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

Даракчі Юрій Іванович,
студент групи РЗ-161

**КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА**

# Розробка стеганографічного методу з використанням генеративно-змагальних нейронних мереж

Спеціальність:
125 Кібербезпека

Керівник:
Кушніренко Наталія Ігорівна,
к.т.н., доцент

Одеса – 2021

Міністерство освіти і науки України
Державний університет «Одеська політехніка»
Інститут інформаційної безпеки, радіоелектроніки та телекомунікацій
Кафедра кібербезпеки та програмного забезпечення
Рівень вищої освіти перший (магістерський)
Спеціальність 125 – Кібербезпека
Освітня програма Кібербезпека

ЗАТВЕРДЖУЮ
Завідувач кафедри КБПЗ

_____
д.т.н.,проф. А.А.Кобозєва
____ _____202_р.

**ЗАВДАННЯ**
**НА КВАЛІФІКАЦІЙНУ РОБОТУ**

*Даракчі Юрію Івановичу*

1.Тема роботи: *Розробка стеганографічного методу з використанням генеративно-змагальних нейронних мереж*,
керівник роботи: *Кушніренко Наталія Ігорівна, к.т.н., доцент каф.* КБПЗ,
затверджені наказом ректора університету від „25” жовтня 2021 р. № 372-в
2. Зміст роботи*: аналіз проблемної області, постановка задачі, опис алгоритму статастичного стеганоаналізу на основі нейронних мереж, аналіз принципів побудови генеративно-змагальних мереж, реалізація програмного забезпечення для підвищення стійкості стеганосистеми до статистичного стеганоаналізу шляхом адаптивної модифікації контейнера, охорона праці.*
3. Перелік ілюстративного матеріалу: *структурна схема стегосистеми, схема генеративно-змагальної мережі, ілюстрація результату вбудовування за алгоритмом F5, знімки процесу роботу з розробленою програмою.*

4. Консультанти розділів роботи

| Розділ | Консультант | Підпис, дата | |
|---|---|---|---|
| | | Завдання видав | Завдання прийняв |
| *Охорона праці* | ктн. доц. Ярова І. А. | | |

5. Дата видачі завдання “_____”_____20___ р.

## КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів кваліфікаційної роботи | Строк виконання | Примітка |
|---|---|---|---|
| *1* | *Аналіз джерел з теми випускної кваліфікаційної роботи* | *13-09-2021* | *виконано* |
| *2* | *Огляд методів стеганоаналізу* | *25-09-2021* | *виконано* |
| *3* | *Аналіз принципів роботи генеративно-змагальних мереж* | *30-09-2021* | *виконано* |
| *4* | *Аналіз особливостей імплементації стеганографічного адгоритма F5* | *15-10-2021* | *виконано* |
| *5* | *Розробка статистичного класифікатора* | *20-10-2021* | *виконано* |
| *6* | *Розробка генеративно-змагальної мережі* | *02-11-2021* | |
| *7* | *Дизайн та реалізація інтерфейсу програмного продукту* | *17-11-2021* | *виконано* |
| *8* | *Підготовка тексту роботи* | *25-11-2021* | *виконано* |
| *9* | *Підготовка презентації та доповіді* | *30-11-2021* | |
| *9* | *Попередній захист* | *03-12-2021* | *виконано* |
| *10* | *Нормоконтроль, рецензування* | *20-12-2021* | *виконано* |
| *11* | *Занесення роботи в електронний архів* | | |
| *12* | *Допуск до захисту у завідувача кафедри* | | |

**Здобувач вищої освіти** _____ *Даракчі Ю. І.*

**Керівник роботи** _____ *Кушніренко Н. І.*

**ЗАВДАННЯ**

на розробку розділу «Охорона праці та безпека в надзвичайних ситуаціях»

Даракчі Юрія Івановича

Інститут інформаційної безпеки, радіоелектроніки та телекомунікацій
Кафедра кібербезпеки та програмного забезпечення

Тема роботи: Розробка стеганографічного методу з використанням генеративно-змагальних нейронних мереж

Зміст розділу:

1. Аналіз умов праці і вибір заходів і засобів захисту від небезпечних і шкідливих виробничих факторів.
2. Аналіз техногенних небезпек і вибір заходів і засобів забезпечення безпеки у надзвичайних ситуаціях.
3. Проектування системи освітлення.

Керівник роботи
к.т.н., доцент Кушніренко Н. І.

_____
(підпис)
«__»_____2021 р.

Консультант з охорони праці та БНС
к.т.н., доцент Ярова І.А.

_____
(підпис)
«__»_____2021 р.

# АНОТАЦІЯ

Кваліфікаційна робота на тему «Розробка стеганографічного методу з використанням генеративно-змагальних нейронних мереж» на здобуття другого (магістерського) рівня вищої освіти за спеціальністю 125 – Кібербезпека, спеціалізація, освітня програма: Кібербезпека, містить 18 рисунків, 2 таблиці, 1 додаток, 27 літературні джерела за переліком посилань. Робота виконана на 66 сторінках загального тексту і 55 сторінках основного тексту.

Метою роботи є підвищення стійкості стеганосистеми до статистичного стеганоаналізу шляхом розробки методу адаптивної модифікації контейнера з використанням генеративно-змагальних мереж.

У роботі проведено аналіз деяких перспективних напрямків у розвитку стеганографічних алгоритмів, етапів роботи генеративно-змагальних мереж та базових принципів проектування стеганоаналітичних методів з використанням нейронних мереж.

У результаті виконання кваліфікаційної роботи було розроблено генеративно-змагальну мережу для адаптивної модифікації стеганографічного контейнера перед вбудовуванням повідомлення, а також створено відповідне програмне забезпечення на її основі. Попередня модифікація дозволить уникнути детектування повідомлення сучасними статистичними стеганоаналітичними методами або значно зменшить дану вірогідність.

Отримані результати можуть бути використані для підвищення стійкості та покращення ефективності роботи стеганографічних алгоритмів вбудовування у коефіцієнти ДКП.

ІНФОРМАЦІЙНА БЕЗПЕКА, ЦИФРОВА СТЕГАНОГРАФІЯ, JPEG, ДИСКРЕТНО-КОСИНУСНЕ ПЕРЕТВОРЕННЯ, СТАТИСТИЧНИЙ СТЕГАНОАНАЛІЗ, ГЕНЕРАТИВНО-ЗМАГАЛЬНА МЕРЕЖА.

# ANNOTATION

Qualification work on the theme "Development of steganographic method using generative adversarial neural networks" for obraining the second (master’s) level of higher education in specialty 125 – Cybersecurity specialization, educational program: Cybersecurity, contains 18 figures, 2 tables, 1 attachment, 27 literature sources under the list of references. The work is done on 66 pages of general text and 55 pages of main text.

The objective of the work is to increase the resistance of the steganographic system to statistical steganoanalysis by developing a method of adaptive modification of the container using generative adversarial networks.

The paper analyzes some promising directions in the development of steganographic algorithms, stages of generative adversarial networks and the basic principles of constructing steganoanalytical methods using neural networks.

As a result of the qualification work, a generative adversarial network was developed for the adaptive modification of the steganographic container before embedding the message, and the appropriate software based on it was created. Preliminary modification will avoid the detection of the message by modern statistical steganoanalytical methods or significantly reduce this probability.

The obtained results can be used to increase the stability and improve the efficiency of steganographic algorithms for embedding in DCT coefficients.

INFORMATION SECURITY, DIGITAL STEGANOGRAPHY, JPEG, DISCRETE COSINE TRANSFORM, STATISTICAL STEGANALYSIS, GENERATIVE ADVERSARIAL NETWORK.

ЗМІСТ

# ВСТУП

Завдання захисту від несанкціонованого доступу вирішувалося за всіх часів протягом історії людства. Стеганографія — це один з найдавніших і одночасно найперспективніших сучасних напрямів захисту інформації. Його особливістю є те, що на відміну від криптографії, він не передбачає прямого оголошення факту існування таємного повідомлення. У процесі стеганографування приховується повідомлення або додаткова інформація (ДІ), вона вбудовується в об’єкт або контейнер, що не привертає увагу, який потім передається по відкритому каналу зв’язку. На сучасному етапі розвитку технологій у якості контейнерів зазвичай використовуються цифрові сигнали: зображення (ЦЗ), аудіо, відео.

З моменту становлення Інтернету як загальнодоступної телекомунікаційної мережі і понині перше місце за поширеністю займають саме цифрові зображення. Серед них особливо вирізняються зображення формату JPEG, який активно використовують майже 75% усіх сайтів, відомих пошуковим системам [1].

Формат JPEG зараз є найпопулярнішим форматом фотографій та кольорових зображень [2]. Він широко застосовується в цифрових фотоапаратах для збереження знятих знімків, тому що дозволяє отримати найбільше стиснення кольорового зображення. На момент виходу JPEG був єдиним міжнародним стандартом стиснення, заснованим на моделі вільного розповсюдження. Ще однією його перевагою була універсальність та відмінна адаптація як до апаратного, так і до програмного забезпечення. Формат JPEG мав нові ітерації – наприклад, JPEG XT, який відрізнявся підтримкою HDR-контенту та широким колірним простором. Це було дуже важливо з точки зору виробника та споживача, тому що ринок не любить частих змін у великих масштабах, про що свідчить неослабна популярність цього формату і сьогодні, 28 років після його створення. Це явище зробило його найперспективнішим кандидатом на роль стеганографічного контейнера.

За ці роки було розроблено безліч стеганографічних методів, спроектованих спеціально для роботи з даним форматом. Однак разом з цим також розроблялися

методи стеганоаналізу. Особливо добре себе показали методи так званого статистичного стеганоаналізу, які базуються на дослідженні статистичного розподілу пар коефіцієнтів дискретного косинусного перетворення (ДКП), яке застосовується для стиснення ЦЗ. Сутність даного виду аналізу полягає у тому, що кількість пар коефіцієнтів ДКП в оригінальному зображенні та стеганоповідомленні (СП) будуть відрізнятись. Ще кращі результати показують нейронні мережі, які використовують статистичні показники розподілу для встановлення залежностей.

Проблемним аспектом даного підходу є той факт, що ефективність стеганоаналізу значною мірою залежить від об’єму збурень, які вносить стеганоалгортм при вбудовуванні повідомлення. Отож, якщо зменшити вплив на ЦЗ, найімовірніше, статистичні стеганоаналітичні методи будуть неефективними.

Чим більше інформації циркулює в мережі Інтернет, тим важливіше стає її захист від втручання шляхом отримання несанкціонованого доступу, порушення авторських прав, спотворення даних, а також витоку інформації з обмеженим доступом. Одним з актуальних напрямків у предметних галузях безпеки інфокомунікаційних систем є питання формування та експлуатації прихованих каналів на основі стеганографічних методів перетворення інформації. У зв'язку з цим є актуальним завданням є підвищення стійкості стеганографічних алгоритмів до стеганоаналізу з використанням новітніх технологій.

Мета даної роботи — є підвищення стійкості стеганосистеми до статистичного стеганоаналізу шляхом розробки методу адаптивної модифікації контейнера з використанням генеративно-змагальних мереж.

Для досягнення поставленої мети необхідно розв’язати наступні задачі:

1. Проаналізувати принципи роботи сучасних стеганографічних методів та реалізувати стеганографічний алгоритм F5.
2. Визначити переваги та недоліки статисничних методів стеганоаналізу, а також проаналізувати сучасні методи побудови та архітектури нейронних мереж. Побудувати ефективний статистичний класифікатор на основі

проведених досліджень. Протестувати ефективність роботи отриманого класифікатора.

3. Докладно розібрати принципи і особливості роботи етапів генеративно-змагальних мереж. Побудувати генеративно-змагальну мережу.
4. Створити програмну реалізацію з використанням спроектованих класифікатора та ГЗМ. Переконатися в працездатності створеного програмного продукту.

Об’єкт дослідження — застосування методів машинного навчання у стеганографії.

Предмет дослідження — стеганографічний метод на основі ГЗМ для підвищення стійкості до статистичного стеганоаналізу.

Новизна кваліфікаційної роботи полягає в удосконаленні методу статистичного стеганоаналізу на основі аналізу пар коефіцієнтів ДКП, а також розробці методу адаптивної модифікації контейнера для зменшення ймовірності детектування ДІ на основі генетивно-змагальних мереж (ГЗМ).

Отримані результати можуть бути використані для підвищення стійкості та покращення ефективності роботи стеганографічних алгоритмів вбудовування у коефіцієнти ДКП.

Результати досліджень були подані у науковій статті «Розробка методу для підвищення стійкості до статистичного стеганоаналізу з використанням генеративно-змагальних мереж», що опублікована у журналі «Інформатика та математичні методи в моделюванні» [3].

# 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

## 1.1 Аналіз принципів сучасної стеганографії

З кожним роком людство все більше поринає у цифрову еру з її майже безмежними можливостями. Триває оцифрування документів, предметів мистецтва та музейних експонатів. Інтернет месенджери та відеоконференції все більше замінюють живе спілкування і ми захоплено занурюємося у світ цифрових послуг та товарів. Чим більше інформації циркулює в мережі Інтернет, тим важливіше стає її захист від втручання шляхом отримання несанкціонованого доступу, порушення ліцензійних угод та авторських прав на інтелектуальну власність, підробки або руйнування даних, а також витоку інформації, до якої було отримано доступ. Дані проблеми є актуальними як в державній сферах, так і комерційній сферах людської діяльності.

Зараз у всьому світі назріло питання розробки нових та удосконалення наявних методів інформаційної безпеки цифрових даних, серед яких важливе місце займають методи стеганографії.

Стеганографія — це наука про таємну передачу інформації шляхом приховування самого факту передачі. А саме слово стеганографія походить від грец. στεγανός «прихований» + γράφω «пишу» і буквально перекладається як «тайнопис» [4].

Цифрова стеганографія — це один з напрямків класичної стеганографії, який базується на приховуванні або запровадження додаткової інформації у цифрові об’єкти, спричиняючи при цьому певні спотворення цих об’єктів [5]. Як правило, дані об’єкти є мультимедійними (зображення, відео-потоки, аудіодані, текстури 3D-об’єктів) і внесення спотворень, що знаходяться нижче порогу чутливості органів чуття середньостатистичної людини, не призводить до помітних їх змін. У оцифрованих даних, що мають аналогову природу, завжди присутній деякий додатковий шум — шум квантування, а при відтворенні цих даних з’являється ще аналоговий шум внаслідок нелінійних спотворень в апаратурі, що сприяє більшій непомітності прихованої інформації.

У цифрову стеганографію включають такі напрямки:

- вбудовування інформації, щоб передати її потай;
- вбудовування ідентифікаційних номерів (fingerprinting);
- вбудовування заголовків (captioning);
- вбудовування цифрових водяних знаків (ЦВЗ) (watermarking).

Загальною рисою методів стеганографії є приховування секретної інформації шляхом вбудовування її у деякий об’єкт, що не буде привертати увагу потенційних супротивників, — контейнер, або основне повідомлення. Отриманий об’єкт відкрито транспортується адресату каналом зв’язку або зберігається в такому вигляді.

Стеганографічна система або стегосистема – це сукупність засобів і методів, які використовуються для формування прихованого каналу зв’язку для передачі інформації. Основним завданням будь-якого алгоритму стеганографії є забезпечення збереження в секреті наявності таємного каналу передачі, тому при побудові стегосистеми повинні виконуватись наступні положення [6]:

- противник може мати повне уявлення про деталі реалізації стеганографічної системи. Єдина інформація, що є невідомою потенційному супротивнику — це ключ, завдяки якому лише той, хто володіє даним ключем може достеменно встановити факт наявності додаткової інформації та зміст потайного повідомлення;
- метод стеганографії повинні забезпечувати цілісність та автентичність даних, у яких ховається секретне повідомлення;
- якщо противник якимось чином дізнається про факт існування прихованого повідомлення, це не повинно дозволити йому витягти приховану інформацію з інших контейнерів до тих пір, поки ключ зберігається в таємниці;
- потенційний супротивник не повинен мати будь-які технічні чи інші переваги у розпізнаванні або розкритті змісту таємних повідомлень.

Дещо спрощена та узагальнена структурна схема стегосистеми представлена на рис. 1.1. Повідомлення — це будь-яка інформація, що підлягає таємній передачі. У якості повідомлення може бути використано будь-який вид інформації: текст, зображення тощо.

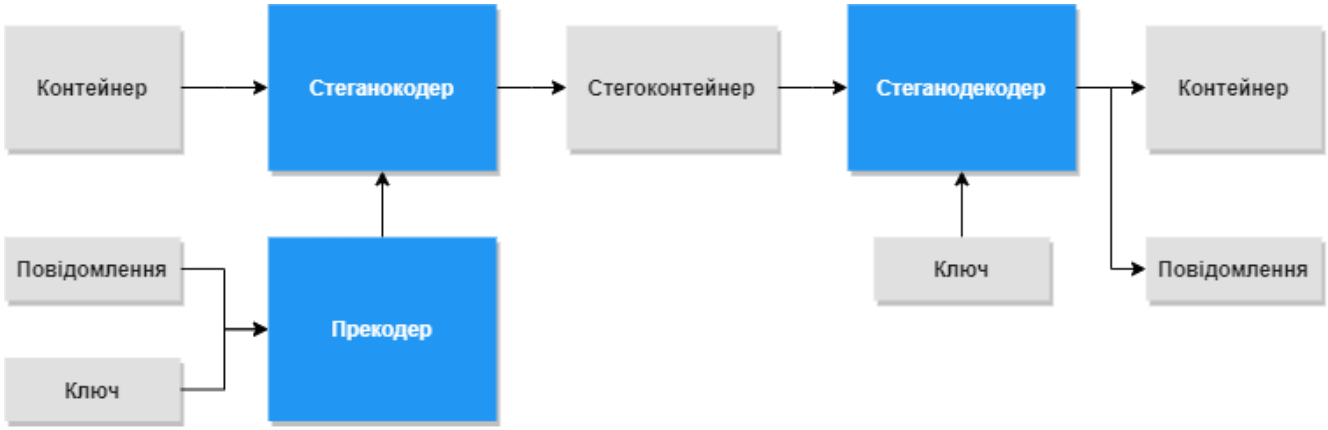


Рисунок 1.1 — Спрощена структурна схема стегосистеми

Надійність таких систем визнається насамперед надійністю ключа або ключів, кількість яких залежить від кількості рівнів захисту у стегосистемі.

Незважаючи на різноманітність варіантів вбудовування та рівнів захисту, які пропонують автори у своїх роботах, основним завданням стеганографії, як і раніше, залишається пошук компромісу між рівнем непомітності при передачі та обсягом переданої інформації. Отже, для більшості сучасних стеганографічних методів існує зворотна залежність між надійністю системи та обсягом вбудованих даних (рис. 1.2).


Рисунок 1.2 — Графік залежності надійності приховування від об’єму повідомлення

Дана залежність вказує, що зі збільшенням обсягу приховуваних таємних даних збільшується вірогідність детектування секретного повідомлення. Звідси випливає, що контейнер, який використовується у стегосистемі, здатний обмежувати розмір впроваджуваних секретних даних.

## 1.2 Огляд перспективних методів статастичного стеганоаналізу

Стегоаналіз або стеганоаналіз — це розділ стеганографії; наука про виявлення факту передачі прихованої інформації в повідомленні, що аналізується. У деяких випадках під стегоаналізом розуміють також вилучення прихованої інформації з повідомлення і (якщо це необхідно) подальше її дешифрування. Останнє визначення слід використовувати з відповідним застереженням.

Методи стеганоаналізу за принципом аналізу діляться на три основні класи:

- візуальні,
- сигнатурні,
- статистичні.

Візуальні методи стеганоаналізу цілком засновані на здатності органів чуття середньостатистичної людини аналізувати об’єкти, зіставляти зображення та

виявляти відмінності між зоровими образами. Метод візуального аналізу є найпростішим способом аналізу графічних файлів. Він є ефективними лише за умови майже повного заповнення контейнера, відповідно зі зменшенням його заповнення зменшуються збурення та очам людини все важче помітити зміни серед елементів оригінального зображення. Крім того, дані методи можуть протидіяти лише стеганографічним методам приховування даних у частотній області.

Сигнатурні методи стеганоаналізу призначені для роботи з форматними методами приховування інформації, які в процесі вбудовування залишають специфічні маркери (сигнатури), за якими і вдається детектувати приховане вкладення.

Клас статистичних методів стеганоаналізу представляє введення в контейнер прихованої інформації як порушення статистичних відносин оригінальних контейнерів і дає ймовірнісну характеристику. Аналізуються статистичні характеристики певної бітової послідовності і встановлюється, чи корелює вона з характеристиками порожніх стеганоконтейнерів того ж типу. Якщо збіг достатній, то прихованої передачі інформації немає, якщо вона схожа на характеристики заповнених контейнерів, то виявлено факт існування вкладення.

Даний підхід використовує безліч статистичних характеристик, такі як коефіцієнти кореляції, оцінка ентропії, ймовірності появи та залежності між елементами послідовностей, розрізнення розподілів за критерієм Хі-квадрат та багато інших. До переваг даної групи стеганоаналітичних методів належить необмежена сфера застосування. Основним недоліком методів цього класу є припущення про існування деякого оригінального контейнера.

Розглянемо аналіз розподілу значень з урахуванням критерію Хі-квадрат . У методі використовується аналіз гістограми, отриманої за елементами зображення та оцінка розподілу пар значень цієї гістограми. Метод Хі-квадрат є універсальним, оскільки підходить для аналізу зображень, створених різними

засобами приховування. Проте результати роботи методу за критерієм Хі-квадрат значною мірою залежить від способу приховування даних. При послідовному запису у найменші значущі біти (НЗБ) елементів контейнера метод забезпечує гарні результати, а при псевдовипадковому виборі молодших біт та розсіюванні повідомлення по всій довжині контейнера метод не спрацьовує. Крім того, існує можливість вибору окремих областей зображення для подальшого аналізу. Такий підхід з певною ймовірністю дозволяє виявляти наявність інформації, прихованої псевдовипадковим чином.

На рис. 1.3 зображено результат роботи алгоритму при заповнені контейнера на 9%. По осі X — відсоток перевірених коефіцієнтів, Y — ймовірність вбудовування. Суцільна лінія – звичайний Хі-квадрат, пунктир – блоковий Хі-квадрат. [7].

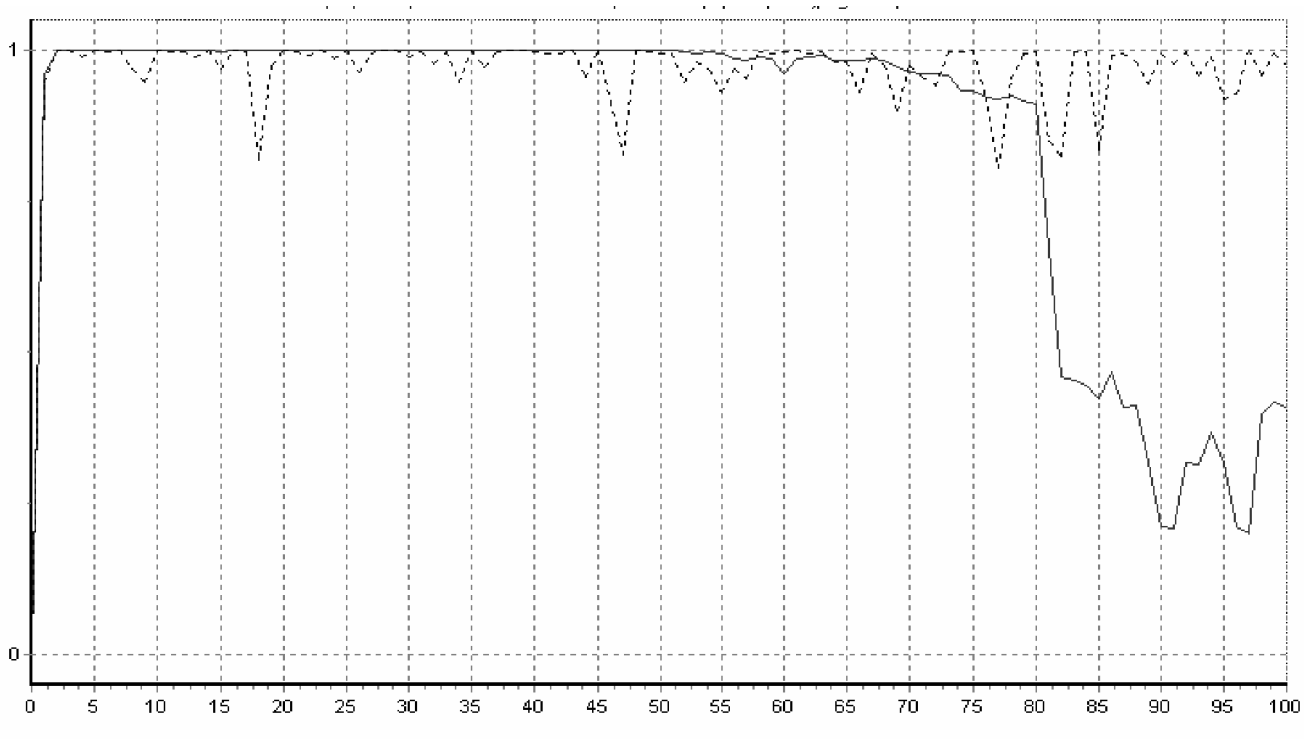


Рисунок 1.3 — Графіки, що показують ймовірність наявності приховування інформації у зображенні при заповнені на 9%

На рис. 1.4 зображено результат роботи алгоритму при заповнені контейнера на 3%.

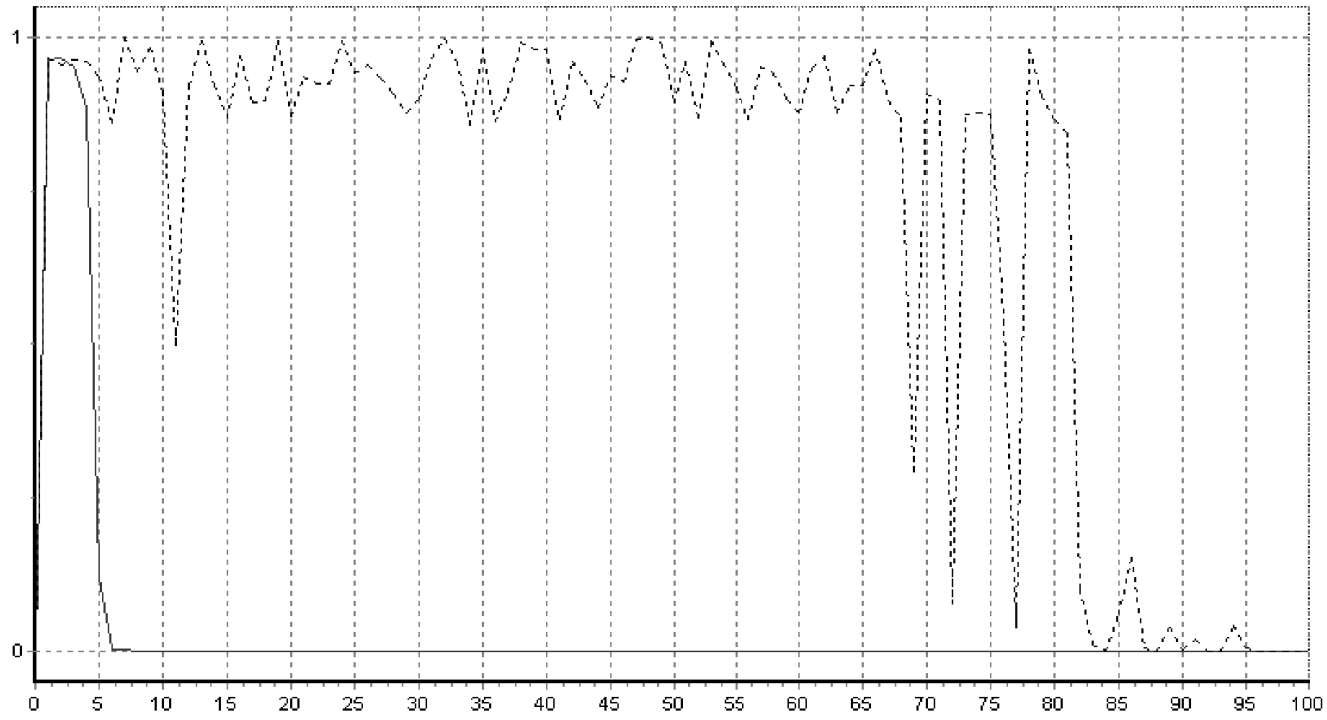


Рисунок 1.4 — Графіки, що показують ймовірність наявності приховування інформації у зображенні при заповнені на 3%

Наступним популярним статистичним методом є аналіз гістограм частот елементів зображення. З його допомогою можна оцінити рівномірність розподілу елементів аналізованого зображення, і навіть встановити частоту появи окремих елементів. Якщо розподіл частот появи елементів у колірних складових зображення прагне до нуля, то контейнер з великою вірогідністю містить додаткову інформацію. У іншому випадку контейнер вважається незаповненим. Для цифрових зображень формату JPEG будується гістограма частот відквантованих коефіцієнтів ДКП. Було експериментально встановлено, що гістограми порожнього зображення мають більш гладкий характер порівняно із гістограмами зображень, що містять стеганографічне вкладення.

Більшість стеганографічних алгоритмів, що працюють з JPEG, приховують дані в молодші біти коефіцієнтів відмінних від 0 та 1. Як наслідок частоти 0-х і 1-х коефіцієнтів не змінюються, у той час як всі інші частоти або зменшуються, або збільшуються залежно від алгоритму вбудовування. При значних обсягах гістограми, що приховується, часто набувають характеру нетипово для звичайних

JPEG зображень. Це добре помітно на рис 1.5. Зліва — стеганоконтейнер, справа — оригінальне зображення.

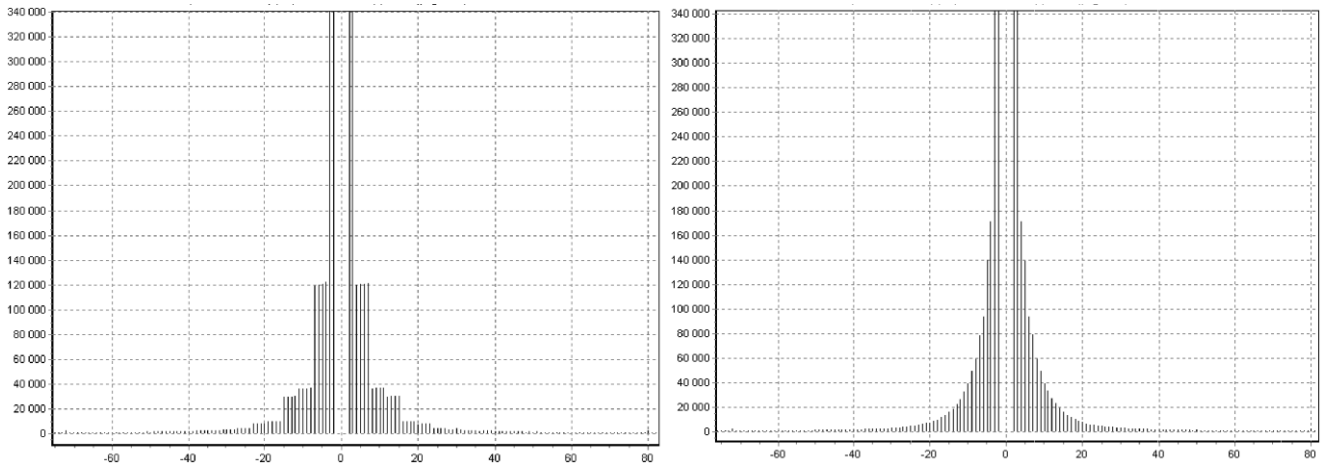


Рисунок 1.5 — Гістограми розподілу коефіцієнтів ДКП

Розробка математичної моделі, що дозволяє за видом гістограм судити про наявність прихованої в JPEG-файлі інформації, не є перспективною зважаючи на випадковість одержуваних результатів для різних зображень, і різних алгоритмів приховування інформації у JPEG. Однак даний метод продемонстрував непогані результати при роботі з Jsteg та подібними до нього алгоритмами.

## 1.3 Аналіз принципів проектування генеративно-змагальних мереж

Останніми роками генеративно-змагальні мережі стали активною темою досліджень. Директор з досліджень штучного інтелекту компанії Meta (Facebook) Янн ЛеКун назвав змагальне навчання в галузі машинного навчання «найцікавішою ідеєю за останні 10 років» [8].

Генеративно-змагальні мережі (англ. Generative Adversarial Nets, GAN) — алгоритм машинного навчання, що входить до сімейства породжувальних моделей і побудований на комбінації з двох нейронних мереж: генеративна мережа G, яка будує наближення розподілу даних, і дискримінативна мережа D, що оцінює ймовірність того, що зразок надійшов із тренувальних даних, а не

згенерований моделлю G. Навчання для моделі G полягає у максимізації ймовірності помилки дискримінатора D (звідси і «змагальний» характер моделі). Вперше такі мережі були представлені у 2014 році експертом з глибинного навчання Іеном Гудфеллоу та його колегами. Їх спрощена архітектура зображена на рис. 1.6.

Дискримінаційні моделі — це моделі, які використовуються у більшості завдань навчання з учителем для класифікації чи регресії.

Проте генеративні моделі, такі як GAN, навчені описувати, як створюється набір даних у термінах імовірнісної моделі. Використовуючи вибірку з генеративної моделі, ви можете створювати нові дані. У той час, як дискримінаційні моделі використовуються для навчання з вчителем, генеративні моделі часто використовуються з немаркованими наборами даних і можуть розглядатися як форма навчання без вчителя.

Для виведення нових вибірок генеративні моделі зазвичай розглядають стохастичний або випадковий елемент, який впливає на вибірки, згенеровані моделлю. Випадкові вибірки, що використовуються для керування генератором, отримані з прихованого простору, в якому вектори представляють своєрідну стиснену форму згенерованих вибірок.

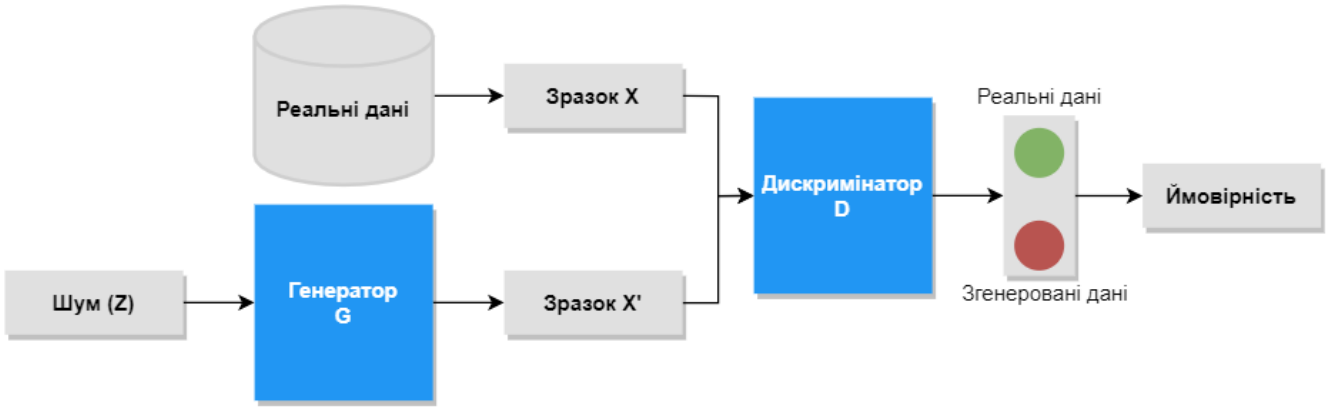


Рисунок 1.6 — Оригінальна архітектура генеративно-змагальної мережі

Як було зазначено раніше, ми бажаємо навчити дві моделі: генеративну та дискримінативну. Оскільки найзручніше використовувати багатошарові

перцептрони для навчання змагальної моделі, будемо використовувати саме їх для детального опису роботи. Щоб вивести ймовірнісний розподіл генератора $p_g$ над набором даних $X$, визначимо апріорну ймовірність шуму $p_z(z)$ і представимо генератор, як відображення $G(z, \gamma_g)$, де $G$ диференційована функція, представлена багатошаровим перцептроном з параметром $\gamma_g$. Аналогічно представимо другий багатошаровий перцептрон $D(z, \gamma_d)$, який на вихід подає одне скалярне значення - ймовірність того, що $x$ прийшло з тренувальних даних, а не $p_g$. Під час тренування $D$ ми прагнемо максимізувати ймовірність правильної ідентифікації об'єктів із тренувальної та згенерованої вибірок. І у той же час тренуємо $G$ так, щоб мінімізувати $log\left(1 - D\left(G(z)\right)\right)$. Іншими словами, $D$ та $G$ грають у «мінімакс гру» [9]:

$$\min_G \max_D V(D,G) = \underset{xp_{data}}{E}[logD(x)] + \underset{zp_z}{E}\left[log\left(1 - D\left(G(z)\right)\right)\right]$$

Для більшості ГЗМ існують наступні проблеми:

- Схлопування мод розподілу (англ. mode collapse): генератор колапсує, тобто видає обмежену кількість різних зразків.
- Проблема стабільності навчання (англ. non-convergence): параметри моделі дестабілізуються та не сходяться.
- Зникаючий градієнт (англ. diminished gradient): дискримінатор стає занадто «сильним», а градієнт генератора зникає і навчання не відбувається.
- Проблема заплутування (англ. disentanglement problem): виявлення кореляції в ознаках, які не пов'язані (слабко пов'язані) у реальному світі.
- Висока чутливість до гіперпараметрів.

# 2 РОЗРОБКА СТЕГАНОМЕТОДУ З АДАПТИВНОЮ МОДИФІКАЦІЮ КОНТЕЙНЕРА

## 2.1 Імплементація стеганографічного алгоритму F5

Для проведення експериментів зі зниження вразливості до статистичного стеганоаналізу був обраний алгоритм F5.

F5 — це один із алгоритмів стеганографії для вбудовування даних у зображення JPEG. Стеганоаналіз даного алгоритму залишається актуальним напрямом завдяки наступним його особливостям [10]:

- забезпечує велику пропускну здатність;
- висока ефективність (вставляйте більше бітів за одну зміну) завдяки матричному кодуванню;
- добре протидіє візуальним атакам;
- має високу стійкість до статистичних атак, зокрема до Хі-квадрат;
- використовує розповсюджений формат для контейнерів (JPEG);
- має відкритий код.

Зазвичай стеганоалгоритми пропонують відносну стійкість до візуальних і статистичних атак в обмін на досить невеликі обсяги інформації, що передається. Зі свого боку F5 є досить збалансованим і здатний вбудовувати великі обсяги інформації, зберігаючи при цьому стійкість до різних типів стеганоаналізу.

Стеганографічний алгоритм F5 був запропонований німецькими дослідниками Пфіцманом і Вестфельдом у 2001 році [11]. Метою їхнього дослідження була розробка концепцій та практичного методу вбудовування у зображення формату JPEG, які забезпечили б високу пропускну стеганографічну здатність без шкоди для безпеки. Керуючись атакою Хі-квадрат, вони кинули виклик прийнятій парадигмі заміни бітів інформації контейнера на секретне повідомлення, запропонувавши іншу парадигму. Замість того, щоб замінити останні біти окремих квантованих коефіцієнтів ДКП бітами повідомлення, абсолютне значення коефіцієнта зменшується на одиницю. Автори стверджують, що цей тип вбудовування не може бути виявлений за допомогою статистичної

атаки Хі-квадрат. Алгоритм F5 вбудовує біти повідомлення у випадково вибрані коефіцієнти ДКП і використовує матричне кодування, що мінімізує необхідну кількість змін для вбудовування повідомлення певної довжини.

Необхідність перестановки обумовлена тим, що в багатьох випадках повідомлення, що вбудовується, не вимагає повної ємності контейнера. Тому частина файлу залишається невикористаною, а зміни накопичуються на початку файлу (рис. 2.1), що значно знижує стійкість до стеганоаналітичних атак.

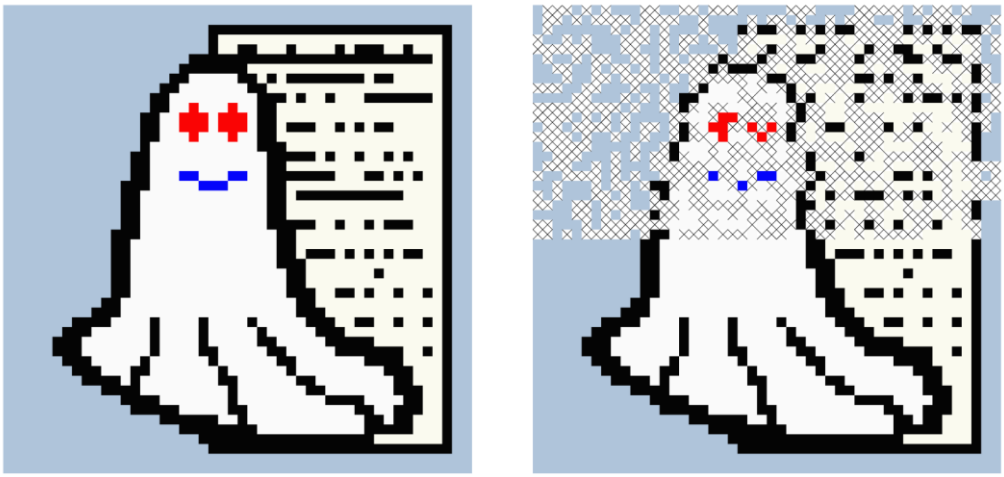

Рисунок 2.1 — Послідовне вбудовування повідомлення

Механізм рознесеної перестановки F5 спочатку перемішує всі коефіцієнти, використовуючи перестановку (рис. 2.2). Вона залежить від ключа, отриманого на основі псевдовипадкового генератора та заданого користувачем пароля, і має лінійну складність виконання. F5 відновлює розташування стеганографічно змінених коефіцієнтів перед кодуванням Гаффмана. При правильному ключі, одержувач може повторити перестановку та витягти повідомлення.

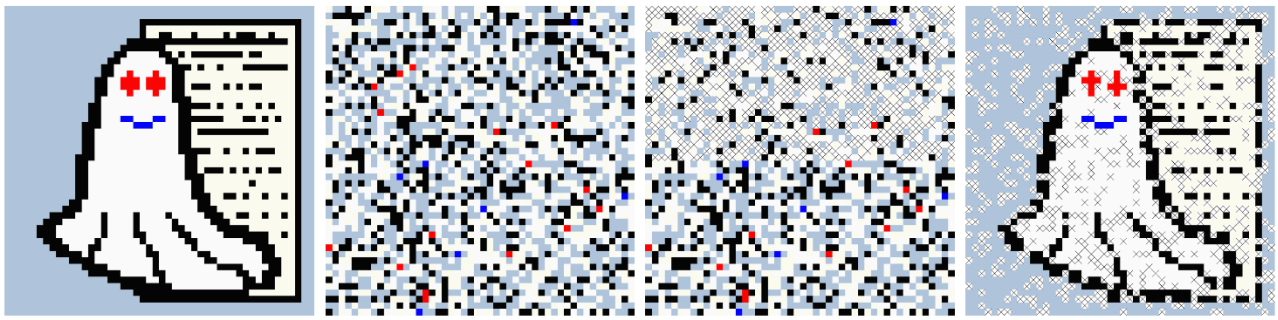

Рисунок 2.2 — Вбудовування з використанням рознесеної перестановки

Поняття матричного кодування, як метод збільшення ефективності стеганографічного вбудовування, було введено Роном Кренделом [12]. Матричне кодування здатне значно зменшити обсяг збурень контейнера при частковому його заповнені.

Без матричного кодування ефективність вбудовування алгоритму становить 2 біта за зміну. Наприклад, якщо ми спробуємо вбудувати повідомлення обсягом 217 байт (1736 біт), алгоритм F4, який не використовує матричне кодування, змінить 1157 позицій контейнера. F5 буде достатньо лише 459, тобто його ефективність буде становити 3.8 біти за одну зміну.

Для вбудовування двох бітів повідомлення $m_1$ та $m_2$ у три біти контейнера $p_1, p_2, p_3$, змінюючи при цьому не більше одного із них, F5 використовує наступний підхід:

$$m_1 = p_1 \otimes p_3, m_2 = p_2 \otimes p_3 \Rightarrow \text{не змінюється};$$
$$m_1 \neq p_1 \otimes p_3, m_2 = p_2 \otimes p_3 \Rightarrow \text{інкремент } p_1;$$
$$m_1 = p_1 \otimes p_3, m_2 \neq p_2 \otimes p_3 \Rightarrow \text{інкремент } p_2;$$
$$m_1 \neq p_1 \otimes p_3, m_2 \neq p_2 \otimes p_3 \Rightarrow \text{інкремент } p_3.$$

Вбудовування повідомлення виконується під час стиснення JPEG відповідно до рис. 2.3.

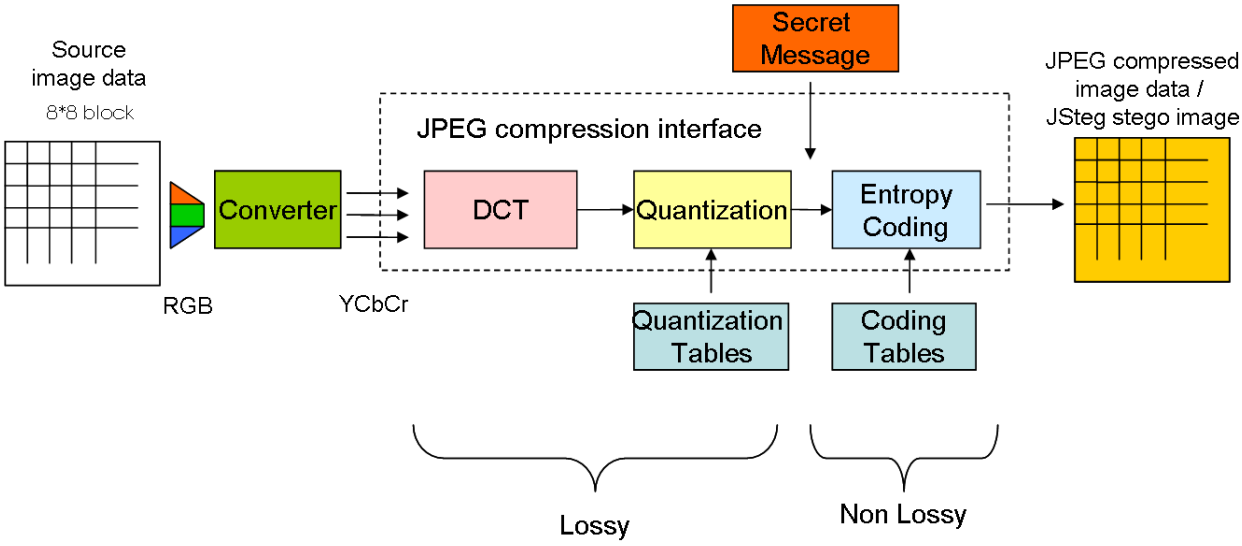


Рисунок 2.3 — Місце вбудовування повідомлення під час JPEG кодування

Для вбудовування за алгоритмом F5 потрібно виконати наступні кроки [13]:

- Почати стиснення JPEG. Зупинитись після квантування коефіцієнтів.
- Ініціалізувати стійкий генератор випадкових чисел за допомогою ключа, отриманого з пароля.
- Створити перестановку із двома параметрами — генератор випадкових чисел та кількість коефіцієнтів (включаючи нульові коефіцієнти).
- Визначити параметр $k$, який залежить від ємності несучого середовища, а також довжини секретного повідомлення.
- Обчислити довжину кодового слова $n = 2^k - 1$.
- Вбудувати секретне повідомлення за допомогою $(1, \mathrm{n}, \mathrm{k})$ кодування матриці.
- Заповнити буфер $n$ ненульовими коефіцієнтами.
- Хешувати буфер (згенероване значення хеш-функції складається з $k$ біт).
- Додати наступні $k$ бітів повідомлення до значення хешу (за допомогою операції XOR).
- Якщо сума дорівнює 0, буфер залишається незмінним. Інакше сума вказує на елемент буфера, абсолютне значення якого потрібно зменшити на одиницю.

- При отриманні нуля перевіряється скорочення. Якщо скорочення має місце, налаштувати буфер (виключити 0, прочитавши ще один ненульовий коефіцієнт). Якщо скорочення не відбулося, здійснюється перехід до нових коефіцієнтів фактичного буфера.
- Продовжити стиснення JPEG (кодування Гаффмана тощо).

У експерименті в якості контейнерів було використано набір даних «Linnaeus 5 256X256» [14], формату JPEG загальною кількістю 8000 цифрових зображень. Розмір кожного із зображень 256×256 пікселів. У якості таємного повідомлення було використано псевдовипадкову бітову послідовність із рівномірним розподілом.

## 2.2 Побудова статистичного класифікатора

З метою розробки більш універсального підходу було обрано метод стеганоаналізу, що належить до так званих евристичних методів, які зазвичай базуються на вирішенні завдань бінарної класифікації із застосуванням методів машинного навчання. Ключовим елементом методу є оцінка статистики розподілу пар коефіцієнтів ДКП зображення та стегозображення.

Евристичні методи стеганоаналізу є дуже перспективним напрямом з огляду на їх універсальність, оскільки вони не прив’язані до якогось конкретного алгоритму впровадження прихованої інформації, хоч і дещо менш точні в цілому. Окремим, досить цікавим та перспективним напрямком розвитку евристичних методів стеганоаналізу є штучні нейронні мережі (ШНМ), біологічним прототипом яких є нервова система живих організмів.

Штучні нейронні мережі (англ. artificial neural networks, ANN), або конективістські системи (англ. connectionist systems) — це суперпозиція великої кількості лінійних нейронів. Суперпозиція у математиці — це функція від функції від функції [15]. У даному разі під функціями слід розуміти нейрони. Існують нейрони, які приймають на вхід ознаки об’єкта. Вони утворюють перший шар

мережі (рис. 2.4). На виході кожен такий нейрон дає одне число, тому всі вони формують вектор нових ознак об’єкта. Розмірність цього вектора дорівнює числу нейронів у першому шарі. Продовжуючи діяти так само, можна побудувати другий шар — це нейрони, які вхід приймають не оригінальні ознаки, а сформовані першим шаром. Для кожної конкретної задачі кількість шарів та нейронів може відрізнятись. Зазвичай чим складніше завдання, тим більше шарів нам знадобиться, але і даних теж потрібно буде більше для навчання всіх вагових коефіцієнтів у всіх нейронах.

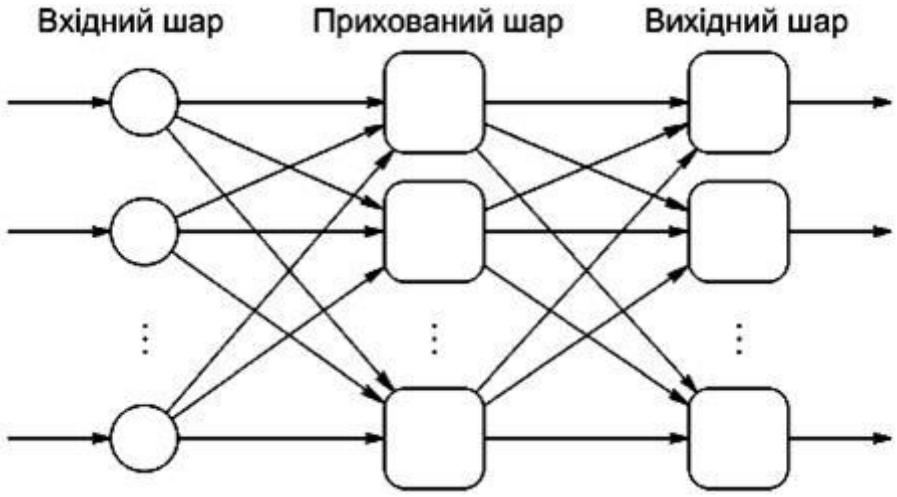


Рисунок 2.4 — Узагальнена топологія нейронних мереж

Машинне навчання загалом побудовано на вирішенні оптимізаційних завдань. Ми задаємо критерій, який хочемо мінімізувати, наприклад сумарну помилку мережі на векторах навчальної вибірки, і запускаємо алгоритм, який крок за кроком змінює коефіцієнти по всій мережі, поступово розгортаючи гіперплощину кожного нейрона в найкраще положення. Зазвичай це робиться методом градієнтного спуску (gradient descent).

Кожен шар нейронів перетворює вхідний вектор у вихідний, причому розмірність вихідного вектора дорівнює числу нейронів у шарі. Таких перетворень послідовно можна виконати багато, і це називається багатошаровою (глибокою) нейронною мережею. Кожен шар — це набір лінійних класифікаторів,

які спеціалізуються на одному відносно простому перетворенні вектора. Разом вони утворюють щось на кшталт конвеєра.

В експерименті розглядається проблема бінарної класифікації, для її вирішення було використано мережу типу багатошаровий перцептрон із трьома прихованими шарами та зворотнім поширенням помилки. На вхід до мережі подається два масиви:

- масив статистики розподілу пар коефіцієнтів ДКП;
- масив очікуваних вихідних значень нейронної мережі. Якщо контейнер заповнений, то мережа видає значення «1», в іншому випадку – «0».

На момент виконання цього кроку ми маємо 8000 цифрових зображень, у які була вбудована додаткова інформація за алгоритмом F5 та таку ж кількість зображень, яка не піддавалася стеганографічному вбудовуванню.

Масив розподілу значень пар коефіцієнтів ДКП формується за наступним алгоритмом:

1. Із набору даних обирається монохромне цифрове зображення $Im_{(n,m)}$, де $n, m$ — його ширина та висота.
2. Виконується ДКП пікселів зображення, у результаті якого отримуємо масив коефіцієнтів тієї ж розмірності.
3. Після виконання квантування коефіцієнтів отримуємо цілі значення.
4. Формується масив зсувів з 10 елементів типу $(i', j')$, де $i', j'$ — координатний зсув для обрання парного коефіцієнта.
5. Будується нульова матриця $A$, розмірність якої дорівнює подвоєному максимальному коефіцієнту ДКП після квантування (без урахування DC коефіцієнтів).
6. Для кожного коефіцієнта $K$ з координатами $i, j$, який не є нульовим чи DC коефіцієнтом, визначаємо парний коефіцієнт $K_{(i+i', j+j')}$. Далі будемо позначати пари як $(p, q)$.
7. Виконується інкремент значень $A_{(p,q)}$, для кожної пари коефіцієнтів.

8. Виконуємо нормалізацію відносно кількості блоків ДКП (використовуємо блоки 8×8):

$$A_{\text{н}} = A/\left(\frac{n}{8} + \frac{m}{8}\right)$$

9. Повторюємо кроки 5-8 для кожного з 10 зсувів.
10. Об’єднуємо отримані матриці в одну, яка й буде характеризувати зображення.

## 2.3 Перевірка ефективності статистичного класифікатора

Перед тренуванням нейронної мережі дані розбиваються на тренувальну та тестову. Після завершення процесу навчання на основі тренувальній множині виконується перевірка результату на тестових даних. Встановлюється порогове значення. Для оцінки ефективності запропонованого алгоритму на тренувальних та тестових даних використані ймовірності помилок першого та другого роду. Чим менше значення помилок, тим вище ефективність отриманої мережі. У якості вихідної гіпотези $H_0$ виступає припущення, що контейнер порожній, а $H_1$, відповідно, що у контейнері є приховане повідомлення. Ймовірності вибору та вірності цих гіпотез наведено у табл. 2.1.

Таблиця 2.1 — Ефективність класифікатора

| Гіпотеза, що була обрана мережею | Правильна гіпотеза | |
|---|---|---|
| | $H_0$ | $H_1$ |
| $H_0$ | 0,949 | 0,039 |
| $H_1$ | 0,051 | 0,961 |

З наведених результатів видно, що ймовірність успішного виявлення наявності додаткової інформації складає 96%, а детектування порожнього

контейнера майже 95%. Відповідно помилок першого роду — 5% та другого 4%. Такі значення дозволяють мережі надійно виявляти факт вбудовування додаткової інформації.

## 2.4 Побудова генеративно-змагальної мережі

Генеративно-змагальні мережі складаються з двох нейронних мереж, одна навчена генерувати дані, а інша навчена відрізняти підроблені дані від реальних даних. Хоча ідея структури для генерації даних не нова, коли справа доходить до створення зображень та відео, мережі ГЗМ дають вражаючі результати.

Можливості ГЗМ у стеганографії можна розглядати з різних боків, наразі існують три основні напрями розвитку: змагальна гра, генератор або функція відображення. Вони узгоджуються із класифікацією основних стратегій у стеганографії, тобто модифікації, синтезу та селекції [16].

1. Для вирішення задачі протидії класифікатору необхідно вирішити проблему зміни статистичних показників контейнеру. Модифікація зображення, заснована на ГЗМ, фокусується на змагальній грі між стеганографом і стеганалізатором [17]. Даний підхід використовує генератор, навчений для побудови різних ключових елементів. Цей підхід дозволяє створювати більш захищене від стеганалізу стегоповідомлення, яке може передавати по відкритому каналу зв’язку. Фактично генеративно-змагальна мережа складається з трьох мереж: генератор G, дискримінатор D та класифікатор S стеганалізатору (рис. 2.5).

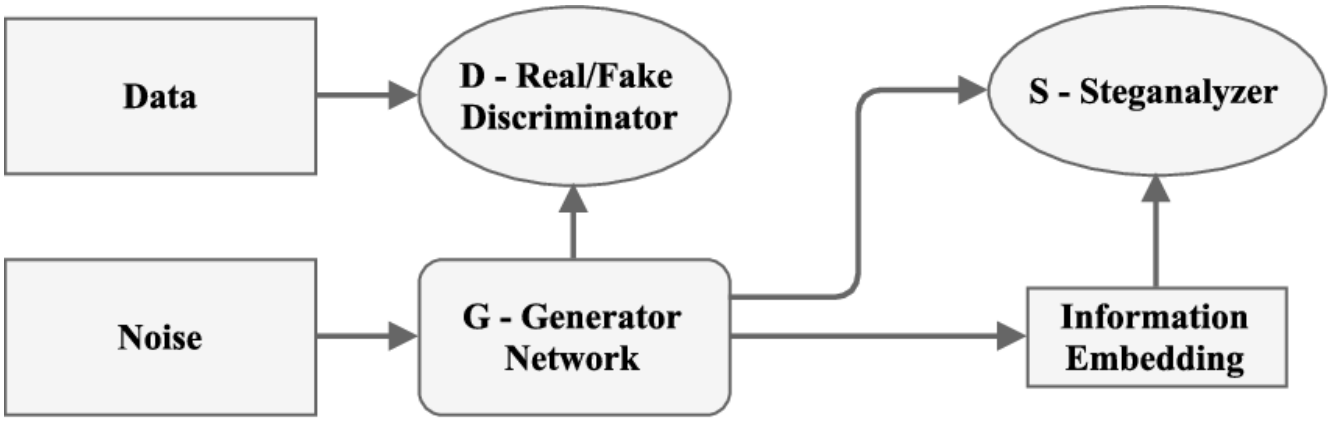

Рисунок 2.5 — Структурна схема SGAN

На рис. 2.6 зображена запропонована схема навчання ГЗМ для модифікації контейнера з використанням стеганографічного методу F5. Модифікація контейнера перед вбудовуванням прихованого повідомлення проводиться таким чином, щоб після вбудовування за статистичними та візуальними характеристиками контейнер був максимально наближений до порожнього.

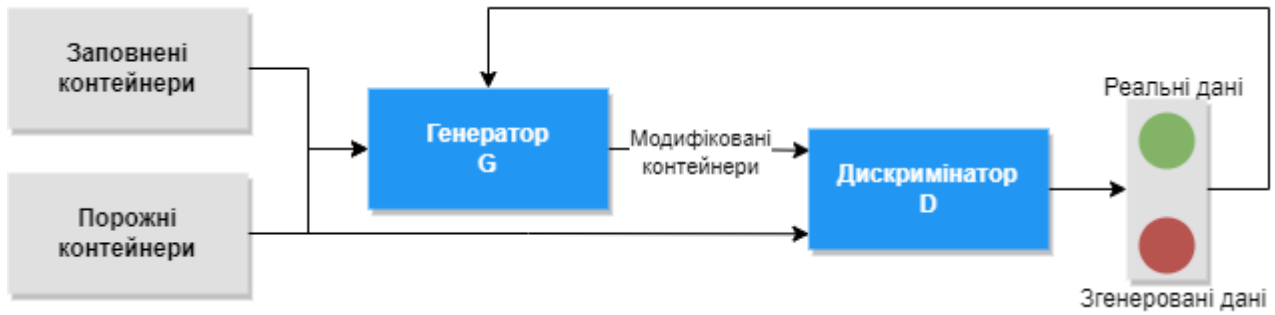


Рисунок 2.6 — Структурна схема запропонованої ГЗМ

Схема ГЗМ (рис. 1.6) була модифікована наступним чином: замість деякого латентного вектору, на вхід генератора подаються порожній та заповнений контейнери. Створені таким чином модифіковані контейнери після вбудовування секретного повідомлення будуть демонструвати кращі результати, але це можливо лише при заздалегідь відомому повідомленні.

На рис. 2.6 продемонстрована спрощена схема роботи стеганографічної системи. На рисунку можна побачити, що після вбудовування повідомлення в контейнер статистичний класифікатор (стеганоаналізатор) перевіряє контейнер з повідомленням. Якщо на виході він не виявляє факт вбудовування ДІ, то вважаємо, що повідомлення вбудовано успішно та готово для подальшої передачі по відкритому каналу зв’язку. Якщо класифікатор виявляє повідомлення, то застосовуємо ГЗМ, якій передаємо оригінальний та заповнений контейнери для

генерації модифікованого контейнера. Після чого здійснюємо повторне вбудовування, але вже у модифікований контейнер.

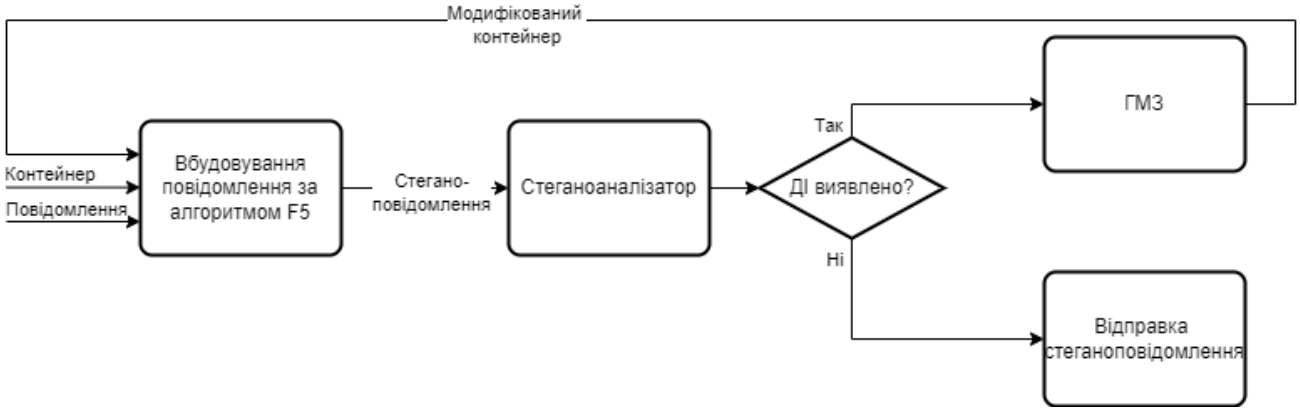


Рисунок 2.6 — Структурна схема запропонованої стеганографічної системи

Мережа дискримінатора є стандартною згортковою мережею, яка може класифікувати зображення. Генератор у певному сенсі є зворотною мережею згортки.

Згорткою називається нейрон, який здійснює зважене усереднення яскравостей по сусідніх пікселях. Якщо згортку однаково застосувати до всіх пікселів зображення, то може вийти трохи розмите зображення або зображення, на якому яскравіше висвітлилися певні переходи кольорів. Як правило, до одного зображення застосовуються одночасно десятки або сотні різних згорток, а виході отримують масив зображень. Шари згорткових нейронів чергуються з шарами пулінгу, що є більш примітивними нейронами, які не мають вагових коефіцієнтів: вони зменшують розмір зображень у кілька разів.

Кожна наступна пара шарів згортка + пулінг зменшує розмір зображення, підвищує розмірність вектора у кожному пікселі і виділяє більші елементи зображень, ніж попередня пара. На виході з конвеєра розмір зображення стягується в точку 1×1, однак вона представляється вектором досить великої розмірності, наприклад 4000, у якому закодована інформація про всі «важливі» об’єкти на зображенні. У процесі градієнтної оптимізації згорткові шари, що

утворюють конвеєр, підлаштовуються один під одного так, щоб фінальний вектор зображення був максимально «корисний» для безпомилкового розпізнавання.

Тренування ГЗМ виконується наступним чином:

1. Для тренування генератора знову використовується набір зображень «Linnaeus 5 256X256», який розбивається на 3 множини: тренувальну, тестову та валідаційну. На тестову та валідаційну вибірки відводиться по 20% від тренувальної, тобто 1600 та 1280 відповідно.
2. Будується генеративно-змагальна мережа.
3. Під час її тренування за допомогою алгоритму F5 проводиться вбудовування повідомлення у кожний контейнер. Отримані стеганоповідомлення разом із порожніми контейнерами подаються на вхід генеративної мережі.
4. Дискримінатор приймає як реальні, так і модифіковані зображення і повертає ймовірності від 0 до 1, причому 1 являє собою справжнє зображення і 0 представляє фальшиве.

## 2.5 Тестування ефективності створеної генеративно-змагальної мережі

Для оцінки ефективності запропонованого алгоритму знову будемо використовувати ймовірності помилок першого та другого роду.

Помилки першого роду демонструють, з якою ймовірністю класифікатор під час аналізу порожнього контейнера покаже, що він заповнений. Помилки другого роду демонструють, з якою ймовірністю класифікатор покаже, що цей контейнер є порожнім. Чим ближче значення помилок до 0.5, що відповідає випадковому вгадуванню, тим менш дієвим буде стеганоаналіх, і тим більшою буде ефективність запропонованої ГЗМ.

Таблиця 2.2 — Ефективність запропонованого методу

| Гіпотеза, що була обрана мережею | Правильна гіпотеза |
|---|---|

| | $H_0$ | $H_1$ |
|---|---|---|
| $H_0$ | 0,599 | 0,401 |
| $H_1$ | 0,401 | 0,599 |

З отриманих результатів видно, що ймовірність успішного виявлення значно зменшилась у порівняння із вбудовуванням без попередньої модифікації. Ймовірність помилок першого та другого роду збільшилась на ~36%. Такі значення більше не дозволяють класифікатору надійно виявляти факт вбудовування секретного повідомлення.

# 3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ МОДИФІКАЦІЇ КОНТЕЙНЕРА

## 3.1 Вибір мови та середовища програмування

Для тренування нейронних мереж був використаний сервіс від Google — Google Colab. Програмний продукт був розроблений на мові Python за допомогою середовища Pycharm.

Google Colab (також відомий як Colaboratory) — це безкоштовний хмарний сервіс на базі Jupyter Notebook [18]. Colab являє собою передове операційне середовище для швидкого виконання науково-технічних розрахунків будь-якої складності. Він надає все необхідне для машинного навчання прямо у браузері (рис. 3.1), дає безкоштовний доступ до швидких GPU та TPU, але з деякими обмеженнями. Використовувати відеокарту можна 12 годин на добу, тому для складніших завдань варто розглянути інші варіанти.

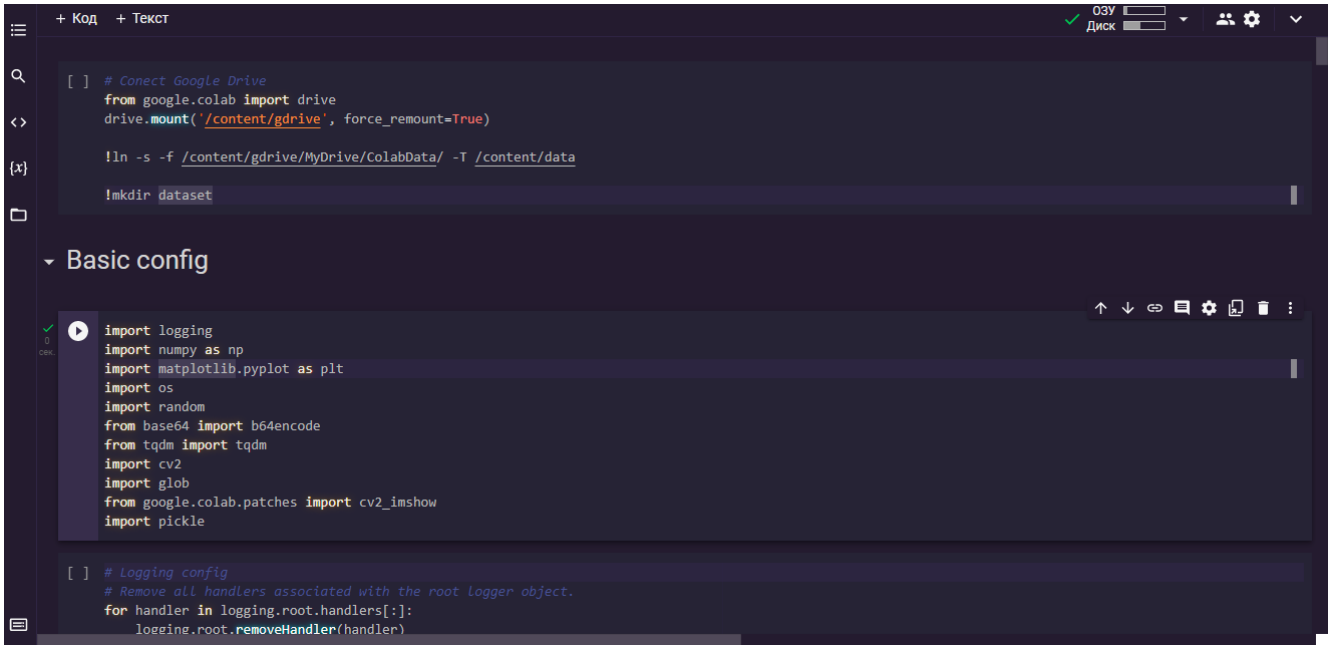


Рисунок 3.1 — Інтерфейс Google Colab

Colab розпочинався як внутрішній проект Google, згодом була зроблена спроба [19] перенести кодову базу на відкриту основу, що призвело до розробки

розширення «Відкрити у Colab» для Google Chrome [20] але цьому так і не судилося статися і розробка Colab продовжилася всередині компанії Google. Станом на грудень 2021 року інтерфейс Colaboratory дозволяє створювати блокноти лише на основі Python 3.

Jupyter Notebook — командна оболонка для інтерактивних обчислень на Python. Проект Jupyter був відокремлений з інтекративної оболонки IPython у 2014 році Фернандо Пересом і Браяном Грейнджером [21]. Назва проекту Jupyter є посиланням на три основні мови програмування, які підтримує Jupyter, а саме: Julia, Python і R.

Розробка програм з використанням нейронних мереж відрізняється від звичайної розробки. Для роботи з ними потрібен спеціальний стек технологій і спеціальні навички. Крім того, створення програм на основі нейронних мереж потребує глибоких досліджень. Для реалізації ідей, пов’язаних із застосуванням машинного навчання, зокрема, нейронних мереж, знадобиться надійна, гнучка мова програмування з багатим інструментарієм.

Python допомагає розробникам працювати продуктивно та впевнено, причому на всіх стадіях проекту, від розробки до підтримки. Ця мова має певні характеристики, які роблять її найкращим вибором для ML- та II-проектів: він простий і логічний, гнучкий і мультиплатформний, має відмінні бібліотеки та фреймворки для машинного навчання, а ще за ним стоїть численна спільнота розробників. Завдяки цьому Python є однією з найпопулярніших мов програмування у світі.

Python досить простий для роботи та читання: його використання знижує вартість розробки та обслуговування програм. Крім простоти, Python має ще один плюс — він досить легко взаємодіє з іншими мовами, особливо з C і C++.

Одна з основних причин, чому Python використовується для машинного навчання, зокрема генеративно-змагальних мереж, полягає в тому, що у нього є безліч фреймворків і окремих бібліотек, які спрощують процес написання коду, наприклад:

- Keras, TensorFlow та Scikit-learn — для машинного навчання;
- Pandas — для загального аналізу даних;
- Seaborn — для візуалізації даних;
- NumPy — для високопродуктивних наукових обчислень та аналізу даних;
- SciPy — для просунутих обчислень;
- Scikit-learn пропонує різні алгоритми класифікації, регресії та кластеризації, включаючи допоміжні векторні машини, випадкові ліси, підвищення градієнта та DBSCAN. Ця бібліотека призначена для роботи з числовими та науковими бібліотеками Python NumPy та SciPy.

За допомогою цих готових рішень є можливість створювати свої продукти набагато швидше.

Наступна перевага Python у машинному навчанні полягає у його гнучкості: наприклад, розробник має вибір між об’єктно-орієнтованим підходом і скриптами. Python допомагає поєднувати різні типи даних. Більш того, Python особливо зручний для тих розробників, які більшу частину коду пишуть за допомогою IDE.

Мультиплатформенність (у нашому випадку) — це властивість мови програмування або фреймворку, що дозволяє розробникам переносити програмне забезпечення на різні машини з мінімальними змінами або без змін зовсім [22].

Однією з причин популярності Python є те, що ця мова від платформ не залежить, тому що підтримується багатьма з них, включаючи Linux, Windows та MacOS. Код Python може використовуватися для створення програм для більшості операційних систем, а це означає, що програмне забезпечення Python легко розповсюджувати та використовувати в цих системах без спеціальних інтерпретаторів.

## 3.2 Реалізація графічного інтерфейсу

Для вбудовування секретного повідомлення, виконання статистичного стеганоаналізу та модифікації цифрових зображень на мові програмування Python було реалізовано користувацький інтерфейс, представлений на рисунку 3.2.

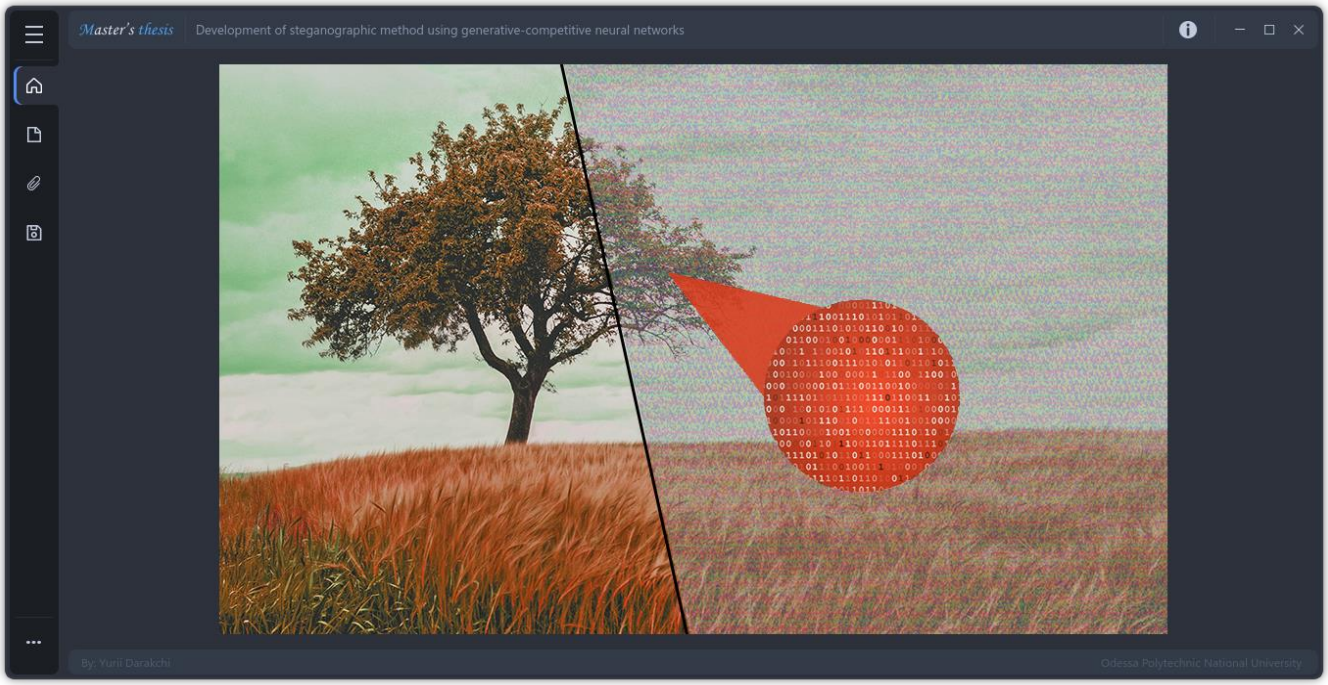


Рисунок 3.2 — Головна сторінка створеного додатку

Даний інтерфейс було розроблено засобами PyQt, а точніше його повністю безкоштовної альтернативи — PySide.

PySide — прив’язка мови Python до інструментарію Qt, сумісна на рівні API з PyQt. На відміну від PyQt, PySide доступна для вільного використання як у відкритих, так і закритих, зокрема, комерційних проектах, оскільки ліцензована LGPL [23]. PyQt — це прив’язка Python з відкритим вихідним кодом для віджет-інструментарію Qt, який також функціонує як крос-платформне середовище розробки програм [24]. Qt — це популярне середовище C++ для написання програм за допомогою графічного інтерфейсу для всіх основних настільних, мобільних та вбудовуваних платформ (підтримує Linux, Windows, MacOS, Android, iOS, Raspberry Pi та багато інших).

PyQt розробляється та підтримується Riverbank Computing, компанією, що базується в Англії, а Qt — фінською фірмою The Qt Company.

PyQt доступний у двох редакціях: PyQt4 та PyQt5. PyQt5 не має зворотної сумісності зі застарілими модулями більш ранньої версії. Крім цих двох версій, Riverbank Computing також надає PyQt3D – прив'язки Python для інфраструктури Qt3D. Qt3D — це прикладне середовище, що використовується для створення систем моделювання е реальному часі з 2D/3D рендерингом.

У центрі знаходиться область для відображення поточного цифрового зображення. Під час зчитування будь-яке зображення автоматично перетворюється на зображення у градаціях сірого.

Зліва знаходиться меню із наступними кнопками:

- «Home» — кнопка, що допомагає повернутись на головну сторінку та закриває усі додаткові меню.
- «Choose container» — кнопка, що призначена для обрання графічного файлу, який буде використано у якості контейнера, за допомогою файлового меню (рис. 3.3)

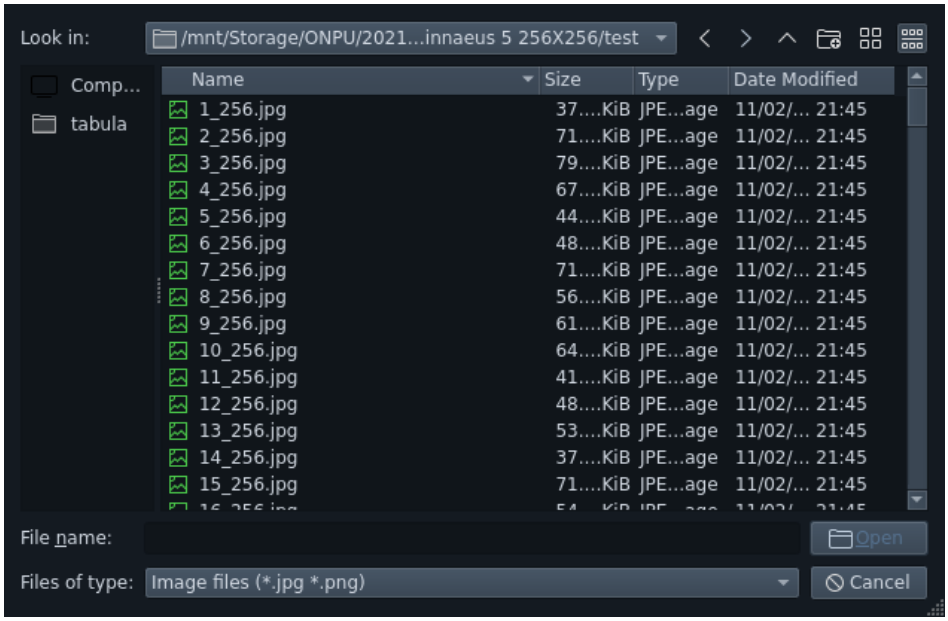


Рисунок 3.3 — Файлове меню

- «Choose message» — кнопка, яка дозволяє користувачеві обрати текстовий файл із повідомленням. Надалі дане повідомлення може бути вбудоване у контейнер після натискання відповідної кнопки.
- «Save» — кнопка, яка зберігає поточне стеганоповідомлення чи модифікований контейнер в обраній директорії.

Справа зверху знаходиться кнопка для відкриття інформаційного меню, що містить (рис. 3.4):

- Назву зображення-контейнера
- Розмір контейнера
- Кількість каналів цифрового зображення
- Ім’я файлу повідомлення
- Розмір повідомлення у байтах
- Візуальну критерії оцінки отриманого стеганоповідомлення

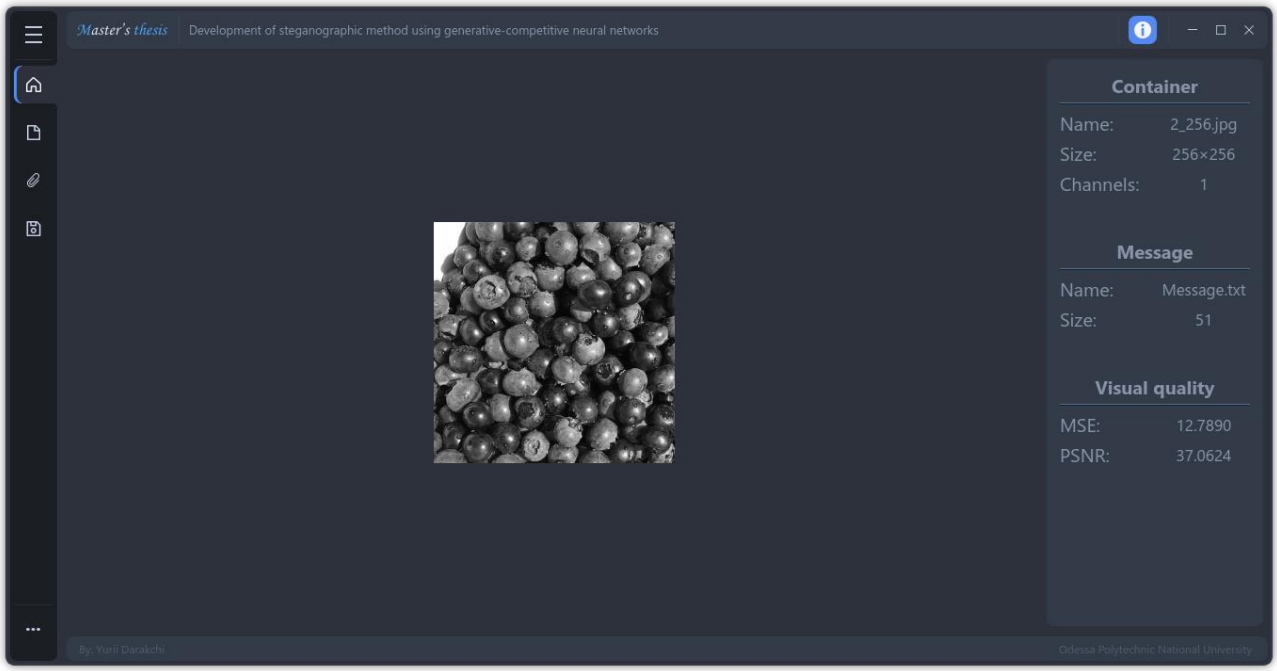


Рисунок 3.4 — Інформаційне меню

У нижньому лівому куті знаходиться кнопка для модифікації контейнера. Вона відкриває меню, зображене на рис 3.5.

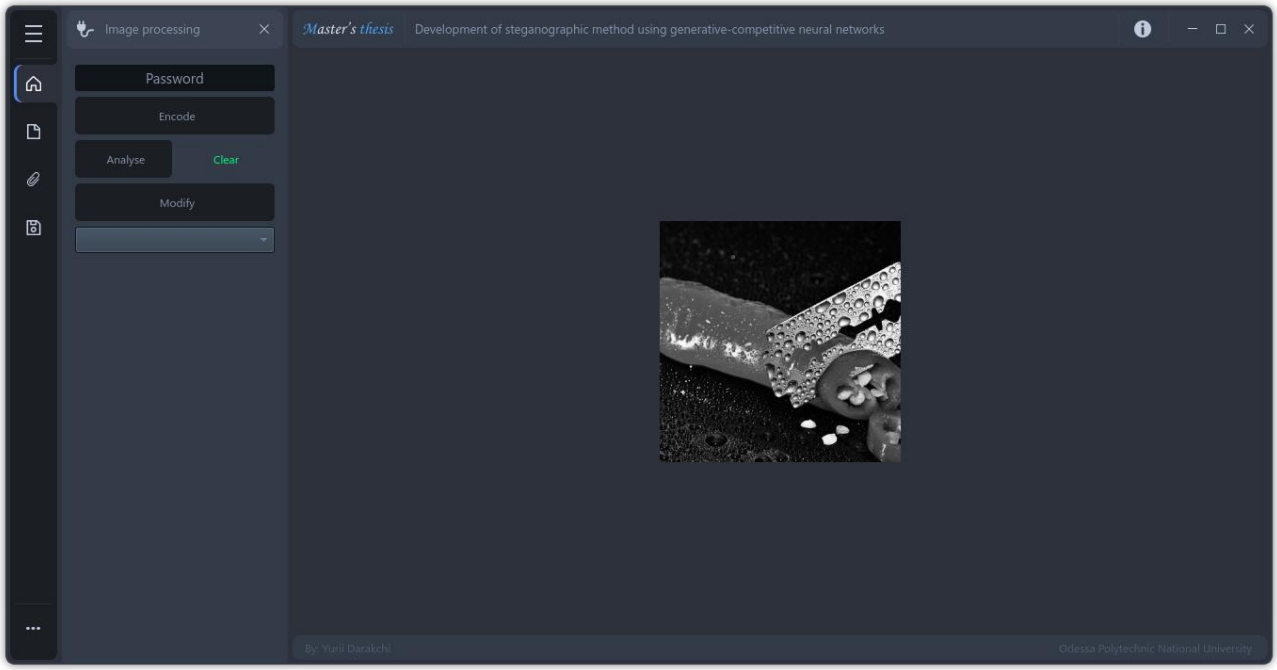


Рисунок 3.4 — Меню модифікації зображення

Дане меню містить наступні елементи:

- Текстове поле для введення пароля, який буде використовувати алгоритм F5.
- Кнопка «Encode» для вбудовування повідомлення у поточний контейнер.
- Кнопка «Analyse» для перевірки контейнеру за допомогою класифікатора, описаного у розділі 2.2.
- Поле, що повідомляє про результат аналізу.
- Кнопка «Modify» для модифікації контейнера за допомогою генеративно-змагальної мережі.
- Випадний список для вибору типу представлення зображення.

Даний метод організації дозволяє забезпечити мінімалістичний інтерфейс користувача, що одночасно буде повністю виконувати поставлену задачу та не буде занадто складним. Загалом інтерфейс надає можливість користувачу вбудовувати текстове повідомлення у цифрове зображення алгоритмом F5,

провести детектування наявності стеганоповідомлення та модифікувати контейнер за допомогою ГЗМ. Код застосунку наведено у Додатку А.

# 4 ОХОРОНА ПРАЦІ І БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

## 4.1 Аналіз умов праці і вибір заходів і засобів захисту від небезпечних і шкідливих виробничих факторів

Охорона праці, як галузь практичної діяльності, спрямована на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності. В основу даної системи покладено закон «Про охорону праці», що містить норми і вимоги з техніки безпеки і виробничої санітарії тощо, а також законодавчі норми Конституції України (розділ II, стаття 43).

Робота з програмним забезпеченням, спроектованим і розробленим під час виконання даної магістерської кваліфікаційної роботи, буде проводиться фахівцем з кібербезпеки у середньостатистичному офісному приміщенні. Робоче місце людини, яка працює зі стеганографічними методами приховування інформації - їх розробкою, модифікацією, впровадженням – передбачає наявність та використання ПК.

Розглядається офісне приміщення площею 30 м$^2$, висота стелі — 3.5 м. У приміщенні знаходиться 4 робочих місця операторів ПК. Кожне з них відповідно обладнане усім необхідним устаткуванням для продуктивної роботи: робочим столом, ящиками з документацією, кріслом та персональним комп'ютером, що складається з монітора, відеокамери, клавіатури, мікрофона, миші, принтера та системного блоку. Відповідно до ДСанПіН 3.3.2.007-98 площа одного такого робочого місця становить не менше ніж 6.0 м$^2$, об'єм не менше ніж 20.0 м$^3$, а устаткування відповідає гігієнічним вимогам до організації і обладнання робочих місць з ВДТ ЕОМ та ПЕОМ (обладнання не містить та не використовує полімерні матеріали, що виділяють у повітря шкідливі хімічні речовини та ін.)

При розробці оптимальних умов праці фахівця з кібербезпеки необхідно враховувати і за можливості усувати потенційно небезпечні та деструктивні фактори, пов'язані зі специфікою робочого процесу. При систематичному впливі даних факторів на організм значно зростає ймовірність виникнення нещасних випадків та професійних захворювань. До найпоширеніших захворювань фахівців

з кібербезпеки належать: синдром зап’ястного каналу, анемія, мігрень, захворювання хребта, очей та серцево-судинної системи.

Поглиблений аналіз умов праці в офісному приміщенні показує, що, згідно з ГОСТ 12.0.003-74, у такому приміщенні на фахівця з кібербезпеки можуть негативно впливати наступні фізичні, біологічні та психофізіологічні фактори:

- занадто висока вологість повітря;
- підвищена або знижена температура повітря робочої зони;
- недостатня освітленість робочого місця, зокрема нестача природного світла;
- підвищений рівень електростатичного поля та електромагнітного випромінювання;
- висока ймовірність ураження струмом при замиканні фази на корпус комп’ютера;
- підвищений рівень шуму у робочій зоні;
- нервово-психічні перевантаження (розумова перенапруга);
- фізичні перевантаження (одноманітна чи незручна поза).

Під час аналізу впливу внутрішнього середовища робочої зони на організм працівника розглядають наступні параметри: відносну вологість, температуру повітря, швидкість руху повітря, інтенсивність теплових (інфрачервоних) опромінень. Допустимі нормативні значення мікроклімату встановлюються відповідно до ГОСТ 12.1.005-88 «Загальні санітарно-гігієнічні вимоги до повітря робочої зони», чинність якого було відновлено 24 квітня 2019 року, та ДСН 3.3.6.042-99 «Санітарні норми мікроклімату виробничих приміщень». Дані показники залежать від пори року і категорії (ступеня важкості) виконуваних робіт. Робота фахівця з кібербезпеки в офісному приміщенні за енерговитратами класифікується як робота першої категорії. Це сидяча робота на постійному робочому місці, яка не потребує значного фізичного напруження, енерговитрати при цьому не перевищують 120 ккал/год. У даному випадку згідно «Санітарним нормам мікроклімату виробничих приміщень» ДСН 3.3.6.042-99 повинні дотримуватися наступні вимоги.

У холодний період року:

- оптимальна температура повітря – 22-24°С;
- оптимальна відносна вологість – 40-60%;
- швидкість руху повітря не більш 0,1 м/с.
- У теплий період року:
- оптимальна температура повітря – 23-25°С;
- оптимальна відносна вологість – 40-60%;
- швидкість руху повітря не більш 0,1 м/с.

Для підтримання у приміщенні в холодний період року оптимальних параметрів температури повітря використовуються опалювальні радіатори, а у теплий період кондиціонування.

Освітлення у приміщенні може бути штучним, природним або комбінованим. Приміщення з робочими місцями, обладнаними персональними комп'ютерами, повинні мати як природне освітлення, так і бути облаштоване системою штучного освітлення (відповідно до пункту 2.4 ДСанПіН 3.3.2.007-98).

Природне освітлення забезпечують прямі сонячні промені й дифузне світло небосхилу та здійснюється через світлові отвори в стінах (вікна) або/та через ліхтарі та отвори в дахах і перекриттях. Нормою природного освітлення згідно ДБН В.2.5-28:2018 є коефіцієнт природного освітлення, що має відповідати розряду виконуваних зорових робіт та бути не нижче ніж 1,5 %. При роботі фахівця з кібербезпеки мінімальний еквівалентний розмір об'єкта розрізнення складає близько 1.0 мм (найменший символ на клавіатурі), а це робота середньої точності (IV розряд).

Штучне освітлення забезпечується за рахунок світильників. При загальному освітленні світильники розміщуються у верхній зоні приміщення не нижче 2.5 м над підлогою. Для місцевого освітлення робочих місць використовуються світильники з відбивачами. При цьому важливо, щоб елементи, які світяться, не влучали в поле зору працюючих, а яскравість джерел світла, що опиняються у полі зору працівника, не має перевищувати 200 кд/м$^2$. Застосування лише

місцевого освітлення, а також без розсіювачів та екрануючих гратів не допускається з огляду на небезпеку збільшення ймовірності професійних захворювань та виробничого травматизму. Для даної категорії зорових робіт мінімальна освітленість штучним освітленням складає 300 люкс. Недостатня освітленість робочої зони може стати причиною виникнення серйозних проблем із зором.

Наступним небажаним фактором є підвищений рівень електростатичного поля. Він відноситься до одних з основних шкідливих факторів, що можуть впливати на організм людини під час роботи з комп'ютером. Головним джерелом даного поля є монітори, вони також є джерелом м'якого рентгенівського, ультрафіолетового, інфрачервоного та електромагнітного випромінювання. Електростатичне поле великої напруженості здатне змінювати і переривати клітинний розвиток, а також викликати катаракту з наступним помутнінням кришталика. Нині більшість виробників ПК установлюють жорсткі нормативи до якості дисплеїв, але це не гарантує повне позбавлення від негативного впливу електростатичного поля. Щоб убезпечити себе від нього необхідно використовувати комп'ютери з елементами захисту моніторів та маркуванням про низький рівень випромінювання LR — low radiation, за відсутності такого маркування необхідно встановити відповідних захисний екран, а також дотримуватися регламентованих режиму праці та відпочинку. Якщо зовнішня поверхня екрана заземлена, його електростатичний потенціал знижується: при сухому повітрі на 50%, при вологому більш ніж на 50%. Також необхідним є захисне покриття для стінок монітора за відповідними вимогами, а якщо моніторів у приміщенні багато, то відстань між їх тильними повехнями повинна бути не менше 2,5 м та 1,2 м між бічними [25].

Наступним небезпечним виробничим фактором є підвищене значення напруги. Оскільки у приміщенні оператора ПК усі пристрої підключені до мережі 220В, то є ризик зазнати удару струмом. Щоб уникнути нещасних випадків та небажаних травм слід дотримуватись наступних правил:

- корпуса пристроїв, що знаходяться під напругою мають бути виготовлені із діелектриків, щоб унеможливити поразку струмом при дотику до них;
- уся електропроводка має бути закритою, штепсельні розетки мають знаходитись не нижче метра від підлоги;
- проводка та всі електроприлади повинні бути не ближче ніж 50 см від печей, опалювальних приладів, труб та інших джерел тепла.

Наступним негативним фактором може стати підвищений рівень шуму. Як було вказано раніше, у приміщення знаходиться 4 робочих місць з ПК, кожне з яких устатковане монітором, системним блоком (жорсткий диск, 3 вентилятори, привід DVD-дисків) мишею, принтером, камерою з мікрофоном та клавіатурою. Крім того поряд може працювати інша периферійна техніка. Таким чином у приміщенні мають місце змішані шуми із підсиленням при роботі принтерів. Рівень шуму при цьому не має перевищувати допустимий для робочого місця оператора ПК, який дорівнює 65 дБ (ДБН В.1.1-31:2013).

Для додаткового зниження рівня шуму стіни і стеля приміщень, де встановлені комп’ютери, можуть бути облицьовані звукопоглинальними матеріалами. Також можна видати працівникам навушники.

Напружена та монотонна робота з часом можуть призвести до нервово-психічних перевантажень (перенапруження аналізаторів, розумова перенапруга). У свою чергу вони призведуть до розвитку фізичної та емоційної перевтоми організму, порушення координації, втрати пильності та відчуття реальності. Працівник стає дратівливим, зменшується концентрація, знижується рівень відчуття зовнішніх подразників та небезпеки. Протистояння нервово-психічним перевантаженням зводиться до дотримання, перерахованих вище, санітарних норм, а це зниження шуму і опромінення, достатнє освітлення робочих місць, забезпечення надходження свіжого повітря, підтримання належного рівня вологості і температури. Також слід пам’ятати про правильну організацію робочого місця та робочого процесу. Ритм повинен бути розміреним, у працівників має бути раціональний режим праці і відпочинку, певна емоційна

стимуляція під час робочого процесу, а також усе необхідне дня існування в межах офісу протягом робочого дня тощо.

## 4.2 Аналіз техногенних небезпек, вибір заходів і засобів забезпечення безпеки у надзвичайних ситуаціях

Техногенна небезпека – це внутрішній стан технічної системи, що реалізується у вигляді негативних впливів на людину і навколишнє середовище при виникненні надзвичайної ситуації, або у вигляді опосередкованої чи прямої шкоди для людини і навколишнього середовища, викликаних дією або бездіяльністю людини під час нормальної експлуатації цієї системи. До надзвичайних ситуацій техногенного характеру відносяться аварії, які супроводжуються викидами небезпечних хімічних речовин, пожежі, вибухи, аварії в інженерних мережах, раптове руйнування будівель та споруд тощо [26].

Згідно із загальновідомою статистикою найбільш ймовірною техногенною загрозою в офісних приміщеннях, зокрема у якому розміщені робочі місця фахівців з кібербезпеки, є пожежі.

За пожежною небезпекою приміщення класифікуються відповідно до ДСТУ Б В.1.1-36:2016. До матеріалів, які несуть пожежну небезпеку відносять: технологічне обладнання, електрокабелі, меблі, документи тощо.

Робоче місце фахівця з кібербезпеки знаходиться в офісному приміщенні, де наявні усі перераховані вище небезпечні матеріали. Зважаючи на це, віднесемо дане приміщення до категорії В.

Поширені причини виникнення пожеж в офісних приміщеннях:

- Короткі замикання. Вони виникають через перенапруження у мережі, пошкоджену ізоляцію. Особливо часто з цієї причини пожежі трапляються у будівлях зі старою проводкою або порушеннями правил її експлуатації.
- Використання несправного електроустаткування. Вимикачі, розетки, техніка із пошкодженою ізоляцією або несправні прилади.

- Застосування обігрівачів відкритого типу. Їх особливо небезпечно використовувати у приміщеннях, де зберігається велика кількість паперових (документація, архіви та ін.), горючих і легкозаймистих матеріалів.
- Паління у недозволених місцях.
- Використання піротехнічних пристроїв, особливо під час проведення корпоративних заходів, інших свят.

Для запобігання перерахованих причин пожежна безпека в офісному приміщенні має забезпечуватися шляхом проведення технічних та організаційних заходів, спрямованих на підтримання стану захищеності персоналу, а також мінімізацію негативних наслідків. Для До них можна віднести регулярну профілактику системи електропроводки, проведення інструктажів щодо поводження з горючими матеріалами або джерелами займання і поведінкою по настанню пожежі, застосування протипожежної сигналізації тощо.

Згідно закону «Про затвердження Правил пожежної безпеки в Україні» діяльність із забезпечення пожежної безпеки є невід'ємною частиною трудової діяльності працівників підприємств та окремих об'єктів, а також посадових осіб [27]. За забезпечення пожежної безпеки відповідає особисто керівник підприємства, установи або призначена ним особа, якщо інше не передбачено відповідним договором. На кожному підприємстві наказом має бути встановлений протипожежний режим. Також має бути організована система оповіщення про пожежу, яка повинна забезпечувати передавання сигналів оповіщення по всій будівлі. Порядок використання систем оповіщення необхідно визначати в інструкціях з їх експлуатації та в планах евакуації, де потрібно також указувати осіб, котрі мають право приводити систему в дію.

Вогнегасники та інші засоби пожежогасіння, повинні знаходитись на видному та легкодоступному місці. Щоб забезпечити швидку та безпечну евакуацію людей, що знаходяться у приміщенні, евакуаційні шляхи завжди мають бути вільні, а евакуаційні виходи незамкненими (двері евакуаційних виходів можуть зачинятися лише на внутрішні запори, які відчиняються зсередини без

ключа). Більш докладно заборони викладено у вимогах пункту 4.3.11 Правил пожежної безпеки в Україні:

- не допускається розташовувати на шляхах евакуації пристрої, які перешкоджають вільній евакуації людей: турнікети, будь-які виступи або пороги, двері розсувні, підйомні, такі, що обертаються, та інші;
- заборонено замикати на болтові з'єднання, навісні замки та інші запори, що відчиняються зсередини, а також зварювати зовнішні евакуаційні двері будівель;
- заборонено перешкоджати вільному відкриттю зовнішніх евакуаційних дверей шляхом зварювання, замикання на навісні замки, болтові з'єднання та інші запори, що відчиняються зсередини.

Нормативні характеристики евакуаційних шляхів та виходів наведені у ГОСТ 12.6.004-76 та у законі України «Про затвердження Правил пожежної безпеки в Україні». Усі шляхи евакуації мають бути забезпечені відповідним евакуаційним освітленням. Нормативні характеристики евакуаційних шляхів:

- висота дверних отворів — не менше 2 м;
- ширина дверних отворів — не менше 0.8 м;
- ширина проходу для евакуації — не менше 1 м;
- ширина евакуаційного коридору — не менше 2 м;
- ширина сходів — не менше 1 м;
- висота перил сходів — не менше 1 м.

## 4.3 Проектування системи освітлення

Згідно із вимогами охорони праці, одним із базових засобів колективного захисту, призначеним для підвищення продуктивності праці, попередження зорової втоми та підтримання психологічного тонусу працівників у приміщеннях із обчислювальною технікою є нормалізація освітлення робочих місць, зокрема системи загального штучного освітлення.

Вихідні дані для її проектування:

- фактична освітленість на робочому місці: $E_ф$ = 140 лк;
- розміри приміщення: довжина $A$ = 24 м; ширина $B$ = 12 м;
- розрахункова висота підвісу світильника: $H$ = 3 м;
- найменший розмір об'єкта розрізнення: 0,6 мм;
- контраст об'єкта розрізнення з фоном: середній, фон темний;
- система освітлення: загальна, виконується світильниками ППД з лампами розжарення та косинусною кривою сили світла;
- концентрація пилу в повітрі: 3,5 мг/м$^3$;
- коефіцієнти відображення: 0,5 – 0,3 – 0,1.

За найменшим розміром об'єкта розрізнення визначаємо розряд зорової роботи: IV(б) розряд, роботи середньої точності. Для IV розряду зорової роботи із використанням ламп розжарення норма освітленості на робочому місці становить: $E_н$ = 150 лк. Таким чином, фактична освітленість на робочих місцях недостатня: $E_ф$ = 140 лк, що менше нормативної освітленості $E_н$ = 150 лк. Тому існує необхідність модернізувати систему освітлення. Виконуємо її розрахунок:

Площа приміщення:

$$S = A \cdot B = 24 \cdot 12 = 288 \text{ м}^2$$

Індекс приміщення:

$$i = \frac{S}{(A + B) \cdot H} = \frac{288}{(24 + 12) \cdot 3} = 2{,}67$$

Еквівалентна площа:

$$S_e = \frac{S \cdot K \cdot z}{\eta} = \frac{288 \cdot 1{,}5 \cdot 1{,}15}{0{,}61} = 814{,}43 \text{ м}^2$$

$$\eta = \frac{i^2}{(i^2 + c) \cdot b} = \frac{7{,}13}{(7{,}13 + 0{,}56) \cdot 1{,}5} = 0{,}61$$

Оптимальна кількість світильників:

$$N_o = \frac{S}{{L_o}^2} = \frac{288}{17{,}64} = 16{,}33 \text{ шт.}$$

$$L_o = \lambda_o \cdot H = 1{,}4 \cdot 3 = 4{,}2 \text{ м}$$

Приймаємо $N_o$ = 17 шт.

Потрібний світовий потік світильників:

$$\Phi_c = \frac{S_e \cdot E_н}{N_o} = \frac{814{,}43 \cdot 150}{17} = 7186{,}15 \text{ лм}$$

Освітленість $E_1$, яка створюється одним світильником:

$$E_1 = \frac{n \cdot \Phi}{S_e} = \frac{1 \cdot 8300}{814{,}43} = 10{,}19 \text{ лк}$$

Попередня кількість світильників:

$$N_n = \frac{E_н}{E_1} = \frac{150}{10{,}19} = 14{,}72 \text{ шт.}$$

Приймаємо кількість світильників для монтажу $N$ = 15 шт.

Попередня відстань між світильниками:

$$L = \sqrt{\frac{S}{N_n}} = \sqrt{\frac{288}{15}} = 4{,}38 \text{ м}$$

Попереднє значення відношення відстані між світильниками до розрахункової висоти підвісу світильників:

$$\lambda = \frac{L}{H} = \frac{4{,}38}{3} = 1{,}46$$

Число рядів світильників вздовж сторони А:

$$N_A = \frac{A}{L} = \frac{24}{4{,}38} = 5 \text{ шт.}$$

Число рядів світильників вздовж сторони В:

$$N_B = \frac{N_n}{N_A} = \frac{15}{5} = 3 \text{ шт.}$$

Загальна кількість світильників:

$$N = N_A \cdot N_B = 5 \cdot 3 = 15 \text{ шт.}$$

Мінімальна освітленість

$$E = N \cdot E_1 = 15 \cdot 10{,}19 = 152{,}85 \text{ лк}$$

Відносне відхилення мінімальної освітленості від нормованої:

$$\varepsilon = \left(\frac{E}{E_{\text{Н}}} - 1\right) \cdot 100 = \frac{152{,}85}{150} \cdot 100 = 1{,}9\ \%$$

Відстань між світильниками вздовж сторони А:

$$L_A = \frac{A}{N_A} = \frac{24}{5} = 4{,}8 \text{ м}$$

Відстань між світильниками вздовж сторони В:

$$L_B = \frac{B}{N_B} = \frac{12}{3} = 4 \text{ м}$$

Згідно з розрахунком у приміщенні необхідно встановити загальну систему штучного освітлення з 15 світильників типу ПДД, кожен із 1 лампою розжарення, які рекомендується розташувати на стелі у 5 рядів по 3 світильники.

# ВИСНОВКИ

Під час проведення дослідження сучасних проблем та загроз інформаційної безпеки була розглянута потенційна можливість застосування генеративно-змагальних нейронних мереж у якості інструмента для підвищення стійкості до статистичного стеганоаналізу.

Були проаналізовані деякі поширені методи стеганоаналізу, розглянуті їх основні переваги та недоліки. З огляду на результати даного дослідження, був обраний стеганографічний метод F5 та запропонований метод його стеганоаналізу на основі дослідження відношень пар коефіцієнтів ДКП.

Наступним кроком були розглянуті базові принципи побудови генеративно-змагальних мереж. Проаналізовані принципи роботи та об’єднання нейронних мереж.

Зважаючи на здобуті знання, було розроблено, який здатен з встановити наявність таємного повідомлення у контейнера, а також генеративно-змагальну мережу, яка спроможна приховати факт вбудовування. Проведена перевірка ефективності їх роботи.

Заключним етапом стала розробка і перевірка програмного продукту для застосування алгоритму F5, перевірки наявності вкладеного повідомлення та зменшення ймовірності його виявлення.

# ПЕРЕЛІК ПОСИЛАНЬ


1. Usage statistics of JPEG for websites. URL: https://w3techs.com/technologies/details/im-jpeg
2. Почему JPEG является самым популярным расширением, каковы его преимущества и недостатки. *Информационный портал систем безопасности.* URL: https://bezopasnik.info/почему-jpeg-является-самым-популярным-ра/
3. Даракчі Ю.І., Кушніренко Н.І. Розробка методу для підвищення стійкості до статистичного стеганоаналізу з використанням генеративно-змагальних мереж. Інформатика та математичні методи в моделюванні. 2021. URL: http://immm.op.edu.ua
4. Стеганография. Матеріал із Вікіпедії – вільної енциклопедії. https://ru.wikipedia.org/wiki/Стеганография
5. Цифровая стеганография. Предмет, терминология, области применения. URL: https://leally.ru/download-soft/cifrovaya-steganografiya-predmet-terminologiya-oblasti-primeneniya/
6. Генне О.В. Основные положения стеганографии. *Защита информации. Конфидент.* 2000. №3. 16 с.
7. Дрюченко, М.А. Алгоритмы выявления стеганографического скрытия информации в jpeg-файлах. *Вестник Воронежского государственного университета. Серия: Системный анализ и информационные технологии*. 2007. №1. С. 20-29.
8. Генеративно-состязательные сети: создаём свои первые модели. *Python - курс молодого бойца*. URL: https://chel-center.ru/python-yfc/2021/01/08/generativnye-sostyazatelnye-seti-sozdajte-svoi-pervye-modeli/
9. Goodfellow I. J., Pouget-Abadie J., Mirza M., Xu B., Warde-Farley D., Ozair S., Courville A., Bengio Y. Generative Adversarial Networks. 2014. URL: https://arxiv.org/abs/1406.2661

10. Westfeld A. F5 – A Steganographic Algorithm: High Capacity Despite Better Steganalysis. Information Hiding, 4th International Workshop, volume 2137 of Lecture Notes in Computer Science. 2001. P. 289-302.
11. Westfeld A., Pfitzmann A. Attacks on steganographic systems. URL: https://link.springer.com/chapter/10.1007/10719724_5
12. Crandall R. Some Notes on Steganography. 1999. URL: https://www.semanticscholar.org/paper/Some-Notes-on-Steganography-Crandall/9d84704f34e594fef44edd2515203a1db44f0af3
13. F5 (Стеганографический алгоритм). *Национальная библиотека им. Н. Э. Баумана*. URL: https://ru.bmstu.wiki/F5_(Стеганографический_алгоритм)
14. Linnaeus 5 dataset. Giorgi Chaladze. URL: http://chaladze.com/l5/
15. Суперпозиция функций. Дискретная математика. URL: https://cde.osu.ru/courses2/temp4/modul1/supfunc/supfunc.html
16. Liu J., Ke Y., Lei Y., Zhang Z., Li J., Luo P., Zhang M., Yang X. Recent Advances of Image Steganography with Generative Adversarial Networks. 2019. URL: https://arxiv.org/abs/1907.01886
17. Tang W., Li B., Tan S., Barni M., Huang J. CNN Based Adversarial Embedding with Minimum Alteration for Image Steganography. 2018. URL: https://arxiv.org/abs/1803.09043
18. Project Jupyter. Матеріал із Вікіпедії – вільної енциклопедії. URL: https://en.wikipedia.org/wiki/Project_Jupyter
19. Nerds rejoice: Google just released its internal tool to collaborate on AI. Quartz. URL: https://qz.com/1113999/nerds-rejoice-google-just-released-its-internal-tool-to-collaborate-on-ai/
20. Open in Colab GitHub repository. GitHub. URL: https://github.com/googlecolab/open_in_colab
21. Perez F., Granger B.. IPython: A System for Interactive Scientific Computing. Computing in Science and Engineering. 2007. № 3. P. 21-29.

22. 10 интересных фактов про Python. IT новости. URL: https://itproger.com/news/10-interesnih-faktov-pro-python
23. PySide. Python Wiki. URL: https://wiki.python.org/moin/PySide
24. Прохоренок Н.А. PyQt. Создание оконных приложений на Python 3. 2011. 243 с.
25. ДСанПіН 3.3.2.007-98. Гігієнічні вимоги до організації роботи з візуальними дисплейними терміналами електронно-обчислювальних машин. 1998. URL: https://zakon.rada.gov.ua/rada/show/v0007282-98
26. Надзвичайні ситуації та їх класифікація. Реферат - Освіта.UA. URL: https://ru.osvita.ua/vnz/reports/bjd/22895/
27. Наказ «Про затвердження Правил пожежної безпеки в Україні». 2020. URL: https://ips.ligazakon.net/document/RE26697?an=1

# Додаток А. Код застосунку

```
import random
import logging
from utils import *

logger: logging.Logger = logging.getLogger('F5')


class F5Encoder:
    def f5_encoding(self, container: np.ndarray, password: str, data:
 str, compression_quality: int = 90,
                    is_dct_blocks: bool = False):
        """
        Returns:
        if is_dct_blocks = True:
          dct blocks with embeded data + diff

        if is_dct_blocks = False:
          reconstructed image (quantization + idct) with embeded data
 + diff
        """

        def encode():
            dct_arr = list(dct_blocks.ravel())
            shuffle = self._permutate(password, dct_arr)
            f5_dct_cficients = np.array(self._matrix_code(data,
 dct_arr[:], shuffle))
            return f5_dct_cficients.reshape((len(f5_dct_cficients) //
 64, 8, 8))

        if not is_dct_blocks:
            quality_table =
 ijg_quantization_tables(compression_quality)
            mod = True if (container[-1, 0] == 57) and (container[0,
 -1] == 173) else False
            dct_blocks = np.array(get_dct_blocks(container,
 quality_table))
            f5_dct_blocks = encode()

            image_blocks = []
            # for each octet/block
            for block in f5_dct_blocks:
                # quantize
                quantized_block = np.round(block * quality_table)
                # perform matlab-like idct2
                dct_block = idct2(quantized_block)
                image_blocks.append(dct_block)
            reconstructed_image =
 reconstruct_from_blocks(np.array(image_blocks, 'int16'),
 container.shape)
```

```
            logger.info(f'cficients changed:
{np.count_nonzero(dct_blocks != image_blocks)}')
            if not mod:
                reconstructed_image[-1,0] = 133
                reconstructed_image[0,-1] = 57
            return reconstructed_image
        else:
            dct_blocks = container
            f5_dct_blocks = encode()
            logger.info(f'cficients changed:
{np.count_nonzero(container != f5_dct_blocks)}')
            return f5_dct_blocks

    def _get_byte(self):
        return random.randint(-128, 127)

    def _get_int(self, max_value):
        ret_val = self._get_byte() | self._get_byte() << 8 |
self._get_byte() << 16 | self._get_byte() << 24
        ret_val %= max_value
        if ret_val < 0: ret_val += max_value
        return ret_val

    def _permutate(self, password, cf):
        f5random = random.seed(password)
        size = len(cf)
        shuffle = [i for i in range(size)]
        for i in range(size):
            index = self._get_int(size)
            size -= 1
            shuffle[size], shuffle[index] = shuffle[index],
shuffle[size]
        return shuffle

    def _create_array(self, default_value=None, *args):
        if len(args) and args[0]:
            return [self._create_array(default_value, *args[1:]) for
i in range(args[0])]
        else:
            return default_value

    def _matrix_code(self, data, cf, shuffle):
        logger.debug('Matrix coding started')
        c_f = len(cf)
        one, zero = 0, 0

        for i, j in enumerate(cf):
            if i % 64 == 0:
                continue
            elif j == 1 or j == -1:
                one += 1
            elif j == 0:
```

```
                zero += 1
        other = c_f - one - zero - c_f // 64
        expected = int(0.5 * one) + other
        byte_to_embed = len(data)
        if byte_to_embed > 0x7fffff: byte_to_embed = 0x7fffff
        for i in range(1, 10):
            n = (1 << i) - 1
            use = (expected // n) * i / 8
            if use < byte_to_embed + 4:
                i = i - 1
                break
        k = i  # №4
        n = (1 << k) - 1  # №5
        if n == 0:
            logger.warning('File not suitable\n')
            n = 1
        else:
            logger.debug('using (1,%d,%d) code' % (n, k))
        logger.debug('totally %d bytes to embed by f5' %
byte_to_embed)
        byte_to_embed |= k << 24
        for i in range(4): byte_to_embed ^= self._get_byte() << i * 8
        bit_to_embed = byte_to_embed & 1
        byte_to_embed >>= 1
        need_to_be_embeded = 31
        data_index = 0
        shuffle_pos = 0
        for j in shuffle:
            shuffle_pos += 1
            if j % 64 == 0 or cf[j] == 0: continue
            if cf[j] > 0 and (cf[j] & 1) != bit_to_embed:
                logger.debug(f'[{j}]: {cf[j]} => {cf[j] - 1}')
                cf[j] -= 1
            elif cf[j] < 0 and (cf[j] & 1) == bit_to_embed:
                logger.debug(f'[{j}]: {cf[j]} => {cf[j] + 1}')
                cf[j] += 1
            if cf[j] != 0:
                if need_to_be_embeded == 0: break;
                bit_to_embed = byte_to_embed & 1
                byte_to_embed >>= 1
                need_to_be_embeded -= 1
        try:
            while True:
                kbits_embed = 0
                for i in range(k):
                    if need_to_be_embeded == 0:
                        if data_index >= len(data): break
                        byte_to_embed = ord(data[data_index]) ^
self._get_byte()
                        data_index += 1
                        need_to_be_embeded = 8
                    bit_to_embed = byte_to_embed & 1
```

```
                    byte_to_embed >>= 1
                    need_to_be_embeded -= 1
                    kbits_embed |= bit_to_embed << i
                refer = self._create_array(0, n)
                for i in range(n):
                    while shuffle[shuffle_pos] % 64 == 0 or
cf[shuffle[shuffle_pos]] == 0:
                        shuffle_pos += 1
                    refer[i] = shuffle[shuffle_pos]
                    shuffle_pos += 1
                while True:
                    fhash = 0
                    for i, j in enumerate(refer):
                        if cf[j] > 0:
                            tmp = cf[j] & 1
                        else:
                            tmp = cf[j] & 1 ^ 1
                        if tmp == 1: fhash ^= i + 1
                    d = fhash ^ kbits_embed
                    if d == 0: break
                    s = refer[d - 1]
                    if cf[s] > 0:
                        logger.debug(f'[{s}]: {cf[s]} => {cf[s] -
1}')
                        cf[s] -= 1
                    elif cf[s] < 0:
                        logger.debug(f'[{s}]: {cf[s]} => {cf[s] +
1}')
                        cf[s] += 1
                    if cf[s] == 0:
                        refer[d - 1:d] = []
                        while shuffle[shuffle_pos] % 64 == 0 or
cf[shuffle[shuffle_pos]] == 0:
                            shuffle_pos += 1
                        refer.append(shuffle[shuffle_pos])
                        shuffle_pos += 1
                    else:
                        break
                if data_index >= len(data) and need_to_be_embeded ==
0: break
        except IndexError:
            raise ValueError('Data is too long')
        return cf

from typing import Optional

import numpy as np
from gui.uis.windows.main_window.ui_main import *
import cv2

from pathlib import Path
from os import sep
```

```
# Last opened file
FILE_NAME: Optional[str] = None
# Container image
CONTAINER: Optional[np.ndarray] = None
# Text message
MESSAGE: Optional[str] = None
# New Image
NEW_CONTAINER: Optional[np.ndarray] = None


class File:
    def __init__(self):
        super().__init__()

        # Import widgets
        self.ui = UI_MainWindow()
        self.ui.setup_ui()

    # Load container
    def load_container(self):
        global FILE_NAME, CONTAINER
        # file dialog
        dlg_file = QFileDialog.getOpenFileName(
            parent=self,
            caption=self.tr("Select an image"),
            filter=self.tr("Image files (*.jpg *.png)")
        )
        # read file
        if dlg_file:
            img = cv2.imread(dlg_file[0], cv2.IMREAD_GRAYSCALE)
            CONTAINER = img
            # Show image
            height, width = img.shape[:2]
            q_img = QImage(img, width, height, img.strides[0],
 QImage.Format_Indexed8)
            q_img = QPixmap.fromImage(q_img)

            FILE_NAME = dlg_file[0]

            self.ui.load_pages.container_image.setPixmap(q_img)


 self.ui.right_column.image_name.setText(FILE_NAME.split(sep)[-1])

 self.ui.right_column.image_size.setText(f"{height}×{width}")
            self.ui.right_column.channels.setText(str(img.shape[3] if
 len(img.shape) > 2 else 1))

    # Load message
    def load_message(self):
        global FILE_NAME, MESSAGE
```

```
        # file dialog
        dlg_file = QFileDialog.getOpenFileName(
            parent=self,
            caption=self.tr("Select a file"),
            filter=self.tr("Text files (*.txt)")
        )
        # read file
        if dlg_file:
            print(f"Loading message: {dlg_file[0]}")
            with open(dlg_file[0], encoding='utf-8') as file:
                text = file.read()

            FILE_NAME = dlg_file[0]
            MESSAGE = text


 self.ui.right_column.message_name.setText(FILE_NAME.split(sep)[-1])
            self.ui.right_column.message_size.setText(str(len(text)))

    def save(self):
        file_name = Path("out.png")
        folder_name = Path(QFileDialog.getExistingDirectory(
            parent=self,
            caption=self.tr("Select a folder"),
        ))

        print(f"Saving file: {Path.joinpath(folder_name,
 file_name)}")
        cv2.imwrite(str(Path.joinpath(folder_name, file_name)),
 NEW_CONTAINER)

import numpy as np
from PySide6.QtWidgets import QMessageBox
from scipy.fftpack import dct, idct
from math import log10, sqrt


def calc_psnr(original_image: np.ndarray, stego_image: np.ndarray) ->
 tuple:
    """
    :param original_image:
    :param stego_image:
    :return: tuple(psnr, mse)
    """
    #original_image, stego_image = np.array(original_image),
 np.array(stego_image)
    mse = np.mean((original_image - stego_image) ** 2)
    if (mse == 0):  # MSE is zero means no noise is present in the
 signal .
        # Therefore PSNR have no importance.
        return 100
    max_pixel = 255.0
```

```
    psnr = 20 * log10(max_pixel / sqrt(mse))
    return psnr, mse

def reconstruct_from_blocks(blocks_arr: np.ndarray, original_shape:
 tuple) -> np.ndarray:
    _, M, N = blocks_arr.shape
    m, n = original_shape
    return blocks_arr.reshape(m // M, n // N, M, N).swapaxes(1,
 2).reshape(m, n)


def idct2(block: np.ndarray) -> np.ndarray:
    """2D IDCT as in Matlab `idct2` function"""
    return idct(idct(block.T, norm='ortho').T, norm='ortho')


def dct2(block: np.ndarray) -> np.ndarray:
    """2D DCT as in Matlab `dct2` function"""
    return dct(dct(block.T, norm='ortho').T, norm='ortho')


def ijg_quantization_tables(quality: int) -> (np.ndarray,
 np.ndarray):
    """Returns IJG-standard JPEG quantization matrice"""

    ijg_table_y = [[16, 11, 10, 16, 24, 40, 51, 61],
                   [12, 12, 14, 19, 26, 58, 60, 55],
                   [14, 13, 16, 24, 40, 57, 69, 56],
                   [14, 17, 22, 29, 51, 87, 80, 62],
                   [18, 22, 37, 56, 68, 109, 103, 77],
                   [24, 35, 55, 64, 81, 104, 113, 92],
                   [49, 64, 78, 87, 103, 121, 120, 101],
                   [72, 92, 95, 98, 112, 100, 103, 99]]

    ijg_table_y = np.array(ijg_table_y)

    # scale factor
    s = 5000 / quality if quality < 50 else 200 - 2 * quality
    # scale matrices
    q_y = np.floor((ijg_table_y * s + 50) / 100)
    return q_y


def get_dct_blocks(img, quality_table):
    rows, cols = img.shape
    quantized_blocks = []
    # for each octet/block
    # 256/8 = 32, 32*32=1024 blocks
    for i in range(0, rows - 7, 8):
        for j in range(0, cols - 7, 8):
            # extract block form image
            block = img[i:(i + 8), j:(j + 8)]
```

```
            # perform matlab-like dct2
            dct_block = dct2(block)
            # quantize
            quantized_block = np.round(dct_block /
 quality_table).astype('int')
            quantized_blocks.append(quantized_block)
    return quantized_blocks


def dct2image(dct_blocks, quality_table, shape):
    image_blocks = []
    # for each octet/block
    for block in dct_blocks:
        # quantize
        quantized_block = np.round(block * quality_table)
        # perform matlab-like idct2
        dct_block = idct2(quantized_block)
        image_blocks.append(dct_block)
    reconstructed_image =
 reconstruct_from_blocks(np.array(image_blocks, 'int'), shape)
    return reconstructed_image


def show_msb(title, text, icon=QMessageBox.Information):
    msg = QMessageBox()
    msg.setWindowTitle(title)
    msg.setText(text)
    msg.setIcon(icon)
    x = msg.exec_()

import logging
from file_manager import File
from gui.uis.windows.main_window.functions_main_window import *
import sys
import os
from qt_core import *
from gui.core.json_settings import Settings


class MainWindow(QMainWindow):
    def __init__(self):
        super().__init__()

        self.ui = UI_MainWindow()
        self.ui.setup_ui(self)

        settings = Settings()
        self.settings = settings.items


        self.hide_grips = True
        SetupMainWindow.setup_gui(self)
```

```
        self.show()


    def btn_clicked(self):

        btn = SetupMainWindow.setup_btns(self)

        if btn.objectName() != "btn_modify":
            self.ui.left_menu.deselect_all_tab()

        top_settings = MainFunctions.get_title_bar_btn(self,
"btn_top_settings")
        top_settings.set_active(False)

        if btn.objectName() == "btn_home":

            self.ui.left_menu.select_only_one(btn.objectName())

            MainFunctions.set_page(self, self.ui.load_pages.page_1)

        if btn.objectName() == "btn_container":
            File.load_container(self)

        if btn.objectName() == "btn_message":
            File.load_message(self)

        if btn.objectName() == "btn_save":
            File.save(self)

        if btn.objectName() == "btn_info":

            if not MainFunctions.left_column_is_visible(self):

self.ui.left_menu.select_only_one_tab(btn.objectName())

                MainFunctions.toggle_left_column(self)

self.ui.left_menu.select_only_one_tab(btn.objectName())
            else:
                if btn.objectName() == "btn_close_left_column":
                    self.ui.left_menu.deselect_all_tab()

                    MainFunctions.toggle_left_column(self)

self.ui.left_menu.select_only_one_tab(btn.objectName())

            if btn.objectName() != "btn_close_left_column":
                MainFunctions.set_left_column_menu(
                    self,
                    menu=self.ui.left_column.menus.menu_2,
```

```
                    title="Info tab",
                    icon_path=Functions.set_svg_icon("icon_info.svg")
                )

        if btn.objectName() == "btn_modify" or btn.objectName() ==
"btn_close_left_column":

            if not MainFunctions.left_column_is_visible(self):

                MainFunctions.toggle_left_column(self)

self.ui.left_menu.select_only_one_tab(btn.objectName())
            else:
                if btn.objectName() == "btn_close_left_column":
                    self.ui.left_menu.deselect_all_tab()

                    MainFunctions.toggle_left_column(self)

self.ui.left_menu.select_only_one_tab(btn.objectName())

            if btn.objectName() != "btn_close_left_column":
                MainFunctions.set_left_column_menu(
                    self,
                    menu=self.ui.left_column.menus.menu_1,
                    title="Image processing",

icon_path=Functions.set_svg_icon("icon_widgets.svg")
                )
        if btn.objectName() == "btn_top_settings":

            if not MainFunctions.right_column_is_visible(self):
                btn.set_active(True)

                MainFunctions.toggle_right_column(self)
            else:
                btn.set_active(False)

                MainFunctions.toggle_right_column(self)

            top_settings = MainFunctions.get_left_menu_btn(self,
"btn_modify")
            top_settings.set_active_tab(False)

        logger.debug(f"Button {btn.objectName()}, clicked!")

    def btn_released(self):

        btn = SetupMainWindow.setup_btns(self)

        logger.debug (f"Button {btn.objectName()}, released!")
    def resizeEvent(self, event):
        SetupMainWindow.resize_grips(self)
```

```
    def mousePressEvent(self, event):

        self.dragPos = event.globalPos()

def config_logging():
    for handler in logging.root.handlers[:]:
        logging.root.removeHandler(handler)
    logger = logging.getLogger(__name__)
    logging.basicConfig(format='%(asctime)s - %(name)s -
 <%(levelname)s> - %(message)s', datefmt='%d-%b-%y %H:%M:%S',
                        level=logging.INFO)

if __name__ == "__main__":
    config_logging()

    app = QApplication(sys.argv)
    app.setWindowIcon(QIcon("icon.ico"))
    window = MainWindow()

    sys.exit(app.exec_())
```