Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
“____” _____ 2026 р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за першим (бакалаврським) рівнем вищої освіти
на тему
“Програмне забезпечення системи інтелектуального розпізнання облич при виконанні OSINT-аналізу”

Виконав здобувач вищої освіти
IV курсу, групи КН-22
ОПП «Комп’ютерні науки»
спеціальності 122 «Комп’ютерні науки»
_____ Кузнєцов М.О.
«____» _____ 2026 р.

Керівник проекту
кандидат технічних наук, доцент
_____ Коваленко А.С.
«____» _____ 2026 р.
Рецензент _____
_____

м. Кропивницький

Центральноукраїнський національний технічний університет

Факультет *Механіко-технологічний*

Кафедра *Кібербезпеки та програмного забезпечення*

Освітній ступінь *бакалавр*

Галузь знань ._12 _____*“Інформаційні технології”*_____

Спеціальність 122 *“Комп’ютерні науки”*

Освітньо-професійна (освітньо-наукова) програма _*“Комп’ютерні науки”*_

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____ Олексій СМІРНОВ
« 06 » лютого 2026 року

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ ЗА ПЕРШИМ (БАКАЛАВРСЬКИМ) РІВНЕМ ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Кузнєцову Миколі Олександровичі
(прізвище, ім’я, по батькові)

1. Тема роботи *Програмне забезпечення системи інтелектуального розпізнання облич при виконанні OSINT-аналізу*

2. Керівник роботи *Коваленко Анна Степанівна, канд. техн. наук, доцент*
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 116-02 від 06.02.2026 року

3. Строк подання студентом роботи до захисту *23.05.2026 р.*

4. Мета та завдання випускної кваліфікаційної роботи*: Метою роботи є розробка програмного забезпечення системи інтелектуального розпізнання облич при виконанні OSINT-аналізу*

5. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Призначення та область використання.

2.Перегляд аналогічних існуючих систем.

3. Опис і обґрунтування проектних рішень.

4. Етапи програмування системи.

5. Впровадження системи в промислову експлуатацію.

6. Висновки

6. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

Структурна схема системи	*1 аркуш*
Функціональна схема системи	*1 аркуш*
Діаграма процесів	*1 аркуш*
Блок-схема алгоритму роботи додатку	*2 аркуша*

7. Дата видачі завдання « 06 » лютого 2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.03.2026 р.	
2.	Постановка задачі, оформлення ТЗ	15.03.2026 р.	
3.	Розробка моделі компонента	20.03.2026 р.	
4.	Розробка структур даних	25.03.2026 р.	
5.	Розробка алгоритмів зв’язку та відображення	30.03.2026 р.	
6.	Програмування алгоритмів	10.04.2026 р.	
7.	Оформлення ПЗ	17.04.2026 р.	
8.	Попередній захист роботи	23.05.2026 р.	

Дата видачі завдання
« 06 » лютого 2026 р.

Підпис керівника
_____ Коваленко А.С.
(прізвище та ініціали)

Завдання прийнято до виконання
« 06 » лютого 2026 р.

Підпис здобувача
_____ Кузнєцов М.О.
(прізвище та ініціали)

АНОТАЦІЯ

Кузнєцов М.О. Програмне забезпечення системи інтелектуального розпізнання облич при виконанні OSINT-аналізу. 122 Комп'ютерні науки. Центральноукраїнський національний технічний університет. Кропивницький. 2026.

В даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи інтелектуального розпізнання облич при виконанні OSINT-аналізу.

Метою розробки є програмне забезпечення системи інтелектуального розпізнання облич при виконанні OSINT-аналізу.

Результат роботи – програмна реалізація системи інтелектуального розпізнання облич при виконанні OSINT-аналізу.

В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

Розроблено зручний інтерфейс користувача. Наведені інструкції по роботі з програмними засобами.

Програма може використовуватися на ПЕОМ з ОС Windows 10/11.

Програму розроблено в середовищі Python.

Ключові слова: комп'ютерні науки, інтелектуальне розпізнання облич, OSINT-аналіз

ABSTRACT

Kuznietsov M.O. Software for the intelligent face recognition system when performing OSINT analysis. 122 Computer Science. Central Ukrainian National Technical University. Kropyvnytskyi. 2026.

In this final qualification work for the first (bachelor's) level of higher education, software has been developed, which is intended for the intelligent face recognition system when performing OSINT analysis.

The purpose of the development is the software for the intelligent face recognition system when performing OSINT analysis.

The result of the work is the software implementation of the intelligent face recognition system when performing OSINT analysis.

In the process of working on the software model, an analysis of existing hardware and software was performed. All components of the developed software are fully described.

A convenient user interface has been developed. Instructions for working with software are provided.

The program can be used on a PC with Windows 10/11.

The program is developed in the Python environment.

Keywords: computer science, intelligent facial recognition, OSINT analysis

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

БД	–	база даних
ПЗ	–	програмне забезпечення

ВСТУП

Актуальність теми. Аналіз фотографій і соціальних зв'язків може стати величезною підмогою при соціальній інженерії. Якщо ми знаємо, ким є персона й коло знайомих цієї персони, то цілком можемо виявити зв'язки усередині компанії, що згодом дозволить реалізувати витончені цільові атаки з використанням соціальної інженерії. Термін OSINT або по іншому розвідка на основі відкритих джерел охоплює такі сфери, як аналіз, оцінку й застосування привселюдно доступної інформації. Щоб одержати щось коштовне, не обов'язково намагатися дістати винятково конфіденційну інформацію. Розуміння навколишньої реальності приходить через читання блогів і спеціальних журналів, перегляд телевізора й вивчення інших загальнодоступних джерел.

Крім тексту загальнодоступна інформація зберігається в інтернеті у вигляді фотографій, які, в основному, фільтруються й обробляються людьми вручну за допомогою графічного редактора з метою наступного аналізу отриманих результатів. Однак з появою програмного забезпечення, заточеного під розпізнавання осіб, багато операцій стало можливим автоматизувати.

Із приводу технології розпізнавання осіб уже з'явилося багато статей і дискусій, наприклад, відносно біометрії або того, як ця технологія впливає на інформаційні масиви й приватне життя. Незважаючи на те, що розпізнавання осіб в основному використовується на таких платформах як Facebook або правоохоронними органами по усьому світі, будь-який бажаючий може скористатися схожими безкоштовними додатками з відкритим вихідним для рішення своїх власних завдань.

Мета й завдання дослідження. Метою роботи є програмне забезпечення системи інтелектуального розпізнання облич при виконанні OSINT-аналізу.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

– Огляд існуючих систем інтелектуального розпізнання облич при виконанні OSINT-аналізу.

– Дослідження системи інтелектуального розпізнання облич при виконанні OSINT-аналізу.

– Програмна реалізація системи інтелектуального розпізнання облич при виконанні OSINT-аналізу.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі інтелектуального розпізнання облич при виконанні OSINT-аналізу.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи інтелектуального розпізнання облич при виконанні OSINT-аналізу, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Система призначена для розпізнавання осіб. Будь-який інструмент можна використовувати з добрими або злими намірами. Навіть звичайною мітлою можна не тільки підмітати підлогу, але й ударити кого-небудь по голові. Чим могутніший інструмент, тим більше користі або збитку він може принести. В останні кілька місяців про це багато говорять, обговорюючи комп'ютерні технології розпізнавання осіб, що дозволяють ідентифікувати людину по її фотографії, а також через вуличні відеокамери. Ці технології можна використовувати в благо: наприклад, для сортування приватних фотоальбомів і для пошуку родичів. Але є й інша можливість – зловживання з боку приватних компаній і державних органів.

Технології розпізнавання осіб піднімають питання, пов'язані з фундаментальними правами людини, включаючи право на конфіденційність і волю вираження думок. Можливість зловживань підвищує відповідальність, покладену на IT-компанії, що створюють відповідну продукцію. На наш погляд, ця можливість також вимагає продуманого державного регулювання й розробки норм прийнятного використання.

У демократичній країні пошуком компромісу між суспільною безпекою й нашими демократичними свободами повинні займатися вибрані представники народу. Рішення проблеми розпізнавання осіб зажадає участі – і дій – як державного, так і приватного сектора.

Технології розпізнавання осіб швидко розвивалися в останнє десятиліття. Якщо ви бачили, як Facebook або інша соціальна мережа пропонує відзначити людини на фотографії, правильно вказавши його ім'я, то ви знаєте, як працює розпізнавання осіб. Багато які IT-компанії, включаючи Microsoft, уже кілька років

використовують такі технології, щоб перетворювати стомлюючу роботу з каталогізації фотографій у більш швидку й результативну.

Так що ж змінилося? По-перше, розширилися здатності комп'ютерного зору, і машини стали розпізнавати особи людей точніше й швидше. Почасти ці поліпшення пов'язані з удосконалюванням відеокамер, сенсорів і алгоритмів машинного навчання. Крім того, поліпшенням сприяло появу усе більших наборів даних, що розширюються в міру того, як люди завантажують в Інтернет усе більше своїх зображень. Свою роль зіграла й можливість використання хмар для зв'язку всіх цих даних і технологій розпізнавання осіб з постійно працюючими камерами, які одержують зображення осіб і намагаються ідентифікувати людей у безлічі різних місць, у реальному часі.

Передові технології більше не стоять особняком від суспільства; вони глибоко інтегруються в наше особисте й професійне життя. Це значить, що є багато можливостей для застосування технологій розпізнавання осіб. Найпростіший приклад – каталогізація й пошук фотографій. І цей тільки початок. Деякі технології вже підвищують безпеку комп'ютерів, наприклад, дозволяють розблокувати iPhone або ноутбук з Windows, розпізнаючи особу власника без уведення пароля. Можливо, у майбутньому по такому ж принципі будуть працювати банкомати й інші пристрої.

Деякі передові можливості можуть зробити на суспільство ще більший вплив. Представте, що можна знайти дитину, що втратилася, яка проходить повз вуличну камеру. Представте, що поліція може виявити терориста, що направляється на переповнений стадіон. Представте, що смартфон може повідомляти сліпій людині, імена людей, що заходять у кімнату.

Але не все так райдужно. Представте, що уряд стежить за вами без вашого ведена й знає в дрібних подробицях, де й коли ви перебували протягом цього місяця. Представте, що є база даних з інформацією про усіх, хто був присутній на політичному мітингу, виражаючи волю слова. Представте, що продавці в торговому центрі, не запитуючи у вас дозволу, використовують технологію

розпізнавання осіб, щоб ділитися один з одним інформацією про кожну полицю, що ви переглядаєте, і про продукти, які ви купуєте. Подібне здавна було в сюжетах книг і фільмів – «Особлива думка» («Minority Report»), «Ворог держави» («Enemy of the State») і навіть «1984», – але тепер все це може стати реальністю.

Технології розпізнавання осіб (видимо, як і всі інші передові технології) піднімають найважливіше питання: яку роль у повсякденному житті суспільства Microsoft готові їм віддати?

Прийняття рішень утрудняється тим, що технології розпізнавання осіб швидко розвиваються, але усе ще далекі від досконалості. Як широко обговорювалося в останні місяці, у роботі декількох технологій, що застосовуються на практиці, виявлена упередженість: особи білих чоловіків розпізнаються точніше, ніж білих жінок, а особи зі світлою шкірою – точніше, ніж з темною. Дослідники ретельно працюють над рішенням цієї проблеми й уже домоглися значних успіхів. Але, як показало одне важливе дослідження, упередженість як і раніше зберігається. Відносна незрілість технологій робить питання з боку суспільства ще більш актуальними.

Навіть якщо позбутися від упередженості й домогтися однакових результатів у розпізнаванні осіб всіх людей, імовірність помилок залишиться. Технології розпізнавання осіб, як і багато інших технологій штучного інтелекту, звичайно видають деяку частку помилок, навіть якщо працюють без упередженості. А проблеми розпізнавання осіб виходять далеко за рамки упередженості, змушуючи задуматися про наші фундаментальні волі.

В останні тижні в політиці США приділяється багато уваги IT-розробкам, що ведуться на Західному узбережжі. У середині червня 2018 року питання, пов'язані з розпізнаванням осіб, устали перед керівництвом Microsoft особливо гостро. Країну захлиснули суперечки про відділення дітей іммігрантів від їхніх родин на південній границі, і твіт про маркетинговий пост Microsoft, опублікованому в січні, швидко розійшовся по соцмережам і викликав жваві дебати. У тому пості обговорювався контракт із імміграційною й митною

службою США (ICE) і говорилося, що Microsoft досягла високого порога безпеки. Крім іншого, там повідомлялося про можливість ICE використовувати розпізнавання осіб.

Як Microsoft вже повідомляло, даний контракт не має ніякого відношення до технологій розпізнавання осіб Microsoft. Крім того, Microsoft не брала участь у яких-небудь проектах уряду США, пов'язаних з відділенням дітей від їхніх родин на границі, і Microsoft категорично проти подібної практики. Предметом цього контракту була підтримка застарілих застосунків: електронної пошти, календарів, месенджерів і систем документообігу. Подібна робота йде в кожному державному агентстві США, як і практично в будь-який інший державній, комерційній або некомерційній організації у світі. Проте в Microsoft надходили пропозиції розірвати контракт і припинити будь-яке співробітництво з ICE.

Наступне обговорення перейшло на більше широкі питання, що хвилюють IT-сектор. Ці питання не унікальні для Microsoft: на початку цього року вони піднімалися в Google і інших IT-компаніях. Недавно група співробітників Amazon подала протест проти контракту з ICE, знову підтверджуючи стурбованість суспільства застосуванням правоохоронними органами технологій розпізнавання осіб, про що повідомляв Американський союз захисту цивільних воль (American Civil Liberties Union, ACLU). Співробітники Salesforce задавали ті ж питання щодо політики імміграційних служб і використання ними продуктів Salesforce. До IT-компаній всі частіше пред'являються вимоги по обмеженню можливостей використання їхніх технологій розпізнавання осіб (а також деяких інших технологій) державними органами.

Ці проблеми нікуди не зникнуть. Вони відбивають швидко, що розширюються можливості, нових технологій, які повною мірою виявляться в найближче десятиліття. Зараз мова йде про розпізнавання осіб, але очевидно, що в майбутньому ті ж проблеми будуть пов'язані з іншими новими технологіями. Тим більше важливо відповісти на сьогоднішні питання й вибрати правильний напрямок подальшого розвитку подій.

Єдиний ефективний спосіб керування використанням технологій урядовими органами – контроль із боку самого уряду. І якщо є побоювання із приводу широкого застосування якоїсь технології в суспільстві, той тільки уряд може регулювати це широке застосування. Microsoft вважає, що сьогодні необхідно саме це: урядова ініціатива по регулюванню використання технологій розпізнавання осіб, створена при участі двопартійної експертної комісії.

Microsoft цінує ініціативу деяких людей, які сьогодні призивають IT-компанії самостійно приймати такі рішення, і визнає явну необхідність у відповідальному підході до цього питання. Але, як докладно пояснюється нижче, Microsoft вважає, що це неадекватна заміна прийняттю рішень громадськістю і її представниками в демократичній державі. ми живемо в країні, що підкоряється законам, тому уряд повинне відігравати важливу роль у регулюванні технологій розпізнавання осіб. У цілому представляється більше розумним просити вибраний уряд регулювати компанії, чим просити не вибрані компанії, щоб вони регулювали уряд.

Такий підхід, швидше за все, буде набагато більш ефективним для досягнення суспільних цілей. Зрештою навіть якщо одна або декілька IT-компаній змінять свої принципи, але інші продовжать працювати як і раніше, проблему не вдасться вирішити. Якщо не буде загальної нормативної бази, то конкуренція між американськими IT-компаніями, не говорячи вже про компанії з інших країн, дозволить урядам продовжувати здобувати й використовувати нові технології таким чином, що суспільство буде невдоволено.

Може здатися незвичайним, що компанія просить уряд регулювати її продукти, але існує багато ринків, де продумане регулювання сприяє розвитку більше здорової динаміки, вигідної для споживачів і виробників. В XX столітті автомобільна індустрія витратило десятиліття, пручаючись закликам до регулювання, але сьогодні всі визнають, наскільки важливим став вплив регулювання на поширення ременів і подушок безпеки й на підвищення ефективності використання палива. Те ж саме справедливо для безпеки

повітряних польотів, харчових і фармацевтичних продуктів. Завжди будуть суперечки про якісь деталі, і ці деталі важливі. Однак світ із твердим регулюванням корисних, але потенційно небезпечних продуктів краще, ніж світ без правових норм взагалі.

Тому компанія Microsoft призивала до прийняття законодавства про національну безпеку США в 2005 році й підтримала Загальний регламент по захисту даних у Європейському союзі. У споживачів буде більше впевненості в тому, як компанії використовують конфіденційну особисту інформацію, якщо з'являться чіткі правила, обов'язкові для всіх. Хоча нові проблеми, пов'язані з розпізнаванням осіб, виходять за рамки конфіденційності, Microsoft вважає, що аналогія доречна.

Продумане державне регулювання технологій розпізнавання осіб представляється особливо важливим, якщо врахувати його широкі соціальні наслідки й можливості для зловживань. Без продуманого підходу державні органи можуть покладатися на помилкові або упереджені комп'ютерні методи прийняття рішень щодо того, за ким треба стежити, про кого збирати відомості й навіть кого заарештовувати за злочини. Уряди можуть спостерігати за політичною й іншою суспільною діяльністю такими способами, які суперечать очікуванням демократичного суспільства, знижуючи бажання громадян брати участь у політичних подіях, підриваючи основні волі й порушуючи право на збори й вираження особистих думок. Аналогічно цьому, компанії можуть використовувати розпізнавання осіб, щоб автоматично (без участі людини) приймати рішення, що впливають на доступність товарів, на можливість одержання кредиту або роботи. Всі ці сценарії піднімають важливі питання конфіденційності, волі слова, волі на збори й навіть життя.

Отже, які питання варто вирішити за допомогою державного регулювання? Це одне з найважливіших питань, що вимагають відповіді в першу чергу.

Як відправну точку Microsoft пропонує наступні питання:

– Чи повинно суспільство контролювати використання правоохоронними органами технологій розпізнавання осіб, у тому числі обмежувати використання результатів автоматичного розпізнавання осіб як доказ провини або невинності в злочині?

– Аналогічно цьому, чи варто забезпечувати цивільний нагляд і підзвітність використання технологій розпізнавання осіб у рамках державних програм забезпечення національної безпеки?

– Які юридичні міри можуть перешкоджати розпізнаванню осіб для расової дискримінації й порушення інших прав людини, але при цьому зберігати можливість корисного застосування технологій?

– Чи належні технології розпізнавання осіб, використовувані державними органами або іншими організаціями, мати певний мінімальний рівень точності?

– Чи належний закон вимагати, щоб роздрібні магазини публікували на видному місці повідомлення про використання ними технології розпізнавання осіб у громадських місцях?

– Чи належний закон вимагати, щоб компанії діставали попередню згоду, перш ніж збирати зображення людей для розпізнавання їхніх осіб? Якщо так, то в яких ситуаціях і місцях це повинне застосовуватися? І яким образом можна запросити й одержати таку згоду?

– Чи належні Microsoft гарантувати людям право на знання того, які їхні фотографії були зібрані, збережені й зіставлені з їхніми іменами й особами?

– Чи належні Microsoft створювати процедури, що забезпечують юридичні права громадян, які думають, що вони були помилково ідентифіковані системою розпізнавання осіб?

Цей список, жодним чином не вичерпний, демонструє широту й важливість розглянутих питань.

Ще одне важливе питання, що вимагає першочергової відповіді: як уряди повинні вирішувати ці проблеми? У США це національне питання, яким повинні

займатися вибрані народом представники. Це означає, що лідерство повинен взяти на себе Конгрес. Хоча деякі сумніваються в тому, що члени Конгресу мають достатні знання в технологічних областях, Microsoft уважає, що Конгрес здатний ефективно вирішувати такі проблеми. Для цього необхідно використовувати правильні механізми збору експертних рекомендацій, необхідних для прийняття обґрунтованих рішень.

У багатьох випадках Конгрес створює двопартійні експертні комісії для оцінки складних питань і надання рекомендацій з виробітку законодавчих мір. Як відзначила торік Дослідницька служба Конгресу (Congressional Research Service, CRS), ці комісії є «офіційними групами, створеними для надання незалежних консультацій, винесення рекомендацій зі змін у державній політиці, вивчення конкретної проблеми, питання або події або виконання інших обов'язків». Створена Конгресом «Комісія 9/11» зіграла важливу роль в оцінці національної трагедії, що відбулася 11 вересня. За минулі десять років Конгрес створив 28 таких комісій для консультацій по самих різних питаннях, від захисту дітей під час катастроф до визначення майбутнього армії.

Microsoft вважає, що Конгресу варто створити двопартійну експертну комісію для оцінки найкращого способу регулювання технологій розпізнавання осіб у США. При оцінці цих проблем і розробці більше чітких етичних принципів використання технологій комісія повинна враховувати недавні академічні роботи, а також роботи, що ведуться в державному й частці секторах. Така комісія повинна надавати Конгресу рекомендації про те, які нові закони необхідні, а також пропонувати більше ефективні методи забезпечення належного нагляду над цими технологіями в рамках всієї виконавчої влади.

Проблеми, пов'язані з розпізнаванням осіб, актуальні не тільки для США. Перераховані вище питання, як і багато хто інші, стануть важливими аспектами світової державної політики, вимагаючи активної участі урядів, учених, IT-компаній і цивільного суспільства на міжнародному рівні. З огляду на глобальний характер самих технологій, імовірно, буде зростати й потреба у взаємодії й координації між регулювальними органами різних країн.

Необхідність державного керівництва не звільняє ІТ-компанії від власних етичних обов'язків. З огляду на важливість і широту проблем розпізнавання осіб, Microsoft, як і весь інший ІТ-сектор, відповідає за те, щоб ці технології були орієнтовані на інтереси людей і розроблялися відповідно до загальноприйнятих суспільних цінностей. Треба визнати, що багато хто із цих питань нові, і ні в кого немає всіх відповідей на них. Деякі проблеми ще тільки має бути визначити. Коротко говорячи, усім нам має бути багатьому навчитися. Проте, деякі первісні виводи вже ясні.

По-перше**,** борг ІТ-сектора – продовжувати важливу роботу зі зниження ризику виникнення упередженості в технологіях розпізнавання осіб. Ніхто не виграє від впровадження незрілої технології, що часто помиляється, особливо при розпізнаванні осіб жінок і темношкірих людей. Тому наші дослідники й розроблювачі прагнуть прискорити прогрес у цій області, і саме тому один з наших пріоритетів – робота внутрішнього комітету Microsoft Aether, що надає рекомендації з деяких питань штучного інтелекту й етики.

Microsoft розуміє важливість співробітництва з академічним співтовариством і іншими компаніями, у тому числі в складі таких груп, як Partnership for AI. Нам важливо не тільки створювати набори даних, що відбивають розмаїтість світу, але й приймати на роботу самих різних фахівців з більшим досвідом, здатних ефективно знижувати ризик виникнення упередженості. Це жадає від Microsoft і інших ІТ-компаній постійних зусиль по підтримці розмаїтості співробітників і інвестицій у розвиток талантів, необхідних для роботи в майбутньому. Microsoft зосереджені на досягненні прогресу в цих областях, але усвідомлюємо, що має бути ще дуже багато роботи.

По-друге, у більше широкому змісті Microsoft визнає необхідність прийняття принципового й прозорого підходу до розробки й застосування технологій розпізнавання осіб. Microsoft проводить роботу з оцінки й розробки додаткових принципів для керування нашими технологіями розпізнавання осіб. Microsoft вже використовували схожий підхід в інших випадках, включаючи

принципи довіри до хмарних сервісів, прийняті в 2015 році, частково підтримувані центрами прозорості й інших об'єктів по усьому світі й що дозволяють перевіряти наш вихідний код і інші дані. Крім того, раніше цього року Microsoft опублікували загальний набір етичних принципів, якими Microsoft керуємося в розробці всіх наших технологій штучного інтелекту (ШІ).

Просуваючись уперед, Microsoft прагне створити прозорий набір принципів для технологій розпізнавання осіб, якою Microsoft поділилося із громадськістю. Частково він буде опиратися на наші більше широкі зобов'язання по розробці продуктів і керуванню послугами, що відповідають Керівним принципам ООН в області бізнесу й прав людини. Ці принципи, прийняті в 2011 році, стали глобальним стандартом забезпечення корпоративної поваги прав людини. Microsoft періодично проводимо оцінку впливу наших продуктів і послуг на права людини (HRIA), і в цей час робимо те ж саме для наших технологій ШІ.

Microsoft буде продовжувати цю роботу, частково опираючись на досвід і внесок наших співробітників, але Microsoft також визнаємо важливість активного зовнішнього моніторингу й участі. Тому Microsoft враховує думки зовнішніх зацікавлених сторін, включаючи клієнтів, учених і правозахисників, зосереджених на конкретних питаннях, пов'язаних з розпізнаванням осіб. Ця робота займе кілька місяців, але Microsoft прагне завершити її в найкоротший термін.

Одне з важких запитань, які нам необхідно вирішити – проведення границі між розвитком наших технологій розпізнавання осіб і використанням нашої більше широкої IT-Інфраструктури третіми сторонами, які створюють і розгортають свої власні технології розпізнавання осіб. Компанії складніше регулювати використання її інструментів і інфраструктури третіми сторонами, чим при повнім обслуговуванні або використанні власних консультантів, коли можна пред'являти більше тверді вимоги. Ці розходження й нюанси зажадають окремого розгляду.

По-третє, Microsoft усвідомлює, що при розгортанні повного спектра технологій розпізнавання осіб важливо не приймати поспішних рішень. Багато інформаційних технологій (у відмінність, наприклад, від фармацевтичних продуктів) поширюються дуже швидко й широко для прискорення темпів інновацій і впровадження. Девіз «рухайся швидко, трощи все на своєму шляху» став чимсь начебто мантри в Кремнієвій долині в цьому десятилітті. Але якщо Microsoft буде занадто поспішати із впровадженням розпізнавання осіб, то можемо виявити, що порушуються основні права людини.

Тому Microsoft прийняли рішення, частково засноване на рекомендаціях комітету Aether, більш вдумливо підходити до контрактам і консультаціям в області розпізнавання осіб. Це привело до того, що Microsoft відмовилися від деяких контрактів на розгортання цих технологій, придя до виводу, що існують більші ризики для прав людини. Microsoft продовжуємо розробляти більше довгострокові принципи й будемо стежити за використанням наших технологій розпізнавання осіб, щоб запобігати порушення прав людини.

Крім того, Microsoft мають намір ділитися додатковою інформацією із клієнтами, які розглядають можливість розгортання технологій розпізнавання осіб. Microsoft продовжує надавати клієнтам і іншим користувачам ту інформацію, що допоможе їм глибше зрозуміти існуючі можливості й обмеження технологій розпізнавання осіб і те, як ці функції можуть і повинні використовуватися, а також ризики неправильного використання.

По-четверте, Microsoft готові до повноцінної й відповідальної участі в обговореннях державної політики, що стосується розпізнавання осіб. Чиновники, цивільні організації й широка громадськість зможуть оцінити всі наслідки впровадження нових технологій тільки в тому випадку, якщо у творців цих технологій добре налагоджений обмін інформацією зі споживачами. Microsoft прагне спонукати уряд діяти, так що Microsoft повинні зробити крок уперед і поділитися всією необхідною інформацією. Microsoft робить це й виступає за етичне використання розпізнавання осіб і інших нових технологій як у США, так і в усьому світі.

Microsoft розуміє, що може з'явитися додаткова відповідальність, що повинні будуть прийняти на себе компанії IT-сектора. Microsoft пропонує наведений вище список не тому, що вважаємо, що в нього більше нема чого додати, а в надії, що він послужить гарним початком для просування вперед.

Нарешті, міркуючи про постійно, що розвиваються способах, застосування технологій, Microsoft визнає, що перед нами відкривається непросте майбутнє. Урядове агентство, що сьогодні робить щось неоднозначне, завтра може зробити щось гідне похвали. Тому нам потрібний принциповий підхід до технологій розпізнавання осіб, втілена в законі, що переживе його адміністрацію, що прийняла, і сіюхвиліні політичні вигоди.

Навіть у часи усе більше поляризованої політики Microsoft віримо в наші фундаментальні демократичні інститути й цінності. У Конгресі працюють вибрані народом представники, які мають у своєму розпорядженні інструменти, необхідними для оцінки нових технологій разом з усіма їхніми наслідками. Нам допомагає система сдержек і противаг Конституції, з якої Microsoft успішно пройшли від епохи свічкового освітлення до епохи штучного інтелекту. Як і в багатьох випадках у минулому, Microsoft повинні зробити так, щоб нові винаходи служили нашим демократичним свободам відповідно до верховенства закону. З огляду на глобальний охват технологій, ці проблеми необхідно вирішувати на міжнародному рівні, прислухаючись до думки інших шановних сторін. Міжнародне співтовариство повинне об'єднати зусилля, і Microsoft охоче виконуємо свою частину роботи.

1.2 Область застосування

Областю застосування є OSINT аналіз. Реалії сучасного світу такі, що наш співрозмовник \ клієнт \ співробітник \ цілком може виявитися зовсім не тим, за кого він себе видає. Приміром, він може виявитися професійним шахраєм, що прагне заволодіти нашою інформацією, грошима або ж власністю. Як же

убезпечити себе від подібних “контактів”? У цьому нам допоможуть всі ті ж реалії сучасного світу, а саме – відкриті джерела інформації.

OSINT – це акронім Open Source Intelligence (англ. – розвідка на основі відкритих джерел), що узвичаївся в епоху Інтернету й простоти доступу до даних.

OSINT використовується як у приватному секторі, так і у військових і розвідувальних службах у плині багатьох років, підходи й джерела інформації були відібрані й упорядковані фахівцями з Ленглі.

Суть усього процесу OSINT полягає в пошуку й аналізі інформації, отриманої з відкритих джерел. Можна одержати аналітичну інформацію із заданої теми досліджуючи безліч медіа й онлайн джерел. Ця інформація може бути екстрапольована в значимі відомості про компанію, особи, групи або країнах, які вони очолюють. Більшість таких підходів до збору інформації (harvesting) прив'язані до онлайн движків аналізу, що попереджає (Silobreaker.com і Basistech), які нібито можуть “пророчити майбутні дії”, як вони затверджують. Однак, базова ідея OSINT – це збір інформації для наступного аналізу звітів про Об'єкт.

Аналіз даних і погрішність/упередженість

Перед тим як позначити інструменти й методи OSINT, варто розповісти про “Аналіз”. Може так трапиться, що більша частина зусиль буде витрачена на збір інформації, що скоріше спантеличує або попросту є “дезінформацією”. Необхідно вміти відсівати факти, коментарі й іншу інформацію, а потім взяти те, що було зібрано й ретельно проаналізувати на предмет ключової інформації. Неопрацьована інформація повинна бути проаналізована й аналітик повинен вирішити що вірно, а що ні, а також призначити кожному джерелу інформації свій коефіцієнт.

Головне – не бути упередженим у своєму мисленні при проведенні OSINT аналізу. Принаймні, щоб створити повну картину про Об'єкт аналізу, необхідно порівнювати й зіставляти кожну зібрану інформацію й призначати їй свій коефіцієнт. Важливо зберігати широкий кругозір, і не дозволяти своєму

мисленню зациклюватися й плисти за течією. У противному випадку, зібрана інформація, швидше за все буде не коректною.

OSINT може бути зв'язаний як з особами, так і з організаціями. У теж час, люди на ділі можуть бути частиною руху або групи, що порівнянно реальної компанії, у такий спосіб макро й мікро дослідження дуже зв'язані.

Для підтвердження інформації може придатися й безпосередня взаємодія з Об'єктом. Процес OSINT живий, і аналітик повинен бути готів до такої взаємодії. Треба додержуватися підказок, задавати питання, звістки докладні записи, щоб потім скористатися їхнім умістом. Ключовим є перевірка даних і джерел, подібно гарним детективам і репортерам.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи інтелектуального розпізнання облич при виконанні OSINT-аналізу, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

2.1 Огляд існуючих систем, технологій, архітектур, програмних рішень за профілем теми випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти

Google

Google Search може надати багато інформації для OSINT. Хоча треба стати адептом “Google Hacking”, щоб використовувати всі можливості Google, тобто навчитися володіти ключами й запитами, які дозволяють одержати більше детальні результати.

Можна використовувати Google Alerts для автоматизованого пошуку по ключових словах. Сервіс інформує поштою про результати при кожному новому виявленні роботами. Зручно те, що результат приходить прямо в руки й немає необхідності здійснювати ручний пошук. Пошук застосуємо не тільки для рядків, але й для цілі вираження (наприклад у випадку пошуку плагіату).

Google Cache

Google Cache надає архівну інформацію активних сайтів, архіви відключених сайтів доступні на сайтах типу Wayback Machine (http://archive.org/web/), створений для пошуку інформації із сайтів, власники яких не хочуть більше публікувати свою інформацію.

Пошук по соціальних мережах

Twitter, Instagram, Facebook, та інші соціальні мережі – це відмінне джерело інформації, де люди, компанії й організації викладають безліч інформації, що не варто було б розміщати.

Представлені нижче сайти збирають подібну інформацію за допомогою пошукових систем і пропонують її (іноді в графічній формі).

WHOIS і подібні інструменти

ROBTEX

Власник інформації може спробувати сховати факт володіння доменом. Ця міра може виявитися запізнілою, тому що за інформацією про домен можна сказати досить багато чого. Є багато відповідних інструментів, вони легко перебувають Google'ом. Частина з них надає зв'язану інформацію, наприклад Robtex.

Robtex гарний тим, що надає інформацію про домен, про IP-адресу, на якому перебуває ресурс, про власника домену, а також про те, які ще домени використовують цей же серверний простір.

InfoSniper

InfoSniper – це розвідувач із “геолокацією” для IP адрес і доменів, що може вказати де сервер перебуває фізично. Такий пошук стає важливим у випадку розслідувань, де важлива юрисдикція.

Maltego

Maltego – це метапошукова система й графічна/реляційна утиліта для аналізу бази даних, що називають швейцарським ножем для збору даних і OSINT. За допомогою щоденних відновлень, можна одержати безліч даних, які можуть бути оброблені аж до готового результату.

Відмінна штука в Maltego – це наявність маппінгу інформації відповідно до її коефіцієнта (вагою). Це дозволяє дивитися на карту й бачити зв'язку між даними, хто з ким взаємодіє й контактує, як дані співвідносяться між собою. Це те, до чого необхідно звикнути й використовувати в OSINT.

Paterva “Casefile”

Новий продукт компанії Paterva, щось начебто “Maltego Light”, однак є одна серйозна перевага. Це цифрова біла дошка або “дошка вбивств” як у фільмах про поліцію. Можна прикріпити імена й фотографії, створити “case” файли.

Стадія «розвідки»

Перша стадія, стадія «розвідки», починається задовго до того, як атакуючий доторкнеться до першої машини жертви. Від кількості і якості даних, зібраних на цьому етапі, залежить успішність атаки й, саме головне, вартість її проведення.

Зрозуміло, можна стукатися експлойтами на уразливі сервіси, доступні на периметрі (і, наприклад, засвітити сплойти й своя присутність у балках систем захисту), а можна використовувати spear phishing і закріпитися на робочій станції усередині периметра. Результат буде досягнутий в обох випадках, але вартість атаки зовсім різна.

Стадія розвідки – ключова для вибору тактики, технік і тулз (tactics, techniques and procedures, далі TTPs), які будуть використовуватися для досягнення мети. Однак найчастіше завдання розвідки полягає в наступному: знайти якнайбільше потенційних точок входу для доступу до мети й оцінити вартість реалізації виявлених векторів. Для того щоб ускладнити життя атакуючий, котрий проводить розвідку, необхідно розуміти, які TTPs він використовує на даному етапі.

Просканувати, відзначити, повторити

Очевидно, для розуміння того, що розташовано на мережному периметрі, необхідно одержати діапазон IP-адрес, що належить цільової організації. У цьому списку можлива наявність IP-адрес третіх сторін (сервіси-провайдери, підрядники та інші) – атакуючий їх точно включить у скоп, а ти, як аудитор своєї мережі, цього робити не можеш. Отриманий діапазон IP можна занести в port-сканер. Замість **Nmap** я рекомендую використовувати для цієї мети **Masscan** або **ZMap** – це сильно скоротить час сканування.

```
<?xml version="1.0"?>
<!-- masscan v1.0 scan -->
<?xml-stylesheet href="" type="text/xsl"?>
<nmaprun scanner="masscan" start="1509715614" version=
"1.0-BETA"  xmloutputversion="1.03">
<scaninfo type="syn" protocol="tcp" />
<host endtime="1509715614"><address addr="80.169.130.206"
addrtype="ipv4"/><ports><port protocol="tcp" portid="35065"
><state state="open" reason="syn-ack" reason_ttl="236"/>
</port></ports></host>
<host endtime="1509715614"><address addr="80.169.130.199"
addrtype="ipv4"/><ports><port protocol="tcp" portid="37829"
><state state="open" reason="syn-ack" reason_ttl="236"/>
</port></ports></host>
<host endtime="1509715614"><address addr="80.169.130.199"
addrtype="ipv4"/><ports><port protocol="tcp" portid="34004"
><state state="open" reason="syn-ack" reason_ttl="236"/>
</port></ports></host>
<host endtime="1509715614"><address addr="80.169.130.197"
addrtype="ipv4"/><ports><port protocol="tcp" portid="48468"
><state state="open" reason="syn-ack" reason_ttl="236"/>
</port></ports></host>
<host endtime="1509715614"><address addr="80.169.130.205"
addrtype="ipv4"/><ports><port protocol="tcp" portid="14719"
><state state="open" reason="syn-ack" reason_ttl="236"/>
</port></ports></host>
<host endtime="1509715614"><address addr="80.169.130.202"
addrtype="ipv4"/><ports><port protocol="tcp" portid="3059"
><state state="open" reason="syn-ack" reason_ttl="236"/>
</port></ports></host>
<host endtime="1509715614"><address addr="80.169.130.202"
addrtype="ipv4"/><ports><port protocol="tcp" portid="2371"
><state state="open" reason="syn-ack" reason_ttl="236"/>
</port></ports></host>
<host endtime="1509715614"><address addr="80.169.130.208"
addrtype="ipv4"/><ports><port protocol="tcp" portid="2389"
><state state="open" reason="syn-ack" reason_ttl="236"/>
</port></ports></host>
<host endtime="1509715614"><address addr="80.169.130.206"
addrtype="ipv4"/><ports><port protocol="tcp" portid="32927"
```


Рисунок 2.1 – Фрагмент звіту про сканування Masscan

Так, для оцінки точок входу в медичні корпоративні мережі можна вивантажити з RIPE діапазони IP-адрес всіх організацій, у назві яких присутні ключові слова:

– healthcare;

– medic;

– clinic;

– surgery;

– hospit;

– dental;

– pharmacist.

Після цього можна запускати порт-сканер і почекати його видачу кілька днів.

Якщо сканувати за допомогою ZMap, то згодом можна скористатися утилітою ZTag для розміщення тегів по кожному виявленому сервісі. Теги проставляються на основі зібраної бази баннерів. У випадку зі сканом медзакладів отримані сервіси класифікуються в такий спосіб.

Tags	Hosts, %
http/https	87,1
smtp	2,8
dns	2,4
ssh	2,1
ftp	1,3
embedded	1,1
smb	0,9
routers	0,8
telnet	0,6

Tags	Hosts, %
ssh	13.87
infrastructure router	10.29
http	9.86
network	6.07
telnet	5.96
dns	1.52
building automation	0.98
scada	0.98
ftp	0.98
printer	0.98
smb	0.65
nas	0.54
DSL/cable modem	0.33
heartbleed	0.33
camera	0.22
dhe-export	0.22
smtp	0.22

Рисунок 2.2 – Топ сервіси на периметрі медичної інфраструктури

Серед тривіальних речей начебто веб-застосунків і поштових серверів перебувають цікаві застосунки: системи керування будинками (building

management systems; принтери (найчастіше без якої-небудь авторизації до адмін-панелей), сховища NAS (і навіть спеціалізовані PACS-сервери), розумні чайники та інше. Використовуючи кожний зі знайдених сервісів, атакуючий може визначати вектори атаки й оцінювати складність (читай – вартість) їхньої реалізації.

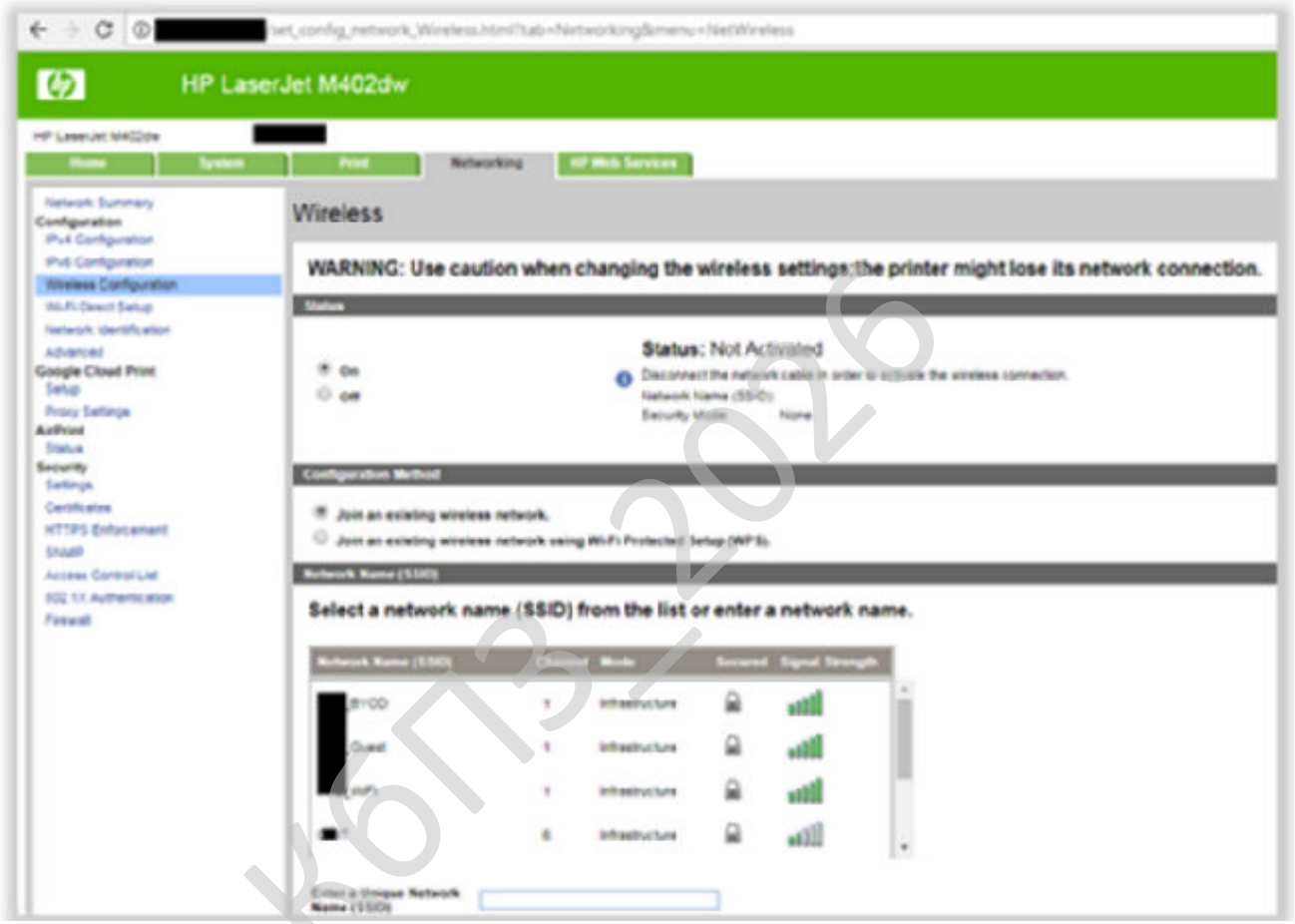

Рисунок 2.3 – Панель керування принтером, у якій, наприклад, видний список сусідніх бездротових мереж

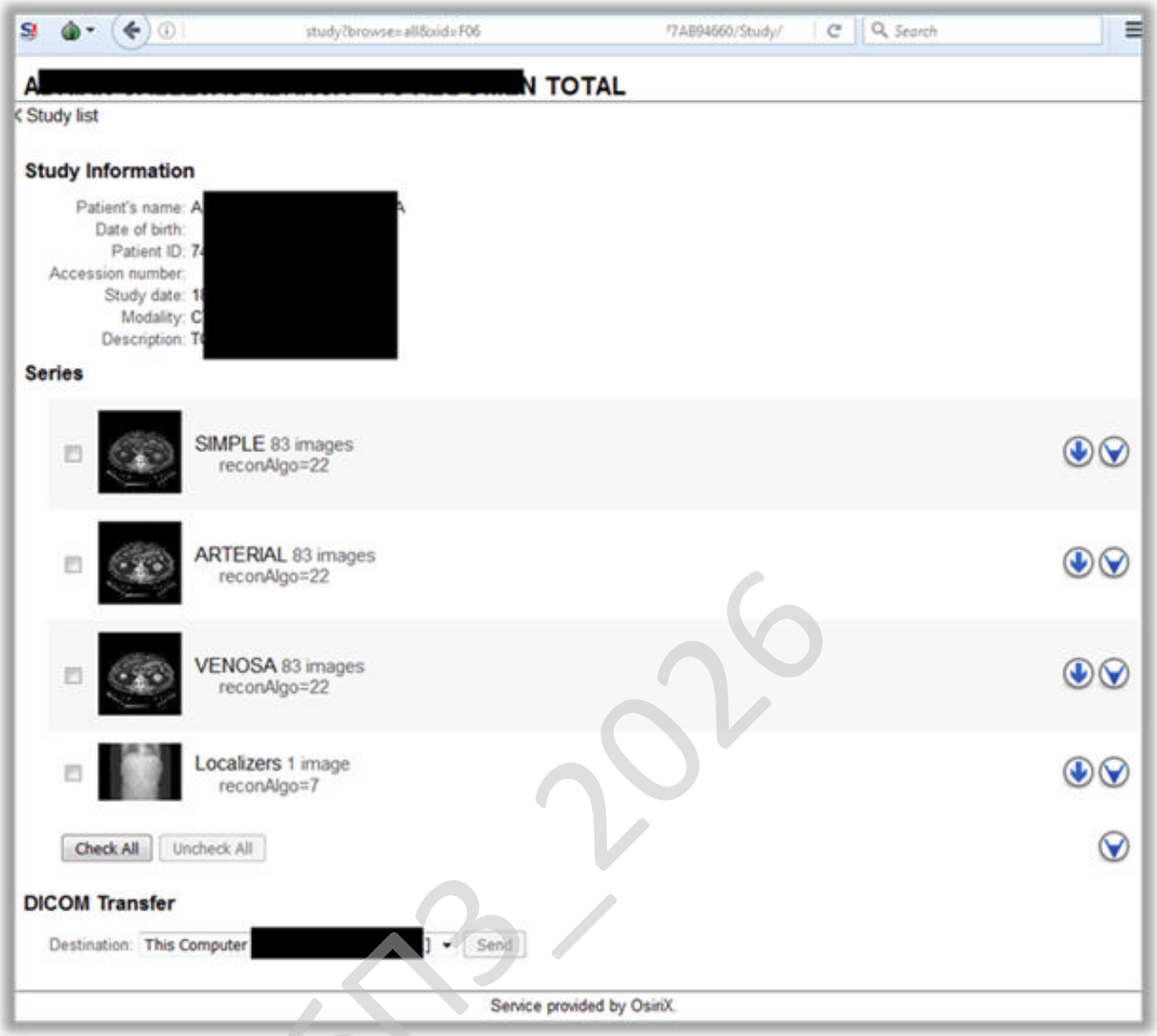

Рисунок 2.4 – Приклад уразливого медпорталу, ведучого до медичних даних

OSINT без інтерактиву

Інший відомий спосіб одержати інформацію про периметр і при цьому ніяк з ним не взаємодіяти – вивчити логи Shodan і аналогічних пошукових систем, роботи яких люб'язно все зробили за атакуючого.

Як було видно з логів вище, у публічному доступі перебувають усілякі сервери, які можуть нести в собі специфіку діяльності цільової організації й

зберігати коштовну інформацію. Приміром, якщо говорити про медичні компанії, те їхній периметр містить DICOM-пристрої й PACS-сервери (picture archiving and communication system). Це медичні системи, засновані на стандарті DICOM (digital imaging and communications in medicine, галузевий стандарт створення, зберігання, передачі й візуалізації медичних зображень і документів обстежених пацієнтів) і складаються з наступних компонентів:

– DICOM-клієнт – медичний пристрій, що володіє можливістю передачі інформації DICOM-серверу;

– DICOM-сервер – програмно-апаратний комплекс, що забезпечує одержання й зберігання інформації від клієнтів (зокрема, до таких пристроїв ставляться PACS-сервери);

– діагностична DICOM-станція й DICOM-принтери – програмно-апаратний комплекс, відповідальний за обробку, візуалізацію й печатку медичних зображень.

Відмінна риса більшості даних систем – наявність веб-інтерфейсу для керування ними через Мережу. Тут можуть виявитися уразливості, які зловмисник може використовувати для одержання доступу до коштовної інформації й процесів. Варто докладніше розглянути ці системи й перевірити, чи доступні вони з інтернету, тобто чи служать потенційною точкою входу для зловмисника.

Пошук DICOM-пристроїв можна почати з найпростішого запиту в розвідувачі Shodan: `DICOM port:104`.

Також можна спробувати знайти діагностичні DICOM-станції – спеціалізовані PACS-системи, які використовуються для обробки, діагностики й візуалізації даних. Приклад запиту для пошукової системи Censys: `pacs and autonomous_system.organization: (hospital or clinic or medical or healthcare)`.

Використовуючи стандартні запити в Shodan на одержання інформації про доступні ресурси на порту 445 (SMB), що атакує іноді може довідатися імена

внутрішніх ресурсів (серверів і робочих станцій), завдяки яким визначити, які вузли в мережі надалі становлять інтерес, а які – немає.

Збір інформації для соціальної інженерії

Закріпитися усередині корпоративного периметра ефективно дозволяє використання різних сценаріїв соціальної інженерії. Наприклад, розсилання фішингових повідомлень, які містять шкідливі вкладення або посилання, що ведуть на фішингові ресурси.

Для реалізації даних сценаріїв атакуючий також повинен зібрати інформацію про об'єкт атаки, щоб збільшити ймовірність, що адресат перейде по посиланню в листі або відкриє вкладення. У цей час служби безпеки великих організацій намагаються підвищувати поінформованість своїх співробітників про шкідливу кореспонденцію, що явно не йде на користь зловмисникам. Тепер лиходіям необхідно не тільки обійти спам-фільтри жертви й доставити «корисне навантаження» в Inbox, але й мотивувати об'єкт атаки виконати необхідні дії й, саме головне, не викликати підозр.

Достаток соціальних мереж, а також «розкутість» типового їхнього користувача надає атакуючу можливість витягти інформацію про свою жертву й скласти переконливий контекст «корисного навантаження»: текст супровідного листа, стиль комунікації.

Це творче завдання, і залежить вона від конкретної ситуації, але як загальний приклад можна привести ресурс, що використовує відкриті API популярних соцмереж для добування коштовної інформації про обліковий запис.

Приміром, що атакує, звернувшись до LinkedIn, зміг визначити ключових співробітників для організації атаки spear-phishing, їхні імена, прізвища, контактні дані (електронну пошту). Використовуючи ці дані, нескладно знайти аккаунти цих співробітників в інших соцмережах, наприклад в Facebook. За допомогою веб-сервісу атакуючий може зібрати цікаву статистику своїх жертв, наприклад де й у яких готелях вони «чекінились». Далі від імені готелю лиходій може відправити повідомлення з нагадуванням про оплату деяких Resort Fee і прикріпленим інвойсом у вигляді PDF-документа.

Recon як мистецтво

Тема добування даних про об'єкт атаки дуже велика, і про одну тільки розвідку на основі відкритих джерел написана не одна книга. Із цієї причини я й сфокусував увагу на зборі технічної інформації про периметр – саме в ньому найчастіше втримуються незакриті двері, які не патчаться роками й про які може не знати сам власник.

На додаток до цьому атакуючий, що пробирається через зовнішні ресурси, не взаємодіє з людиною (як це відбувається у випадку із соціальною інженерією), а виходить, єдине перешкода – це всілякі IDS/IPS, WAF і все, що фіксує активність на периметрі. Якщо ці засоби там взагалі є.

2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

Python – це потужна мова програмування, яка проста у вивченні. Він має ефективні структури даних високого рівня та простий, але ефективний підхід до об’єктно-орієнтованого програмування. Елегантний синтаксис і динамічна типізація Python разом з його інтерпретованим характером роблять його ідеальною мовою для створення сценаріїв і швидкої розробки додатків у багатьох сферах на більшості платформ.

Інтерпретатор Python і обширна стандартна бібліотека доступні у вихідному або двійковому вигляді для всіх основних платформ на веб-сайті Python https://www.python.org/ і можуть вільно поширюватися. Цей же сайт також містить дистрибутиви та вказівники на багато безкоштовних сторонніх модулів Python, програм і інструментів, а також додаткову документацію.

Інтерпретатор Python легко розширюється за допомогою нових функцій і типів даних, реалізованих у C або C++ (або інших мовах, які можна викликати з C). Python також підходить як мова розширення для налаштовуваних програм.

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускну кваліфікаційну роботу за першим (бакалаврським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи інтелектуального розпізнання облич при виконанні OSINT-аналізу.

В процесі розробки випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти необхідно виконати наступний обсяг роботи:

а) провести аналіз існуючих систем-аналогів для виявлення їх позитивних і негативних якостей. Результати аналізу врахувати в подальших розробках;

б) вибрати та обґрунтувати методику побудови системи контролю роботи технологічного обладнання на виробництві в автоматизованому режимі. Розробити функціональну та структурну схеми системи;

в) розробити програмне забезпечення системи, що дозволить реалізувати поставлену технічним завданням задачу. Побудувати блок-схеми алгоритмів програми та підпрограми;

г) організувати інтерфейс користувача з метою формування та виводу на екран ЕОМ повідомлень про некоректні дії користувача та нестандартні ситуації в роботі технологічного обладнання;

д) розробити рекомендації по організаційних та методичних заходах, які забезпечать впровадження системи в промислову експлуатацію та її подальшу успішну експлуатацію;

е) провести розрахунки по визначенню економічної ефективності розробленої системи;

ж) розробити заходи по охороні праці при впровадженні та експлуатації системи, а також розробити заходи з цивільного захисту;

з) сформувати висновки про виконаний обсяг робіт та одержані результати.

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

Наскільки добре спрацьовують пристрою ідентифікації по особі?

Комп'ютери здатні робити усе більше разючі речі, але, проте, вони не є чарівним засобом. Якщо самі люди часто не можуть довідатися суб'єкта, зображеного на фотографії, то якої надійності можна чекати від комп'ютера? Людський мозок найвищою мірою пристосований до розпізнавання осіб, приміром, маленькі діти запам'ятовують обличчя набагато краще, ніж будь-які інші форми. Крім усього іншого, людський мозок володіє набагато більшої, ніж комп'ютери, здатністю компенсації змін освітленості й кута зору. Справа в тому, що особи мають надзвичайно складний рисунок і часто відрізняються друг від друга тільки невловимими деталями, тому нерідко як людині, так і комп'ютеру буває неможливо зіставляти зображення при наявності різниці у освітленні, кута огляду камери, не говорячи вже зміни зовнішнього вигляду самої особи. Тому не дивно, що проведені урядом США дослідження програмних засобів розпізнавання по особі встановили високий відсоток помилкового розпізнавання безневинних людей і ідентифікації їх з фотографіями інших осіб, що перебувають у базі даних, а також нездатність цих засобів розпізнати справжніх злочинців, навіть якщо їхній фото є в наявності в базі даних.

Приміром, дослідження, проведене Національним інститутом стандартів і технологій (NIST), установило, що рівень помилкової ідентифікації або нерозпізнавання суб'єктів, чиї фотографії були зроблені всього 18 місяців назад, дорівнює 43%. При цьому фотографії, використані в дослідженні, були зняті в ідеальних умовах, що досить важливо, тому що програми розпізнавання по особі дуже погано справляються з оцінкою зміни освітленості або кута нахилу камери. "Важкі" для них і фотографії із жвавим тлом. Дослідження NIST також дійшло

висновку, що зміна кута нахилу камери на 45 градусів робить такі програми практично марними. Щонайкраще технологія розпізнавання особи спрацьовує в строго контрольованих умовах, коли суб'єкт дивиться прямо в камеру при яскравої освітленості. Правда, інше дослідження, проведене Міністерством оборони США, виявило високий рівень помилкової ідентифікації навіть при таких ідеальних умовах. Таким чином, від знімків із зернистим зображенням або старими фотографіями, типу тих, які зберігаються в особистих справах, буде дуже мало користі.

Крім цього, піднімалися питання про те, наскільки добре спрацьовують програми відносно темношкірих людей, риси особи яких можуть не розрізнятися об'єктивами, оптимізованими для зйомки людей зі світлою шкірою.

У розпорядженні уряду перебуває величезна, готова до використання база даних зображень осіб громадян – це фотографії водійських посвідчень. За законом, уряд не може продати ці фотографії приватним компаніям, однак не передбачено ніяких обмежень для використання цих знімків самим урядом для цілей відеоспостереження й ідентифікації. Федеральний уряд США приступилося до фінансування експериментальних проектів по використанню фотографій з водійських посвідчень у базах даних систем біометричної ідентифікації.

Недоторканність приватного життя

Одна з потенційних погроз полягає в тому факті, що розпізнавання по рисунку особи в сполученні з усе більше широким застосуванням відеоспостереження згодом буде проникати в усі нові й нові області. Після установки подібного роду систем спостереження їхнє застосування рідко обмежується цілями, для яких вони були призначені споконвічно. Виникають всі нові способи їхнього використання, а організації й оператори таких систем часто не можуть устояти від спокуси розширити з їхньою допомогою свою владу й вплив, наносячи тим самим ще один удар по правах громадян. В остаточному підсумку, погроза полягає в тому, що широко застосовуване відеоспостереження змінить характер, сприйняття і якість повсякденного життя громадян.

Іншою погрозою є можливість зловживання. Застосування ідентифікації по особі в місцях суспільного користування, таких як аеропорти, залежить від широкого впровадження відеомоніторингу – проникаючої форми спостереження, за допомогою якої особистий і приватний характер поводження може реєструватися графічними засобами. Досвід показує, що відеоспостереженням будуть зловживати. Системи відеокамер управляються звичайними людьми, які привносять на місце своєї роботи всі свої наявні упередження й упередженість. Наприклад, у Великобританії, де дуже широко використовуються камери відеоспостереження в публічних місцях, було виявлено, що оператори невиправдано зосереджують увагу на темношкірих, оператори-чоловіки часто схильні "підглядати" за жінками.

Хоча поліцейське відеоспостереження не так широко поширене в США, розслідування, проведене газетою Detroit Free Press показало, які види зловживань можуть мати місце. Перевіряючи, як використовувалася база даних, доступна правоохоронним органам Мічигану, журналісти газети виявили, що поліцейські застосовували її для того, щоб допомогти своїм друзям або навіть колегам по роботі стежити за жінками, загрожувати водіям, відслідковувати подружжя, що живуть окремо, і навіть залякувати політичних опонентів. Неминуча істина полягає в тому, що чим більше людей мають доступ до бази даних, тим більше ймовірність, що цією базою даних будуть зловживати.

Розпізнавання по особі особливо піддано зловживанням, тому що дана технологія може використовуватися пасивним образом, тобто для її застосування не потрібне згода або участь суб'єкта, за яким ведеться спостереження. Камеру можливо встановити в будь-якому місці й "натренувати" її на людей. Сучасні телекамери легко забезпечують перегляд осіб з відстані більше 90 м. Люди поводяться по-іншому, якщо знають, що за ними спостерігають, і мають право знати про те, що їхні рухи і їхнє поводження відслідковуються.

3.2 Розробка структурної схеми

У цьому розділі ми розглянемо, як використовувати створену в даній роботі бібліотеку Face Recognition OSINT на базі Python для пошуку й аналізу зображень. Крім того, ми розглянемо пакетну обробку фотографій і методи одержання статистики з метою того, щоб згодом зробити виводи про окрему персону або цілий вебсайт.

Крок 1: Установка залежностей

Бібліотека Face Recognition OSINT, що ми будемо використовувати, доступна прямо в менеджері пакетів Pip в Linux і macOS. Незважаючи на те, що цей пакет в Windows офіційно не підтримується, доступний посібник з установки й зконфігурований образ віртуальної машини.

Перед початком установки перевірте, присутні чи останні відновлення у вашій операційній системі. У дистрибутивах на базі Debian (наприклад, Ubuntu і Kali) відновлення й установка нового програмного забезпечення робиться за допомогою наступної команди:

```
sudo apt-get update && apt-get upgrade
```

У деяких дистрибутивах Pip доступний за замовчуванням. У противному випадку останню версію цього менеджера можна встановити за допомогою наступної команди:

```
sudo apt-get install python-pip
```

Далі встановлюємо бібліотеку Face Recognition OSINT за допомогою наступної команди:

```
sudo pip install face_recognition_OSINT
```

Тепер бібліотека готова до використання.

Крок 2: Організація інформації для аналізу

Якщо бібліотека для розпізнавання осіб реалізована на Python, виходить, ми можемо відносно просто впровадити цю бібліотеку в інші застосунки. Тобто, по суті, ми можемо розширити функціонала веб-камер, пошукових ботів і інших

застосунків, де доступні особи в цифровому форматі. Хоча з іншої сторони нам може знадобитися написання додаткового коду. Щоб спростити завдання ми буде використовувати утиліту, керовану з командного рядка, у зв'язуванні з іншими системними інструментами для виконання схожих функцій, але без необхідності в написанні нових скриптів.

Утиліта, що йде разом з бібліотекою, дозволяє працювати із двома наборами картинок. В одній папці будуть перебувати фотографії для аналізу, в іншій – колекція зображень, які користувач хоче зрівняти з першим набором.

Щоб створити масив картинок для фільтрації на предмет присутності певних осіб, ми можемо рекурсивно завантажити весь сайт, показаний нижче, за допомогою Wget. У реальному житті пентестер або дослідник як джерело зображень може використовувати сайти компаній, галереї, соціальні мережі й інші джерела.

Cwget можна працювати з командного рядка й завантажувати як окрему сторінку або картинку, так і цілі веб-сайти. Команда, показана нижче, дозволяє пройтися по кожному посиланню усередині заданого домену й завантажити весь доступний уміст. Щоб примусово припинити завантаження, необхідно нажати комбінацію клавіш Ctrl+C. У противному випадку процес весь процес завершиться після завантаження всього вмісту.

```
wget -p -r -E -e robots=off ---i convert-links -U mozilla ---ilevel 1whitehouse.gov
```

Прапор -p пропонує, що додатковий контент у вигляді картинок, стилів і скриптів також буде завантажуватися.

Прапор -E пропонує, що кожний завантажувати^ся файл, що, буде зберігатися разом з повним розширенням, щоб згодом ми легко змогли відфільтрувати картинки.

Прапор -r пропонує, що завантаження повинна бути рекурсивної. Відповідно, інформація з інших посилань і в інших директоріях також буде завантажуватися.

Прапор ---ilevel1 задає рівень рекурсії. Тобто в цьому випадку інформація буде завантажена тільки по посиланнях зі стартової сторінки. Щоб збільшити глибину рекурсії, необхідно або збільшити значення цього ключа, або забрати ключ зовсім (тоді глибина рекурсії буде нескінченною).

Прапор domain пропонує, що завантаження повинна здійснюватися винятково в рамках заданого домену.

Пам’ятайте, що рекурсивні запити можуть бути заблоковані або стати причиною проблем на деяких серверах. Тобто бажано або обмежити глибину рекурсії або скористатися прапором-w, щоб між запитами була затримка.

Прапор -e robots=off дозволяє уникнути ігнорування або блокування запиту як від небажаного пошукового бота.

Те ж завдання вирішує прапор -U mozilla, що визначає параметр useragent, щоб наші запити сприймалися як від легітимного браузера.

Прапор ---i convert-links конвертує посилання, щоб згодом ми змогли використовувати скачанні сторінки без яких-небудь проблем. Цей прапор прямо не пов'язаний з фільтрацією зображень, але спрощує завдання пошуку місцезнаходження картинок.

Як тільки ми налаштували збирач картинок протягом певного часу, то можемо створити дублікати файлів у новій директорії для спрощення наступної роботи. Спочатку створимо нову папку за допомогою наступної команди:

```
mkdir unknown_faces
```

Потім копіюємо всі зображення, скачанні за допомогою wget, у нову директорію за допомогою двох команд, показаних нижче. Спочатку копіюємо всі JPG-Файли, потім – всі PNG-Файли. Рядок urlfolder повинна бути замінена на папку, куди wget зберіг завантажений уміст. Звичайна назва цієї директорії пов'язане з ім'ям домену.

```
cp urlfolder**/*.jpg unknown_faces/
cp urlfolder**/*.jpg unknown_faces/
```

Команда cp копіює всі файли, що задовольняють фільтру й перебувають усередині зазначеної директорії (у нашім випадку -whitehouse.gov). Подвійна зірочка, використовувана в назві папки, визначає рекурсію. Одинарна зірочка пропонує, що повинні бути скопійовані всі файли з розширенням JPG або PNG. Останній параметр пропонує, що всі файли повинні бути скопійовані в папку unknown_faces.

Після того як картинки переміщені в нову директорію, ми можемо переглянути вміст цієї папки за допомогою команди ls.

```
ls unknown_faces/
```

Ми також можемо за допомогою оператора | перенаправляти цей список на вхід іншої команди, щоб порахувати кількість завантажених фотографій. Ця фішка може придатися для з'ясування, наскільки часто певна особа зустрічається серед набору зображень. У команді, показаній нижче, результат попередньої команди перенаправляється на вхід команди wc із прапором -l з метою підрахунку кількості рядків.

```
ls unknown_faces/ | wc -l
```

У результаті з'ясувалося, що ми завантажили 178 картинок. Як тільки зображення готові до використання, можна приступати до розпізнання осіб.

Крок 3: Підготовка списку для ідентифікації

Перед підготовкою списку осіб або інших ознак, які ми хочемо знайти в масиві завантажених зображень, потрібно створити нову директорію поза папкою, створеної wget. Перемістимося на рівень вище за допомогою команди `cd ../`, створимо папку known_faces, використовуючи команду `mkdir known_faces`, і перейдемо в нову директорію.

Потім у нову директорію потрібно скопіювати зображення, які ми хочемо знайти на завантажених фотографіях. В ідеалі потрібно використовувати фотографії осіб те саме що тим, які часто використовуються на веб-сторінках з контактами. На кожній фотографії, використовуваної для порівняння, повинне бути тільки одна особа. Для зручності можна перейменувати файл так, щоб ім'я

відповідало персоні, зображеної на фото, оскільки у випадку збігу, буде вертатися ім'я файлу.

Як тільки фотографії в обох директоріях підготовлені, приступаємо до використання утілити face_recognition_OSINT.

Крок 4: Розпізнавання осіб

Утиліта, що дозволяє працювати з командного рядка, що йде разом з бібліотекою Face Recognition OSINT, відносно проста у використанні й вимагає всього два параметри. У першому параметрі потрібно вказати директорію із зображеннями, які ми хочемо знайти (known_faces), у другому параметрі потрібно вказати директорію із зображеннями, серед яких буде здійснюватися пошук (unknown_faces). У підсумку команда виглядає так:

```
face_recognition_OSINT known_faces/ unknown_faces/
```

По кожній фотографії буде виведений один з наступних результатів: «ім'я одного з файлів» (виходить, знайдена відома особа), «невідома особа» або «не знайдена жодного особи». Щоб згодом автоматично обробити отриману інформацію, можна перенаправляти результати виконання в текстовий файл за допомогою оператора >.

```
face_recognition_OSINT known_faces/ unknown_faces/ > results.txt
```

Надалі ми буде використовувати цей текстовий файл для аналізу результатів пошуку.

Крок 5: Аналіз результатів пошуку

Один з можливих варіантів аналізу результатів роботи утиліти face_recognition_OSINT – одержання списку файлів, у яких знайдено одне з осіб. Наприклад, за допомогою утиліти grep, на вхід якої подається файл results.txt, одержимо перелік всіх файлів, де найшлася особа з файлу з ім'ям Anna Kovalenko.

```
grep "Anna Kovalenko" results.txt
```

Ми також можемо використовувати комбінацію інших системних утиліт для обчислення частотності кожного з результатів.

```
cat results.txt | sed 's/.*,//' | sort | uniq -c
```

Спочатку команда `cat results.txt` виводить уміст файлу results.txt у стандартний потік уведення. Потім отримані результати відправляються в потоковий редактор sed, що видаляє все до й включаючи кому. Далі список групується за допомогою утиліти sort. У самому кінці виробляється підрахунок кількості входжень за допомогою команди `uniq -c`.

У результаті виконання команди вище ми одержуємо список унікальних збігів і частотність кожного збігу.

Якщо ми хочемо одержати список певних зображень, знайдених утилітою face_recognition_OSINT, то можемо використовувати схожу техніку. Наприклад, за допомогою команди нижче створимо новий текстовий файл із переліком результатів пошуку, що задовольняють певному критерію.

```
grep "Anna Kovalenko" results.txt | sed 's/,.*$//' > identified.txt
```

Спочатку за допомогою grep у файлі results.txt шукаємо всі результати, де є рядок Anna Kovalenko. Отримані результати відправляються в sed, де в кожному рядку весь уміст після коми, включаючи саму кому, віддаляється. Підсумкові результати зберігаються у файлі identified.txt (робимо перенапрямок за допомогою оператора >).

Якщо ми запустимо команду head identified.txt, те побачимо тільки список файлів, що, наприклад, можна використовувати як вхідний параметр ще одного скрипту для копіювання всіх цих файлів у нову папку.

Новий скрипт можна створити в редакторі Nano за допомогою команди `nano identified.sh`.

```
#!/bin/bash
mkdir identified
while read line; do
filename=$(echo $line | sed 's/.*\///')
cp $line identified/$filename
```

```
done < identified.txt
```

У першому рядку, `#!/bin/bash`, декларується, що скрипт призначено для оболонки Bash.

У другому рядку створюється нова директорія identified, куди будуть переміщені всі знайдені файли.

Третій і останній рядок утворять цикл читання рядків з файлу identified.txt. Кожний рядок файлу модифікується усередині конструкції while.

У першому рядку циклу (четвертий рядок коду) створюється змінна filename. Цієї змінної привласнюється ім'я файлу, отримане після обробки рядка потоковим редактором sed. У кожному рядку віддаляється весь уміст до слешу.

Ця змінна використовується в останньому рядку циклу в команді, що копіює файл у директорію identified і призначає скопійованому файлу ім'я зі змінної filename.

Після того як редагування закінчене, зберігаємо скрипт, нажавши комбінацію клавіш Ctrl+O. Для виходи з редактора натискаємо Ctrl+X. Після того як скрипт збережений у файлі identified.sh, додаємо права на виконання за допомогою команди chmod +x.

Далі запускаємо скрипт за допомогою простої команди `./identified.sh`.

Після завершення виконання скрипта, у директорії identified будуть перебувати всі зображення, які задовольняють вашому фільтру. Як ви могли переконатися, весь процес автоматизований і серйозно заощаджує час у порівнянні з обробкою списку фотографій вручну.

Як захиститися від технік, пов'язаних з розпізнаванням особи

На жаль, надійно захиститися від розпізнавання вашої особи на фотографіях можна лише одним способом. А конкретно – перестати викладати свої фотографії в мережу і як можна рідше з'являтися в публічних місцях. Крім того, можна виставляти більше строгі настроювання у ваших аккаунтах, не

використовувати соціальні мережі або приховувати свою особу з появою в публічних місцях. В основному ці методи доставляють багато незручностей.

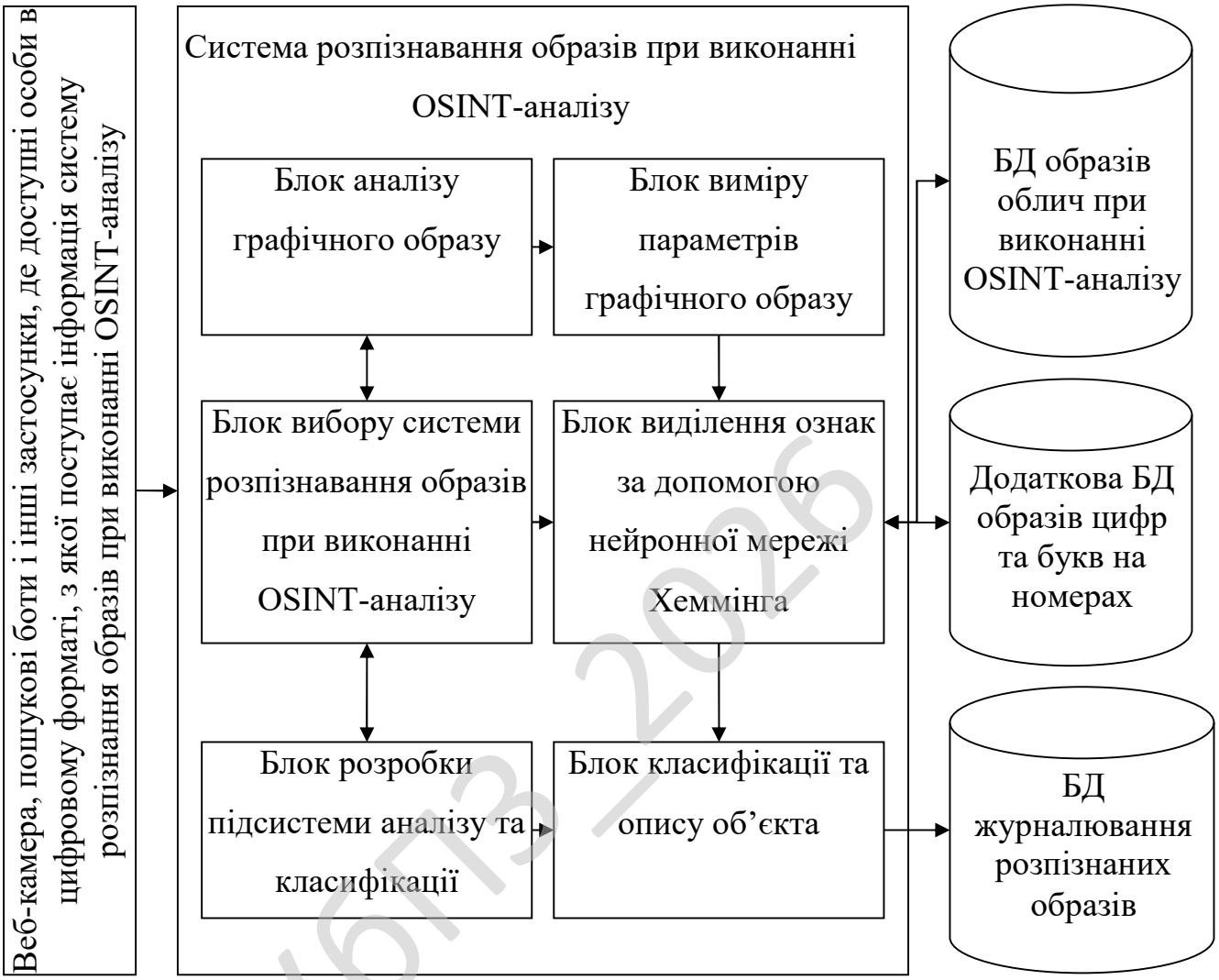

Рисунок 3.1 – Структурна схема системи

У цілому, небезпека, пов'язана з розпізнаванням осіб, більше ставиться до даних, прив'язаним до конкретного профілю або аккаунту, а не сама фотографія. Пам'ятайте, що якщо ви виклали свої фотографії або поділилися іншою інформацією в мережі, те вже не можна бути впевненими в повній захищеності.

Структурна схема системи наведена на рисунку 3.1. Структурно система складається з наступних частин:

1. База даних журналювання розпізнаних образів.

2. База даних образів облич при виконанні OSINT-аналізу.

3. База даних образів цифр та букв на номерах автомобілів.

4. Веб-камера, пошукові боти і інші застосунки, де доступні особи в цифровому форматі, з якої поступає інформація систему розпізнання образів.

5. Система розпізнання образів.

3.3 Розробка функціональної схеми

На рисунку 3.2 зображена функціональна схема системи. Нижче розглянемо її більш докладно.

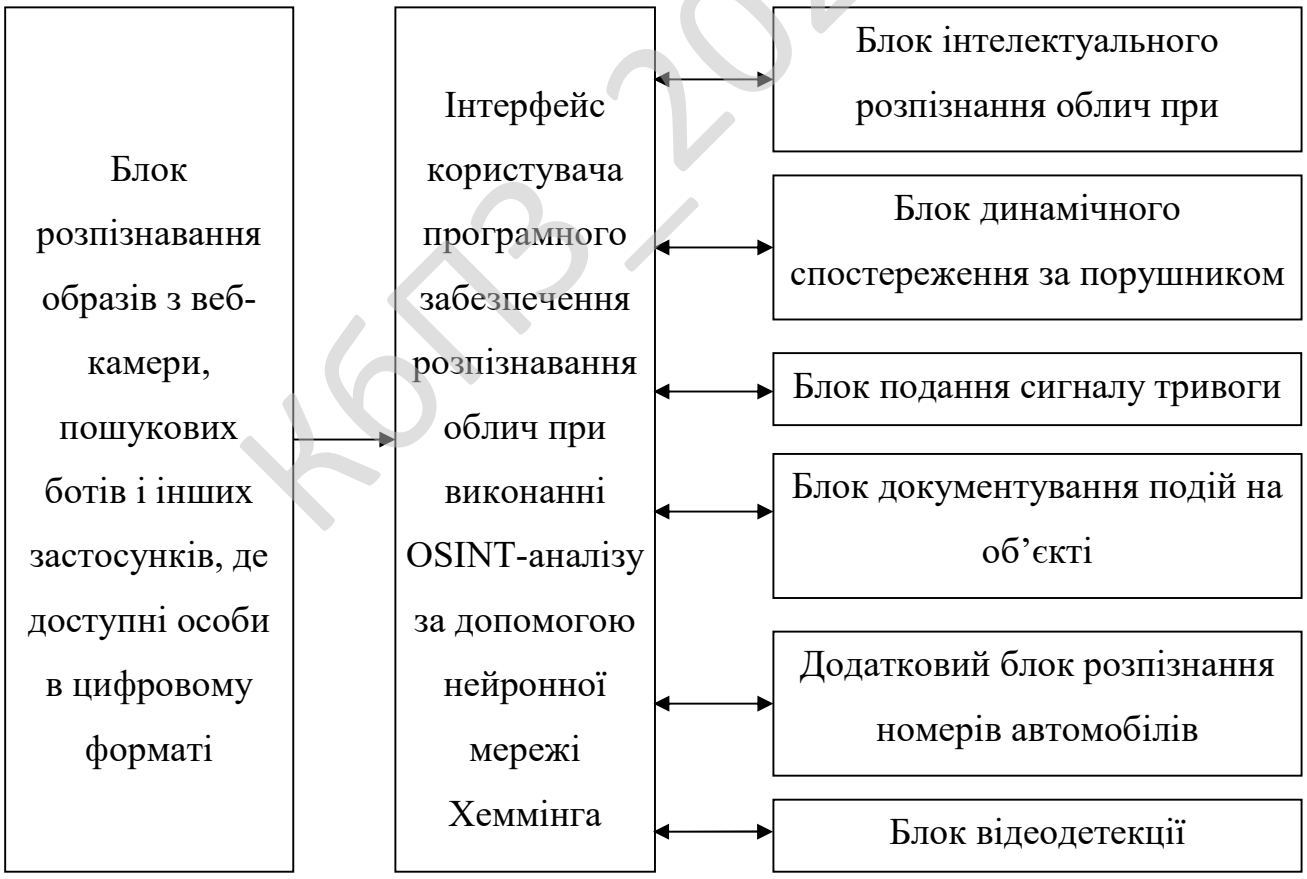

Рисунок 3.2 – Функціональна схема системи

Функціональна схема системи включає в себе наступні функціональні блоки:

1. Блок розпізнання образів з веб-камери, пошукових ботів і інших застосунків, де доступні особи в цифровому форматі.
2. Блок розпізнавання облич при виконанні OSINT-аналізу.
3. Блок відеодетекції.
4. Блок динамічного спостереження за порушником.
5. Блок подання сигналу тривоги.
6. Додатковий блок розпізнавання номерів автомобілів.
7. Блок документування подій на об’єкті.

Розглянемо ці блоки більш детально.

Блок розпізнання образів з веб-камери, пошукових ботів і інших застосунків, де доступні особи в цифровому форматі

Виділимо етапи при рішенні завдання розпізнавання зображень:

– Сприйняття поля зору.

– Сегментація.

– Нормалізація виділених об'єктів.

– Розпізнавання.

Виходячи із цього, використовуються наступні основні принципи:

– Принцип цілісності – розпізнаваний об'єкт розглядається як єдине ціле, що складається зі структурних частин, зв'язаних між собою просторовими відносинами.

– Принцип двонаправленості – створення моделі ведеться від зображення до моделі й від моделі до зображення.

– Принцип передбачення. Полягає у формуванні гіпотези про зміст зображення.

– Принцип цілеспрямованості, що включає сегментацію зображення й спільну інтерпретацію його частин.

– Нічого не робити без процедури розуміння (сприйняття поля зору).

– Принцип максимального використання моделі проблемного середовища, використання заздалегідь відомих, апріорних параметрів.

Етап інтерпретації зображення не позначений чіткими границями й включається частково в процес сегментації й остаточно завершується на етапі розпізнавання.

Природно задатися питанням: а чи не можна брати зображення й послідовно порівнювати його з еталонами по ряду яких-небудь ознак? Але отут виникає ряд проблем і складностей:

– Тло. Як правило, зображення пред'являються на складному динамічному тлі.

– Орієнтація. Зображення еталона й вхідних зображень відрізняються положенням у поле зору.

– Перешкоди. Вхідні зображення не збігаються з еталонами за рахунок випадкових і локальних перешкод.

– Висвітлення. Відмінності вхідних і еталонних зображень виникає за рахунок зміни освітленості, підсвічування.

– Перетворення. Еталони й зображення можуть відрізняти складні геометричні перетворення.

Додатковий блок розпізнавання номерів автомобілів

Система використовується автоматичної реєстрації й розпізнавання автомобільних номерів на контрольно-пропускних пунктах на підприємствах, платних стоянках і гаражних комплексах, на постах ДПС. Захист із системою контролю доступу дозволяє автоматизувати контрольно-пропускний режим. Система дозволяє вести розшук автомобілів у викраденні.

Функціональні можливості:

– автоматична реєстрація автомобільних номерів і розпізнавання;

– збереження номера й відеозапису проїзду транспортного засобу в базі даних із вказівкою дати й часу;

– автоматичне зіставлення автомобільного номера з наявними базами даних (наприклад, автомобілів, що мають право допуску на територію підприємства, або автомобілів, що перебувають у викраденні) і видача відповідного повідомлення операторові;

– автоматизація контрольно-пропускного режиму при використанні із пристроями контролю доступу;

– пошук у базі даних за номером, датою, часом;

– формування звітів за номером, датою, часом.

Характеристики:

– Ширина контрольованої зони проїзної частини – від 1,5 до 3,5 метрів.

– Глибина зони контролю – 10 метрів (для КПП: 2-4 метри).

– Кількість оброблюваних кадрів (з обліком 100 % потокової завантаженості на всіх камерах):

– канал – 25 к/с, до 150 км/год;

– каналу й більше – 16 к/с (сумарне кіл-у кадрів), до 75 км/ч.

– Імовірність розпізнавання – не менш 90%.

– Припустимі кути установки веб-камери, пошукових ботів і інших застосунків, де доступні особи в цифровому форматі.

– По вертикалі – не більше 30°.

– По горизонталі – не більше 20°.

– Крен номерного знака ±15°.

– Освітленість – не менш 50 люкс.

– Розпізнавані номери – 7 типів (тло білий і жовтий).

– Вимога до бази даних – підтримка інтерфейсу ADO.

Блок розпізнавання облич при виконанні OSINT-аналізу

Система захвата осіб дозволяє виділяти тільки особи людей, вибирати найбільш виразне зображення з декількох варіантів і зберігати їх у базі даних. Створення бази осіб людей на прохідних підприємств, у місцях масового

скупчення людей (культурно-розважальні центри, вокзали, стадіони й т.д.) полегшує роботу з архівами при розслідуванні позаштатних ситуацій.

Далі система розпізнавання особи ідентифікує особистість і автоматизує пошук зображень у базах даних.

Блок відеодетекції

Виявлення переміщення в зоні спостереження (відеодетекція). Такі пристрої часто вбудовуються в стандартні мультиплексори. При цьому оператор може задавати зону на екрані монітора, рух у якій викликає сигнал тривоги.

Блок динамічного спостереження за порушником

Системи динамічної цілевказівки аналізують зміни координат характерних точок об'єкта, наприклад центра ваги, кольору.

Розглянувши усі блоки функціональної схеми перейдемо до розгляду діаграми взаємодії процесів, які відбуваються у системі.

Розглянувши усі блоки функціональної схеми перейдемо до розгляду діаграми взаємодії процесів, які відбуваються у системі.

Блок подання сигналу тривоги

Веб-камера, пошукові боти і інші застосунки, де доступні особи в цифровому форматі використовується разом з технічним засобом охорони для підтвердження факту спрацьовування останнього.

Блок документування подій на об’єкті

Матеріал відеоархівів може виявитися корисним як доказова база при розслідуванні несанкціонованих дій.

Розглянувши усі блоки функціональної схеми перейдемо до розгляду діаграми взаємодії процесів, які відбуваються у системі.

3.4 Розробка діаграми процесів

Розглянемо розроблену діаграму процесів яка зображена на рисунку 3.3. Основна будова діаграми процесів полягає у графічному представленні складу

сукупностей даних, що характеризуються як співвідношення різних частин кожної з сукупностей. Склад статистичної сукупності графічно може бути представлений як за допомогою абсолютних, так і відносних показників.

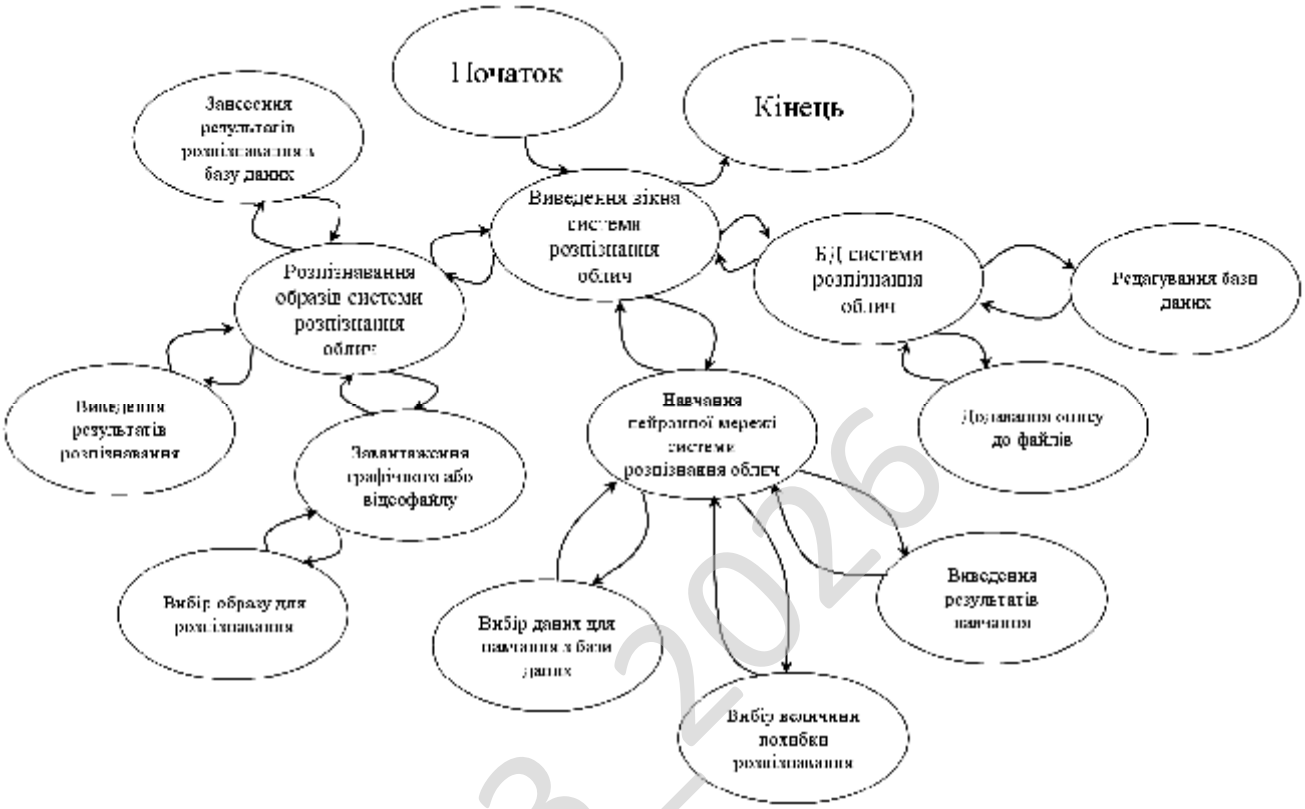

Рисунок 3.3 – Діаграма взаємодії процесів

Графічне зображення складу сукупності по абсолютними і відносними показниками сприяє проведенню більш глибокого аналізу і дозволяє проводити аналіз системи. Діаграма взаємодії процесів використовується для візуалізації процесів обробки даних (структурне проектування). Для розробника вважається звичним спочатку креслити діаграму взаємодії процесів даних рівня контексту, завдяки чому буде показано взаємодію системи. Ця діаграма в подальшому підлягає уточненню шляхом деталізації процесів та потоків даних з метою показати систему що розробляється. При детальному її розгляді можна побачити як саме проходе взаємодія у розробленій системі. Використовується модель проектування, графічне представлення «потоків» даних в інформаційній системі.

Діаграми потоків даних містять чотири типи елементів:

– Зовнішні по відношенню до системи сутності.

– Потоки даних між елементами трьох попередніх типів.

– Процеси які являють собою трансформацію даних в рамках описуваної системи.

– Сховища даних (репозиторії).

Таким чином, розглянувши опис системи, структурну, функціональну схеми системи, та діаграму взаємодії процесів перейдемо до опису блок-схем основної програми, та підпрограм, які використовуються, для реалізації системи.

4 РЕАЛІЗАЦІЯ ПРОЕКТУ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ПРАВИЛЬНІСТЬ ПРОЕКТНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

Розглянемо реалізацію бакалаврської дипломної роботи. Були проведені розрахунки і підібрані набори тестових даних для перевірки правильності реалізації проектних рішень.

Було створено блок-схеми роботи системи. Перед їх розглядом необхідно провести роз’яснення який саме тип блок-схем використовується. Блок-схема це представлення задачі для її аналізу або розв'язування за допомогою спеціальних символів (геометричних образів), які позначають такі елементи, як операції, потік, дані тощо.

Блок вхідних та вихідних даних прийнято позначати паралелограмом, блок обчислень (обробки) даних – прямокутником, блок прийняття рішень – ромбом, еліпсом – початок та кінець алгоритму.

У інформаційних технологіях функціональна схема складається з функціональних блоків, які являють собою конструктивно відособлені частини (елементи або пристрої) автоматичних систем, які виконують певні функції. Функціональні блоки на схемі позначають прямокутниками, всередині яких надписують їх найменування відповідно до функцій, що виконуються. Зв'язки між функціональними блоками (внутрішні впливи) позначаються лініями зі стрілками, які вказують напрям впливів.

Функціональні схеми можуть виконуватися в укрупненому і розгорненому вигляді. У першому випадку на схемі зображають найважливіші блоки системи і зв'язки між ними.

У другому варіанті схема відображається більш детально, що полегшує її читання та ілюструє принцип роботи.

Основні елементи схем алгоритму це термінатор, процес, рішення, зумовлений процес (підпрограма), дані та з'єднувач. Термінатор це елемент відображає вхід із зовнішнього середовища або вихід з неї (найчастіше застосування – початок і кінець програми). Всередині фігури записується відповідна дія.

Процес це виконання однієї або кількох операцій, обробка даних будь-якого виду (зміна значення даних, форми подання, розташування). Всередині фігури записують безпосередньо самі операції. Рішення це показує рішення або функцію перемикального типу з одним входом і двома або більше альтернативними виходами, з яких тільки один може бути обраний після обчислення умов, визначених всередині цього елементу.

Вхід в елемент позначається лінією, що входить зазвичай у верхню вершину елементу. Якщо виходів два чи три то зазвичай кожен вихід позначається лінією, що виходить з решти вершин (бічних і нижній). Якщо виходів більше трьох, то їх слід показувати однією лінією, що виходить з вершини (частіше нижній) елемента, яка потім розгалужується.

Відповідні результати обчислень можуть записуватися поруч з лініями, що відображають ці шляхи. Зумовлений процес (підпрограма) це символ відображає виконання процесу, що складається з однієї або кількох операцій, що визначені в іншому місці програми (у підпрограмі, модулі). Всередині символу записується назва процесу і передані в нього дані.

Дані це перетворення у форму, придатну для обробки (введення) або відображення результатів обробки (виведення). Цей символ не визначає носія даних (для вказівки типу носія даних використовуються специфічні символи). З'єднувач це символ відображає вихід в частину схеми і вхід з іншої частини цієї схеми.

Використовується для обриву лінії та продовження її в іншому місці (приклад: поділ блок-схеми, що не поміщається на листі). Відповідні сполучні символи повинні мати одне (при тому унікальне) позначення.

Блок-схеми показують весь процес роботи системи з підсистемами та частково доказують правильність вибраних проектних рішень. Тому від точності і детальної блок-схеми залежить результат всієї програми.

При виборі початкової точки відліку при побудові схем було враховано, що виходячи з вибору мови програмування і інших технічних засобів, програма буде об’єктно-орієнтована що вимагає високого рівня декомпозиції задач на класи.

На рисунку 4.1 зображена основна блок-схема програми, на рисунку 4.2 зображено роботу підпрограми.

З яких видно що робота основної програми складається з початкових етапів ініціалізації ПЗ, перевірки наявності ресурсів системи, блоку початку основного циклу з чеканням запиту від користувача в якому відбувається виклик підсистеми та останньої стадії – перевірка поточного стану з завершенням роботи розробленого ПЗ. При роботі підпрограми виконується основний функціонал системи з циклічними послідовностями, перевіркою поточного стану та поверненням в основну програму прапорів стану виконання.

Було використано підходи з використанням UML, це уніфікована мова моделювання, використовується у парадигмі об'єктно-орієнтованого програмування. Є невід'ємною частиною уніфікованого процесу розробки програмного забезпечення. UML є мовою широкого профілю, це відкритий стандарт, що використовує графічні позначення для створення абстрактної моделі системи, називаної UML-моделлю.

UML був створений для визначення, візуалізації, проектування й документування в основному програмних систем. UML не є мовою програмування, але в засобах виконання UML-моделей як інтерпретованого коду можлива кодогенерація.

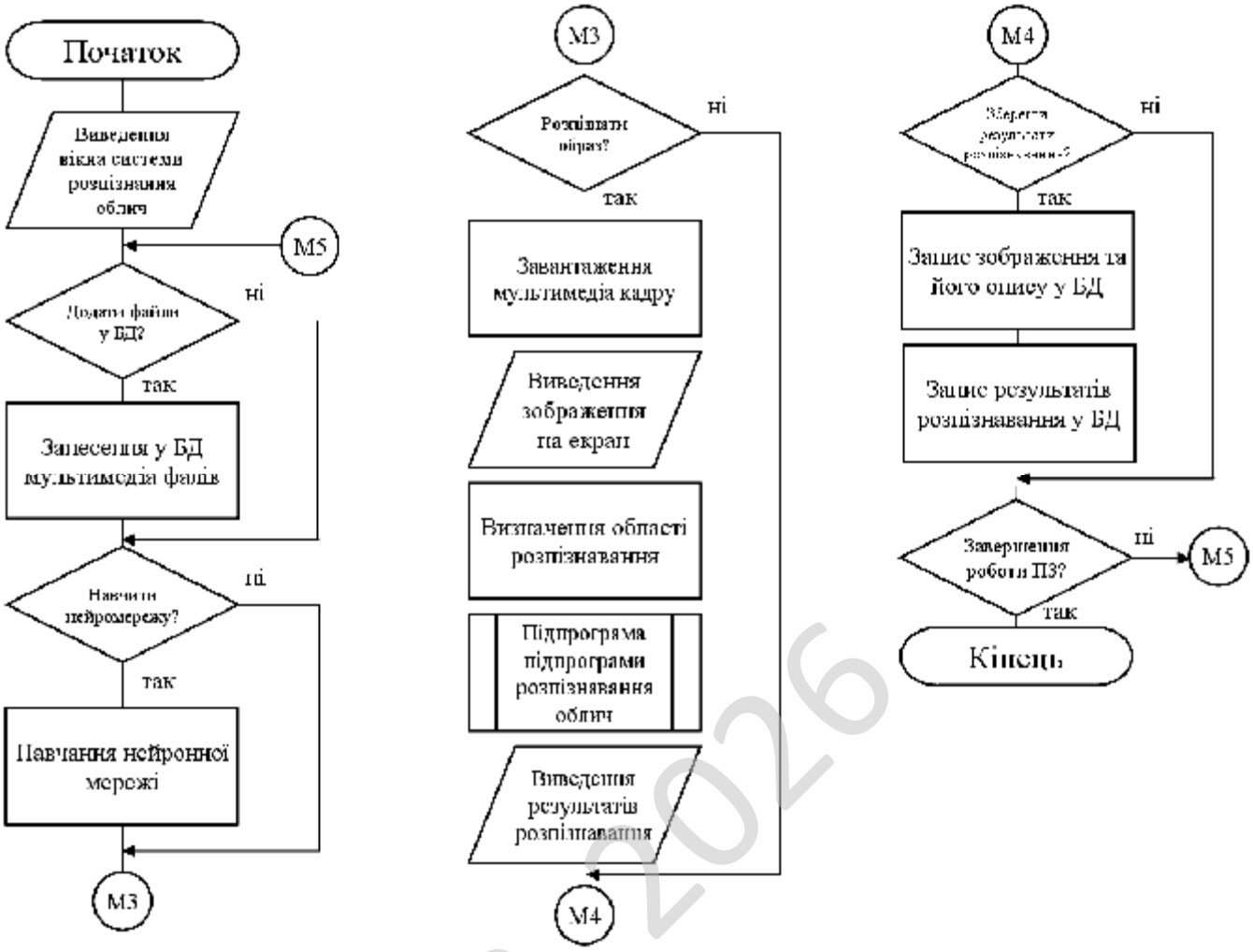

Рисунок 4.1 – Блок-схема основної програми

Було використано наступні підходи UML: діаграма діяльності (діаграми поведінки типу); діаграма прецедентів (діаграми поведінки типу).

Діаграма діяльності. Це візуальне представлення графу діяльностей. Граф діяльностей є різновидом графу станів скінченного автомату, вершинами якого є певні дії, а переходи відбуваються по завершенню дій. Дія є фундаментальною одиницею визначення поведінки в специфікації. Дія отримує множину вхідних сигналів, та перетворює їх на множину вихідних сигналів.

Одна із цих множин, або обидві водночас, можуть бути порожніми. Виконання дії відповідає виконанню окремої дії. Подібно до цього, виконання діяльності є виконанням окремої діяльності, буквально, включно із виконанням

тих дій, що містяться в діяльності. Кожна дія в діяльності може виконуватись один, два, або більше разів під час одного виконання діяльності. Щонайменше, дії мають отримувати дані, перетворювати їх та тестувати, деякі дії можуть вимагати певної послідовності.

Специфікація діяльності (на вищих рівнях сумісності) може дозволяти виконання декількох (логічних) потоків, та існування механізмів синхронізації для гарантування виконання дій у правильному порядку.

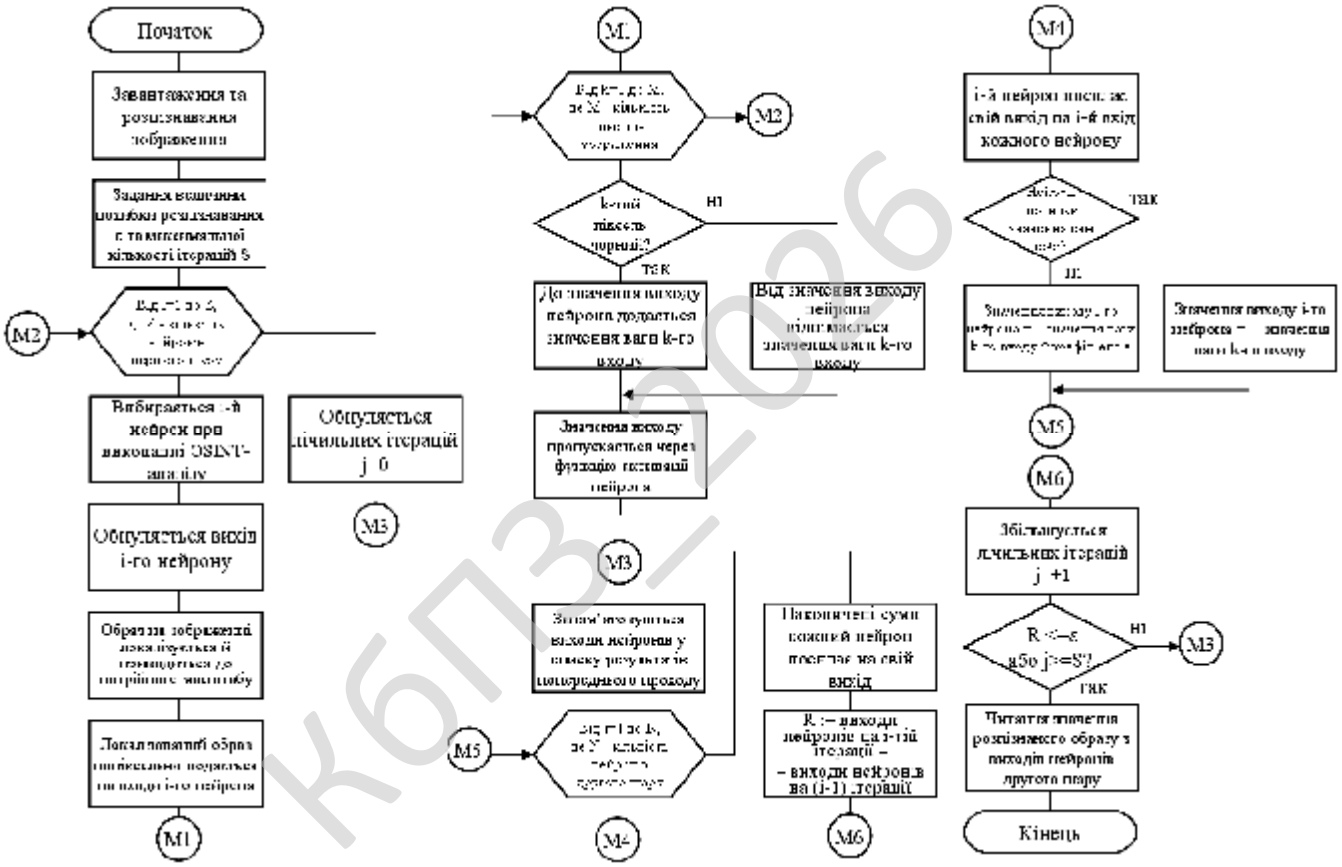

Рисунок 4.2 – Блок-схема роботи підпрограми

Діаграма прецедентів це діаграма, на якій зображено відношення між акторами та прецедентами в системі. Також, перекладається як діаграма варіантів використання.

Діаграма прецедентів є графом, що складається з множини акторів, прецедентів (варіантів використання) обмежених границею системи

(прямокутник), асоціацій між акторами та прецедентами, відношень серед прецедентів, та відношень узагальнення між акторами. Діаграми прецедентів відображають елементи моделі варіантів використання.

Суть даної діаграми полягає в наступному: проектована система представляється у вигляді безлічі сутностей чи акторів, що взаємодіють із системою за допомогою так званих варіантів використання. Варіант використання (use case) використовують для описання послуг, які система надає актору. Іншими словами, кожен варіант використання визначає деякий набір дій, який виконує система при діалозі з актором.

При цьому нічого не говориться про те, яким чином буде реалізована взаємодія акторів із системою.

У мові UML є кілька стандартних видів відношень між акторами і варіантами використання:

– асоціації (association relationship);

– включення (include relationship);

– розширення (extend relationship);

– узагальнення (generalization relationship).

При цьому загальні властивості варіантів використання можуть бути представлені трьома різними способами, а саме – за допомогою відношень включення, розширення і узагальнення.

Відношення асоціації – одне з фундаментальних понять у мові UML і в тій чи іншій мірі використовується при побудові всіх графічних моделей систем у формі канонічних діаграм.

Включення (include) у мові UML – це різновид відношення залежності між базовим варіантом використання і його спеціальним випадком. При цьому відношенням залежності (dependency) є таке відношення між двома елементами моделі, при якому зміна одного елемента (незалежного) приводить до зміни іншого елемента (залежного).

Відношення розширення (extend) визначає взаємозв'язок базового варіанта використання з іншим варіантом використання, функціональна поведінка якого задіюється базовим не завжди, а тільки при виконанні додаткових умов.

Розглянемо діаграму об'єктів в UML це діаграма, що відображає об'єкти та їх зв'язки в певний момент часу. Діаграма об'єктів може розглядатись як окремий випадок діаграми класів, на якій можуть бути представлені як класи, так і екземпляри (об'єкти) класів. Схожою за змістом є діаграма взаємодії (collaboration diagram).

Діаграми об'єктів не мають власної нотації. Оскільки діаграми класів можуть відображати об'єкти, то діаграма класів, на якій відображено лише об'єкти, та не відображено класи, може вважатись діаграмою об'єктів.

Діаграма об'єктів відображає об'єкти та зв'язки в певний момент роботи програми. Об'єкти можуть містити інформацію про власні значення а не про описання. Для відображення загальних шаблонів об'єктів та зв'язків, що можуть багаторазово створюватись під час роботи програми, слід використовувати діаграму взаємодії, яка може відображати характеристики об'єктів та зв'язків. Екземпляр діаграми взаємодії створює діаграму об'єктів.

Діаграма об'єктів не відображає еволюцію системи під час роботи. Натомість, слід використовувати діаграми взаємодії з повідомленнями, або діаграми послідовності.

Діаграма розгортання (deployment diagram) це діаграма в UML, на якій відображаються обчислювальні вузли під час роботи програми, компоненти, та об'єкти, що виконуються на цих вузлах. Компоненти відповідають представленню робочих екземплярів одиниць коду.

Компоненти, що не мають представлення під час роботи програми на таких діаграмах не відображаються; натомість, їх можна відобразити на діаграмах компонент.

Діаграма розгортання відображає робочі екземпляри компонент, а діаграма компонент, натомість, відображає зв'язки між типами компонент.

Опис системи

Розроблена система інтелектуального розпізнання облич при виконанні OSINT аналізу являє собою локальне ПЗ, яке забезпечує накопичення, обробку, кластеризацію, пошук, верифікацію та звітність за зображеннями облич у межах окремих аналітичних кейсів.

Система працює як настільний ПЗ на основі Tkinter, використовує SQLite для зберігання даних, OpenCV для компютерного зору, NumPy для чисельної обробки, Pillow для перегляду зображень та стандартні бібліотеки Python для організації файлової підсистеми, журналювання, безпеки доступу й експорту результатів.

ПЗ орієнтується на локальну обробку власних або службово дозволених медіа матеріалів і не виконує звернення до зовнішніх веб ресурсів, зовнішніх баз даних чи хмарних сервісів. Таке рішення зменшує ризики витоку даних, підвищує контрольованість обробки та робить систему придатною для навчального і дослідницького середовища закладу вищої освіти. Код прямо фіксує цю концепцію в інтерфейсі, де користувач бачить повідомлення про локальний характер аналізу.

Архітектура системи

Архітектура системи має модульний характер і поєднує рівень збереження даних, рівень бізнес логіки, рівень обробки зображень та рівень графічного інтерфейсу. У верхній частині розташовується головний клас Application, який створює конфігурацію, ініціалізує базу даних, служби авторизації та OSINT аналізу, допоміжний модуль перегляду зображень, а також запускає механізм взаємодії між потоками через чергу повідомлень.

Після входу в систему керування передається класу MainFrame, який формує багатовкладковий інтерфейс користувача.

Рівень даних реалізує клас DatabaseManager. Він створює структуру бази, забезпечує зєднання з SQLite, виконує запити, масові операції, читання одного або кількох записів, а також веде аудит дій користувача. На цьому рівні

створюються таблиці користувачів, кейсів, осіб, медіафайлів, виявлених облич та журналу аудиту. Кожна сутність має чітко визначені звязки, а зовнішні ключі забезпечують цілісність інформації.

Рівень безпеки доступу реалізує клас AuthService, який відповідає за вхід до системи, створення користувачів та зміну пароля. Шифрування паролів здійснюється не простим зберіганням тексту, а через стійку до перебору процедуру на основі hashlib.pbkdf2_hmac. Це свідчить про коректне проєктне рішення, оскільки навіть у локальному дослідницькому застосунку дані облікових записів не зберігаються у відкритому вигляді.

Рівень обробки зображень формує клас ImageProcessor. Саме він виконує перевірку наявності бібліотек, зчитування файлів, вилучення метаданих, виявлення облич, обрізання фрагментів, оцінювання якості, побудову векторів ознак, обчислення схожості, збереження копій зображень та збереження обрізаних областей. На цьому рівні зосереджується інтелектуальна частина системи.

Рівень прикладної логіки реалізує клас OSINTService. Він створює кейси, веде перелік осіб, імпортує набори зображень, виконує кластеризацію облич, пошук, статистичний аналіз, верифікацію еталонного зразка, формує звіти у форматах CSV і JSON, а також надає дані для аудиту. Фактично саме цей клас поєднує БД, обробку зображень і графічний інтерфейс в єдину функціональну систему.

Рівень відображення реалізують класи LoginFrame, CasesTab, ImportTab, SearchTab, VerifyTab, StatsTab, ReportsTab, AdminTab та MainFrame. Кожен із цих модулів відповідає за окремий сценарій взаємодії користувача із системою.

Організація файлового простору і конфігурація

На початку коду задаються глобальні константи APP_NAME, BASE_DIR, DATA_DIR, DB_PATH, LOG_PATH, ORIGINALS_DIR, CROPS_DIR, EXPORTS_DIR. Таке рішення забезпечує єдину точку налаштування та уніфікований доступ до всіх службових каталогів.

Система автоматично створює потрібні директорії при запуску, тому розгортання не потребує ручної підготовки файлової структури.

Конфігурація застосунку описується dataclass класом AppConfig. У ньому визначаються розмір обличчя для нормалізації 96 на 96 пікселів, мінімальний розмір обличчя для детекції 60 на 60 пікселів, поріг схожості 0,92, максимальний розмір превю 360 на 240 та перелік підтримуваних форматів зображень. Використання окремого конфігураційного класу підвищує гнучкість системи, спрощує супровід і робить код більш керованим.

Модуль авторизації та захисту доступу

Клас PasswordManager реалізує дві функції. Функція hash_password генерує сіль і формує криптографічний геш пароля. Функція verify_password перевіряє, чи відповідає введений пароль збереженому гешу. Для формування гешу використовується 200000 ітерацій алгоритму PBKDF2 HMAC SHA256. Це означає, що одна перевірка пароля виконує 200000 раундів обчислень замість одного, тому вартість грубого перебору зростає у 200000 разів відносно простого одноразового гешування. Для навчального настільного засобу такий рівень захисту є цілком обґрунтованим.

Клас DatabaseManager у методі _ensure_default_admin створює початковий обліковий запис адміністратора. Це дозволяє одразу увійти в систему після запуску, а потім змінити пароль через відповідну вкладку. Клас AuthService додатково перевіряє мінімальну довжину логіна й пароля, контролює унікальність імені користувача та фіксує факт успішного входу в аудиті. Унаслідок цього підсистема доступу не лише аутентифікує користувача, а й підтримує підзвітність усіх критичних дій.

Структура бази даних

У БД створюються таблиці users, cases, persons, media_files, faces, audit_logs. Таблиця користувачів містить імям користувача, геш пароля, сіль, роль, дату створення та дату останнього входу. Таблиця кейсів зберігає назву, опис, правову підставу, автора та час створення.

Таблиця осіб прив’язує псевдоніми до конкретного кейсу. Таблиця медіафайлів містить початковий шлях, локальну копію, геш файлу, розміри, метадані та час імпорту. Таблиця облич містить координати прямокутника, оцінку якості, вектор ознак, код кластера, шлях до обрізаного фрагмента та можливу привязку до особи. Таблиця аудиту фіксує всі ключові дії.

Важливим проєктним рішенням є наявність індексів за часом створення, ідентифікаторами кейсів, гешами медіафайлів, кодами кластерів і привязкою облич до осіб. Це зменшує час доступу до інформації при пошуку, статистиці, побудові звітів і повторному аналізі даних.

Модуль обробки зображень

Клас ImageProcessor є центральним компонентом інтелектуальної частини системи. Метод validate_environment перевіряє наявність OpenCV, NumPy, Pillow та каскадного класифікатора Haar. Метод hash_file обчислює SHA256 для кожного файлу. Метод load_image читає зображення через OpenCV. Метод extract_metadata формує словник із базовими атрибутами файла та за наявності Pillow вилучає EXIF дані.

Виявлення облич реалізує метод detect_faces, який переводить зображення у відтінки сірого та передає його у CascadeClassifier.detectMultiScale. Обране рішення є раціональним для кваліфікаційної роботи, оскільки Haar каскад не потребує потужного графічного прискорювача, швидко запускається на звичайному персональному компютері та добре підходить для локального автономного засобу.

Метод crop_face вирізає знайдену область, а метод save_face_crop зберігає її на диску. Метод save_original_copy копіює оригінальний файл у локальне сховище. Таким чином система не залежить від вихідної папки після завершення імпорту та зберігає власну керовану колекцію матеріалів.

Оцінювання якості фрагмента обличчя

Метод quality_score формує інтегральну оцінку якості на основі різкості, яскравості та контрасту.

Різкість обчислюється через дисперсію лапласіана, яскравість через середнє значення пікселів, контраст через стандартне відхилення. Підсумкова формула використовує ваги 0,5 для різкості, 0,3 для контрасту та 0,2 для яскравості. Сума коефіцієнтів дорівнює 1,0, тому оцінка залишається нормованою і зрозумілою для інтерпретації. Таке співвідношення ваг є логічним, оскільки саме різкість найсильніше впливає на придатність обличчя до порівняння, контраст визначає виразність структурних деталей, а яскравість відіграє допоміжну роль.

Якщо прийняти типовий приклад, коли відносний показник різкості дорівнює 0,80, контрасту 0,60, а яскравості 0,70, то інтегральний бал становить $0,80 \times 0,5 + 0,60 \times 0,3 + 0,70 \times 0,2 = 0,72$. Отже система віддає перевагу чітким обличчям навіть тоді, коли освітлення не є ідеальним. Для реальних колекцій OSINT матеріалів це виправдано, оскільки зображення часто мають неоднорідне освітлення, але саме втрата різкості найбільш критично знижує інформативність.

Формування вектора ознак

Метод compute_feature переводить фрагмент обличчя у відтінки сірого, нормалізує до розміру 96 на 96, після чого формує три групи ознак. Перша група це гістограма інтенсивностей на 32 інтервали. Друга група це LBP гістограма на 32 інтервали, яка описує локальну текстуру. Третя група це гістограма напрямків градієнта на 18 інтервалів.

Загальна розмірність вектора ознак становить $32 + 32 + 18 = 82$ компоненти. Для представлення у форматі float32 це дає $82 \times 4 = 328$ байт на один вектор без урахування службових накладних витрат. Для 10000 фрагментів потрібно приблизно 3,28 МБ сирої числової інформації, що є дуже помірним обсягом.

Навіть якщо вектор зберігається у JSON і фактичний розмір одного запису зростає приблизно до 0,8 або 1,0 КБ, для 10000 облич сумарний обсяг становить лише близько 8 або 10 МБ, що також є прийнятним для SQLite бази навчального масштабу.

Цей розрахунок підтверджує правильність рішення використовувати компактний опис ознак замість великих нейромережевих дескрипторів.

LBP модуль працює з нормалізованим зображенням 96 на 96 пікселів і формує карту розміром 94 на 94, тобто аналізує 8836 локальних позицій. Це достатньо для стійкого опису текстури без надмірного збільшення обчислювальних витрат. Модуль градієнтів використовує 18 інтервалів по 20 градусів, а отже повністю покриває коло 360 градусів. Така деталізація є збалансованою між точністю та швидкодією.

Після обєднання гістограми нормалізуються за довжиною вектора. Це робить подальше косинусне порівняння коректним і зменшує вплив абсолютного масштабу окремих ознак.

Механізм кластеризації

Кластеризація реалізується в методі _find_cluster_for_face. Для кожного нового обличчя система бере до 2000 останніх записів поточного кейсу, відновлює їх вектори ознак із JSON, обчислює косинусну схожість і вибирає найкращий збіг.

Якщо найкраще значення не менше 0,92, нове обличчя відноситься до наявного кластера. Якщо ж поріг не досягається, система створює новий код кластера виду CL із випадковим шістнадцятковим ідентифікатором.

Поріг 0,92 є достатньо строгим. Для двох нормованих векторів евклідова відстань повязана з косинусною схожістю співвідношенням $\sqrt{(2 \times (1 - 0{,}92))} = 0{,}4$. Це означає, що об’єднання відбувається лише для дуже близьких описів і допомагає зменшити хибне злиття різних людей в один кластер. Для OSINT сценарію, де помилкове ототожнення є критичнішим за появу зайвих дрібних кластерів, таке проєктне рішення є правильним.

Якщо оцінити лише скалярний добуток, одна перевірка нового фрагмента проти 2000 кандидатів потребує $2000 \times 82 = 164000$ парних множень.

Це невелике навантаження для сучасного персонального компютера. Навіть якщо врахувати додаткові операції для норм і сервісну обробку JSON,

система зберігає достатню швидкодію на локальному рівні. Отже обмеження вибірки до 2000 останніх фрагментів є раціональним компромісом між швидкістю та якістю кластеризації.

Імпорт і аналіз колекції зображень

Метод import_folder реалізує один із найважливіших сценаріїв. Він перевіряє середовище, шукає у вказаній папці всі підтримувані файли, обчислює для кожного SHA256, перевіряє дублікати в межах кейсу, копіює оригінал у локальне сховище, формує метадані, виконує детекцію облич, обчислює ознаки, визначає належний кластер, зберігає обрізане обличчя та реєструє результат у БД.

Дуже важливим є використання SHA256 для контролю дублювання. Оскільки для одного кейсу геш файла має бути унікальним, повторний імпорт тієї самої фотографії не створює надлишкових записів. Це економить дисковий простір, зменшує повторне навантаження на алгоритм і не спотворює статистику. Для навчальної інформаційної системи така перевірка є принципово правильною.

Імпорт виконується у фоновому потоці, а повідомлення передаються через чергу queue.Queue. Завдяки цьому графічний інтерфейс не блокується під час тривалого аналізу великих наборів зображень. Це підтверджує продуману архітектуру взаємодії між інтерфейсом та обчислювальними процедурами.

Пошук і верифікація

Метод search_faces дозволяє відбирати записи за кейсом, псевдонімом особи та назвою файла. Для цього формується гнучкий SQL запит із умовами, що додаються лише за потреби. Результат містить не тільки службові поля, а й шлях до фрагмента, координати рамки, кластер і назву кейсу. Це робить підсистему пошуку придатною і для оперативного огляду, і для аналітичної роботи.

Метод verify_probe реалізує верифікацію контрольного зразка. Система читає вибране зображення, знаходить усі обличчя, бере найбільше з них, обчислює ознаки і порівнює з усіма фрагментами вибраного кейсу. Далі результати агрегуються так, щоб для кожної мітки або кластера лишався

найкращий збіг. Після цього список сортується за спаданням схожості й обрізається до 20 позицій.

Якщо кейс містить, наприклад, 5000 фрагментів, то лише частина скалярного добутку потребує 5000 × 82 = 410000 парних множень. Для локального застосунку це також є прийнятним навантаженням. Отже рішення не використовувати зовнішній пошуковий індекс або окремий векторний сервер є виправданим для дипломного проєкту навчального масштабу.

Статистика і звітність

Метод get_statistics обчислює кількість медіафайлів, кількість облич, кількість іменованих осіб, середню оцінку якості, найчисельніші кластери та найактивніші мітки осіб. Такий набір показників є достатнім для оперативного оцінювання стану кейсу й ефективності накопиченої колекції.

Експорт реалізують методи export_case_csv та export_case_json. У форматі CSV система формує табличне подання для подальшої обробки в електронних таблицях або аналітичних пакетах. У форматі JSON зберігається ширший набір відомостей, включно зі статистикою, кластерами та переліком осіб. Це підвищує універсальність програмного засобу і робить його придатним для дослідницької документації, архівування і подальшого аналізу результатів.

Графічний інтерфейс

Клас LoginFrame створює вікно входу, де користувач вводить логін і пароль. Після успішної автентифікації відкривається головне вікно, яке формує клас MainFrame. У верхній частині відображається назва системи та відомості про поточного користувача. Нижче розташовується вкладковий інтерфейс.

Вкладка CasesTab забезпечує створення кейсу, перегляд наявних кейсів, створення псевдонімів осіб та привязку кластера до конкретної особи. Саме тут користувач переходить від автоматичної кластеризації до змістовного аналітичного маркування.

Вкладка ImportTab відповідає за вибір кейсу, вибір папки, запуск імпорту та перегляд журналу поточного процесу. У цій частині особливо важливим є

виведення проміжних повідомлень, які дають змогу контролювати хід аналізу великих колекцій.

Вкладка SearchTab надає пошук за кейсом, псевдонімом і назвою файла, відображає табличні результати та показує превю обраного фрагмента. Також тут система виводить структуровану службову інформацію у текстове поле.

Вкладка VerifyTab забезпечує перевірку еталонного зразка щодо вмісту вибраного кейсу. Вона демонструє дослідницьку цінність системи, оскільки дає змогу оцінити, з якими кластерами або іменованими особами зразок має найбільшу схожість.

Вкладка StatsTab відображає агреговану статистику і допомагає швидко оцінити наповнення кейсу. Вкладка ReportsTab формує файли звітності. Вкладка AdminTab надає засоби зміни власного пароля, створення нових користувачів і перегляду журналу дій.

Допоміжні компоненти

Клас PreviewHelper створює мініатюри зображень для інтерфейсу. Він зменшує їх до допустимого розміру та переводить у формат, придатний для Tkinter. Це знижує навантаження на інтерфейс і робить перегляд стабільним навіть при роботі з великими файлами.

Журналювання налаштовується через модуль logging. Усі суттєві події, включно з помилками та службовими повідомленнями, записуються у файл application.log. Такий підхід спрощує діагностику, аудит експлуатації та аналіз виняткових ситуацій.

Обґрунтування правильності проєктних рішень

Обрана архітектура є правильною для випускної кваліфікаційної роботи, оскільки вона поєднує завершеність функціоналу, зрозумілу модульність, локальну автономність і помірні вимоги до апаратних ресурсів. Застосування SQLite є виправданим через невеликий або середній обсяг наборів даних та простоту розгортання.

Розрахунок обсягу векторних ознак показує, що навіть десятки тисяч фрагментів не створюють критичного навантаження на сховище.

Використання каскаду Haar для детекції облич є обґрунтованим з погляду доступності, швидкодії та відсутності потреби в спеціалізованому обладнанні. Нормалізація до 96 на 96 пікселів забезпечує достатню деталізацію. Співвідношення 96 до 60 дорівнює 1,6, тому навіть мінімально допустиме обличчя після масштабування не втрачає базової структури, необхідної для побудови гістограмних і текстурних ознак.

Комбінація трьох груп ознак у 82 вимірах є вдалою, оскільки поєднує глобальний розподіл яскравостей, локальний текстурний опис і напрямкову інформацію контурів. Це підвищує стійкість до часткових змін освітлення, невеликих поворотів та варіацій якості. Водночас вектор залишається компактним, а отже обчислення косинусної схожості не створює надмірного навантаження.

Поріг 0,92 у кластеризації зменшує ризик хибного обєднання різних осіб. Для OSINT аналізу це особливо важливо, оскільки помилкове злиття даних різних людей може призвести до неправильних висновків. Вибір жорсткішого порога робить систему більш консервативною, але саме така стратегія є доречною в умовах навчального дослідження з акцентом на коректність результату.

Рольова модель доступу, журналювання, перевірка дублювання за SHA256, багатопотоковий імпорт та експорт результатів демонструють, що розробка виходить за межі простого лабораторного прикладу й являє собою цілісну інформаційну систему. Вона містить засоби безпеки, зберігання, аналітики, пошуку, звітності та адміністрування, що підтверджує правильність обраної структури програмного рішення.

Розглянемо роботу системи Redmine яка дозволила створити ПЗ. Це вільне серверне ПЗ для управління проектами та відстежування помилок. До системи входить календар-планувальник та діаграми Ганта для візуального представлення ходу робіт за проектом та строків виконання. Redmine написано на мові Ruby і є

ПЗ розробленим з використанням відомого веб-фреймворку Ruby on Rails, що означає легкість в розгортанні системи та її адаптації під конкретні вимоги. Для кожного проекту можна вести свої вікі та форуми.

Функціональні можливості:

– Ведення декількох проектів.

– Гнучка система доступу з використанням ролей.

– Система відстеження помилок.

– Діаграми Ганта та календар.

– Ведення новин проекту, документів та управління файлами.

– Сповіщення про зміни за допомогою RSS-потоків та електронної пошти.

– Власна Wiki для кожного проекту.

– Форуми для кожного проекту.

– Облік часових витрат.

– Налаштування власних (custom) полів для задач, затрат часу, проектів та користувачів.

– Легка інтеграція із системами керування версіями (SVN, CVS, Git, Mercurial, Bazaar и Darcs).

– Створення записів про помилки на основі отриманих листів

– Підтримка LDAP автентифікації.

– Можливість самореєстрації нових користувачів.

– Багатомовний інтерфейс (у тому числі українська мова).

– Підтримка СКБД: MySQL, PostgreSQL, SQLite.

Діаграма Ганта (*Gantt chart*, також стрічкова діаграма, графік Ганта) – це популярний тип діаграм, який використовується для ілюстрації плану, графіка робіт за будь-яким проектом. Є одним з методів планування та управління проектами.

Діаграма Ганта являє собою відрізки (графічні плашки), розміщені на горизонтальній шкалі часу. Кожен відрізок відповідає окремому завданню або підзадачі. Завдання і підзадачі, складові плану, розміщуються по вертикалі.

Початок, кінець і довжина відрізка на шкалі часу відповідають початку, кінцю і тривалості завдання. На деяких діаграмах Ганта також показується залежність між завданнями.

Діаграма може використовуватися для представлення поточного стану виконання робіт: частина прямокутника, що відповідає завданню, заштриховується, відзначаючи відсоток виконання завдання; показується вертикальна лінія, що відповідає моменту «сьогодні».

Часто діаграма Ганта використовується спільно з таблицею зі списком робіт, рядки якої відповідають окремо взятій задачі, зображеній на діаграмі, а стовпці містять додаткову інформацію про задачу.

Система відстеження помилок Багтрекер – прикладна програма для допомоги розробникам програмного забезпечення (програмістам, тестувальникам тощо) враховувати і контролювати помилки, знайдені у програмах, питання щодо функціональності, рішення та оновлення, побажання користувачів, а також стежити за процесом їх виконання.

Кожному, хто розробляв програмні продукти, добре знайоме співвідношення «20/80» – останні 20 % роботи тривають 80 % часу.

Як це не парадоксально, але нічого дивного в цій пропорції немає, адже саме на завершальній стадії починається тестування проекту, коли виявляються помилки, і що більший проект, то більше буде знайдено помилок.

Водночас досить часто виявляється, що більшість цих помилок були відомі та могли бути виправлені з меншими витратами на попередніх стадіях роботи, але не були вчасно описані, а потім загубилися серед інших важливих завдань.

Отже, система відстеження помилок у найпростішому варіанті – це процес, що включає в себе виявлення помилки, її опис, виправлення і перевірку цього виправлення, тобто процес «стеження» за багом протягом всього як його життєвого циклу, так і життєвого циклу розробки в цілому.

Сукупність інформації про дефект. Головний компонент такої системи – база даних, що містить відомості про виявлені дефекти. Ці відомості можуть включати в себе:

– номер (ідентифікатор) дефекту;

– хто повідомив про дефект;

– дата і час виявлення дефекту;

– версія продукту, в якій виявлено дефект;

– серйозність (критичність) дефекту та пріоритет рішення;

– опис кроків для відтворення дефекту (неправильної поведінки програми);

– відповідальний за усунення дефекту;

– обговорення можливих рішень та їх наслідків;

– поточний стан виправлення дефекту;

– версії продукту, в якій дефект виправлений.

Крім того, розвинені системи надають можливість прикріплювати файли, які допомагають описати проблему, наприклад, дамп пам'яті або скріншот.

Використання. Основна перевага систем відстеження помилок полягає в забезпеченні чітких централізованих оглядів, запитів на розробку (включаючи помилки і виправлення) та їх стан. У корпоративному середовищі, системи відстеження помилок можуть бути використані для генерації звітів по продуктивності програмістів виправлення помилок. Однак, це може іноді приводити до неточних результатів, тому що різні помилки можуть мати різні ступені пріоритету та серйозності, що пов'язано з складністю їх фіксації.

Життєвий цикл дефекту. Як правило, система відстеження помилок використовує той чи інший варіант «життєвого циклу» помилки, стадія якого визначається поточним станом помилки.

Типовий життєвий цикл дефекту:

1. Новий – дефект зареєстрований тестувальником;

2. Призначений – призначений відповідальний за виправлення дефекту;

3. Дозволений – дефект переходить назад у сферу відповідальності тестувальника. Як правило, супроводжується резолюцією, наприклад:

– Виправлено (виправлення включені у версію таку-то);

– Дубль (повторює дефект, що вже знаходиться в роботі);

– Не виправлено (працює відповідно до специфікації, має занадто низький пріоритет, виправлення відкладено до наступної версії тощо);

– «В мене все працює» (запит додаткової інформації про умови, в яких дефект проявляється).

4. Далі тестувальник проводить перевірку виправлення, залежно від чого дефект або знову переходить у стан «Призначений» (якщо він описаний як виправлений, але не виправлений), або у стан «Закрито».

5. Відкрито повторно – дефект знайдено знову в іншій версії.

Система може надавати адміністраторові можливість налаштування користувачі, які можуть переглядати і редагувати помилки залежно від їх стану, переводити їх в інший стан або видаляти.

У корпоративному середовищі, система відстеження помилок може використовуватися для отримання звітів, що показують продуктивність програмістів при виправленні помилок. Однак, часто такий підхід не дає достатньо точних результатів через те, що різні помилки мають різну ступінь серйозності та складності. При цьому серйозність проблеми прямо не стосується складності її усунення.

Таког була використана водоспадна (каскадна) модель життєвого циклу ПЗ (waterfall model) – послідовний метод розробки програмного забезпечення, названий так через діаграму схожу на водоспад.

Ця модель розробки запозичена з системної інженерії у виробництві та будівництві – областях, в яких зміни на пізніх етапах дуже дорогі, або неможливі. Наприклад, для створення складних інженерних конструкцій (споруд, літаків, мостів і т.п.). Зміни в проекті фундаменту будинку після того, як покладений дах коштують дуже дорого, тому перфекціонізм на початкових етапах проектування

просто необхідний. Інженери, які починали займатись розробкою програмного забезпечення перейшовши з інших галузей, просто адаптували звичну модель, тому що на ранніх етапах розвитку комп'ютерної техніки не було методологій створених саме для програмування. Проте, схожі методології застосовуються для програмного забезпечення й далі, у випадках коли вимоги фіксовані, і вимагається висока якість та надійність, наприклад в системах для військових чи медичних потреб.

Перший формальний опис водоспадної моделі, після якої вона стала популярною був здійснений В. В. Ройсом у 1970. Попри те, що стаття містить переважно критику методу, на неї часто посилаються.

Переваги методу:

– Ніяких переробок.

– Гарна специфікація перетікає в гарну документацію.

– Зрозуміла модель.

– Розробники можуть мати низьку кваліфікацію.

Недоліки:

– Необхідний перфекціонізм на кожному етапі.

– Важко вносити зміни (якщо взагалі можливо).

– Надлишкове проектування.

– Поділ розробників на "perfect" та "code monkeys".

Модифікації. Через те що цей метод погано підходить для розробки саме ПЗ, частіше використовують його модифікації.

Найвідоміша модифікація – Sashimi. Названа так через японську страву сашимі (суші нарізане і сервіроване так, що складені рядочком шматочки накладаються один на одного). В моделі розробки Сашимі фази життєвого циклу йдуть одна за одною, але при цьому перекриваються одна з одною в часі.

Розглянемо в яку саме БД зберігаються образи облич – MySQL це вільна система керування реляційними базами даних. Розробка та підтримка сайта MySQL здійснює корпорація Oracle, яка отримала права на торговельну марку

разом з поглиненої Sun Microsystems, яка раніше придбала шведську компанію MySQL AB.

Продукт поширюється як під GNU General Public License, так і під власною комерційною ліцензією. Крім цього, розробники створюють функціональність за замовленням ліцензійних користувачів. Саме завдяки такому замовленню майже в найраніших версіях з'явився механізм реплікації.

MySQL є рішенням для малих і середніх додатків. Входить до складу серверів WAMP, AppServ, LAMP і в портативні збірки серверів Денвер, XAMPP, VertrigoServ. Зазвичай MySQL використовується як сервер, до якого звертаються локальні або видалені клієнти, проте в дистрибутив входить бібліотека внутрішнього сервера, що дозволяє включати MySQL в автономні програми.

Гнучкість СКБД MySQL забезпечується підтримкою великої кількості типів таблиць: користувачі можуть вибрати як таблиці типу MyISAM, що підтримують повнотекстовий пошук, так і таблиці InnoDB, що підтримують транзакції на рівні окремих записів. Більш того, СКБД MySQL поставляється із спеціальним типом таблиць EXAMPLE, що демонструє принципи створення нових типів таблиць. Завдяки відкритій архітектурі і GPL-ліцензуванню, в СКБД MySQL постійно з'являються нові типи таблиць.

26 лютого 2008 року Sun Microsystems придбала MySQL AB за 1 млрд доларів, 27 січня 2010 року Oracle придбала Sun Microsystems за 7,4 млрд доларів і включила MySQL в свою лінійку СКБД.

Спільнотою розробників MySQL створені різні відгалуження коду, такі як Drizzle (англ.), OurDelta, Percona Server і MariaDB. Всі ці відгалуження вже існували на момент поглинання компанії Sun корпорацією Oracle.

MySQL має подвійне ліцензування. MySQL може розповсюджуватися відповідно до умов ліцензії GPL. Але за умовами GPL, якщо якась програма використовує бібліотеки MySQL, то вона теж повинна розповсюджуватися за ліцензією GPL. Проте це може розходитися з планами розробників, які не бажають відкривати сирцеві тексти своїх програм. Для таких випадків

передбачена комерційна ліцензія компанії Oracle, яка також забезпечує якісну сервісну підтримку.

В разі використання та розповсюдження програмного забезпечення з іншими вільними ліцензіями, такими як BSD, Apache, MIT та інші, MySQL дозволяє використання бібліотек MySQL за ліцензією GPL.

MySQL виникла як спроба застосувати mSQL до власних розробок компанії: таблиць, для яких використовувалися ISAM – підпрограми низького рівня для індексного доступу до даних. У результаті був вироблений новий SQL-інтерфейс, але API-інтерфейс залишився в спадок від mSQL. Звідки походить назва «MySQL» – достеменно не відомо. Розробники дають два варіанти: або тому, що практично всі напрацювання компанії починалися з префікса My, або на честь дівчинки на ім'я My, дочки Майкла Монті Віденіуса, одного з розробників системи.

Логотип MySQL у вигляді дельфіна носить ім'я «Sakila». Він був обраний з великого списку запропонованих користувачами «імен дельфіна». Ім'я «Sakila» було відправлено Open Source-розробником Ambrose Twebaze.

В січні-лютому 2008 Sun Microsystems придбала розробника системи керування базами данних MySQL за $1 млрд. Після поглинання у 2009 році Sun Microsystems компанією Oracle Corporation MySQL стала власністю Oracle.

За час розвитку під орудою Oracle дедалі більше відокремлює MySQL від спільноти і робить процес розробки все менш прозорим. Наприклад, повернута практика поставки власницьких розширених функцій в Enterprise-версії MySQL, спостерігається приховування інформації про вразливості, зі складу виключений тестовий набір, закритий доступ до більшої частини системи відстеження помилок та припинено публікація згрупованого логу змін, що дозволяє судити про прив'язку патчів до конкретних змін.

Можливості

MySQL – компактний багатопотоковий сервер баз даних. Характеризується високою швидкістю, стійкістю і простотою використання.

MySQL вважається гарним рішенням для малих і середніх застосувань. Сирцеві коди сервера компілюються на багатьох платформах. Найповніше можливості сервера виявляються в UNIX-системах, де є підтримка багатопоточності, що підвищує продуктивність системи в цілому.

Можливості сервера MySQL:

1. Простота у встановленні та використанні.

2. Підтримується необмежена кількість користувачів, що одночасно працюють із БД.

3. Кількість рядків у таблицях може досягати 50 млн.

4. Висока швидкість виконання команд.

5. Наявність простої і ефективної системи безпеки.

4.2 Захист розробленого програмного забезпечення

Захист розробленого програмного забезпечення буде відбуватися за допомогою IDEA – симетричний блоковий алгоритм шифрування даних, запатентований швейцарською фірмою Ascom. Відомий тим, що застосовувався в пакеті програм шифрування PGP. У листопаді 2000 року IDEA був представлений як кандидат у проекті NESSIE в рамках програми Європейської комісії IST (англ. Information Societes Technology, інформаційні громадські технології).

Першу версію алгоритму розробили в 1990 році Лай Сюецзя (Xuejia Lai) і Джеймс Мессі (James Massey) зі Швейцарського інституту ETH Zürich (за контрактом з Hasler Foundation, яка пізніше влилася в Ascom-Tech AG) як заміна DES (англ. Data Encryption Standard, стандарт шифрування даних) і назвали її PES (англ. Proposed Encryption Standard, запропонований стандарт шифрування). Потім, після публікації робіт Біхамом і Шаміра по диференціальному криптоаналізу PES, алгоритм був поліпшений з метою посилення криптостійкості і названий IPES (англ. Improved Proposed Encryption Standard,

покращений запропонований стандарт шифрування). Через рік його перейменували в IDEA (англ. International Data Encryption Algorythm).

Так як IDEA використовує 128-бітний ключ і 64-бітний розмір блоку, відкритий текст розбивається на блоки по 64 біт. Якщо таке розбиття неможливо, останній блок доповнюється різними способами певною послідовністю біт. Для уникнення витоку інформації про кожному окремому блоці використовуються різні режими шифрування. Кожен вихідний незашифрований 64 – біт ний блок ділиться на чотири підблока по 16 біт кожен, так як всі алгебраїчні операції, що використовуються в процесі шифрування, відбуваються над 16-бітними числами. Для шифрування і розшифрування IDEA використовує один і той же алгоритм.

Позначення операцій:

– $\boxplus$ Додавання за модулем 2^{16}.

– $\odot$ Множення за модулем $2^{16}+1$.

– $\oplus$ Побітова виключна диз'юнкція.

Фундаментальним нововведенням в алгоритмі є використання операцій з різних алгебраїчних груп, а саме:

Додавання за модулем 2^{16}.

Множення за модулем $2^{16}+1$.

Побітова виключна диз'юнкція (XOR).

Ці три операції несумісні в тому сенсі, що ніякі дві з них не задовольняють дистрибутивному закону, тобто:

$$a \odot (b \oplus c) \neq (a \odot b) \oplus (a \odot c).$$

Застосування цих трьох операцій ускладнює криптоаналіз IDEA в порівнянні з DES, який базується виключно на операції виключає АБО, а також дозволяє відмовитися від використання S-блоків і таблиць заміни. IDEA є модифікацією мережі Фейстеля.

Генерація ключів

З 128-бітного ключа для кожного з восьми раундів шифрування генерується по шість 16-бітних подключів, а для вихідного перетворення

генерується чотири 16-бітних підключа. Всього буде потрібно 52 = 8 x 6 + 4 різних подключів по 16 біт кожен. Процес генерації п'ятдесяти двох 16-бітних ключів полягає в наступному:

Насамперед, 128-бітний ключ розбивається на вісім 16-бітних блоків. Це будуть перші вісім подключів по 16 біт кожен – $(K_1^{(1)}K_2^{(1)}K_3^{(1)}K_4^{(1)}K_5^{(1)}K_6^{(1)}K_1^{(2)}K_2^{(2)})$

Потім цей 128-бітний ключ циклічно зсувається вліво на 25 позицій, після чого новий 128-бітний блок знову розбивається на вісім 16-бітних блоків. Це вже наступні вісім подключів по 16 біт кожен – $(K_3^{(2)}K_4^{(2)}K_5^{(2)}K_6^{(2)}K_1^{(3)}K_2^{(3)}K_3^{(3)}K_4^{(3)})$

Процедура циклічного зсуву і розбивки на блоки триває до тих пір, поки не будуть згенеровані всі 52 16-бітних підключа.

Шифрування

Структура алгоритму IDEA показана на рисунку 4.3.

Процес шифрування складається з восьми однакових раундів шифрування і одного вихідного перетворення. Вихідний незашифрований текст ділиться на блоки по 64 біта. Кожен такий блок ділиться на чотири підблока по 16 біт кожен. На рисунку ці підблоки позначені D_1, D_2, D_3, D_4. У кожному раунді використовуються свої підключі згідно з таблицею підключів. Над 16-бітними підключами і підблока незашифрованого тексту проводяться наступні операції:

– Множення за модулем $2^{16}+1 = 65537$, причому замість нуля використовується 2^{16}.

– Додавання за модулем 2^{16}.

– Побітове виключне АБО.

В кінці кожного раунду шифрування є чотири 16-бітних підблоки, які потім використовуються як вхідні підблоки для наступного раунду шифрування. Вихідна перетворення являє собою скорочений раунд, а саме, чотири 16-бітних підблоки на виході восьмого раунду і чотири відповідних підключа піддаються операціям:

– Множення за модулем $2^{16}+1$.

– Додавання за модулем 2^{16}.

Після виконання вихідного перетворення конкатенація підблоків D_1', D_2', D_3' і D_4' являє собою зашифрований текст. Потім береться наступний 64-бітний блок незашифрованого тексту і алгоритм шифрування повторюється. Так продовжується до тих пір, поки не зашифрують всі 64-бітові блоки вихідного тексту.

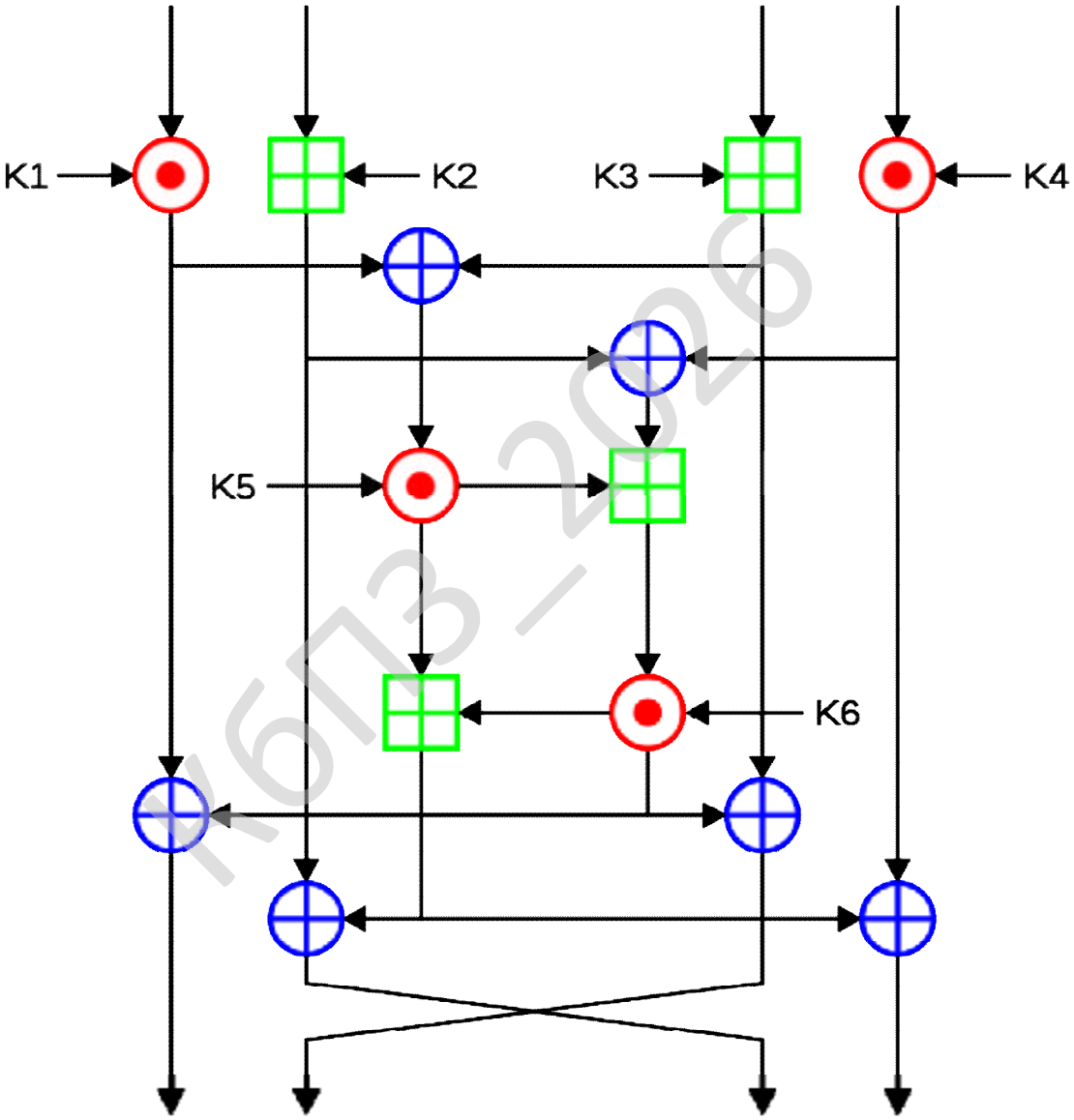

Рисунок 4.3 – Структура алгоритму IDEA

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

На рисунку 5.1 зображено розроблене у бакалаврської дипломної роботі програмне забезпечення системи інтелектуального розпізнання облич при виконанні OSINT-аналізу. З рисунку можна побачити що інтерфейс головного вікна розподілено на наступні функціональні розділи:

– Функціональних кнопок ПЗ.

– Навігаційного меню яке викликається натисканням правої клавіші маніпулятора миші.

– Розділу виведення результату роботи системи – обробка номера.

– Функції представлені у графічному вигляді.

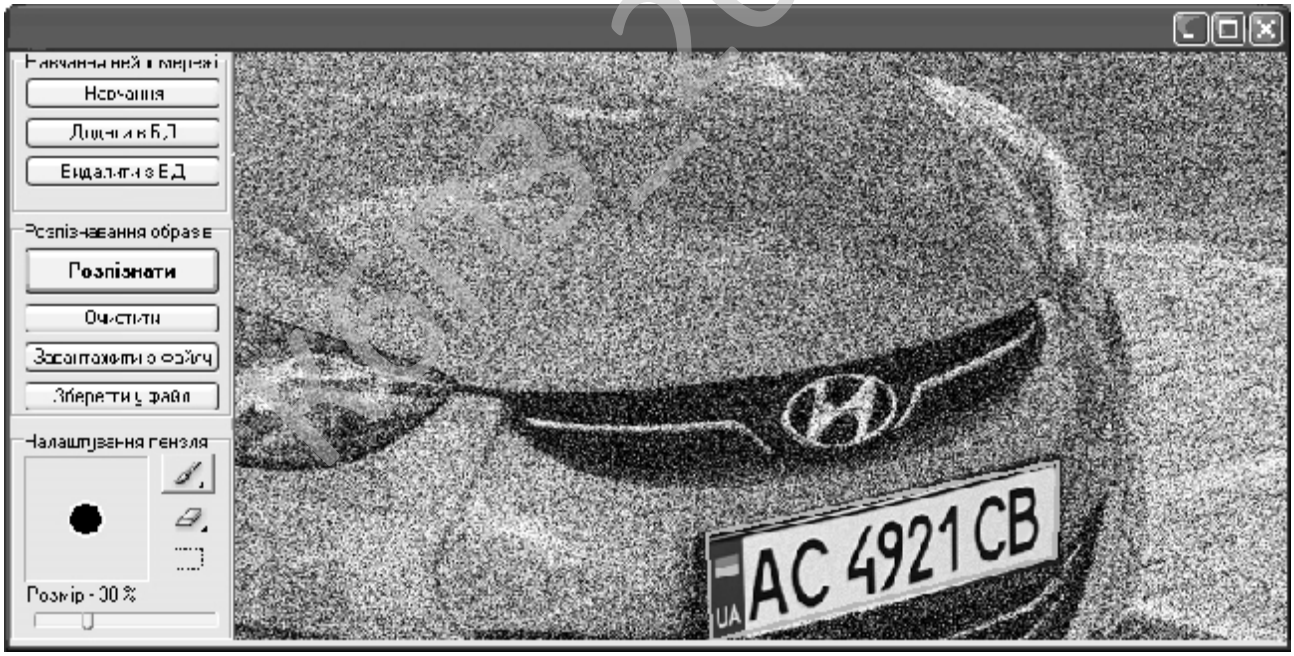

Рисунок 5.1 – Головне вікно ПЗ

Областю застосування є OSINT аналіз. Реалії сучасного світу такі, що наш співрозмовник \ клієнт \ співробітник \ цілком може виявитися зовсім не тим, за

кого він себе видає. Приміром, він може виявитися професійним шахраєм, що прагне заволодіти нашою інформацією, грошима або ж власністю. Як же убезпечити себе від подібних “контактів”? У цьому нам допоможуть всі ті ж реалії сучасного світу, а саме – відкриті джерела інформації.

OSINT – це акронім Open Source Intelligence (англ. – розвідка на основі відкритих джерел), що узвичаївся в епоху Інтернету й простоти доступу до даних.

OSINT використовується як у приватному секторі, так і у військових і розвідувальних службах у плині багатьох років, підходи й джерела інформації були відібрані й упорядковані фахівцями з Ленглі.

Розроблена програма має дуже простий і зрозумілий інтерфейс з користувачем. Кожен, хто в достатньому обсязі володіє операційним середовищем Windows без особливих складностей освоїть і цю програму, оскільки її інтерфейс інтуїтивно зрозумілий. Якщо програма не видала ніяких помилок, і працює, то можна використовувати, інакше слід слідувати інструкціям, які пропонує програма.

На рисунку 5.2 зображено авторські дані розробленого програмного забезпечення.

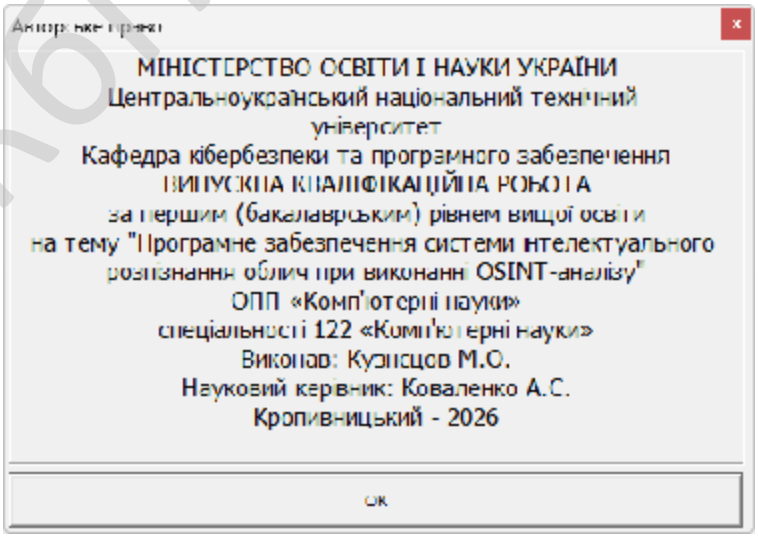

Рисунок 5.2 – Авторське право

Розглянемо процес впровадження програмного забезпечення, це процес налаштування програмного забезпечення під певні умови використання, а також навчання користувачів роботі з програмним продуктом. Впровадження програмного забезпечення це усі дії, що роблять розроблену програмну систему готовою до використання. Даний процес є частинною життєвого циклу програмного забезпечення.

Загалом процес розгортання складається з кількох взаємопов'язаних дій із можливими переходами між ними. Ця активність може відбуватися як з боку виробника так і з боку споживача. Оскільки кожна програмна система є унікальною, то усі процеси та процедури під час розгортання важко передбачити. Тому, "розгортання" можна трактувати як загальний процес відповідно до певних вимог та характеристик. Розгортання може здійснюватись програмістом і в процесі розробки програмного забезпечення.

До діяльностей пов'язаних із розгортанням програмного забезпечення відносять:

– Випуск.

– Встановлення та активація.

– Деактивація.

– Адаптація.

– Обновлення.

– Вмонтування.

– Відстежування версій.

– Видалення.

– Вилучення з обігу.

При впровадженні програмного забезпечення потрібно урахувати наступні дії:

– Виділення критичних, з точки зору загального результату, процедур в діяльності організації. Коли набір таких процедур визначений, необхідно в першу чергу використовувати IT рішення для автоматизації операцій усередині саме цих

процедур. Таким чином, розроблене ІТ рішення автоматично стає життєво важливим і затребуваним для організації, а також буде забезпечена публічність процесу впровадження;

– Розширення нормативної бази організації шляхом включення до неї регламентів, що описують порядок виконання процедур автоматизованих процесів. В іншому випадку є небезпека виникнення неузгодженості між автоматизованими процедурами та іншими процесами організації.

– Виконання робіт з загальної стандартизації існуючої діяльності організації, коли виділяються кращі практики виконання процедур і включаються в ІТ рішення за принципом найбільшої корисності для більшості учасників. Відсоток таких процедур щодо загального обсягу автоматизації може бути невеликий, але це надає процесу побудови рішення вагу в організації за рахунок збільшення його необхідності.

Під час роботи над програмою було проведено тестування програмного забезпечення, тобто технічне дослідження, призначене для виявлення інформації про якість продукту відносно контексту, в якому воно має використовуватись.

Тестування включає як процес пошуку помилок або інших дефектів, так і випробування програмних складових з метою їх оцінки.

Проводилась оцінка:

– відповідності поставленим вимогам;

– правильна відповідь для усіх можливих вхідних даних;

– виконання функцій за прийнятний час;

– практичність;

– сумісність з ОС та стороннім ПЗ.

Оскільки число можливих тестів для програмних компонент практично нескінченне, тому стратегія тестування полягала в тому, щоб провести всі можливі тести з урахуванням наявного часу та ресурсів.

Як результат ПЗ тестувалось стандартним виконанням програми з метою виявлення помилок або інших дефектів.

Проводилось тестування форматом білої скриньки засноване на аналізі керуючої структури програми. Програма вважається повністю перевіреною, якщо проведено вичерпне тестування маршрутів (шляхів) її графа управління.

У цьому випадку формуються тестові варіанти, в яких:

– Гарантується перевірка всіх незалежних маршрутів програми.

– Знаходяться гілки True, False для всіх логічних рішень.

– Виконуються всі цикли (у межах їхніх кордонів та діапазонів).

– Аналізується правильність внутрішніх структур даних.

Недоліки тестування "білої скриньки":

– Кількість незалежних маршрутів може бути дуже велика.

– Повне тестування маршрутів не гарантує відповідності програми вихідним вимогам до неї.

– У програмі можуть бути пропущені деякі маршрути.

– Не можна виявити помилки, поява яких залежить від даних.

Переваги тестування "білої скриньки" пов'язані з тим, що принцип «білої скриньки» дозволяє врахувати особливості програмних помилок:

– Кількість помилок мінімально в «центрі» і максимально на «периферії» програми.

– Попередні припущення про ймовірність потоку керування або даних у програмі часто бувають некоректними. У результаті типовим може стати маршрут, модель обчислень за яким опрацьована слабо.

– При записі алгоритму програмного забезпечення у вигляді тексту на мові програмування можливе внесення типових помилок трансляції (синтаксичних та семантичних).

– Деякі результати в програмі залежать не від вихідних даних, а від внутрішніх станів програми.

Обрано умови розповсюдження – Freeware. Це власницьке програмне забезпечення, котре можна Безоплатно використовувати протягом необмеженого терміну без обмежень у функціональності, і поширюване без сирцевих кодів.

Автори такого програмного забезпечення, як правило, хочуть «дати щось спільноті», але хочуть також контролювати його подальшу розробку. Іноді, коли програмісти вирішують припинити розробку, вони передають сирцевий код іншим програмістам, або ж спільності як вільне програмне забезпечення.

Дуже часто плутають поняття «безплатне програмне забезпечення» та «вільне програмне забезпечення», хоча вони суттєво відрізняються.

Безплатне програмне забезпечення можна безоплатно встановлювати та використовувати (іноді з певними обмеженнями, як, наприклад, «безплатне для домашнього або некомерційного вжитку»), в той час як вільне програмне забезпечення можна продавати за будь-яку суму, але при тому, у користувача, котрий його отримує, повинні бути права на вивчення, модифікацію та поширення сирцевих кодів одержаної програми.

6 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти, призначено для системи інтелектуального розпізнання облич при виконанні OSINT-аналізу.

В межах України в недостатній мірі представлені вітчизняні розробки в цій області.

Рішення завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем інтелектуального розпізнання облич при виконанні OSINT-аналізу.

– Досліджена система інтелектуального розпізнання облич при виконанні OSINT-аналізу.

– На основі отриманих результатів досліджень створена програмна реалізація системи інтелектуального розпізнання облич при виконанні OSINT-аналізу.

Розроблені під час виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання інтелектуального розпізнання облич при виконанні OSINT-аналізу.

Розроблене програмне забезпечення має простий, дружній та зручний інтерфейс користувача, що забезпечує легкість у освоєнні роботи програмного продукту, зручність у використанні, і не потребує особливих спеціальних знань.

При створені програмного забезпечення було використано об’єктно-орієнтований підхід, що відповідає сучасним тенденціям у галузі розробки комерційних програмних систем.

Програма реалізована на мові високого рівня Python. Дана мова програмування дозволяє найбільш ефективно обробляти дані призначені для системи інтелектуального розпізнання облич при виконанні OSINT-аналізу. Це

дозволило мінімізувати строк розробки програмного забезпечення, і, як слід, зменшити витрати на його розробку. Запропоноване програмне забезпечення ділиться на загальне програмне забезпечення, що поставляється із засобами обчислювальної техніки й спеціальне програмне забезпечення, що спеціально розроблене для даної конкретної системи й включає програми, що реалізують її функції.

Програма призначена для виконання під управлінням багатозадачної операційної системи Windows 10/11.

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

Для підвищення рівня безпеки запропоновано застосовувати алгоритм IDEA.

В цілому створене програмне забезпечення підтверджує правильність використаних проектних рішень та повністю відповідає вимогам технічного завдання. Створене програмне забезпечення має потенційну можливість для подальшого вдосконалення і застосування у різних галузях.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Aggarwal C.C. Neural Networks and Deep Learning: A Textbook 1st ed. 2018 Edition. – Springer, 2018. – 520 p

2. Aho A.V., Hopcroft J.E., Ullman J.D. Data Structures and Algorithms. – Pearson, 2001. – 620 c.

3. Brink H., Richards J., Fetherolf M. Real-World Machine Learning. – Manning, 2016. – 474 p.

4. Cormen T.H., Leiserson C.E., Rivest R.L., Stein C. Introduction to Algorithms, 3rd Edition (The MIT Press) 3rd Edition – The MIT Press, 2019. – 1292 p.

5. Fenner M. Machine Learning with Python for Everyone (Addison-Wesley Data & Analytics Series) 1st Edition, Kindle Edition. - Addison-Wesley Professional, 2019. – 586 p.

6. Foreman J.W. Data Smart: Using Data Science to Transform Information into Insight 1st Edition. – Wiley, 2013. – 432 p.

7. Hurbans R. Grokking Artificial Intelligence Algorithms. – Manning, 2020. – 631 p.

8. Gusfield D. Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology 1st Edition. – Cambridge University Press, 2008. – 556 p.

9. Kotu V., Deshpande B. Data Science: Concepts and Practice. – Elsevier Science, 2018. – 953 p.

10. Knowledge Base A Complete Guide - 2021 Edition // The Art of Service - Knowledge Base Publishing, 2020. – 306 p.

11. Knuth D. The Art of Computer Programming, Vol. 1: Fundamental Algorithms, 3rd Edition 3rd Edition. – Addison-Wesley Professional, 2019. – 672 p.

12. Mattmann C. Machine Learning with TensorFlow, Second Edition. – Manning, 2020. – 1124 p.

13. Mueller J.P., Massaron L. Machine Learning For Dummies. – Wiley, 2016. – 714 p.

14. Teofili T. Deep Learning for Search. – Manning, 2019. – 695 p.

15. Rungta K. TensorFlow in 1 Day: Make your own Neural Network. – Publishdrive, 2019. – 587 p.

16. Weidman S. Deep Learning from Scratch: Building with Python from First Principles. – O'Reilly. 2019. – 252 p.

17. Rajasekaran S., Vijayalakshmi Pai G.A. Neural networks, fuzzy logic, and genetic algorithms: synthesis and applications (with cd-rom) Kindle Edition. – PHI, 2013. – 628 p.

18. Вінтенко Б.Ю., Миронець І.В., Смірнов О.А. «Модель комп'ютерно-орієнтованої процедури системи підтримки оперативного персоналу АЕС». *IV Міжнародна науково-практична Інтернет-конференція «Інновації та перспективні шляхи розвитку інформаційних технологій (ІПШРІТ-2025)»* м.Черкаси 25 листопада 2025 року – Черкаси: ЧДТУ.– 2025. – С.101-103.

19. Kuznetsov O., Frontoni E., Kuznetsova Y., Chevardin V., Smirnov O. «Architectural foundations for adaptive security in edge computing systems». *Cybersecurity Defensive Walls in Edge Computing,* 2025. pp. 21-61.

20. Вінтенко, Б.Ю., Миронець, І.В., Смірнов, О.А., Коваленко, О.В., Усік, П.С., Буравченко, К.О., Лисенко, І.А. «Логіко-структурна модель комп'ютерно-орієнтованої процедури системи підтримки оперативного персоналу АЕС». *Кібербезпека: освіта, наука, техніка.* 2025. Том 2 № 30. С. 413-427, 2025.

21. Смірнова, Т.В. «Дослідження методів, моделей та сучасних ІТ-рішень для підтримки технологічних процесів у критичній інфраструктурі держави». *Кібербезпека: освіта, наука, техніка.* 2025. Том 2 № 30. С.195-208, 2025.

22. Усік, П.С., Смірнова, Т.В., Буравченко, К.О., Смірнов, О.А., Улічев, О.С., Смірнов, С.А. «Дослідження технологій забезпечення кібербезпеки

банківських систем з використанням штучного інтелекту». *Кібербезпека: освіта, наука, техніка.* 2025. Том 1 № 29. С.704–716, 2025

23. Kuznetsov, O., Frontoni, E., Kuznetsova, K., Arnesano, M., Smirnov, O. «A secure biometric authentication architecture for blockchain-driven cyber-physical systems». *Security and Privacy of Cyber Physical Systems Emerging Trends Technologies and Applications*, 2025, pp. 193–224.

24. Kuznetsov, O., Smirnov, O., Akhmetov, B., Alimseitova, Z., Imoize, A.L. «Deep Learning Frontiers in Copy-Move Forgery Detection: Advances, Challenges, and Future Directions». *Advancements in Cybersecurity Next Generation Systems and Applications,* 2025. 202-229.

25. Вінтенко Б., Смірнов О., Миронець І., Смірнова Т., Смірнов С. «Імітаційна модель шляхів вхідних даних комп'ютерної інтелектуальної системи підтримки оператора енергоблоку АЕС». *Комбінаторні конфігурації та їхні застосування: Матеріали XXVII Міжнародного науково-практичного семінару, присвяченого 125-річчю Національного університету «Запорізька політехніка»* (Запоріжжя-Кропивницький-Київ, 4-6 червня 2025 р.). Запоріжжя: НУ «Запорізька політехніка», 2025. С.82-91.

26. Al-Azzeh, J., Ayyoub, B., Mesleh, A., Smirnova, T., Gnatyuk, S., Drieiev, O., Smirnov, O., Dorenskyi, O. «Cloud-Based Information System for Evaluating Caverns in the Process of Blasting Metal Surfaces of Details». *International Review on Modelling and Simulations* 18 (1), 2025. pp. 32-42.

27. Вінтенко Б.Ю., Смірнов О.А., Миронець І.В., Смірнова Т.В., Коваленко О.В., Мацуй А.М. «Модель шляхів отримання вхідних даних комп'ютерної інтелектуальної системи підтримки оперативного персоналу АЕС». *Центральноукраїнський науковий вісник. Технічні науки.* 2025. Вип. 11(42), ч. II. С.52-62.

28. Вінтенко Б.Ю., Смірнов О.А., Миронець І.В., Смірнова Т.В. «Методи забезпечення відмовостійкості інтелектуальних систем підтримки оператора». *VIII міжнародна науково-практична конференція "Інформаційна*

безпека та комп'ютерні технології", м. Кропивницький. 24-25 квітня 2025 р. – Кропивницький: ЦНТУ. – 2025. – С. 44-46.

29. Смірнов, О.А., Константинова, Л.В., Конопліцька-Слободенюк, О.К., Козірова, Н.В, Якименко, Н.М., Доренський, О.П., Буравченко, К.О. «Дослідження інструментів штучного інтелекту для роботи з базами даних та аналізу даних». *Кібербезпека: освіта, наука, техніка.* 2025. №3(27), С. 429–448.

30. Lakhno, V., Malyukov, V., Smirnov, O., Bebeshko, B., Chubaievskiy, V., Zhumadilova, M., Malyukova, I., Smirnov, S. «Multifactorial Model for Targeted Attacks Counteracting Within the Framework of a Multi-Step Quality Game with Fuzzy Information». *8th International Symposium on Intelligent Informatics, ISI 2023*, 2025. vol 389. pp 377-389. Springer, Singapore.

31. Smirnov O., Fedorov E., Neskorodieva A., Neskorodieva T. «Intellectual Classification method of Gymnastic Elements Based on Combinations of Descriptive and Generative Approache». *CEUR Workshop Proceedings* Volume 3664, 2024, Pages 11-23.

32. Kuznetsov, O., Kryvinska, N., Ilchenko, O., Smirnova, T., Ulianovska, Y. «Comparative Analysis of Cryptocurrency Trading Platforms Using the Analytic Hierarchy Process». *CEUR Workshop Proceedings*, 2023, 3628, pp. 106-115.

33. Malyukov V., Bebeshko B., Lakhno V., Smirnov O., Malyukova I., Mohylnyi H. «Managing the Purchase-Sale Process of Digital Currencies Under Fuzzy Conditions». *Lecture Notes in Networks and Systems,* 2023, 729 LNNS, pp. 104–112.

34. Al-Mudhafar Aqeel, A.M., Smirnova, T., Buravchenko, K., Smirnov, O. «The method of assessing and improving the user experience of subscribers in software-configured networks based on the use of machine learning». *Advanced Information Systems*, 2023, 7(2), pp. 49-56.

35. Smirnov, O., Sydorenko, V., Aleksander, M., Zhyharevych, O., Yenchev, S. «Simulation of the cloud IoT-based monitoring system for critical infrastructures». *CEUR Workshop Proceedings,* Volume 3530, 2023, pp. 256-265.

36. Аль-Мудхафар Акіл Абдулхуссейн М., Смірнова Т.В., Буравченко К.О., Смірнов О.А. «Метод оцінки та підвищення користувальницького досвіду абонентів в програмно-конфігурованих мережах на основі використання машинного навчання». Сучасні інформаційні системи, 2023, том 7, № 2, С. 49-56.

37. Smirnov, O., Karapetyan, A., Fedorov, E., «Creating Neural Network and Single Solution Human-Based Metaheuristic Methods of Solving the Traveling Salesman Problem». CEUR Workshop Proceedings, Volume 3312, 2022, pp. 47-58.

38. Smirnov, O., Neskorodieva, T., Fedorov, E., Rudakov, K., Neskorodieva, A. «Method Detection Audit Data Anomalies on Basis Restricted Cauchy Machine» CEUR Workshop Proceedings, Volume 3187, 2022, pp. 1-12.

39. Smirnov O., Smirnova T., Anas M. Al-Oraiqat, Drieiev O., Polishchuk L., Sheroz Khan, Yassin M. Y. Hasan, Aladdein M. Amro, Hazim S. AlRawashdeh «Method for Determining Treated Metal Surface Quality Using Computer Vision Technology». Sensors (Basel, Switzerland) Volume 22, Issue 16, 6223, 2022.

40. Smirnov, O., Lakhno, V., Akhmetov, B., Chubaievskyi, V., Khorolska, K., Bebeshko, B. «Selection of a Rational Composition of İnformation Protection Means Using a Genetic Algorithm». In: Rajakumar, G., Du, KL., Vuppalapati, C., Beligiannis, G.N. (eds) Intelligent Communication Technologies and Virtual Mobile Networks. Lecture Notes on Data Engineering and Communications Technologies, vol 131. 2023. Springer, Singapore. pp. 21-34.

41. Kuznetsov, A., Oleshko, I., Chernov, K., Bagmut, M., Smirnova, T. «Biometric authentication using convolutional neural networks». Lecture Notes in Networks and Systems. Volume 152, 2021, Pages 85-98.

42. Smirnov O., Kuznetsov A., Kryvinska N., Kiian A., Kuznetsova K. «Full Non-Binary Constant-Weight Codes». SN Computer Science, Vol 2, 337, 2021. https://doi.org/10.1007/s42979-021-00739-w.

43. Smirnov O., Neskorodieva T., Fedorov E., Rymar P. «Neural Network Modeling Method of Transformations Data of Audit Production with Returnable Waste». CEUR Workshop Proceedings Volume 3101, 2021, Pages 192-207.

44. Smirnov, O., Kuznetsov, A., Potii, O., Poluyanenko, N., Stelnyk, I., Mialkovsky, D. «Combining and filtering functions in the framework of nonlinear-feedback shift register». International Journal of Computing; 2020, Volume 19, Issue 2 – Research Institute for Intelligent Computer Systems – 2020. – P. 247-256.

45. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova T. «Non-binary constant weight coding technique». CEUR Workshop Proceedings. Volume 2740, 2020, Pages 102-114.

46. Smirnov O., Kuznetsov A., Kiian A., Cherep A., Kanabekova M., Chepurko I. «Testing of code-based pseudorandom number generators for post-quantum application». 2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT), Ukraine, Kyiv, May 14-18. 2020. P. 172-177.

47. Smirnov O., Kuznetsov A., Pushkar'ov A., Serhiienko R., Babenko V., Kuznetsova T., «Representation of Cascade Codes in the Frequency Domain». In: Radivilova T., Ageyev D., Kryvinska N. (eds) Data-Centric Business and Applications. Lecture Notes on Data Engineering and Communications Technologies, vol 48. Springer, Cham. 2021. pp 557-587.

48. Smirnov, O., Drieieva, H., Drieiev, O., Polishchuk, Y., Brzhanov, R., Aleksander, M. «Method of fractal traffic generation by a model of generator on the graph». CEUR Workshop Proceedings Volume 2616, 2020, Pages 366-379.

49. Smirnov, O., Drieieva, H., Drieiev, O., Simakhin, V., Bondar, S., Odarchenko, R. «Managing multifractal properties of the binary sequence generated with the Markov chains», CEUR Workshop Proceedings Volume 2608, 2020, Pages 633-645.

50. Smirnov O. Kuznetsov A., Zaicenko Yu., Pastukhov M., Oleshko O., Kuznetsova K., «Formation of Discrete Signals with Special Correlation Properties». International Conference on Information and Telecommunication Technologies and Radio Electronics, UkrMiCo 2019; Odessa; Ukraine; 9-13 September 2019. P.22-28.