qa тестування | Комп’ютер | 0,9 | 8 | 30 | 347,7 |
| Друк, сканування | Принтер | 0,78 | 2 | 30 | 75,35 |
| Факс | Листуваня | 0,65 | 2 | 30 | 62,79 |
| | | | Всього | | 1 528,94 |

Інші витрати

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «Інші витрати» розраховуються як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$I_в = (З_о + З_р) \cdot \frac{Н_{зп}}{100\%}, \quad (4.17)$$

де $H_{ів}$ – норма нарахування за статтею «Інші витрати».

Інші витрати складатимуть:

$$I_в = (84240 + 32970) * 0{,}5 = 58605 \text{ грн.} \quad (4.18)$$

Витрати на проведення науково-дослідної роботи розраховуються як сума всіх попередніх статей витрат за формулою:

$$В_{заг} = З_о + З_р + З_{дод} + З_н + В_{спец} + В_{спец} + В_{прг} + А_{обл} + В_е + I_В, \quad (4.19)$$

$$В_{заг} = 84240 + 32970 + 11721 + 28364{,}82 + 124850 + 31240 + 11049 + 1528{,}94 + 58605 = 384568{,}76 \text{ грн}, \quad (4.20)$$

Загальні витрати ЗВ на завершення науково-дослідно (науково-технічної) роботи та оформлення її результатів розраховуються за формулою:

$$ЗВ = \frac{В_{заг}}{\mu}, \quad (4.21)$$

де ɲ – коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи. Так, якщо науково-технічна розробка знаходиться на стадії: науково-дослідних робіт, то μ =0,1; технічного проектування, то μ =0,2; розробки конструкторської документації, то μ =0,3; розробки технологій, то μ=0,4; розробки дослідного зразка, то μ =0,5; розробки промисло вого зразка, то μ=0,7; впровадження, то μ =0,9.

$$ЗВ = \frac{446454,9}{0,9} = 427298,6 \text{ грн.} \quad (4.22)$$

## 4.4 Розрахунок ефективності вкладених інвестицій та період їх окупності

У даному підрозділі кількісно спрогнозуємо, яку вигоду можна отримати у майбутньому від впровадження результатів виконаної наукової роботи. Розрахуємо збільшення чистого прибутку підприємства ΔПі, для кожного із років, протягом яких очікується отримання позитивних результатів від впровадження розробки, за формулою

$$\Delta П_i = \sum_{1}^{n} (\Delta Ц_0 \cdot N + Ц_0 \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\upsilon}{100}), \quad (4.23)$$

де $\Delta Ц_0$ – покращення основного оціночного показника від впровадження результатів розробки у даному році;

N - основний кількісний показник, який визначає діяльність підприємства у даному році до впровадження результатів наукової розробки;

ΔN – покращення основного кількісного показника діяльності підприємства від впровадження результатів розробки;

$Ц_o$ – основний оціночний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки;

*n* – кількість років, протягом яких очікується отримання позитивних

результатів від впровадження розробки:

$\lambda$ – коефіцієнт, який враховує сплату податку на додану вартість.

Ставка податку на додану вартість дорівнює 20%, а коефіцієнт $\lambda$ = 0,8333;

$p$ – коефіцієнт, який враховує рентабельність продукту, $p$ = 0,2;

$\upsilon$ – ставка податку на прибуток. У 2021 році — 18%.

Припустимо, що при прогнозованій ціні 9 000 грн. за послугу доступу до сервісу, в перший рік термін збільшення прибутку складе 3 роки. Після завершення розробки і її вдосконалення, можна буде підняти його ціну на 900 грн. Кількість наданих послуг також збільшиться: протягом першого року – на 500 шт., протягом другого року — на 350 шт., протягом третього року на 200 шт. До моменту впровадження результатів наукової розробки реалізації продукту не було:

$$\Delta П_1 = (0 * 500 + (9000 + 900) * 500 * 0{,}8333 * 0{,}2) * (1 - 0{,}18) = 676472{,}9 \text{ грн} \quad (4.24)$$

$$\Delta П_2 = (0 * 500 + (9000 + 900) * (500 + 350) * 0{,}8333 * 0{,}2) * (1 - 0{,}18) = 1150004 \text{ грн} \quad (4.25)$$

$$\Delta П_3 = (0 * 500 + (9000 + 900) * (500 + 350 + 200) * 0{,}8333 * 0{,}2) * (1 - 0{,}18) = 1420593{,}2 \text{ грн} \quad (4.26)$$

Отже, комерційний ефект від реалізації результатів розробки за три роки складе 3 247 070, 1 грн.

Далі розраховують приведену вартість збільшення всіх чистих прибутків ПП, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{1}^{T} \frac{\Delta П_i}{(1 + \tau)^t}, \quad (4.27)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої МКР, грн;

Т – період часу, протягом якого виявляються результати впровадженої МКР, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні; для України цей показник знаходиться на рівні 0,15;

t – період часу (в роках).

$$ПП = \frac{676472,9}{(1+0,15)^1} + \frac{1\ 150\ 004}{(1+0,15)^2} + \frac{1\ 420\ 593,2}{(1+0,15)^3} = \tag{4.28}$$

$$= 588237,3 + 869568,2 + 934063,08 = 2\ 391\ 868,58 \text{ грн.}$$

Далі розраховують величину початкових інвестицій PV , які потенційний інвестор має вкласти для впровадження і комерціалізації науково- технічної розробки.

Для цього можна використати формулу:

$$PV = k_{\text{інв}} \cdot ЗВ \tag{4.29}$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки. $k_{інв} = 2….5$

ЗВ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, грн.

$$PV = 2 \cdot 427298,6 = 854\ 597,2 \text{ грн} \tag{4.30}$$

Тоді абсолютний економічний ефект $Е_{абс}$ або чистий приведений дохід(NPV, Net Present Value) для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$Е_{фбс} = ПП - PV, \tag{4.31}$$

де ПП – приведена вартість всіх чистих прибутків, що їх отримає підприємство (організація) від реалізації результатів наукової розробки, грн.

$$Е_{фбс} = 2391868,58 - 854597,2 = 1537271,38 \tag{4.32}$$

Оскільки $Е_{абс} > 0$, то вкладання коштів на виконання та впровадження результатів роботи може бути доцільним.

Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $Е_в$ за формулою:

$$Е_в = \sqrt[Т_ж]{1 + \frac{Е_{абс}}{PV}} - 1, \tag{4.33}$$

де $E_{абс}$ – абсолютна ефективність вкладених інвестицій, грн.;

PV – теперішня вартість інвестицій PV = ЗВ, грн.;

$T_ж$ – життєвий цикл наукової розробки, роки.

$$E_в = \sqrt[3]{1 + \frac{1\,537\,271{,}38}{854\,597{,}2}} - 1 = 1{,}41 - 1 = 0{,}41 \quad (4.34)$$

Далі слід порівняти розраховану величину $E_в$ з мінімальною (бар'єрною) ставкою дисконтування $\tau_{min}$, яка визначає ту мінімальну дохідність, нижче за яку інвестиції вкладатися не будуть.

У загальному вигляді мінімальна (бар'єрна) ставка дисконтування визначається за формулою:

$$\tau_{min} = d + f, \quad (4.35)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2020 році в Україні d = (0,9...0,12);

f – показник, що характеризує ризикованість вкладень; зазвичай, величина f= (0,05...0,5), але може бути і значно більше.

$$\tau_{min} = 0{,}12 + 0{,}08 = 0{,}2 \quad (4.36)$$

Оскільки $E_в > \tau_{min}$, то інвестор буде зацікавлений у фінансуванні програмного забезпечення пошуку прихованої інформації в цифрових зображеннях на основі штучної нейронної мережі.

Термін окупності вкладених у реалізацію наукового проекту інвестицій $T_{ок}$ можна розрахувати за формулою:

$$T_{ок} = \frac{1}{E_в}, \quad (4.37)$$

Для програмного забезпечення пошуку прихованої інформації в цифрових зображеннях на основі штучної нейронної мережі термін окупності складе:

$$T_{ок} = \frac{1}{0{,}41} = 2{,}44 \text{ р.} \quad (4.38)$$

Оскільки $T_{ок}$ < 3-х років, можна зробити висновок, що фінансування даної наукової розробки буде доцільним.

## 4.5 Висновки до Розділу 4

У даному розділі було проаналізовано економічну доцільність розробки програмного забезпечення пошуку прихованої інформації в цифрових зображеннях на основі штучної нейронної мережі. Опитування експертів показало, що запропонована розробка має високий рівень комерційного потенціалу. Вкладання коштів на виконання та впровадження результатів МКР є доцільним. Дана розробка має високий рівень щорічної ефективності та термін окупності складає 2,44 р.

# ВИСНОВКИ

В даній роботі було проведено удосконалення методу пошуку прихованої інформації в цифрових зображеннях на основі штучної нейронної мережі.

У першому розділі розглянуто алгоритми приховування інформації для файлів формату JPEG, наведено порівняльну характеристику стійкості стеганографічних алгоритмів, здійснено огляд методів пошуку прихованої інформації в графічних зображеннях, наведено їхні переваги та недоліки, описано підхід до класифікації заснований на методу опорних векторів. Також була поставлена задача для проведення роботи.

У другому розділі було детально розглянуто напрямки удосконалення методу стеганоаналізу цифрових зображень за рахунок штучної нейронної мережі, сформовано етапи створення згорткової нейронної мережі та наведено алгоритм роботи програми.

У третьому розділі було проведено обґрунтування вибору мови програмування. Програмно реалізовано параметри згорткової нейронної мережі. Наведено інструкцію роботи з програмним продуктом та представлено аналіз результатів.

В результаті програмної реалізації удосконаленого методу пошуку прихованої інформації в цифрових зображеннях на основі штучної згорткової нейронної мережі були отримані наступні результати: порівняно з вручну підібраними наборами ознак SPAM ймовірність помилки виявлення стеганоконтейнеру є на 2-4% меншою, а також час навчання нейронної мережі для виявлення ознак наявності прихованої інформації для нового стеганографічного алгоритму в порівнянні з часом затраченим на ручний підбір ознак значно менший, різниця між цими часовими проміжками може вимірюватись в добах в залежності від складності стеганографічного алгоритму.

Отримані результати в ході написання магістерської кваліфікаційної роботи свідчать про те, що використання згорткових нейронних мереж для вдосконалення методу статистичного стеганоаналізу є актуальним та цінним для досліджень в цілому.

# ПЕРЕЛІК ВИКОРИСТАНИХ ПОСИЛАНЬ

1. Задірака В.К. Комп'ютерна криптологія : підручник / В.К. Задірака, О.С. Олексюк. – Київ, 2002. – 504 с.

2. E. Adelson. Digital Signal Encoding and Decoding Apparatus. – U.S. Patent. – No. 4,939515 (1990).

3. Fridrich J. Reliable Detection of LSB steganography in grayscale and color images / J. Fridrich, M. Goljan, R. Du // Proc. of the ACM, Special Session on Multimedia Security and Watermarking. – Ottawa, Canada, October 5, 2001. – P. 27-30.

4. Hinton, G. E. and Salakhutdinov, R. R., "Reducing the dimensionality of data with neural networks," Science 313(5786), 504–507 (2006).

5. Salakhutdinov, R. and Hinton, G. E., "Deep boltzmann machines," in [International Conference on Artificial Intelligence and Statistics], 448–455 (2009).

6. Larochelle, H., Bengio, Y., Louradour, J., and Lamblin, P., "Exploring strategies for training deep neural networks," The Journal of Machine Learning Research 10, 1–40 (2009).

7. LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P., "Gradient-based learning applied to document recognition," Proceedings of the IEEE 86(11), 2278– 2324 (1998).

8. Simmons G.J. The prisoner`s problem and the subliminal channel, Proc. Workshop on Communications Security (Crypto`83), 1984, 51-67.

9. Pfitzmann B. Information Hiding Terminology, in Information Hiding, Springer Lecture Notes in Computer Science, v.1174, 1996, 347-350. 7. T. Pevny, Steganalysis by subtractive pixel adjacency matrix / T. Pevny, P. Bas, J. Fridrich // IEEE Transactions on Information Forensics and Security, June 2010. – 5(2). – P. 215 – 224.

10. Pevny, T. Using high-dimensional image models to perform highly undetectable steganography // T. Pevny, T. Filler, P. Bas // Information Hiding, 12th International Workshop. – 2010. – LNCS 6387. – P. 161–177.

11. Cachin C. An Information-Theoretic Model for Steganography / C. Cachin // Proceeding of 2nd Workshop of Information Hiding. – USA, 1998. – May, 13. – 12 p.

12. Разинков Е.В. Стойкость стеганографических систем / Е.В. Разинков, Р.Х. Латыпов // Учёные записки Казан.

гос. ун-та. – Казань, 2009. – Т. 151, № 2. – С. 126-132.

13. Гизунов Д.С. Методика автоматизированного обнаружения скрытой информации в компьютерных файлах / Д.С. Гизунов, О. А. Демченко, Е.И. Никутин // Известия ТРТУ. – 2006. – Т. 71, № 16. – С. 49-53.

14. Girod B. The information theoretical significance of spatial and temporal masking in video signals / B. Girod Proc. Of the SPIE Human Vision, Visual Processing, and Digital Display. – 1989. – Vol. 1077. – P. 178-187.

15. Алиев А. Т. Оценка стойкости систем скрытой передачи информации / А. Т. Алиев, А. В. Балакин // Известия ТРТУ. Тематический выпуск. Материалы VII Международной научно-практической конференции «Информационная безопасность». – Таганрог: Изд-во ТРТУ, 2005. – № 4 (48). – С. 199-204.

16. Алиев А. Т. О применении стеганографического метода LSB к графическим файлам с большими областями монотонной заливки / А.Т. Алиев // Вестник ДГТУ. – Ростов-на-Дону, 2004. – Т. 4, № 4 (22). – С. 454-460.

17. Швидченко И.В. Анализ криптостеганографических алгоритмов / И.В. Швидченко // Проблемы управления и информатики. – 2007. – № 4. – С. 149-155.

18. Грибунин В.Г. Цифровая стеганография / Грибунин В.Г., Оков И.Н., Туринцев И.В. – М.: Солон-Пресс, 2002. – 272 с.

19. Барсуков В.С. Оценка уровня скрытности мультимедийных стеганографических каналов хранения и передачи информации / В.С. Барсуков, А.П. Романцов // Специальная Техника. – 2000. – № 1.

20. Кустов В.Н., Параскевопуло А.Ю. Простые тайны стегоанализа / В.Н. Кустов, А.Ю. Параскевопуло // Защита информации, INSIDE. – 2005. – № 4. – С. 72-78.

21. Голуб В.А. Комплексный подход для выявления стеганографического скрытия в JPEG-файлах / В.А. Голуб, М.А. Дрюченко // Инфокоммуникационные технологии. – 2009. – Т. 7, № 1. – С. 44-50.

22. Иванов М. А. Теория, применение и оценка качества генераторов псевдослучайных последовательностей / М. А. Иванов, И.В. Чугунков. – М.: КУДИЦ-ОБРАЗ, 2003.

23. Westfeld A. Attacks on Steganographic Systems: Breaking the Steganographic Utilities EzStego, Jsteg, Steganos and STools-and Some Lessons Learned / A. Westfeld, A. Pfitzmann // 3rd International Workshop on Information Hiding (2000).

24. Дрюченко М.А. Алгоритмы выявления стеганографического скрытия информации в jpeg-файлах / М.А. Дрюченко // Вест. Воронеж. гос. ун. Системный анализ и информационные технологии. – 2007. – № 1. – С. 21-30.

25. Fridrich J. Steganalysis of JPEG Images: breaking the / J. Fridrich, M. Goljan, D. Hogea // F5 algorithm, 5th Information Hiding Workshop, Noordwijkerhout, The Netherlands, 7 – 9 October 2002. – Режим доступа: http://www.ws.binghamton. edu/fridrich/Research/f5.pdf

26. How to Implement Artificial Intelligence for Solving Image Processing Tasks [Електронний ресурс] – Режим доступу до ресурсу: https://www.apriorit.com/dev-blog/599-ai-for-image-processing

27. Boroumand M, Chen M, Fridrich J. Deep Residual Network for Steganalysis of Digital Images. IEEE Transactions on Information Forensics and Security, vol. 14, issue 5, 2019, pp. 1181-1193.

28. Kodovsky J, Fridrich J. Rich models for steganalysis of digital images. IEEE Transactions on Information Forensics and Security, vol. 7, issue 3, 2012, pp. 868-882.

29. Yedroudj M, Comby F, Chaumont M. Yedrouj-Net: An Efficient CNN for Spatial Steganalysis. In Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing, 2018, pp. 2092-2096.

30. Lerch-Hostalot D, Megias D. Detection of Classiffer Inconsistencies in Image Steganalysis. In Proc. of the ACM Workshop on Information Hiding and Multimedia Security. 2019, pp.222 - 229.

31. Свёрточная нейронная сеть с нуля. Часть 0. Введение [Электронный ресурс] – Режим доступа: URL: https://programforyou.ru/poleznoe/convolutional-network-from-scratch-part-zero-introduction.

32. Shafkat I. Intuitively Understanding Convolutions for Deep Learning. URL: https://towardsdatascience.com/intuitively-understanding-convolutions-for-deep-learning-1f6f42faee1.

33. Saha S. A. Comprehensive Guide to Convolutional Neural Networks – the ELI5 way. URL: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way3bd2b1164a53.

34. Шелухин, О. И. Стеганография. Алгоритмы и программная реализация. / О. И. Ше-лухин, С. Д. Канаев; под ред. профессора О. И. Шелухина, – Горячая линия - Телеком, 2017. – 592 с.

35. Грибунин, В. Г. Цифровая стеганогра-фия / В. Г. Грибунин, И. Н. Оков, И. В. Туринцев – Москва: Солон-Пресс, 2002. – 272 с.

36. Конахович, Г. Ф. Принципы стегано-графического анализа / Г. Ф. Конахович, А. Ю. Пузыренко // Компьютерная стеганография. Теория и практика. – Москва: МК-Пресс, 2006. – 288 с.

37. Czaplewski, Bartosz. Current trends in the field of steganalysis and guidelines for constructions of new steganalysis schemes / Bartosz Czaplewski // Przegląd Telekomunikacyjny + Wiadomości Tele-komunikacyjne. – 2017. – No 10. – P. 1121–1125. – DOI: 10.15199/59.2017.10.3.

38. Валишин, М. Ф. Повышение эффектив-ности методов противодействия встраиванию скрытой информации в графические файлы / М. Ф. Валишин. – Ульяновск, 2015. – 106 с.

39. Lyu, S. Detecting messages using higher-order statistics and support vector machines [Электронный ресурс] / S. Lyu, H. Farid // Режим доступа: http://hackerzvoice.net/madchat/crypto/stegano/ih02.pdf.

40. Lyu, S. Steganalysis using color wavelet statistics and one-class support vector machines [Электронный ресурс] / S. Lyu, H. Farid // DOI: 10.1117/12.526012. Режим доступа: http://www.ists.dartmouth.edu/library/34.pdf.

41. Pe v ny, T. Steganalysis by subtractive pixel adjacency matrix / T. Pevny, P. Bas, J. Fridrich // IEEE Trans. Information Forensics and Security. – 2010. – V. 5, No 2. – P. 215–224. – DOI: 10.1109/TIFS.2010.2045842.

42. Fridrich, J. Rich models for steganalysis of digital images / J. Fridrich // IEEE Trans. Inform. Forensics and Security. – 2012. – V. 7, No 3. – P. 868–882. – DOI: 10.1109/TIFS.2012.2190402.

43. Holub, V. Random projections of resid-uals for digital image steganalysis /V. Holub, J. Fridrich // IEEE Trans. Inform. Forensics and Security. – 2013.–V. 8, No 12. – P. 1996–2006. – DOI: 10.1109/TIFS.2013.2286682.

44. Bas, P. “Break our steganographic system” the ins and outs of organizing BOSS / P. Bas, T. Filler, T. Pevny // LNCS. – 2011. – V. 6958. – P. 59–70. – DOI: 10.1007/978-3-642-24178-9_5.

45. Pevny, T. Using high-dimensional im-age models to perform highly undetectable steg- anography / T. Pevn´y, P. Bas, T. Filler // LNCS. – 2010. – V. 6387. – P. 161–177.

46. Нагорный, Н. А. Исследование алгоритмов стегоанализа изображений с использованием глубоких нейронных сетей / Н. А. Нагорный А. А. Сирота // Сборник студенческих научных работ факультета компьютерных наук ВГУ. – 2019. – Часть 2. – С.145–151.

47. Полунин, А. А. Использование аппарата сверточных нейронных сетей для стегоанализа цифровых изображений / А. А. Полунин, Э. А. Яндашевская // Труды ИСП РАН. – 2020. – Том 32, вып. 4. – С. 155–164. – DOI: 10.15514/ISPRAS2020 – 32 (4) - 11.

48. Ranzato, M., Huang, F. J., Boureau, Y.-L., and LeCun, Y., “Unsupervised learning of invariant feature hierarchies with applications to object recognition,” in [IEEE Conference on Computer Vision and Pattern Recognition], 1–8, IEEE (2007).

49. Описание библиотеки cuda-convnet2 / [Электронный ресурс] / Режим доступа: https://github.com/akrizhevsky/cuda-convnet2

50. Larochelle, H., Bengio, Y., Louradour, J., and Lamblin, P., “Exploring strategies for training deep neural networks,” The Journal of Machine Learning Research 10, 1–40 (2009).

51. Ciresan, D. C., Meier, U., Masci, J., Maria Gambardella, L., and Schmidhuber, J., “Flexible, high performance convolutional neural networks for image classification,” in [IJCAI Proceedings-International Joint Conference on Artificial Intelligence], 22(1), 1237 (2011).

52. Browne, M. and Ghidary, S. S., “Convolutional neural networks for image processing: an application in robot vision,” in [AI 2003: Advances in Artificial Intelligence], 641–652, Springer (2003).

53. Boureau, Y.-L., Ponce, J., and LeCun, Y., “A theoretical analysis of feature pooling in visual recognition,” in [Proceedings of the 27th International Conference on Machine Learning (ICML-10)], 111–118 (2010).

54. Lee, H., Grosse, R., Ranganath, R., and Ng, A. Y., “Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations,” in [Proceedings of the 26th Annual International Conference on Machine Learning], 609–616, ACM (2009).

55. Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R., “Improving neural networks by preventing co-adaptation of feature detectors,” arXiv preprint arXiv:1207.0580 (2012).

56. Описание библиотеки opennn / [Электронный ресурс] / Режим доступа: http://www.opennn.net/

57. Описание библиотеки fann / [Электронный ресурс] / Режим доступа: http://leenissen.dk/fann/wp/

58. Описание библиотеки convnet / [Электронный ресурс] / Режим доступа: http://conv-net.sourceforge.net/doc/

59. Описание библиотеки alglib / [Электронный ресурс] / Режим доступа: http://www.alglib.net/

60. Описание библиотеки cuda-convnet2 / [Электронный ресурс] / Режим доступа: https://github.com/akrizhevsky/cuda-convnet2

61. Описание библиотеки pybrain / [Электронный ресурс] / Режим доступа: http://pybrain.org/

62. Описание библиотеки theano / [Электронный ресурс] / Режим доступа: http://deeplearning.net/software/theano/

63. Описание библиотеки pylearn2 / [Электронный ресурс] / Режим доступа: http://deeplearning.net/software/pylearn2/

64. Описание библиотеки caffe / [Электронный ресурс] / Режим доступа: http://caffe.berkeleyvision.org/

65. Описание библиотеки gtk+ / [Электронный ресурс] / Режим доступа: https://www.gtk.org/

66. Nair, V. and Hinton, G. E., “Rectified linear units improve restricted boltzmann machines,” in [Proceedings of the 27th International Conference on Machine Learning (ICML-10)], 807–814 (2010)

67. Pevny, T., Bas, P., and Fridrich, J., “Steganalysis by subtractive pixel adjacency matrix,” IEEE Transactions on information Forensics and Security 5(2), 215–224 (2010).

68. Fridrich, J. and Kodovsky, J., “Rich models for steganalysis of digital images,” IEEE Transactions on Information Forensics and Security 7(3), 868–882 (2012).

69. Extractors for Steganalysis / [Електронний ресурс] / Режим доступу: http://dde.binghamton.edu/download/feature_extractors/

70. Наиболее часто используемый алгоритм для глубокого обучения: алгоритм оптимизации Адама / [Електронний ресурс] / Режим доступу: https://russianblogs.com/article/4255252931/

# ДОДАТКИ

**Додаток А. Технічне завдання**

Вінницький національний технічний університет

Факультет менеджменту та інформаційної безпеки

Кафедра менеджменту та безпеки інформаційних систем

**ЗАТВЕРДЖУЮ**

Голова секції “Управління інформаційною
безпекою” кафедри МБІС
д.т.н., професор
_____Ю.Є.Яремчук
“24” вересня 2021 р.

# ТЕХНІЧНЕ ЗАВДАННЯ

до магістерської кваліфікаційної роботи на тему:
«Удосконалення методу пошуку прихованої інформації в цифрових зображеннях на основі штучної нейронної мережі»

08-42.МКР.010.00.000.ТЗ

Керівник магістерської кваліфікаційної роботи

Голова секції УБ кафедри МБІС д.т.н., проф.

_____Яремчук Ю.Є.

Вінниця – 2021 р.

**1. Найменування та область застосування**

Програмний модуль для статистичного стеганоаналізу на основі глибокого навчання, реалізуючий удосконалений за допомогою згорткової нейронної мережі метод пошуку стеганоконтейнерів.

**2.Підстава для розробки**

Розробка виконується на основі наказу ректора ВНТУ №277 від 24.09.2021 р.

**3. Мета та призначення розробки**

3.1 Мета розробки: програмно реалізувати удосконалений метод пошуку прихованої інформації в цифрових зображеннях за рахунок штучної згорткової нейронної мережі.

3.2 Призначення: розроблений програмний засіб здійснює пошук стеганоконтейнерів.

**4. Джерела розробки**

4.1. Fridrich J. Reliable Detection of LSB steganography in grayscale and color images / J. Fridrich, M. Goljan, R. Du // Proc. of the ACM, Special Session on Multimedia Security and Watermarking. – Ottawa, Canada, October 5, 2011. – P. 27-30.

4.2. Hinton, G. E. and Salakhutdinov, R. R., “Reducing the dimensionality of data with neural networks,” Science, 2019 p., 504–507.

4.3. Salakhutdinov, R. and Hinton, G. E., “Deep boltzmann machines,” in [International Conference on Artificial Intelligence and Statistics], 2015 p., 448–455.

4.4. Larochelle, H., Bengio, Y., Louradour, J., and Lamblin, P., “Exploring strategies for training deep neural networks,” The Journal of Machine Learning Research 10, 2019 p., 1–40.

**5. Вимоги до програми**

5.1 Вимоги до функціональних характеристик:

5.1.1 Програмний засіб повинен мати зручний, легкий у використанні інтерфейс користувача;

5.1.2 Реалізація методу не повинна вимагати спеціальних ліцензійних програмних додатків;

5.1.3 Програма має працювати без помилок, а у разі їх виникнення інформувати користувача.

5.2 Вимоги до надійності:

5.2.1 Програмний засіб повинен працювати без помилок, у випадку виникнення критичних ситуацій необхідно передбачити виведення відповідних повідомлень;

5.2.2 Програмний засіб повинен виконувати свої функції.

5.3 Вимоги до складу і параметрів технічних засобів:

– оперативна пам’ять – не менше 512 Mb;

– середовище функціонування – операційна система сімейство Windows;

– вимоги до техніки безпеки при роботі з програмою повинні відповідати існуючим вимогам та стандартам з техніки безпеки при користуванні комп’ютерною технікою.

**6. Вимоги до програмної документації**

6.1 Обов’язкова поетапна інструкція для майбутніх користувачів, наведена у пункті 3.3

**7. Вимоги до технічного захисту інформації**

7.1 Необхідно забезпечити захист розроблюваного програмного засобу від несанкціонованого використання.

7.2 Необхідно забезпечити збереження отриманих результатів.

**8. Техніко-економічні показники**

8.1 Цінність результатів використання даного проекту повинна перевищувати витрати на його реалізацію.

8.2 Має бути реалізований таким чином, щоб підходити для використання широкого загалу.

## 9. Стадії та етапи розробки

| № | Назва та зміст етапу | Термін виконання | | Примітка |
|---|---|---|---|---|
| | | початок | закінчення | |
| 1 | Аналіз предметної області обраної теми | 27.09.2021 | 10.10.2021 | |
| 2 | Розробка алгоритму роботи | 11.10.2021 | 31.10.2021 | |
| 3 | Написання магістерської кваліфікаційної роботи на основі розробленої теми | 01.11.2021 | 24.11.2021 | |
| 4 | Передзахист магістерської кваліфікаційної роботи | 25.11.2021 | 26.11.2021 | |
| 5 | Виправлення, уточнення, корегування магістерської кваліфікаційної роботи | 27.11.2021 | 16.12.2021 | |
| 6 | Захист магістерської кваліфікаційної роботи | 20.12.2021 | 20.12.2021 | |

## 10. Порядок контролю та прийому

10.1 До приймання магістерської кваліфікаційної роботи надається:

– ПЗ до магістерської кваліфікаційної роботи;

– програмний додаток;

– презентація;

– відзив керівника роботи;

– відзив рецензента

Технічне завдання до виконання прийняв _____ Шевченко М.В.

## Додаток Б. Лістинг програми

```
from CNN.forward import *
from CNN.backward import *
from CNN.utils import *

import numpy as np
import pickle
from tqdm import tqdm

def conv(image, label, params, conv_s, pool_f, pool_s):

    [f1, f2, w3, w4, b1, b2, b3, b4] = params

    conv1 = convolution(image, f1, b1, conv_s)
    conv1[conv1<=0] = 0

    conv2 = convolution(conv1, f2, b2, conv_s)
    conv2[conv2<=0] = 0

    pooled = maxpool(conv2, pool_f, pool_s)

    (nf2, dim2, _) = pooled.shape
    fc = pooled.reshape((nf2 * dim2 * dim2, 1))

    z = w3.dot(fc) + b3
    z[z<=0] = 0

    out = w4.dot(z) + b4

    probs = softmax(out)

    loss = categoricalCrossEntropy(probs, label)

    dout = probs - label
    dw4 = dout.dot(z.T)
    db4 = np.sum(dout, axis = 1).reshape(b4.shape)

    dz = w4.T.dot(dout)
    dz[z<=0] = 0
    dw3 = dz.dot(fc.T)
    db3 = np.sum(dz, axis = 1).reshape(b3.shape)

    dfc = w3.T.dot(dz)
    dpool = dfc.reshape(pooled.shape)

    dconv2 = maxpoolBackward(dpool, conv2, pool_f, pool_s)
    dconv2[conv2<=0] = 0

    dconv1, df2, db2 = convolutionBackward(dconv2, conv1, f2, conv_s)
```

```
    dconv1[conv1<=0] = 0

    dimage, df1, db1 = convolutionBackward(dconv1, image, f1, conv_s)

    grads = [df1, df2, dw3, dw4, db1, db2, db3, db4]

    return grads, loss

def adamGD(batch, num_classes, lr, dim, n_c, beta1, beta2, params, cost):
    '''
    update the parameters through Adam gradient descnet.
    '''
    [f1, f2, w3, w4, b1, b2, b3, b4] = params

    X = batch[:,0:-1]
    X = X.reshape(len(batch), n_c, dim, dim)
    Y = batch[:,-1]

    cost_ = 0
    batch_size = len(batch)

    df1 = np.zeros(f1.shape)
    df2 = np.zeros(f2.shape)
    dw3 = np.zeros(w3.shape)
    dw4 = np.zeros(w4.shape)
    db1 = np.zeros(b1.shape)
    db2 = np.zeros(b2.shape)
    db3 = np.zeros(b3.shape)
    db4 = np.zeros(b4.shape)

    v1 = np.zeros(f1.shape)
    v2 = np.zeros(f2.shape)
    v3 = np.zeros(w3.shape)
    v4 = np.zeros(w4.shape)
    bv1 = np.zeros(b1.shape)
    bv2 = np.zeros(b2.shape)
    bv3 = np.zeros(b3.shape)
    bv4 = np.zeros(b4.shape)

    s1 = np.zeros(f1.shape)
    s2 = np.zeros(f2.shape)
    s3 = np.zeros(w3.shape)
    s4 = np.zeros(w4.shape)
    bs1 = np.zeros(b1.shape)
    bs2 = np.zeros(b2.shape)
    bs3 = np.zeros(b3.shape)
    bs4 = np.zeros(b4.shape)

    for i in range(batch_size):
```

```
    x = X[i]
    y = np.eye(num_classes)[int(Y[i])].reshape(num_classes, 1)

    grads, loss = conv(x, y, params, 1, 2, 2)
    [df1_, df2_, dw3_, dw4_, db1_, db2_, db3_, db4_] = grads

    df1+=df1_
    db1+=db1_
    df2+=df2_
    db2+=db2_
    dw3+=dw3_
    db3+=db3_
    dw4+=dw4_
    db4+=db4_

    cost_+= loss

v1 = beta1*v1 + (1-beta1)*df1/batch_size
s1 = beta2*s1 + (1-beta2)*(df1/batch_size)**2
f1 -= lr * v1/np.sqrt(s1+1e-7)

bv1 = beta1*bv1 + (1-beta1)*db1/batch_size
bs1 = beta2*bs1 + (1-beta2)*(db1/batch_size)**2
b1 -= lr * bv1/np.sqrt(bs1+1e-7)

v2 = beta1*v2 + (1-beta1)*df2/batch_size
s2 = beta2*s2 + (1-beta2)*(df2/batch_size)**2
f2 -= lr * v2/np.sqrt(s2+1e-7)

bv2 = beta1*bv2 + (1-beta1) * db2/batch_size
bs2 = beta2*bs2 + (1-beta2)*(db2/batch_size)**2
b2 -= lr * bv2/np.sqrt(bs2+1e-7)

v3 = beta1*v3 + (1-beta1) * dw3/batch_size
s3 = beta2*s3 + (1-beta2)*(dw3/batch_size)**2
w3 -= lr * v3/np.sqrt(s3+1e-7)

bv3 = beta1*bv3 + (1-beta1) * db3/batch_size
bs3 = beta2*bs3 + (1-beta2)*(db3/batch_size)**2
b3 -= lr * bv3/np.sqrt(bs3+1e-7)

v4 = beta1*v4 + (1-beta1) * dw4/batch_size
s4 = beta2*s4 + (1-beta2)*(dw4/batch_size)**2
w4 -= lr * v4 / np.sqrt(s4+1e-7)

bv4 = beta1*bv4 + (1-beta1)*db4/batch_size
bs4 = beta2*bs4 + (1-beta2)*(db4/batch_size)**2
b4 -= lr * bv4 / np.sqrt(bs4+1e-7)

cost_ = cost_/batch_size
```

```
    cost.append(cost_)

    params = [f1, f2, w3, w4, b1, b2, b3, b4]

    return params, cost

def train(num_classes = 10, lr = 0.01, beta1 = 0.95, beta2 = 0.99, img_dim = 28, img_depth
= 1, f = 5, num_filt1 = 8, num_filt2 = 8, batch_size = 32, num_epochs = 2, save_path =
'params.pkl'):

    m =50000
    X = extract_data('train-images-idx3-ubyte.gz', m, img_dim)
    y_dash = extract_labels('train-labels-idx1-ubyte.gz', m).reshape(m,1)
    X-= int(np.mean(X))
    X/= int(np.std(X))
    train_data = np.hstack((X,y_dash))

    np.random.shuffle(train_data)

    f1, f2, w3, w4 = (num_filt1 ,img_depth,f,f), (num_filt2 ,num_filt1,f,f), (128,800), (10, 128)
    f1 = initializeFilter(f1)
    f2 = initializeFilter(f2)
    w3 = initializeWeight(w3)
    w4 = initializeWeight(w4)

    b1 = np.zeros((f1.shape[0],1))
    b2 = np.zeros((f2.shape[0],1))
    b3 = np.zeros((w3.shape[0],1))
    b4 = np.zeros((w4.shape[0],1))

    params = [f1, f2, w3, w4, b1, b2, b3, b4]

    cost = []

    print("LR:"+str(lr)+", Batch Size:"+str(batch_size))

    for epoch in range(num_epochs):
        np.random.shuffle(train_data)
        batches = [train_data[k:k + batch_size] for k in range(0, train_data.shape[0],
batch_size)]

        t = tqdm(batches)
        for x,batch in enumerate(t):
            params, cost = adamGD(batch, num_classes, lr, img_dim, img_depth, beta1, beta2,
params, cost)
            t.set_description("Cost: %.2f" % (cost[-1]))

    to_save = [params, cost]

    with open(save_path, 'wb') as file:
```

```
        pickle.dump(to_save, file)

    return cost

import numpy as np

def convolution(image, filt, bias, s=1):
    '''
    Confolves `filt` over `image` using stride `s`
    '''
    (n_f, n_c_f, f, _) = filt.shape
    n_c, in_dim, _ = image.shape

    out_dim = int((in_dim - f)/s)+1

    assert n_c == n_c_f, "Dimensions of filter must match dimensions of input image"

    out = np.zeros((n_f,out_dim,out_dim))

    for curr_f in range(n_f):
        curr_y = out_y = 0
        while curr_y + f <= in_dim:
            curr_x = out_x = 0
            while curr_x + f <= in_dim:
                out[curr_f, out_y, out_x] = np.sum(filt[curr_f] * image[:,curr_y:curr_y+f,
curr_x:curr_x+f]) + bias[curr_f]
                curr_x += s
                out_x += 1
            curr_y += s
            out_y += 1

    return out

def maxpool(image, f=2, s=2):
    '''
    Downsample `image` using kernel size `f` and stride `s`
    '''
    n_c, h_prev, w_prev = image.shape

    h = int((h_prev - f)/s)+1
    w = int((w_prev - f)/s)+1

    downsampled = np.zeros((n_c, h, w))
    for i in range(n_c):

        curr_y = out_y = 0
        while curr_y + f <= h_prev:
            curr_x = out_x = 0
            while curr_x + f <= w_prev:
                downsampled[i, out_y, out_x] = np.max(image[i, curr_y:curr_y+f, curr_x:curr_x+f])
```

```
            curr_x += s
            out_x += 1
        curr_y += s
        out_y += 1
    return downsampled

def softmax(X):
    out = np.exp(X)
    return out/np.sum(out)

def categoricalCrossEntropy(probs, label):
    return -np.sum(label * np.log(probs))

from CNN.forward import *
import numpy as np
import gzip

def extract_data(filename, num_images, IMAGE_WIDTH):
    print('Extracting', filename)
    with gzip.open(filename) as bytestream:
        bytestream.read(16)
        buf = bytestream.read(IMAGE_WIDTH * IMAGE_WIDTH * num_images)
        data = np.frombuffer(buf, dtype=np.uint8).astype(np.float32)
        data = data.reshape(num_images, IMAGE_WIDTH*IMAGE_WIDTH)
        return data

def extract_labels(filename, num_images):
    print('Extracting', filename)
    with gzip.open(filename) as bytestream:
        bytestream.read(8)
        buf = bytestream.read(1 * num_images)
        labels = np.frombuffer(buf, dtype=np.uint8).astype(np.int64)
    return labels

def initializeFilter(size, scale = 1.0):
    stddev = scale/np.sqrt(np.prod(size))
    return np.random.normal(loc = 0, scale = stddev, size = size)

def initializeWeight(size):
    return np.random.standard_normal(size=size) * 0.01

def nanargmax(arr):
    idx = np.nanargmax(arr)
    idxs = np.unravel_index(idx, arr.shape)
    return idxs

def predict(image, f1, f2, w3, w4, b1, b2, b3, b4, conv_s = 1, pool_f = 2, pool_s = 2):
    conv1 = convolution(image, f1, b1, conv_s)
    conv1[conv1<=0] = 0
```

```
conv2 = convolution(conv1, f2, b2, conv_s)
conv2[conv2<=0] = 0

pooled = maxpool(conv2, pool_f, pool_s)
(nf2, dim2, _) = pooled.shape
fc = pooled.reshape((nf2 * dim2 * dim2, 1))

z = w3.dot(fc) + b3
z[z<=0] = 0

out = w4.dot(z) + b4
probs = softmax(out)

return np.argmax(probs), np.max(probs)
```

## Додаток В. Ілюстративний матеріал

# Удосконалення методу пошуку прихованої інформації в цифрових зображеннях на основі штучної нейронної мережі

Виконала: ст. гр. УБ -20м Шевченко М.В.
Керівник: д.т.н., проф. Яремчук Ю.Є.

## Актуальність:

Оскільки в наш час постійно з'являються нові стеганографічні алгоритми, а також популярність використання зловмисниками даного методу приховування інформації стрімко зростає необхідно розробляти та удосконалювати існуючі методи стеганоаналізу. Використання нейронних мереж може значно покращити якість виявлення прихованої інформації, а також пришвидшити пошук методів виявлення стеганоконтейнерів для невідомих раніше стеганографічних методів, тому даний напрямок удосконалення є доцільним та актуальним.

## Мета роботи:

Удосконалення методу пошуку прихованої інформації в цифрових зображеннях за рахунок штучної нейронної мережі та реалізація удосконаленого методу.

Для досягнення мети необхідно розв'язати такі задачі:

- дослідити та проаналізувати існуючі методи стеганографії та виявлення прихованих повідомлень у рамках статистичного та нейромережевого підходів у стегоаналізі.
- розробити архітектуру згорткової нейронної мережі для потреб стегоаналізу та архітектуру програмного засобу для аналізу зображень.
- розробити програмний продукт для удосконаленого методу.
- провести аналіз ефективності удосконаленого методу порівняно з існуючими методами статистичного стегоаналізу.

## Огляд стеганографічних методів

Основні методи приховування зображень в стеганографії:

- Методи заміни просторової області
- Методи приховування частотної області зображення
- Широкосмугові методи
- Статистичні методи
- Методи спотворення
- Структурні методи

3

## Огляд методів стеганоаналізу для графічних файлів

1. **Методи, призначені до роботи з конкретними заздалегідь відомими стеганографічними алгоритмами**
   - Сигнатурні методи
   - Схемні методи аналізу
2. **Методи, призначені для будь-яких алгоритмів стеганографії**
   - Візуальні методи
     - Методи структурного аналізу
     - Методи сигнатурного аналізу
   - Статистичні методи
     - Метод оцінки числа переходів значень молодших бітів у сусідніх елементах контейнера
     - Метод оцінки частот появи k -бітових серій у потоці НЗБ елементів контейнера
     - Метод аналізу розподілу пар значень на основі критерію $x^2$
     - Метод аналізу гістограм, побудованих за частотами елементів зображення
     - Метод аналізу розподілу елементів зображення на площині
     - Метод глибокого навчання

4

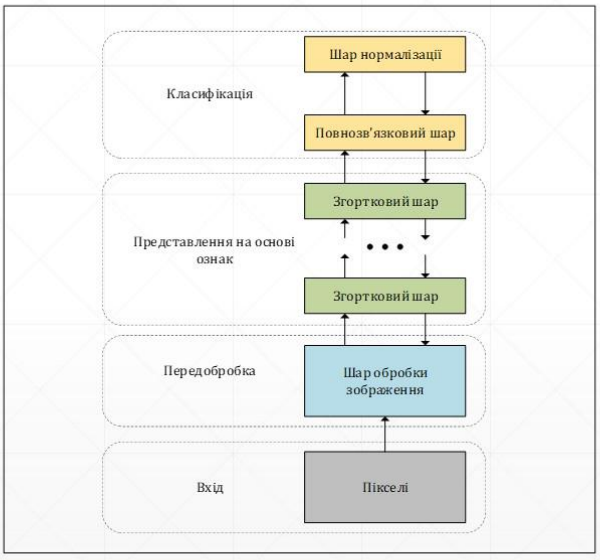
## Використання машинного навчання для стеганоаналізу

| Алгоритм стеганографічного приховування інформації що аналізується | WOW з різним параметром завантаження контейнера | | S-UNIWARD з різним параметром завантаження контейнера | |
|---|---|---|---|---|
| | pl = 0,2 | pl = 0,4 | pl = 0,2 | pl = 0,4 |
| **EC+SRM** | 63 | 74 | 63 | 75 |
| **Yedrouj-Net** | 72 | 86 | 63 | 77 |
| **Xu-Net** | 67 | 79 | 61 | 73 |
| **Ye-Net** | 67 | 77 | 60 | 69 |

Результати порівняння помилок виявлення для глибоких нейронних мережпри різних обсягах впровадження (корисного навантаження) у %

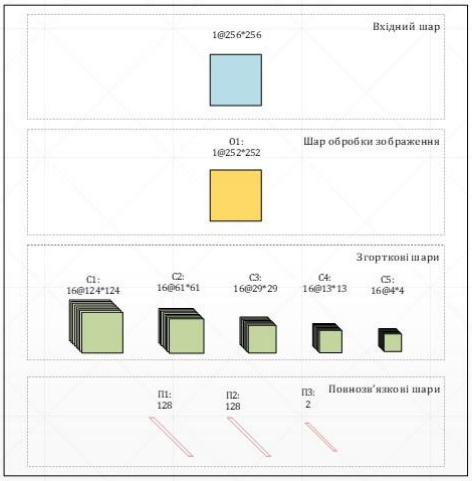
5

## Розроблена модель згорткової нейронної мережі



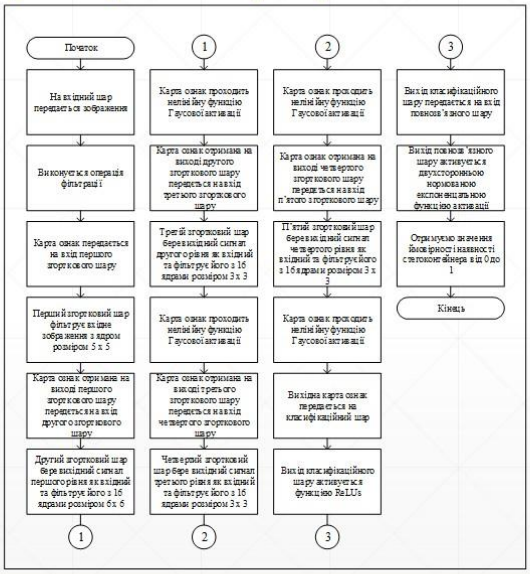

6

## Реалізація параметрів згорткової нейронної мережі



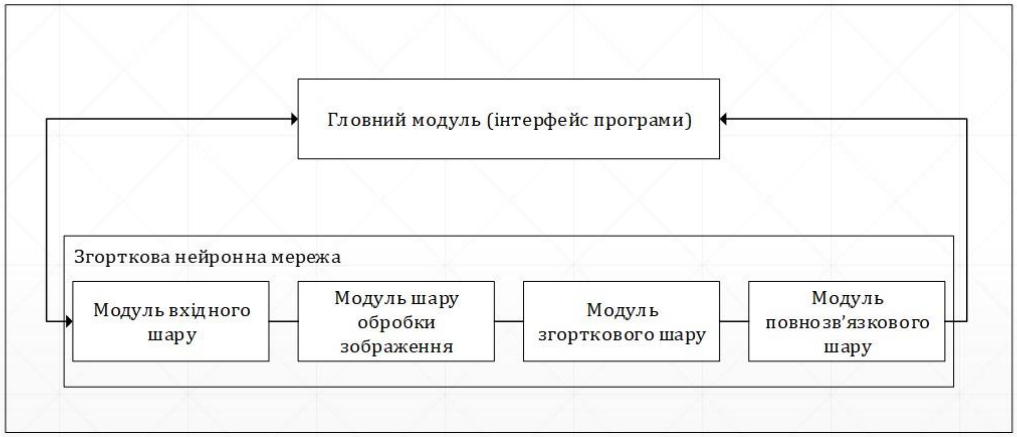

7

## Алгоритм роботи розробленого методу




8

## Архітектура системи




9

## Вибір мови та засобів для розробки програмного засобу





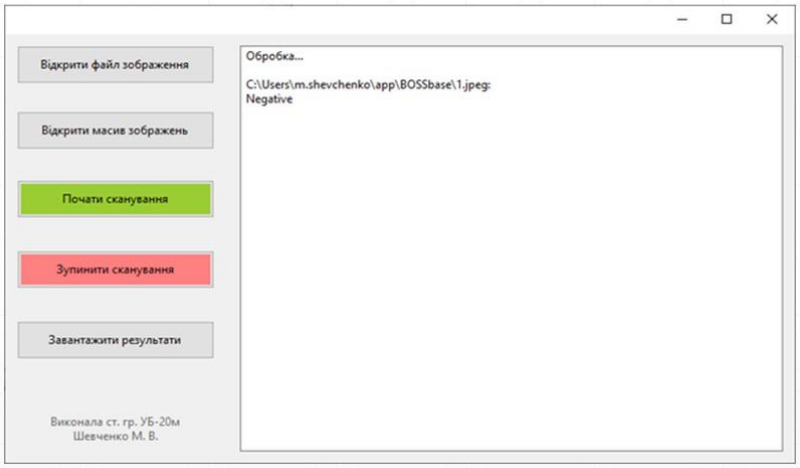
10

## Огляд програмного продукту


Головне вікно програми

11

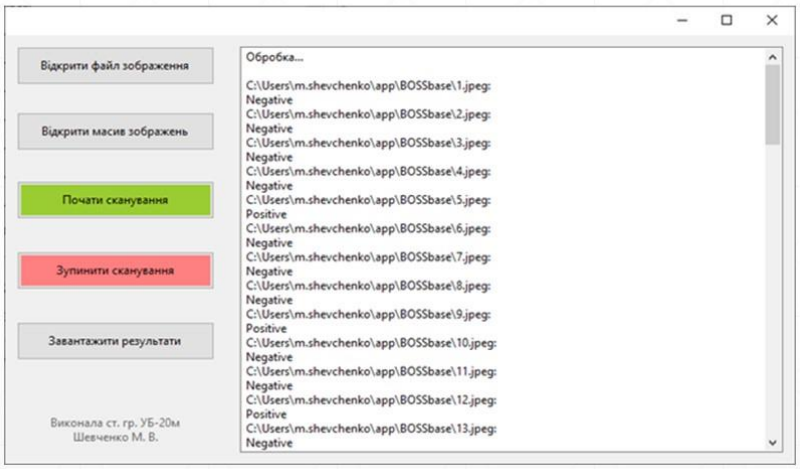
## Огляд програмного продукту




Результат сканування порожнього контейнера

12

## Огляд програмного продукту




Результат сканування масиву зображень

13

## Огляд програмного продукту

| bpp | HUGO | | | WOW | | | S-UNIWARD | | |
|---|---|---|---|---|---|---|---|---|---|
| | ЗНР | SRM | SPAM | ЗНР | SRM | SPAM | ЗНР | SRM | SPAM |
| **0.3** | 33.8% | 29.6% | 42.9% | 34.3% | 31.2% | 42.2% | 35.9% | 31.5% | 40.0% |
| **0.4** | 28.9% | 25.2% | 39.1% | 29.3% | 25.7% | 38.2% | 30.9% | 26.3% | 35.1% |
| **0.5** | 25.7% | 21.4% | 35.7% | 24.8% | 22.1% | 34.9% | 26.3% | 21.4% | 30.6% |

Результати порівняння помилок виявлення для глибоких нейронних мережпри різних обсягах впровадження (корисного навантаження) у %

14

# Дякую за увагу!

## Додаток Г. Протокол перевірки на антиплагіат

## ПРОТОКОЛ ПЕРЕВІРКИ НАВЧАЛЬНОЇ (КВАЛІФІКАЦІЙНОЇ) РОБОТИ

Назва роботи: Удосконалення методу пошуку прихованої інформації в цифрових зображеннях на основі штучної нейронної мережі

Тип роботи: Магістерська кваліфікаційна робота

Підрозділ: Факультет МІБ, кафедра менеджменту та безпеки інформаційних систем, гр. УБ-20м

Науковий керівник Яремчук Ю.Є., Голова секції УБ кафедри МБІС, професор, д.т.н.

Показники звіту подібності

| Plagiat.pl (StrikePlagiarism) | | Unicheck | |
|---|---|---|---|
| КП1 | | | |
| КП2 | | Оригінальність | 82 % |
| Тривога/Білі знаки | / | Схожість | 18 % |

Аналіз звіту подібності (відмінити подібне)

- ☐ Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.
- ☐ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її автора. Роботу направити на доопрацювання.
- ☐ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Заявляю, що ознайомлений(-на) з повним звітом подібності, який був згенерований Системою щодо роботи (додається)

Автор _____ Шевченко М.В.

(підпис) (прізвище, ініціали)

Опис прийнятого рішення

Ступінь оригінальності роботи відповідає вимогам, що висуваються до МКР

Особа, відповідальна за перевірку_____ Коваль Н.П.

(підпис) (прізвище, ініціали)

Експерт _____ _____

(за потреби) (підпис) (прізвище, ініціали)