Хмельницький національний університет
Факультет інформаційних технологій
Кафедра кібербезпеки

КВАЛІФІКАЦІЙНА РОБОТА

бакалавр
Освітній рівень

Система запобігання витоку даних на основі штучного інтелекту
Назва теми

КРКБ 190115.19.01.12 ПЗ
Шифр

Галузь знань 12 «Інформаційні технології»
Шифр, назва

Спеціальність 125 «Кібербезпека»
Шифр, назва

Освітня програма «Кібербезпека»
Назва

Виконав: студент IV курсу, група КБ-19-1 _____ О.В. Яворський
Підпис Ініціали, прізвище

Керівник _____ 6.06.2023 _____ Н.С. Петляк
Підпис, дата Ініціали, прізвище

Нормоконтролер _____ 06.06.23р. _____ С.В. Мостовий
Підпис, дата Ініціали, прізвище

До захисту допускаю:
Зав. кафедри кібербезпеки _____ Ю.П. Кльоц
Підпис Ініціали, прізвище

«6» 06 2023 р.

Хмельницький 2023

Форма	Зона	Позиц	Позначення	Найменування	Кільк.	Прим.
А4		1	КРКБ.190115.19.01.12 ПЗ	Система запобігання	63	
				витоку даних		
				на основі		
				штучного інтелекту		
				Пояснювальна записка		
А4		2	КРКБ. 190115.19.01.12 Е8	Віртуалізація моделі	1	
				штучного інтелекту		
А4		3	КРКБ. 190115.19.01.12 Е8	Схема ініціалізації	1	
				та навчання моделі		
				та токенайзера		
А4		4	КРКБ. 190115.19.01.12 Е8	Алгоритм роботи системи	1	

Зм.	Арк.	№ Докум.	Підп.	Дата	КРКБ.190115.19.01.12 ВП		
Розробив		Яворський О.В.		6.06.23	Система запобігання витоку даних на основі штучного інтелекту Відомість проекту	Літера	Аркуш	Аркушів
Перев.		Петляк Н.С.		6.06.23		н	1	1
Н. контр.		Мостовий С.В.		6.06.23	ХНУ, КБ-19-1			
Затв.		Кльоц Ю.П.		6.06.23				

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра КІБЕРБЕЗПЕКИ

Освітній рівень БАКАЛАВР

Галузь знань 12 ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

Спеціальність 125 КІБЕРБЕЗПЕКА

Освітня програма ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА ПІДГОТОВКИ БАКАЛАВРІВ

ЗАТВЕРДЖУЮ

Зав. кафедри Ю.П. Кльоц

"1" 03 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Яворський О.В

Прізвище, ім'я, по батькові студента

1. Тема роботи Система запобігання витоку даних на основі штучного інтелекту

Керівник роботи Петляк Н.С.

Прізвище, ім'я, по батькові, науковий ступінь, вчене звання

Затверджено наказом ректора університету від 01 березня 2023 р. №5

2. Строк подання студентом роботи на кафедру _____

3. Вихідні дані до проекту (роботи) створити систему для запобігання витоку даних на основі штучного інтелекту. Вибрати середовище розробки та бібліотеку штучного інтелекту, зокрема мову програмування. Визначити об'єкт захисту та дослідити його роботу. Дослідити можливі варіації структури системи .Дослідити датасети та їх види. Згенерувати датасет для навчання моделі на основі даних, які генерує об'єкт захисту. Сформувати вибірки з датасету. Дослідити моделі штучного інтелекту. Впровадити функціонал для форматування вхідних даних в зручний формат для моделі штучного інтелекту. Навчити модель, використовуючи згенерований датасет, та зберегти її в файловій системі для подальшого використання в системі. Протестувати навчену модель. Реалізувати функціонал для безперебійної роботи системи в режимі реального часу. Розробити функціонал, який буде сповіщати адміністратора системи про можливий витік даних, використовуючи електронне листування. Провести розрахунок ефективності роботи системи.

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити) Вступ. Аналіз існуючих підходів, визначення контексту та мети проекту. Деталізація умов використання та вибір засобів реалізації визначених задач. Результати та ефективність програми. Висновки.

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслень): «Віртуалізація моделі штучного інтелекту», «Схема ініціалізації та навчання моделі та токенайзера». «Алгоритм роботи системи»

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 1.03.2023 р.

КАЛЕНДАРНИЙ ПЛАН

№з/п	Назва етапів (розділів) кваліфікаційної роботи	Термін виконання етапів проекту (роботи)	Примітка
1	Вибір і затвердження теми кваліфікаційної роботи	Січень	–
2	Пошук теоретичної інформації про датасети та моделі штучного інтелекту	Січень	–
3	Дослідження існуючих рішень	Лютий	–
4	Постановка задачі	Лютий	–
5	Розробка датасету та навчання моделі	Березень	–
6	Побудова структури системи	Квітень	–
7	Розробка функціоналу для безперебійної роботи системи в режимі реального часу	Квітень\Травень	–
8	Оформлення пояснювальної записки згідно вимог	Травень	–
9	Оформлення графічної частини	Червень	–

Студент _____ Підпис Мордовський О.В. Ініціали, прізвище

Керівник проекту (роботи) _____ Підпис Петляк Н.С. Ініціали, прізвище

АНОТАЦІЯ

Тема кваліфікаційної роботи: «Система запобігання витоку даних на основі штучного інтелекту».

Автор роботи: Яворський Олександр Віталійович.

Керівник роботи: Петляк Наталія Сергіївна.

Пояснювальна записка: 63 с., 2 додатки, 23 рис., 41 джерело.

Графічна частина: 8 презентаційних слайдів.

СИСТЕМА ЗАПОБІГАННЯ ВИТОКУ ДАНИХ НА ОСНОВІ ШТУЧНОГО ІНТЕЛЕКТУ, ШТУЧНИЙ ІНТЕЛЕКТ, СУБД MySQL,

Метою роботи є розробка системи запобігання витоку даних на основі штучного інтелекту, яка дозволить підвищити рівень ефективності захисту даних в СУБД MySQL.

У роботі було досліджено і проаналізовано предметну область, існуючі способи захисту на основі штучного інтелекту, теоретичну інформацію про датасети та моделі та їх створення. Створено і розроблену таку систему, яка дозволяє виявляти витоки даних в СУБД MySQL, використовуючи спроможність штучного інтелекту до навчання та передбачення, а також повідомляти адміністратора через електронний лист про витік даних.

6.06.2023

ANNOTATION

Course project: Data leakage prevention system based on artificial intelligence.

Author of the work: Yavorskyi O. V.

Supervisor: Petliak N. S.

Amount - 63 pages, 2 application, 23 figures, 41 sources.

Graphic part: 8 presentation slides.

DATA LEAKAGE PREVENTION SYSTEM BASED ON ARTIFICIAL INTELLIGENCE

The purpose of the work is to develop a data leakage prevention system based on artificial intelligence, which will allow to increase the level of data protection efficiency in the MySQL DBMS.

This work investigated and analyzed the subject area, existing methods of protection based on artificial intelligence, theoretical information about datasets and models and their creation, as well as created. Developed such a system that allows detecting data leaks in the MySQL DBMS, using the ability of artificial intelligence to learn and predict, and notify the administrator via email of a data leak.

6.06.2023

ЗМІСТ

ВСТУП

У сучасній цифровій епосі витоки даних та кібератаки стають все більш поширеними і частішими явищами. Загрози кібербезпеці викликають зростаюче занепокоєння серед окремих осіб, компаній і урядів країн, оскільки наслідки витоку даних можуть бути нищівними. Хакери та кіберзлочинці постійно знаходять нові способи використання вразливостей комп'ютерних систем та крадіжки конфіденційної інформації, такої як особисті дані, фінансові документи та інтелектуальна власність.

Для протидії цим загрозам були розроблені різноманітні традиційні заходи безпеки, такі як брандмауери, програми шифрування та контролю доступу. Незважаючи на те, що ці заходи мають певну ефективність, вони мають свої обмеження та можуть бути недостатніми перед загрозами кібербезпеки, що постійно розвиваються. Система безпеки є складною структурою, яка об'єднує різні підпрограми та засоби, що працюють у взаємодії один з одним.

Штучний інтелект (ШІ) відкриває нові перспективи у сфері кібербезпеки, здатний змінити її парадигму. З використанням алгоритмів машинного навчання, ШІ може аналізувати великі обсяги даних і виявляти шаблони, які можуть уникнути уваги людських аналітиків. Це дозволяє виявляти та запобігати кібератакам задовго до їх виникнення, а також забезпечує точну та швидку реакцію на інциденти безпеки. Використання ШІ в кібербезпеці має потенціал забезпечити високу ефективність та надійність захисту від постійно зростаючих загроз.

Метою цієї кваліфікаційної роботи є розробка системи запобігання витоку даних з використанням ШІ. Основний акцент системи буде зроблено на захисті бази даних MySQL, яка є популярною відкритою системою управління реляційною базою даних. Розробка цієї системи передбачає збір та аналіз наборів даних, пов'язаних з вразливостями та атаками на MySQL, а також використання моделей машинного навчання для виявлення та запобігання потенційним витокам даних.

Основною одиницею даних для роботи є логи MySQL. Логи MySQL надають цінну інформацію про запити до бази даних, час виконання, а також можуть виявити аномальну активність та потенційні вразливості.

Ця робота базується на ідеї використання сучасних інструментів розробки ШІ для аналізу текстових даних, зокрема логів, з метою покращення загальної безпеки системи. Однією з головних можливостей цієї логіки є сповіщення адміністратора про неправомірне отримання даних з бази даних MySQL.

Проект відображає потенціал ШІ в розвитку кібербезпеки та забезпечує ефективний аналіз логів в базах даних MySQL. Він може стати цінним рішенням для бізнесів та державних установ, які використовують цю реляційну базу даних та приділяють велику увагу безпеці своїх даних і даних своїх клієнтів. Проект дозволяє адміністраторам та правоохоронним органам своєчасно реагувати на інциденти кіберпорушень.

Варто зауважити, що цей проект не претендує на вирішення всіх проблем безпеки системи. Він фокусується на конкретній задачі – виявленні витоку даних шляхом аналізу логів MySQL за допомогою ШІ. Щоб досягти максимальної ефективності та покрити ширший спектр проблем та вразливостей системи, рекомендується комбінувати цей проект з іншими інструментами та підходами до кібербезпеки. Таке поєднання допоможе створити комплексний захист і забезпечити безпеку системи на різних рівнях.

Усе це робить цей проект важливим кроком у напрямку покращення безпеки баз даних та захисту важливої інформації. Враховуючи швидкий розвиток кіберзагроз та постійну необхідність усунення потенційних вразливостей, інтеграція ШІ в системи безпеки стає надзвичайно актуальною та перспективною.

1 АНАЛІЗ ІСНУЮЧИХ ПІДХОДІВ, ВИЗНАЧЕННЯ КОНТЕКСТУ ТА МЕТИ ПРОЕКТУ

1.1 Класифікація та захист від витоків даних

Витік даних – це подія, під час якої чутлива або конфіденційна інформація ненавмисно або навмисно розкривається неавторизованим особам або організаціям [1]. Це може статися різними способами, наприклад хакерством, соціальною інженерією, внутрішніми загрозами або ненавмисними помилками. Витік даних може призвести до крадіжки особистих даних, фінансових втрат, шкоди репутації, правових наслідків і ризиків для національної безпеки. Запобігання витоку даних вимагає впровадження відповідних заходів безпеки, таких як контроль доступу, шифрування та навчання співробітників, а також впровадження систем моніторингу та реагування для швидкого виявлення та пом’якшення будь-яких порушень. Витік може статися різними способами, наприклад через злом, фішинг, соціальну інженерію або внутрішні загрози.

Витік даних може бути надзвичайно шкідливим для окремих осіб, організацій і суспільства в цілому [2]. Наслідки витоку даних можуть бути серйозними та далекосяжними, впливаючи як на окремих осіб, так і на організації. Ось деякі можливі наслідки витоку даних:

– витік даних може розкрити інформацію, таку як імена, адреси та номери соціального страхування, які можуть використовуватися злочинцями для крадіжки особистих даних;

– витік даних може призвести до фінансових втрат для окремих осіб і організацій. Наприклад, викрадені номери кредитних карток можна використовувати для здійснення несанкціонованих покупок або зняття грошей з банківських рахунків;

– витік даних може завдати шкоди репутації організацій і окремих осіб. Наприклад, компанія, яка зазнає витоку даних, може вважатися ненадійною або некомпетентною, що може призвести до втрати клієнтів і доходу;

– витік даних може мати правові наслідки. Наприклад, компанії, які не захищають дані клієнтів, можуть зіткнутися зі штрафами, судовими позовами чи іншими судовими діями;

– витік даних може поставити під загрозу національну безпеку. Наприклад, може статися витік конфіденційної інформації про військові операції чи збір розвідданих, що потенційно може поставити під загрозу життя.

Загалом витік даних може мати серйозні наслідки для окремих осіб, організацій і суспільства в цілому. Важливо вжити заходів, щоб запобігти витокам даних і швидко реагувати, якщо вони все-таки стаються.

Існує кілька типів витоку даних, які можна класифікувати на основі джерела витоку або типу даних. Ось кілька типових прикладів витоку даних:

– зовнішні атаки – цей тип витоку даних відбувається, коли хакери або кіберзлочинці отримують несанкціонований доступ до комп'ютерних систем або мереж і викрадають конфіденційні дані [3]. Це можна зробити за допомогою різних засобів, наприклад використання вразливостей у програмному забезпеченні або використання методів соціальної інженерії, щоб обманом змусити користувачів надати свої облікові дані для входу;

– інсайдерські загрози – цей тип витоку даних відбувається, коли співробітники, підрядники чи інші інсайдери навмисно чи ненавмисно розкривають конфіденційні дані [4]. Наприклад, працівник може випадково надіслати електронний лист із конфіденційною інформацією не тому одержувачу або навмисно викрасти дані для особистої вигоди;

– фізичні порушення – цей тип витоку даних відбувається, коли фізичні активи, такі як ноутбуки, смартфони або USB-накопичувачі, втрачені або викрадені, а конфіденційні дані, що зберігаються на них, скомпрометовані [5];

– зараження зловмисним програмним забезпеченням – цей тип витоку даних відбувається, коли зловмисне програмне забезпечення, наприклад віруси або програми-вимагачі, заражають комп'ютерну систему та викрадають або шифрують конфіденційні дані;

– витік даних у соціальних мережах – цей тип витоку даних відбувається, коли окремі особи чи організації ненавмисно публікують конфіденційну інформацію на платформах соціальних мереж, наприклад, публікуючи фотографії конфіденційних документів або надсилаючи особисту інформацію, яка може бути використана для викрадення особи.

Кожен тип витоку даних створює власний унікальний набір проблем і вимагає різних стратегій запобігання та пом'якшення. Запобігання витоку даних вимагає впровадження відповідних заходів безпеки, таких як контроль доступу, шифрування та навчання співробітників, а також впровадження систем моніторингу та реагування для швидкого виявлення та пом'якшення будь-яких порушень.

Захист даних є однією з найважливіших проблем в сучасному цифровому світі. Витік конфіденційної інформації може мати серйозні наслідки для організацій та осіб, порушуючи приватність, викликаючи фінансові втрати та шкоду репутації. Щоб запобігти витоку даних, необхідно використовувати різноманітні системи та підходи, спрямовані на забезпечення безпеки та конфіденційності. В цьому списку представлені деякі ключові системи та підходи, які використовуються для ефективного запобігання витоку даних. Застосування цих методів допоможе організаціям забезпечити захист своїх цінних даних та зберегти довіру своїх клієнтів і партнерів.

Існує кілька систем і підходів, серед який:

– системи управління доступом (Access Management Systems) контролюють доступ до конфіденційних даних та регулюють права користувачів на їх перегляд, редагування та поширення [6]. Система керування доступом може допомогти захистити організації від втрати даних та порушень безпеки;

– використання криптографічних алгоритмів і протоколів допомагає захистити дані від несанкціонованого доступу та перехоплення. Шифрування, хешування та цифрові підписи – це деякі з криптографічних методів, які забезпечують конфіденційність та цілісність даних;

– моніторинг та аудит безпеки (Security Monitoring and Auditing) відстежують активність в мережі та системах, виявляють підозрілі дії [7]. Також збирають журнали подій для подальшого аналізу та розслідування випадків витоку даних;

– системи виявлення вторгнень (Intrusion Detection Systems) моніторять мережевий трафік та комп'ютерні системи з метою виявлення аномальної або шкідливої активності [8]. Ці дії можуть свідчити про спробу витоку даних;

– управління правами доступу (Rights Management) включає в себе встановлення політик та правил для керування доступом до даних [9]. Також, контроль над можливістю копіювання, друку, редагування та поширення;

– системи контролю руху даних (Data Loss Prevention Systems) виявляють та запобігають ненавмисному або несанкціонованому виходу конфіденційних даних за межі організації [10]. Надають можливість контролювати та блокувати небажаний рух даних;

– одним з важливих аспектів запобігання витоку даних є навчання співробітників про правила безпеки, обізнаність стосовно потенційних загроз та навички впізнавання підозрілої активності або соціальної інженерії.

Впроваджуючи системи захисту потрібно усвідомлювати, що це завжди компроміс між надійним захистом та швидкою простотою. Кожна система чи підхід мають свої переваги та недоліки в експлуатації.

Криптографічні методи забезпечують високий рівень захисту даних, особливо при використанні сильних алгоритмів шифрування. Ключі шифрування дозволяють зберігати дані у безпечному стані. Проте вимагають значних обчислювальних ресурсів для шифрування та розшифрування даних. Можуть бути складні у реалізації та управлінні ключами шифрування. Не вирішують проблеми внутрішнього злочинного доступу до даних.

Аутентифікація та авторизація дозволяють контролювати доступ до даних і ресурсів, що знижує ризик витоку даних. Дозволяють ідентифікувати користувачів та надавати їм відповідні права доступу [11]. Але у той же час можливий підбір паролів або використання слабких аутентифікаційних методів

можуть зробити систему вразливою. Не вирішують проблему внутрішнього несанкціонованого доступу до даних.

Моніторинг та аудит дозволяють виявляти підозрілу або небажану активність, спостерігати за змінами в системі та вчасно реагувати на потенційні загрози безпеки. Не зважаючи на це вони вимагають значних обчислювальних ресурсів та великого обсягу даних для ефективного моніторингу. Можуть виникати помилкові спрацьовування або пропуски у виявленні загроз.

Контроль доступу забезпечує гранульований доступ до даних, що знижує ризик витоку. Дозволяє налаштовувати права доступу для різних користувачів та ролей. Однак, вимагають належного управління правами доступу та регулярного оновлення політик доступу. Можуть бути обхідні шляхи або помилки в конфігурації, які можуть призвести до недостатнього контролю доступу.

Останні великі витоки даних стали серйозними проблемами в сфері кібербезпеки та приватності. Декілька випадків, таких як Facebook-Cambridge Analytica, Marriott International, Capital One і Equifax, набули великої популярності в масмедіа через масштабність витоку та його наслідки.

Ці витоки даних призвели до втрати мільйонів особистих даних користувачів, таких як імена, адреси, соціальні статуси, фінансові дані та інша конфіденційна інформація. Це порушило проблему приватності і викликало серйозне обурення з боку громадськості.

Наслідками цих витоків стали:

– скарги користувачів щодо використання їхньої особистої інформації без їхнього належного дозволу. Це підірвало довіру до організацій, які збирають та обробляють ці дані;

– потрапляння особистих даних користувачів у руки зловмисників, що може спонукати до загрози крадіжки ідентичності, шахрайства та інших видів кіберзлочинності. Це може призвести до серйозних фінансових втрат та негативного впливу на життя постраждалих осіб;

– організації, які були втягнуті в витоки даних, стикаються зі значними наслідками для свого бренду та репутації. Вони можуть втратити довіру клієнтів і партнерів, а також піддаватися правовим наслідкам та санкціям;

– після великих витоків даних зріс тиск на владу із боку громадськості для посилення захисту особистих даних та введення жорсткого законодавства щодо обробки і зберігання даних. Це може призвести до змін у вимогах до організацій та більшої відповідальності за збереження конфіденційності даних.

Великі витоки даних, які сталися недавно, нагадують про критичну потребу у забезпеченні безпеки та захисту особистих даних. Ці події показують, що навіть великі організації можуть бути вразливими перед атаками та витоками інформації, що може призвести до серйозних наслідків.

Безпека даних повинна бути на першому місці для всіх організацій та користувачів. Важливо вдосконалювати заходи безпеки, використовувати найкращі практики, впроваджувати сучасні технології шифрування та механізми контролю доступу. Організації повинні також інвестувати в навчання персонал, щоб уникнути вразливостей та зловживань.

1.2 Існуючі рішення штучного інтелекту для захисту інформації

Наразі, ШІ виявився вирішальною технологією не тільки для технологічних компаній. Його арсенал дозволяє суттєво пришвидшувати та покращувати роботу будь-якої системи. Особливо це важливо для компаній, які ведуть свою діяльність в кібербезпеці. Системи безпеки, які використовують ШІ, можуть виявляти загрози та реагувати на них у реальному часі, з меншим задіянням людей в процесі.

ШІ використовується вже в багатьох сферах життя. Ось декілька прикладів його використання:

– персональні голосові помічники, такі як Siri від Apple, Alexa від Amazon і Google Assistant, використовують ШІ і обробку природної мови, щоб розуміти

голосові команди та відповідати на них [12]. У багатьох випадках користувачі можуть задавати своїм віртуальним асистентам питання, керувати пристроями домашньої автоматизації та відтворенням медіа, а також виконувати інші базові завдання, такі як електронна пошта, список справ і календар – все це за допомогою усних команд;

– ШІ використовується в технології розпізнавання зображень, що дозволяє машинам ідентифікувати та класифікувати об’єкти на зображеннях [13]. Наприклад, ця технологія використовується в безпілотних автомобілях, щоб допомогти їм «бачити» та уникати перешкод;

– використання ШІ для виявлення та запобігання шахрайству шляхом аналізу шаблонів у даних та виявлення аномалій. Це зазвичай використовується у фінансовій та банківській галузях для виявлення шахрайських операцій;

– ШІ можна використовувати для аналізу медичних зображень, таких як рентгенівські знімки та МРТ, щоб виявити потенційні проблеми зі здоров’ям [14]. Крім того, чат–боти на основі ШІ можуть надавати пацієнтам медичну інформацію та поради;

– системи рекомендацій на основі ШІ використовуються такими компаніями, як Amazon і Netflix, щоб пропонувати клієнтам продукти та контент на основі їх минулої поведінки;

– ШІ можна використовувати для виявлення та запобігання кібератакам шляхом аналізу моделей і виявлення аномалій у мережевому трафіку;

– ШІ використовується в обробці природної мови для розуміння та аналізу людської мови, що дозволяє машинам ефективніше спілкуватися з людьми [15]. Ця технологія використовується в чат–ботах, віртуальних помічниках і програмному забезпеченні перекладу.

Це лише кілька прикладів багатьох реальних застосувань ШІ.

Використання ШІ великими компаніями з кібербезпеки дозволяє покращити загальну роботу їхніх продуктів. Технологія ШІ допомагає ідентифікувати та виявляти загрози безпеці швидше й точніше, ніж традиційні засоби безпеки. Аналізуючи величезні обсяги даних у режимі реального часу, рішення з

кібербезпеки на основі ШІ можуть швидко виявляти потенційні загрози та реагувати на них.

Також, алгоритми ШІ можуть аналізувати шаблони та поведінку, які можуть свідчити про атаку, забезпечуючи більш повний аналіз потенційних загроз. Автоматизація процесів безпеки, таких як виявлення загроз і реагування на них, може зменшити потребу в людському втручанні, що потенційно зменшить витрати компанії.

Можливості ШІ дозволяють обробляти великі обсяги даних, уможливлюючи роботу з величезними об'ємами інформації, якими повинні керувати сучасні системи кібербезпеки в реальному часі.

Ще одним плюсом ШІ є те, що він може навчатися та адаптуватися до нових загроз з часом, що робить його більш ефективними у виявленні та запобіганні атакам.

Існує багато програмного забезпечення та інструментів кібербезпеки, які використовують ШІ для покращення своїх можливостей. Ось кілька прикладів:

– Darktrace – це платформа кібербезпеки на основі ШІ, яка використовує машинне навчання для виявлення кіберзагроз і реагування на них у реальному часі [16]. Платформа аналізує мережевий трафік і поведінку користувачів, щоб виявити потенційні загрози та вжити заходів для їх запобігання;

– Cylance – це антивірусне програмне забезпечення на основі ШІ, яке використовує машинне навчання для виявлення та запобігання зловмисному програмному забезпеченню. Програмне забезпечення аналізує файли та код, щоб виявити потенційні загрози та зупинити їх, перш ніж вони можуть завдати шкоди;

– IBM Watson використовує ШІ, щоб допомогти аналітикам безпеки аналізувати кіберзагрози та реагувати на них [17]. Платформа використовує обробку природної мови та машинне навчання для аналізу величезних обсягів даних безпеки та виявлення потенційних загроз;

– McAfee MVISION – це хмарна платформа безпеки, яка використовує ШІ для виявлення та запобігання кіберзагрозам. Платформа використовує машинне

навчання для аналізу мережевого трафіку та поведінки користувачів і виявлення потенційних загроз;

– FireEye – це платформа кібербезпеки, яка використовує ШІ для виявлення та реагування на передові кіберзагрози [18]. Платформа використовує машинне навчання для визначення потенційних загроз і вживає заходів для їх запобігання.

Загалом, використання ШІ в кібербезпеці стає все більш важливим, оскільки кількість і складність кіберзагроз продовжує зростати. Великі компанії з кібербезпеки визнають потребу в передових технологічних рішеннях для захисту своїх клієнтів і випередження загроз, що розвиваються.

ШІ має деякі переваги порівняно з традиційними рішеннями безпеки, коли йдеться про запобігання витоку даних. Ось кілька причин чому:

– виявлення шаблонів та аномалій, які можуть бути пропущені традиційними заходами безпеки. Алгоритми ШІ навчені визначати шаблони та поведінку в даних, які можуть вказувати на потенційне порушення даних. Це дозволяє системі виявляти загрози, які можуть бути неочевидними для аналітика або традиційної системи безпеки;

– адаптація до нових загроз і мінливого середовища. На відміну від традиційних заходів безпеки, які можуть бути розроблені для боротьби з певними типами загроз або атак, ШІ може навчатися та адаптуватися до нових загроз і мінливого середовища. Це дозволяє системі бути більш ефективним у запобіганні витокам даних, навіть якщо ландшафт загроз змінюється з часом;

– ШІ може швидко й точно аналізувати великі обсяги даних: із зростаючим обсягом і складністю даних аналітикам може бути важко своєчасно й точно визначати потенційні загрози. ШІ може швидко й точно аналізувати великі обсяги даних, дозволяючи командам безпеки реагувати ефективніше;

– традиційні заходи безпеки можуть генерувати велику кількість помилкових спрацьовувань, розслідування яких може зайняти багато часу та може відволікати ресурси від більш серйозних загроз. ШІ може допомогти зменшити кількість помилкових спрацьовувань, визначаючи шаблони та поведінку, які, швидше за все, пов’язані з реальною загрозою.

Однак, важливо зазначити, що успішна взаємодія з ШІ в контексті захисту інформації вимагає не тільки потужних алгоритмів і моделей, але і добре налаштованої системної інфраструктури, яка забезпечує доступ до актуальних даних та ресурсів для навчання і оновлення моделей.

Крім того, важливо забезпечити етичне та відповідальне використання ШІ в контексті захисту інформації. Це означає розробку та застосування алгоритмів, які дотримуються принципів приватності, прозорості та справедливості. Потрібно також враховувати можливі наслідки ШІ, зокрема в контексті порушення приватності користувачів або неправильного прийняття рішень на основі алгоритмів.

Взаємодія з ШІ в контексті захисту інформації відкриває нові можливості для ефективного виявлення, захисту та реагування на загрози безпеці даних. Продовження досліджень і розвиток цих технологій важливі для підвищення рівня безпеки в цифровому світі.

Майбутнє кібербезпеки з ШІ обіцяє бути захоплюючим і перспективним. Використання ШІ в цій сфері вже сьогодні демонструє значний потенціал, але його розвиток ще лише починається.

Одним з головних напрямків є виявлення загроз та вразливостей, де ШІ використовується для аналізу великих обсягів даних та ідентифікації потенційних загроз комп'ютерним системам.

Далі, ШІ допомагає прогнозувати поведінку зловмисників, аналізуючи дані про попередні кібератаки. Його застосування також спрямоване на автоматизацію відповіді на загрози, де він може надавати рекомендації щодо заходів безпеки, блокувати підозрілі активності та автоматично відновлювати системи після атак. Крім того, розвиток розумних систем захисту, які використовують ШІ, дозволяє адаптуватися до змінних умов та виявляти аномальну активність, що забезпечує більш ефективний та персоналізований рівень захисту.

Нарешті, ШІ може сприяти полегшенню процесу відновлення після кібератаки шляхом автоматичного виявлення пошкоджень і відновлення даних та

роботи системи. Загалом, використання ШІ в кібербезпеці допоможе забезпечити більш швидку, ефективну та надійну захисту інформації.

1.3 Опис датасету

Для розробки та оцінки ефективності зазначених систем і підходів, використання підходящого датасету є ключовим.

Датасет – це набір даних, які використовуються для тренування моделей машинного навчання та аналізу даних. В датасеті можуть міститися дані з різних джерел, такі як текстові файли, зображення, відео та звукові записи [19].

Інтернет є джерелом великої кількості безкоштовних датасетів, доступних для використання в різних проектах машинного навчання та досліджень. Ці датасети надаються різними організаціями, дослідниками, спільнотами та відкритими репозиторіями з метою популяризації та сприяння розвитку машинного навчання.

Безкоштовні датасети в Інтернеті охоплюють різні галузі, такі як зображення, текст, звук, соціальні мережі, фінанси, медицина та багато інших. Вони можуть містити велику кількість даних, що дозволяє робити більш точні та надійні моделі машинного навчання.

Ці безкоштовні датасети зазвичай постачаються з документацією, яка описує їхню структуру, формат та використання. Це дає можливість дослідникам і розробникам ознайомитись з даними і використовувати їх у своїх проектах без необхідності створення власних датасетів з нуля.

Розмір датасетів може варіюватись від невеликих до великих обсягів даних, залежно від конкретної задачі. Великі датасети можуть містити мільйони або навіть мільярди записів, що дозволяє побудувати більш точні та репрезентативні моделі.

Для ефективного використання датасетів важливо їх підготувати і очистити від некоректних або відсутніх даних. Це може включати видалення дублікатів, заповнення пропущених значень та нормалізацію даних.

Датасет може бути побудований для вирішення конкретної задачі, такої як розпізнавання обличчя або класифікація тексту, і містити відповідні мітки для кожного зразка даних. Також датасет може бути використаний для перевірки та оцінки точності моделей після їх тренування.

Для збереження датасетів використовуються різні формати, залежно від типу даних і специфічних потреб проекту, серед яких:

– CSV (Comma-Separated Values;
– JSON (JavaScript Object Notation);
– HDF5 (Hierarchical Data Format);
– SQL-бази даних;
– TFRecord.

Важливо зазначити, що використання датасетів повинно відбуватися відповідно до правил інтелектуальної власності, етичних принципів та дотримання законодавства щодо захисту особистих даних.

Існують два типи генерації датасетів – штучна та природна. Штучна генерація датасету передбачає створення даних, які не виникають у реальному світі, а генеруються за допомогою алгоритмів та програм. Наприклад, можна створити датасет з фотографій відповідної категорії, використовуючи комп'ютерні програми для створення візуальних зображень, які потім будуть використовуватись для тренування моделі.

Природний датасет містить реальні дані, які можуть бути кориснішими для тренування моделей машинного навчання, оскільки вони відображають реальні умови та фактори, які можуть впливати на результати. Однак, збір даних може бути досить складним та повільним процесом, тому штучна генерація датасету також є важливою складовою машинного навчання.

Датасет може складатися з вибірок. Вибірки – це підмножини даних, які можуть бути витягнуті з датасету [20]. Використання вибірок може бути

корисним, при великому датасеті, або коли потрібно зробити багато ітерацій навчання моделі.

Вибірки можуть бути сформовані різними способами, наприклад, випадковим відбором, згрупуванням за спільними характеристиками, чи за заздалегідь визначеними критеріями [21]. Розмір вибірки може бути змінюваним, в залежності від конкретних вимог і задач, які перед нами стоять.

Зазвичай для побудови ефективної моделі машинного навчання необхідно мати три типи вибірок:

– тренувальні вибірки – це набори даних, які використовуються для навчання моделі. Тренувальні вибірки повинні містити як позитивні, так і негативні приклади, щоб модель мала змогу навчитися розрізняти між ними;

– валідаційні вибірки – це набори даних, які використовуються для перевірки ефективності моделі. Валідаційні вибірки також повинні містити позитивні та негативні приклади, і їхні результати допомагають зрозуміти, наскільки добре працює модель;

– тестові вибірки – це окремий набір даних, які використовуються для оцінки остаточної ефективності моделі після тренування та валідації. Ці вибірки не повинні бути використані в процесі навчання моделі.

Пропорції між тренувальною, валідаційною та тестовою вибірками можуть різнитися в залежності від розміру датасету та конкретного завдання, але деякі загальні рекомендації можуть бути наступними:

– тренувальна вибірка зазвичай більша за інші вибірки та займає від 60% до 80% датасету;

– валідаційна вибірка зазвичай менша за тренувальну вибірку, займає від 10% до 20% датасету. Використовується для налаштування параметрів моделі та оцінки її продуктивності на невідомих даних;

– тестова вибірка зазвичай менша за валідаційну вибірку, займає від 10% до 20% датасету. Використовується для оцінки продуктивності моделі на невідомих даних та перевірки її здатності до узагальнення.

Датасети є невід'ємною частиною багатьох проектів з аналітики даних, машинного навчання та ШІ. Вони забезпечують доступ до реальних або синтетичних даних, які використовуються для розробки та оцінки моделей, проведення.

Під час роботи з датасетами важливо враховувати їхню якість, репрезентативність та надійність. Якісний датасет повинен бути достатньо репрезентативним для потреб проекту, містити різноманітність даних і бути достатньо великим для надійного аналізу. Також важливо враховувати правові аспекти та дотримуватися правил використання датасетів, зокрема щодо конфіденційності та приватності даних.

Ретельний аналіз, вибір та використання датасетів стають ключовими факторами успіху в багатьох проектах. Наявність якісних та відповідних датасетів дозволяє збільшити точність та ефективність моделей, покращити процеси прийняття рішень та сприяти інноваційному розвитку.

З розвитком технологій та збільшенням доступності даних, очікується зростання обсягів датасетів. Це означає, що датасети будуть ставати все більшими та розмаїтими, включати більше змінних та характеристик. Також очікується збільшення розмаїття джерел даних, включаючи сенсорні пристрої, смарт-пристрої та соціальні медіа.

Завдяки розвитку ШІ та аналітичних методів, будуть розроблятися нові техніки обробки та аналізу датасетів, що дозволить отримувати більш точні передбачення та зробить їх використання більш ефективним. Більш активно будуть використовуватися технології автоматичної генерації датасетів та синтетичних даних, що дозволить розширити масштаб і різноманітність доступних даних.

Зростання свідомості щодо етичних питань пов'язаних з даними, приведе до збільшення фокусу на приватність та безпеку даних у датасетах. Остаточно, майбутнє датасетів визначається постійними змінами технологій та потребами суспільства, що вимагає постійного розвитку та інновацій в галузі збору, обробки та використання даних.

1.4 Постановка задачі

Важливо розуміти, що вищенаведені системи та підходи, хоча і мають свої переваги, не можуть гарантувати абсолютну безпеку від витоку даних. Кожна з цих систем має свої обмеження та потенційні недоліки, які можуть бути використані хакерами або зловмисниками для отримання доступу до цінної інформації.

Беручи до уваги наявність вже розроблених систем та підходів, які використовують ШІ, варто зазначити, що на сьогоднішній день жодна з них не здатна повністю задовольнити всі вимоги даної задачі. Тому головною метою цього проекту є створення власної системи запобігання витоку даних, використовуючи потужні можливості ШІ.

Завданнями проекту є:

1. Розробити систему на основі ШІ, яка може виявляти та запобігати витокам даних у режимі реального часу у MySQL;

2. Впровадити алгоритм ШІ, який може аналізувати шаблони даних і поведінку користувачів, щоб виявити потенційні вразливості та ризики, які можуть призвести до витоку даних;

3. Навчити модель ШІ розпізнавати шаблони викрадання даних у різних типах даних, щоб забезпечити комплексний захист від витоку даних;

4. Розробити систему сповіщень сповіщення в реальному часі та докладні звіти про витік даних і події безпеки, щоб допомогти командам безпеки швидко реагувати на потенційні загрози та пом’якшувати їх наслідки.

2 ДЕТАЛІЗАЦІЯ УМОВ ВИКОРИСТАННЯ ТА ВИБІР ЗАСОБІВ РЕАЛІЗАЦІЇ ВИЗНАЧЕНИХ ЗАДАЧ

2.1 Середовище розробки

Вибір оптимального середовища розробки є ключовим для забезпечення продуктивного та ефективного робочого процесу. У цьому розділі буде детально розглянуто середовище розробки проекту, включаючи мову програмування, операційну систему та бібліотеки, які будуть використані. Також, буде розглянуто використані інструменти і методи, а також пояснена їхня роль у процесі розробки проекту.

Мовою програмування було вибрано Python тому, що він:

- має простий та зрозумілий синтаксис;
- є відомим та популярним у всьому світі;
- має велику кількість бібліотек для створення ШІ;
- легко та швидко запускається на будь-якому пристрої.

У ролі середовища розробки використовуватиметься Docker.

Docker – це платформа, яка дозволяє розробникам легко створювати, розгортати та запускати програми в контейнерах [22]. Контейнери – це міні операційні системи, які максимально оптимізовані під окремі сервіси. Ці контейнери містять в собі переважно Linux, як операційну систему, з мінімальним функціоналом та без графічної оболонки, де мають встановлений та налаштований окремий сервіс. Наприклад, python або mysql. Завдяки цьому зменшується споживання ресурсів комп'ютера розробника, що дозволяє розробникам зосередитися на написанні коду, не турбуючись про базову інфраструктуру та залежності. Так, як всі конфігурації сервіса знаходяться в одному місці та є легко доступні, це дозволяє швидко переналагоджувати середовище у випадку помилки зі сторони розробника або програми. Також Docker – це програма, яка не залежить від операційної системи, що є її великою перевагою. Адже, саме несумісність операційних систем є великою завадою для

розповсюдження програмного забезпечення в світі. Ця опція дозволяє розгортати середовища швидко та на будь-якій системі, що буде використано в подальшому.

Також для покращення процесу розробки буде використовуватися утиліта Docker Compose. Docker Compose – це інструмент для налаштування та запуску багато контейнерних програм Docker [23].

Завдяки цій утиліті стає можливим конфігурувати сервіси та мережі в одному файлі, а потім запускати все середовище за допомогою однієї команди. Ця утиліта спрощує процес розгортання середовища на інших комп'ютерах.

Об’єктом, який буде захищатися від витоку даних є – СУБД MySQL.

MySQL – це система керування реляційною базою даних із відкритим вихідним кодом, яка використовує SQL (мову структурованих запитів) для керування та обробки даних [24]. MySQL є популярним вибором для багатьох веб-додатків завдяки своїм характеристикам, таким як:

– надійність та стабільність;

– масштабованість;

– простота використання.

Досвід багатьох років показав, що MySQL довів свою стабільність і надійність шляхом постійних оновлень і вдосконалень.

MySQL має відкритий вихідний код, який можна переглядати, змінювати та розповсюджувати. Також, він має велику спільноту користувачів і розробників, що допомагає забезпечити його надійність і підтримку.

MySQL має простий та інтуїтивно зрозумілий синтаксис для створення, запиту, оновлення та видалення даних у базах даних.

Для розробки буде використовуватися MySQL Docker контейнер. Docker в свою чергу має велику базу з вже готових зразків під різні сервіси, та MySQL також. Ця база називається Docker Hub [25]. Під час розробки використовувалася остання версія MySQL. Незважаючи на це, готова система зможе працювати з MySQL, який знаходиться не тільки в контейнері, а й на локальному середовищі, на окремому сервері, або на віртуальній машині. Також, система потенційно зможе працювати із СУБД MariaDB, адже вона є сумісною з MySQL, бо є його

відгалуженням. Це є можливим завдяки способі авторизації MySQL. Для того, щоб виконати будь-які дії над базою даних, потрібно мати облікові дані користувача, такі як:

– хост;

– логін;

– пароль;

– порт.

Незважаючи на це, для роботи основної програми використання облікових даних не потрібно, адже вона працює лише з файловою системою і текстом. Облікові дані, які були наведені вище, потрібні лише для створення датасету, процес якого буде розкритий в наступному розділі.

За замовчуванням, MySQL не створює жодних файлів логів. Тому, для коректної роботи сервера було створено додатковий конфігураційний файл, який включає генерацію логів та визначає шлях до файла, в який дані будуть записуватися.

На рисунку 2.1 зображений вміст конфігураційного файлу.

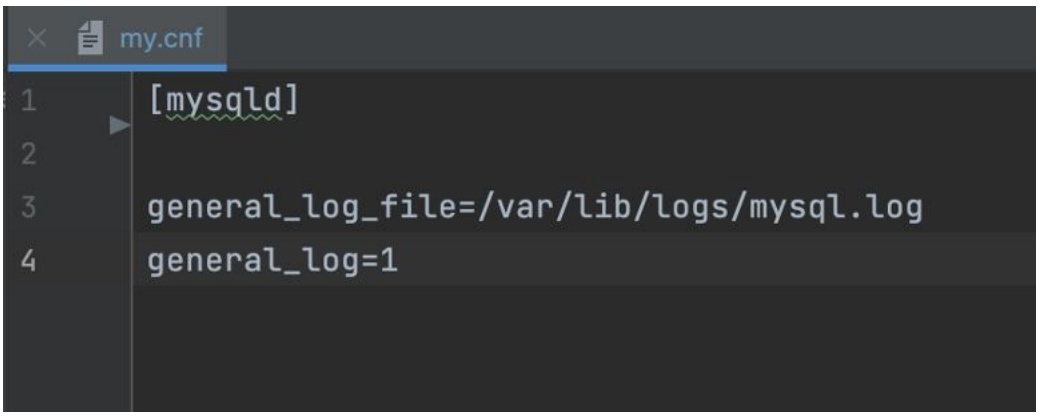


Рисунок 2.1 – Конфігураційний файл для MySQL

Конфігураційні файли MySQL визначають його налаштування та параметри роботи. Вони забезпечують контроль над різними аспектами функціонування сервера, такими як розмір буферів, налаштування безпеки, мережеві налаштування та інші.

Для перемикача поле в конфігурації називається general_log, для шляху – general_log_file [26]. Цей конфігураційний файл може бути інтегрований в MySQL, який знаходиться на різних серверах.

На рисунку 2.2 вказано схематично роботу системи.

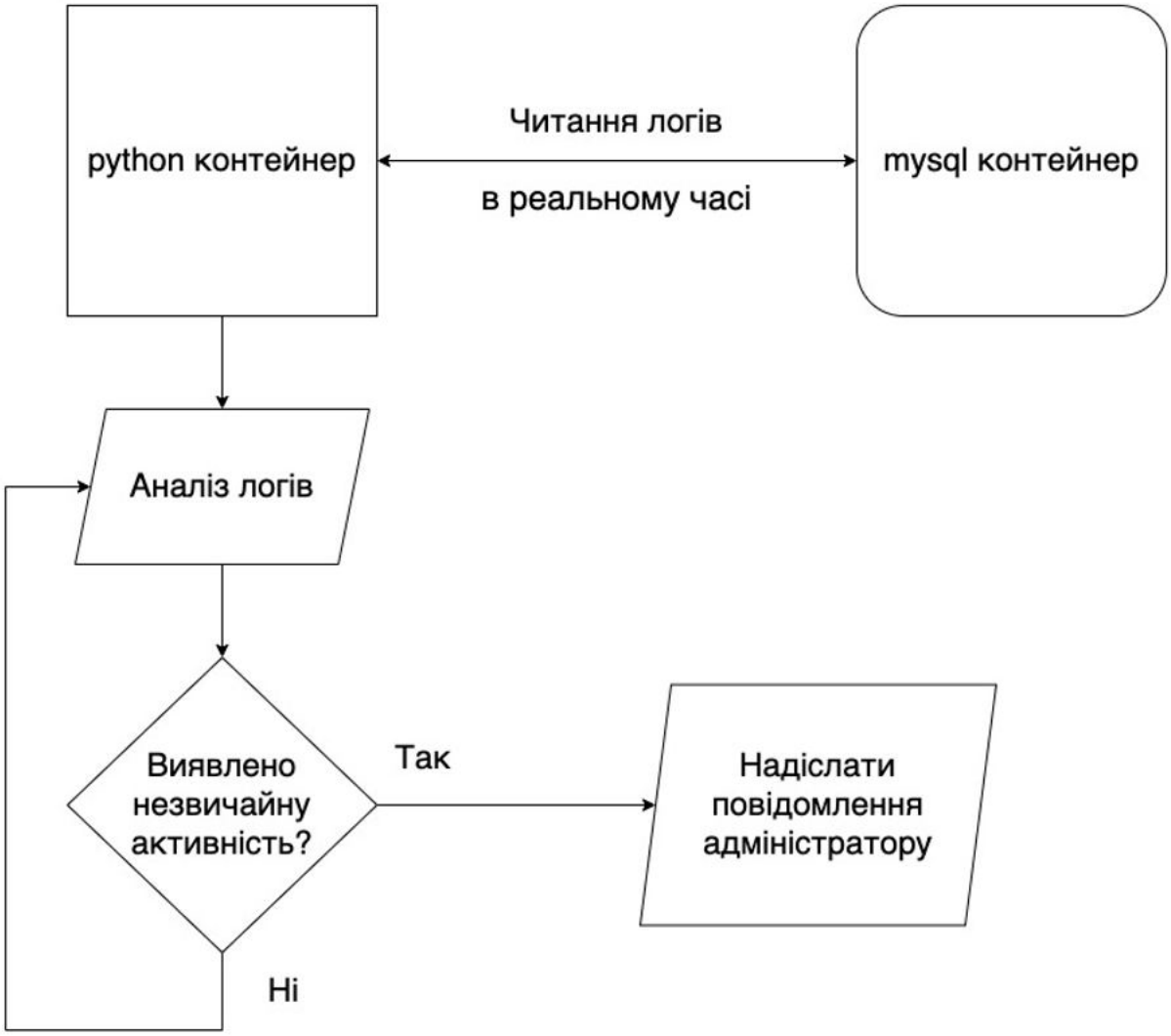


Рисунок 2.2 – Схема відношення між контейнерами python і mysql

Логи – це записи в певному файлі, або декількох файлах, які містять інформацію про події, що сталися в процесі роботи програмі [27]. Вони зазвичай використовуються для відстеження дій користувачів, діагностики помилок та налагоджування програмного забезпечення. У більшості випадків, логи містять

інформацію про події, що відбуваються в реальному часі, а також дату та час, коли вони сталися. Вони можуть також містити рівень важливості події (наприклад, інформаційний, попередження, помилка), її короткий опис та детальну інформацію про те, що сталося. Лог-файли можуть бути проаналізовані з використанням спеціальних програм, щоб знайти помилки та інші проблеми, які можуть виникнути в системі.

Тобто, python контейнер в реальному часі зчитує записи з лог-файлів у контейнері mysql та аналізує їх. Результатом аналізу буде рішення про те чи надсилати повідомлення адміністратору про підозру у витоку даних.

Основним елементом роботи програми є бібліотека ШІ TensorFlow. TensorFlow – це бібліотека машинного навчання з відкритим кодом, розроблена командою Google Brain, яка надає широкий спектр інструментів для створення та розгортання моделей машинного навчання [28]. Він створений для забезпечення швидких числових обчислень і пропонує дуже гнучку та розширену систему для створення та навчання моделей. TensorFlow може працювати на різних платформах, таких як сервери, настільні комп'ютери, мобільні пристрої та Інтернет речей.

Багато великих компаній використовують TensorFlow для своїх проектів машинного навчання. Деякі з них: Google, Airbnb, Uber, Coca-Cola, Intel, Twitter, Snapchat та багато інших. TensorFlow вважається одним з найпопулярніших фреймворків машинного навчання відкритого коду та продовжує знаходити все більш широке застосування у різних галузях.

Доцільно використовувати саме бібліотеку, а не писати логіку ШІ з нуля тому, що алгоритми у ній вже реалізовані та оптимізовані провідними спеціалістами у цій галузі.

Загалом TensorFlow стала однією з найпопулярніших бібліотек машинного навчання завдяки своїй універсальності, простоті використання та широкій підтримці спільноти.

Основною задачею ШІ буде аналіз лог-файлів, тобто звичайного тексту. TensorFlow містить функції та інструменти для обробки тексту, наприклад

бібліотеку TensorFlow Text, яка включає низку інструментів попередньої обробки та токенізації для текстових даних, а також можливість використовувати попередньо навчені мовні моделі для таких завдань, як аналіз настроїв, класифікація тексту та генерація тексту.

2.2 Генерація датасету

Датасет є важливим елементом кожної моделі ШІ. Адже, за допомогою нього модель спроможна навчитися і видавати певний результат. Модель без датасета це лише набір алгоритмів, який не спроможний нічого зробити.

Як було сказано раніше, датасети поділяються на штучні та природні. Кожен вид має свої переваги та недоліки. Проте, в контексті цього проекту доречніше використовувати штучний вид тому, що:

– створення власного датасету дозволяє мати повний контроль над даними, які використовуються для тренування моделей;

– дає можливість працювати з унікальними або специфічними даними;

– генерація власного датасету може бути ефективним способом зекономити час та ресурси.

Опираючись на вище перераховані переваги штучного датасету, вибраний був саме цей вид.

Це дає свої переваги у порівнянні з природним датасетом. По-перше, штучна генерація дозволяє згенерувати більшу кількість даних, ніж доступно в природних джерелах, що може бути корисним при тренуванні складних моделей машинного навчання. Це доступно через те, що розробник сам може контролювати скільки даних він генерує своєю програмою чи скриптом.

В контексті даного проекту, було визначено, що датасет буде поділятися на дві частини: правильні та неправильні відповіді. Кожна частина міститиме 2500 згенерованих одиниць даних, тобто логів.

По-друге, штучна генерація дозволяє контролювати параметри, які важко контролювати в природних джерелах, такі як розмір, розподіл, форма та рівномірність вибірки. Це може допомогти зменшити вплив зовнішніх факторів на якість моделі машинного навчання.

Цей аспект штучної генерації використовується для кращого навчання моделі, тому було прийнято рішення використовувати для кожного сценарію файли, які містять по 13 рядків логів.

На рисунку 2.3 зображено схему генерації штучного датасету.

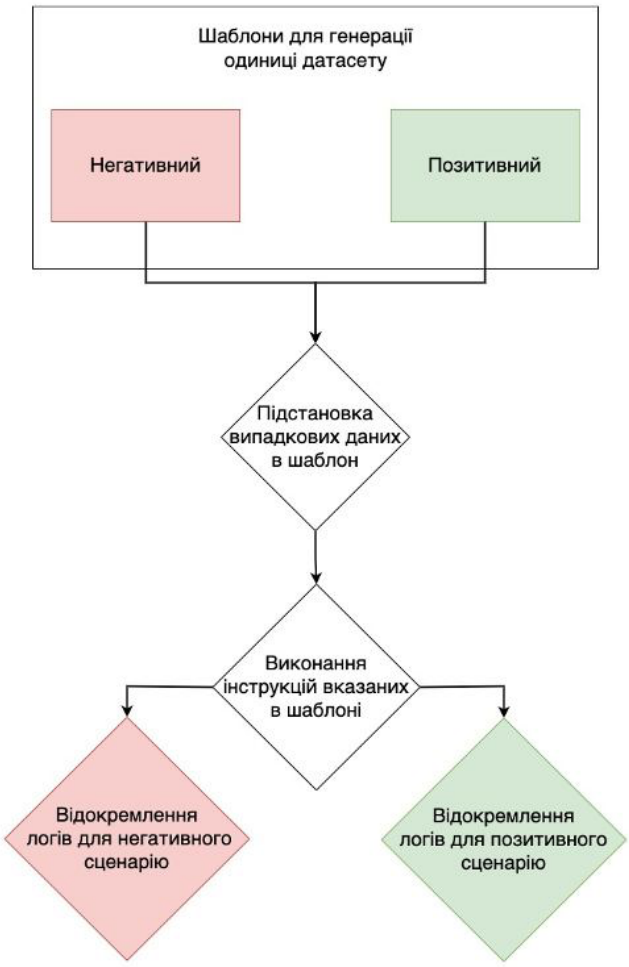


Рисунок 2.3 – Схема генерації датасету

Основу датасету будуть складати шаблони. Шаблони, це наперед визначені інструкції, які виконує програма для отримання відповідних логів.

Основний формат даних, з яким буде взаємодіяти програма – це текст, у вигляді логів. Після кожного виконання команд з шаблону вже із згенерованими випадковими даними, частина логів буде відокремлена в окремий файл та папку і позначена відповідно до типу шаблону. Це в подальшому дозволить ефективніше використовувати датасет.

В контексті генерації датасету існують два поняття: позитивний та негативний сценарії. Вони є важливим аспектом для керованого навчання моделі, бо одразу визначають правильну і неправильну відповідь.

Позитивний сценарій означає, що модель при аналізі даних знайшла логи, які потенційно вказують на витік даних. Негативний сценарій в свою чергу є протилежним до першого, тобто коли співпадінь не знайдено при аналізі логів.

На рисунку 2.4 наведений приклад логів для позитивного сценарію.

a_2023_05_23_09_00_12.log

```
2023-05-23T09:00:12.537321Z   315 Connect   root@192.168.16.3 on  using SSL/TLS
2023-05-23T09:00:12.537656Z   315 Query /*!40100 SET @@SQL_MODE='' */
2023-05-23T09:00:12.539187Z   315 Query /*!40103 SET TIME_ZONE='+00:00' */
2023-05-23T09:00:12.539396Z   315 Query /*!80000 SET SESSION information_schema_stats_expiry=0 */
2023-05-23T09:00:12.539591Z   315 Query SET SESSION NET_READ_TIMEOUT= 86400, SESSION NET_WRITE_TIMEOUT= 86400
2023-05-23T09:00:12.539872Z   315 Query SHOW VARIABLES LIKE 'gtid_mode'
2023-05-23T09:00:12.541478Z   315 Query SELECT LOGFILE_GROUP_NAME, FILE_NAME, TOTAL_EXTENTS, INITIAL_SIZE, ENGINE, EXTRA
2023-05-23T09:00:12.543547Z   315 Query SELECT DISTINCT TABLESPACE_NAME, FILE_NAME, LOGFILE_GROUP_NAME, EXTENT_SIZE, INI
2023-05-23T09:00:12.545557Z   315 Query SHOW VARIABLES LIKE 'ndbinfo\_version'
2023-05-23T09:00:12.547001Z   315 Init DB   a
2023-05-23T09:00:12.547301Z   315 Query show tables
2023-05-23T09:00:12.548298Z   315 Query UNLOCK TABLES
2023-05-23T09:00:12.550244Z   315 Quit
```

Рисунок 2.4 – Приклад логів, які відповідають за позитивний сценарій датасету

Шаблон, який відповідає за позитивний сценарій, тобто коли відбувається спроба викрасти дані, генерується за допомогою команди mysqldump, яка створює файл з даними бази.

mysqldump – це утиліта командного рядка, яка надає можливість резервного копіювання та відновлення бази даних MySQL. Вона дозволяє зробити повний або частковий експорт даних, структури таблиць, процедур, функцій та інших об'єктів бази даних MySQL у текстовий файл.

Приклад логів для цього сценарію наведений на рисунку 2.5.

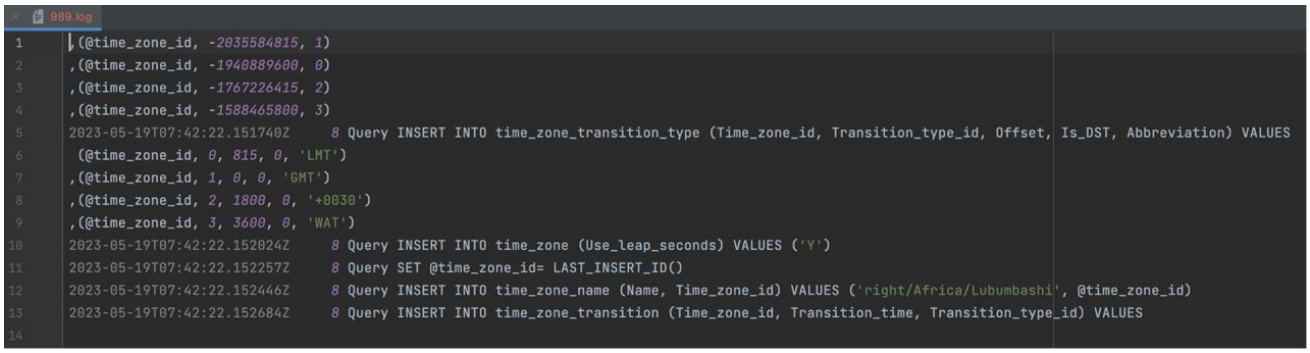


Рисунок 2.5 – Приклад логів, які відповідають за негативний сценарій датасету

Для шаблону, який відповідає за негативний сценарій, тобто при якому не потрібно робити жодних дій, генерується за допомогою вже існуючого файлу логів, який був взятий з працюючого MySQL сервера.

Для зручності та пришвидшення процесу навчання моделі, датасет об'єднується в спільний CSV файл. На рисунку 2.6 зображений цей CSV файл.

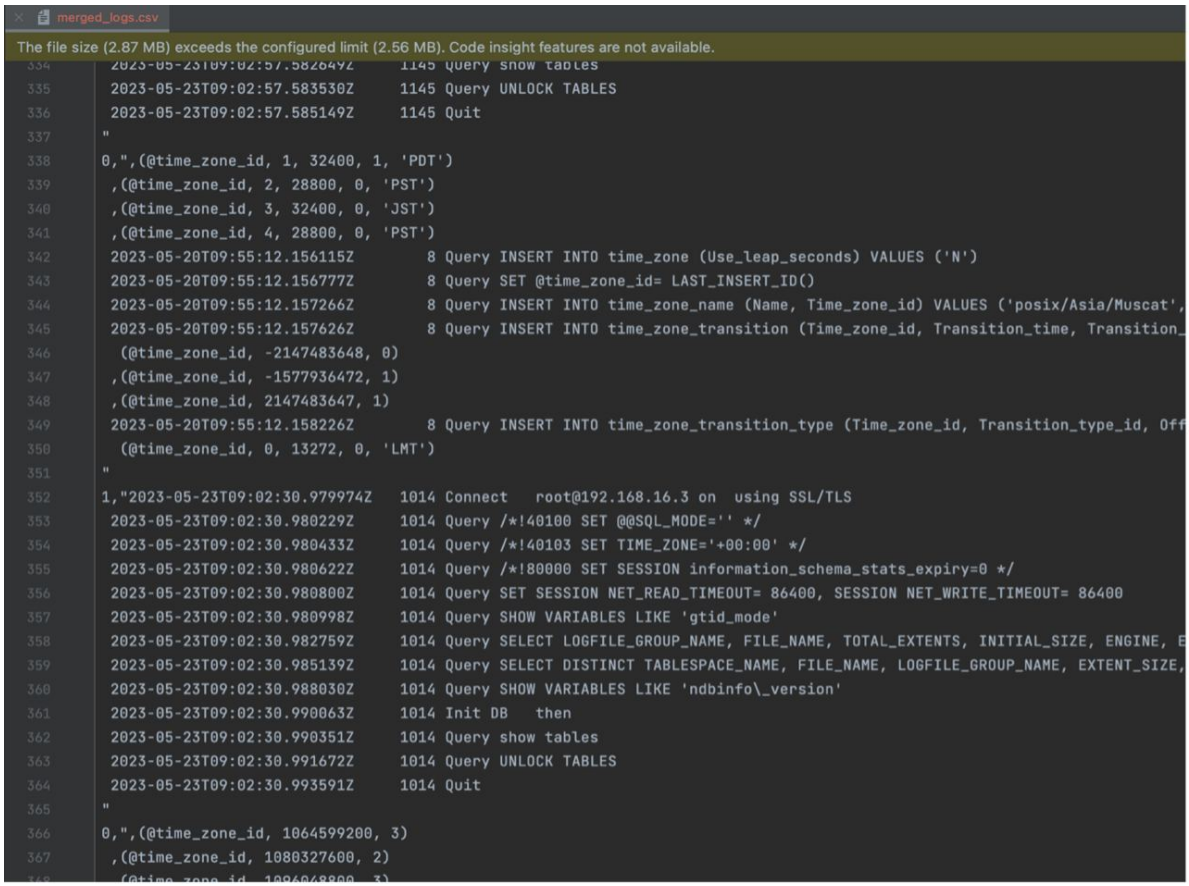


Рисунок 2.6 – Об'єднаний csv файл

CSV файл – це формат файлу, в якому значення розділені за допомогою коми та знаку перенесення на новий рядок. В цьому файлі негативний та позитивний сценарії перемішані та промарковані за допомогою 1 та 0, де 1 – позитивний сценарій, а 0 – негативний. Ці позначки використовуватимуться для визначення правильних та неправильних відповідей для керованого навчання.

Для того, щоб модель краще навчалася їй потрібно декілька вибірок. Рекомендовано використовувати 3 типи вибірок: тренувальну, валідаційну та тестову.

На рисунку 2.7 зображено процентний розподіл датасету на вибірки.

Рисунок 2.7 – Схема розподілу датасету між вибірками

Керуючись рекомендаціями від TensorFlow доцільно буде віднести до тренувальної вибірки 80% датасету, а до валідаційної й тестової – по 10%.

Для розподілу по вибірках буде використовуватися CSV файл, який був описаний раніше. В ньому позитивні та негативні сценарії змішані, тому можна не хвилюватися щодо нерівномірного розподілу.

Так, як дані вже однорідні та перемішані у файлі, тому ніяких додаткових дій перед розподілом робити не потрібно. Тому дані виділятимуться по черзі, тобто перших 80% файлу піде на тренувальну вибірку, а інші 20% будуть розділені між валідаційною та тестовою вибірками.

2.3 Навчання моделі

В попередньому розділі було розглянуто процес генерації власного датасету для навчання ШІ. Далі буде розглянуто можливості використання цього датасету під час навчання мережі. Також буде розглянуто типи навчання та алгоритм основних функцій.

Навчання можна класифікувати за різними критеріями, включаючи спосіб навчання та рівень контролю за процесом навчання.

Кероване навчання – це тип навчання з учителем, в якому модель навчається на основі попередньо підготовлених прикладів вхідних даних та відповідних їм вихідних даних.

Спонтанне навчання – це тип навчання без учителя, в якому модель навчається на основі вхідних даних без будь-яких відповідних вихідних даних або інформації про правильні відповіді.

Навчання з підкріпленням – це тип навчання, в якому модель навчається на основі взаємодії з навколишнім середовищем, отримуючи підсилення або штрафи в залежності від результату своїх дій.

Алгоритм збіжного рекурсивного навчання – це тип навчання, який використовується для навчання штучної нейронної мережі з рекурсивною структурою. В цьому типі навчання модель навчається шляхом поступового зменшення розміру мережі та збільшення її точності під час навчання.

Основний принцип керованого навчання полягає в тому, що модель навчається передбачати відповіді на основі вхідних даних. При цьому, вхідні дані та відповіді повинні бути заздалегідь підготовлені та мітки відповідей повинні бути відомі.

Для навчання штучного навчання доцільно використати метод керованого навчання. Маючи можливість генерувати датасет з правильними та неправильними відповідями дасть кращий результат з використанням керованого навчання, порівняно з іншими видами навчання.

Також варто вказати, що кероване навчання має такі основні плюси для проекту:

– можливість досягнути високої точності в прогнозуванні за допомогою згенерованих даних;

– можливість використання з відносно невеликою кількістю даних, що дозволить зменшити час навчання при сталій результативності.

Процес керованого навчання складається з кількох етапів:

1. Генерація датасету, якщо не був згенерований до цього;
2. Розподілення згенерованого датасету на три вибірки;
3. Токенізація тексту з датасета;
4. Ініціалізація Sequential моделі;
5. Наповнення моделі шарами: embedding, LSTM та Dence;
6. Компіляція моделі з функцією втрат BinaryCrossentropy та оптимізатором Adam;
7. Тренування моделі з використанням токенізованої тренувальної вибірки;
8. Оцінка моделі з використанням токенізованої валідаційної вибірки;
9. Тестування моделі – оцінка якості моделі на тестовій вибірці.

В якості моделі було використано об’єкт моделі типу Sequential [29]. Ця модель є послідовною нейронною мережею, побудованою за допомогою бібліотеки Keras. Вона складається з кількох шарів, розташованих один за одним, де вихід кожного шару передається як вхід до наступного шару. Ця архітектура дозволяє розбити задачу на послідовні етапи обробки даних.

Варто розуміти, що нейронні мережі та машинне навчання є підмножиною ШІ. Нейронні мережі – це підхід до моделювання ШІ, який навчає комп'ютерну систему розрізняти та впізнавати патерни у великих обсягах даних, наслідуючи принципи роботи біологічних нейронних мереж у мозку. Нейронні мережі складаються зі штучних нейронів, які співпрацюють між собою та формують глибокі зв'язки, здатні до вирішення складних задач [30].

Машинне навчання є підходом до ШІ, де комп'ютерні системи навчаються робити висновки з даних та приймати рішення на основі набору правил та шаблонів [31]. Використовуючи алгоритми машинного навчання, системи можуть вдосконалювати свою продуктивність та здатність до самостійного навчання з досвідом. Це дозволяє їм адаптуватися до нових ситуацій, розпізнавати патерни та робити прогнози на основі доступних даних.

Ця модель є ефективним варіантом для бінарної класифікації текстових даних, особливо коли необхідно враховувати контекст та послідовність слів.

Так, як ця модель передбачає наповнення її шарами. То наступним кроком буде ініціалізація шарів для цієї моделі.

Перший шар – це embedding layer, або шар вбудовування. Шар вбудовування є важливою частиною моделей для обробки текстових даних. Він призначений для перетворення категоріальних або дискретних значень (таких як слова або токени) в вектори фіксованої довжини [32]. Це важливо, оскільки багато алгоритмів машинного навчання працюють з числовими векторами, а не з рядками або символами.

Для ініціалізації цього шару потрібно також передати кількість унікальних слів в датасеті та максимальну довжину вхідного рядка, які будуть визначені підчас токенізації датасету. Токенізація буде розглянути далі у цьому розділі.

Наступним у черзі є шар LSTM.

LSTM можна уявити як нейронну мережу з "пам'яттю". Він вміє запам'ятовувати і використовувати інформацію з попередніх кроків часу для генерації прогнозів на наступних кроках [33].

Основна ідея полягає в тому, що LSTM має внутрішні комірки пам'яті, які можуть зберігати та оновлювати інформацію упродовж часу. Кожна комірка має свою структуру з трьох ключових компонентів: входу, виходу та забування.

LSTM використовує вхідний компонент для вирішення, які частини інформації потрібно зберегти в пам'яті. Далі, він оновлює пам'ять, враховуючи нові дані та попередні стани пам'яті. Наостанок, використовується вихідний компонент для генерації вихідного значення на поточному кроці часу.

Цей механізм дозволяє LSTM здійснювати ефективну роботу з послідовними даними, зокрема здійснювати прогнози, зберігати та використовувати контекстуальну інформацію з попередніх кроків. Він особливо корисний при роботі з великими текстовими даними, де важлива взаємодія слів та контексту для зрозуміння смислу тексту.

У контексті даної моделі використовується LSTM шар з 64 нейронами та вказаним параметром dropout, який має значення 0.1. Він вказує на ймовірність відключення випадкового нейрона під час тренування моделі. Dropout використовується для регулювання моделі та запобігання перенавчанню шляхом випадкового вимкнення нейронів, тим самим змушуючи модель навчатися більш узагальненим репрезентаціям. Значення 0.1 вказує на те, що при кожному пакеті тренування 10% нейронів будуть випадково вимкнені.

Наступний шар – це Dence. Він є одним із типів шарів нейронної мережі, в якому кожен нейрон пов'язаний з кожним нейроном попереднього шару [34]. У контексті даної моделі цей шар використовує 1 нейрон з активаційною функцією sigmoid. Функція sigmoid відображає будь-яке вхідне значення на інтервал (0, 1), де значення ближче до 0 відповідає меншій ймовірності, а значення ближче до 1 відповідає більшій ймовірності.

Далі визначається один з головних компонентів моделі – функція втрат. Функція втрат (loss function) є одним з ключових елементів при навчанні моделі машинного навчання. Вона використовується для оцінки різниці між прогнозованими значеннями моделі і фактичними мітками у тренувальному наборі даних [35].

Метою функції втрат є кількісна оцінка того, наскільки добре модель відповідає на поставлену задачу, таку як класифікація, регресія чи сегментація. Вона вимірює величину помилки між прогнозованими значеннями та фактичними значеннями, і ця помилка використовується для налаштування параметрів моделі під час процесу навчання. Для моделі використано функцію під назвою BinaryCrossentropy. Вона використовується у задачах бінарної класифікації. При бінарній класифікації модель спробує передбачити одне з двох можливих значень: 0 або 1.

На рисунку 2.8 зображений процес ініціалізація та навчання моделі.

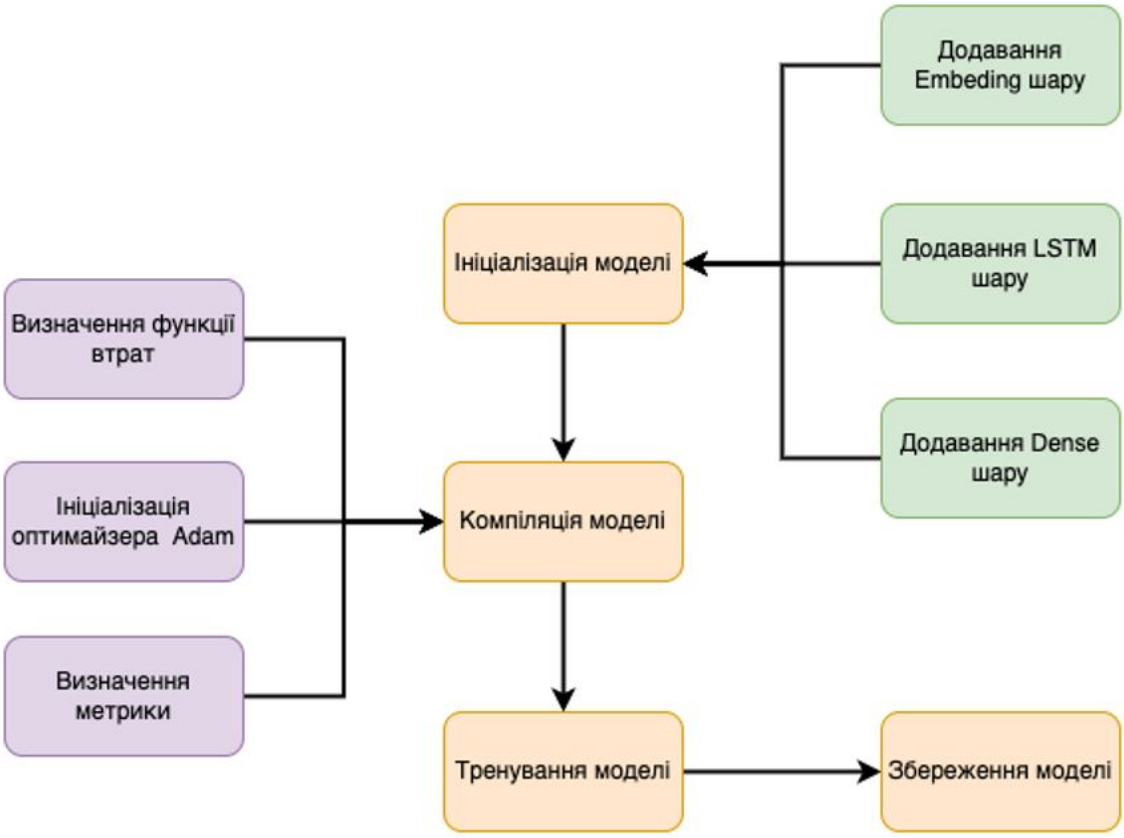


Рисунок 2.8 – Схематична візуалізація навчання

BinaryCrossentropy допомагає оцінити, наскільки добре модель відповідає на цю задачу. Вона порівнює прогнозовані значення моделі з реальними мітками і визначає, наскільки вони відхиляються одне від одного.

Наступним кроком є визначення оптимізатора. Він відповідає за оновлення параметрів моделі, з метою мінімізації функції втрат та покращення ефективності моделі під час навчання.

В якості оптимізатора використаний Adam. Це оптимізатор є одним з популярних алгоритмів оптимізації в глибинному навчанні [36]. Він комбінує метод адаптивної швидкості навчання (adaptive learning rate) та метод моменту (momentum) для ефективної оптимізації. Алгоритм Adam дозволяє швидше знаходити оптимальні значення параметрів моделі, оскільки він ефективно адаптується до змін градієнтів і робить кращі кроки в навчанні.

Також, оптимізатор ініціалізується з параметром lr=0.001, який встановлює швидкість навчання (learning rate). Він визначає, наскільки швидко модель адаптується до змін в градієнтах та оновлює свої параметри під час навчання. Значення 0.001 вказує на низьку швидкість навчання, що може бути корисним для стабільного та точного навчання моделі.

Передостаннім кроком є компіляція моделі. Для компіляції використано 3 компоненти: функція втрат, оптимізатор та метрики. Функція втрат та оптимізатор були описані раніше в цьому розділі. Метрики ж використовуються для оцінки ефективності моделі. Для цієї моделі визначені метрики – accuracy.

Точність (accuracy) вимірює відсоток правильно класифікованих прикладів від загальної кількості прикладів. Вона вказує на те, наскільки добре модель виконує класифікацію на наявних даних. Чим вище значення точності, тим краще модель.

Під час навчання моделі, використовуючи цю метрику, можна спостерігати, як змінюється точність моделі з кожною епохою навчання. Це дозволяє оцінювати якість моделі і визначати, наскільки добре вона працює під час навчання та перевірки.

Останнім кроком є саме навчання моделі. Модель навчалася на тренувальних даних та перевірялася на валідаційних даних протягом 20 епох. Окрім епох, моделі для навчання потрібні токенізовані навчальні та валідаційні дані разом з правильними відповідями.

Важливим етапом навчання моделі є її збереження. Цей крок є необхідним для подальшого використання та навчання моделі.

За допомогою функціоналу Keras можна зберегти всю архітектуру моделі, внутрішні параметри, ваги та конфігурацію оптимізатора. Модель зберігається у форматі HDF5 у файловій системі, що є стандартним форматом збереження моделей в TensorFlow [37].

На рисунку 2.9 зображено ініціалізація та навчання токенайзера.

Рисунок 2.9 – Ініціалізація токенайзера та обробка датасету перед навчанням моделі

Перед усім важливо подбати про дані. В попередньому розділі було зображено генерацію датасету. Проте, цей датасет не є валідним для навчання в своєму вигляді. Для коректного навчання ці данні потрібно обробити. До обробки даних належать:

– токенізація;

– очищення даних від зайвих символів або шуму;

– векторизація;

– нормалізація;

– зменшення обсягу даних.

Токенізація є процесом розбиття тексту на окремі слова або токени. Використання токенізації допомагає зменшити розмір словника і забезпечує числове представлення слів або токенів [38]. На рисунку 2.10 зображено процес обробки текстових даних для моделі.

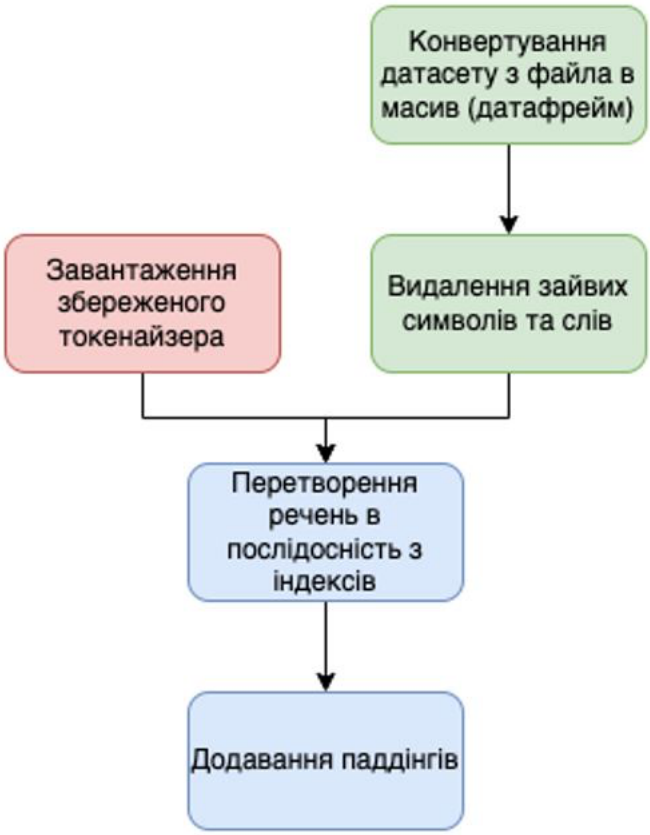


Рисунок 2.10 – Обробка даних для передбачення моделлю

Навчання моделі без токенізації є складним, оскільки модель не матиме інформації про окремі слова або токени, а буде працювати зі вхідним текстом як з одним без порядковим рядком символів. Це може призвести до втрати семантичної інформації та погіршення результатів моделі.

Токенайзер – це один з об’єктів Keras, за допомогою якого виконується токенізація даних. Для початку роботи його потрібно навчити на усіх даних. Протягом цього процесу він визначає всі унікальні слова та визначає їм конкретний індекс. Саме тому важливо зберігати токенайзер разом з моделлю, адже при використанні іншого – робота моделі буде помилковою через те, що індекси однакових слів будуть різні.

Важливим елементом обробки даних є видалення зайвих елементів. До зайвих елементів, які знаходять в моєму датасеті можна віднести дату та час, а також спеціальні слова, які не несуть логічного навантаження.

За векторизацію даних у моделі відповідає шар вбудовування. Нормалізація – це приведення даних, до однакової довжини або стандарту. Для цього було використано функцію pad_sequences з keras [39]. За допомогою неї токенізовані дані приводяться до фіксованого розміру. Для моєї моделі встановлено ліміт в 200 слів на одиницю даних. Ця функція заповнить нулями слова, який не вистачає до фіксованої довжини, а також видалить слова, щоб вміститися в ліміт.

Хоч і навчання та передбачення це два різних процеси, проте вони обидва вимагають обробку вхідних даних для коректної роботи. Незважаючи на те, що ці два процеси майже однакові, вони мають свої відмінності. Першою з яких є те, що при передбаченні токенайзер завантажується з файлової системи. Як було сказано вище, токенайзер зберігається разом з моделлю. Для токенайзера використовується бібліотека python під назвою pickle. Ця бібліотека дозволяє серіалізувати (зберегти) об'єкти Python у байтовий потік і десеріалізувати (відновити) об'єкти з байтового потоку назад у пам'ять.

2.4 Висновки

У цьому розділі було детально розглянути 3 основні аспекти, які є критичними для розробки системи, яка буде вирішувати поставлені перед нею задачі.

В першому підрозділі було вказано засоби середовища розробки та інструменти, за допомогою яких відбувалася розробка системи. Як систему віртуалізації було обрано docker, мовою програмування – python. Було визначено MySQL, як сервіс, що потрібно захищати. Також, були вказані конфігурації, які повинні бути застосовані на СУБД в усіх випадках для коректної роботи системи.

Застосування контейнерів Docker у поєднанні з Python та MySQL дозволило створити зручне та незалежне середовище розробки. Docker дозволяє легко налаштовувати та управляти ізольованими контейнерами, що спрощує розгортання та управління програмними рішеннями.

Tensorflow був обраний у ролі бібліотеки Python для створення ШІ. Використання Tensorflow для навчання моделі показало його потужність у сфері машинного навчання та аналізу даних. За допомогою Tensorflow була побудована та навчена модель, яка здатна розв'язувати поставлені завдання.

В другому підрозділі було обговорено поняття датасету, його види. Прийнято та обґрунтовано рішення про генерацію власного датасету та наведено її алгоритм. Також, були наведені приклади одиниць датасету.

Основний датасет було прийнято розбити на 3 вибірки: навчальна (80%), валідаційна (10%) та тестова (10%).

Генерація датасету виявилася ключовим кроком у процесі розробки. Було створено механізм, який дозволяє збирати та обробляти дані для навчання моделі. Застосування відповідних методів для збору та обробки даних дозволило створити репрезентативний та різноманітний датасет, який був використаний для навчання та оцінки моделі.

В третьому підрозділі було розписано алгоритм навчання моделі. Навчання моделі було проведено з використанням багатошарової нейронної мережі з використанням рекурентного шару LSTM та щільного шару Dense. LSTM шар відповідає за врахування послідовностей вхідних даних та здатність зберігати короткострокову та довгострокову пам'ять. Dense шар, з одним нейроном та функцією активації sigmoid, використовується для здійснення бінарної класифікації.

Під час навчання моделі використовувалася функція втрат BinaryCrossentropy, яка підходить для бінарної класифікації. Оптимізатор Adam був обраний для покращення швидкості та точності навчання моделі. Метрика accuracy використовувалася для оцінки точності класифікації.

Токенізація, з свого боку, є процесом розбиття текстового документа на окремі лексеми або токени. Це дозволяє моделі розуміти текстові дані на рівні окремих слів, фраз або символів. Токенізація може використовувати різні підходи, включаючи розділення тексту за пробілами, використання регулярних виразів або використання спеціалізованих бібліотек та інструментів.

Обидва ці етапи є важливими кроками у розробці і використанні моделей штучного інтелекту для обробки текстових даних. Правильне навчання моделі та ефективна токенізація допомагають досягти точності, швидкості та якості обробки тексту, що є ключовими факторами успіху у багатьох застосуваннях штучного інтелекту, таких як обробка природної мови, машинний переклад, класифікація тексту та багато інших.

В результаті дослідження було показано, що поєднання вищезгаданих технологій та методів може бути дуже ефективним для розробки програмного рішення та досягнення бажаних результатів у сфері машинного навчання та аналізу даних.

3 РЕЗУЛЬТАТИ ТА ЕФЕКТИВНІСТЬ ПРОГРАМИ

3.1 Структура системи

Структура цієї системи включає в себе декілька компонентів, які співпрацюють разом для забезпечення функціональності та ефективності. Основні компоненти цієї системи включають сервер, Docker-контейнери та їхні складові частини.

Сервер є центральним елементом системи, він відповідає за управління та координацію роботи інших компонентів.

Докер містить в собі 2 контейнери: python та mysql. Структура цих контейнерів зображена на рисунку 3.1.

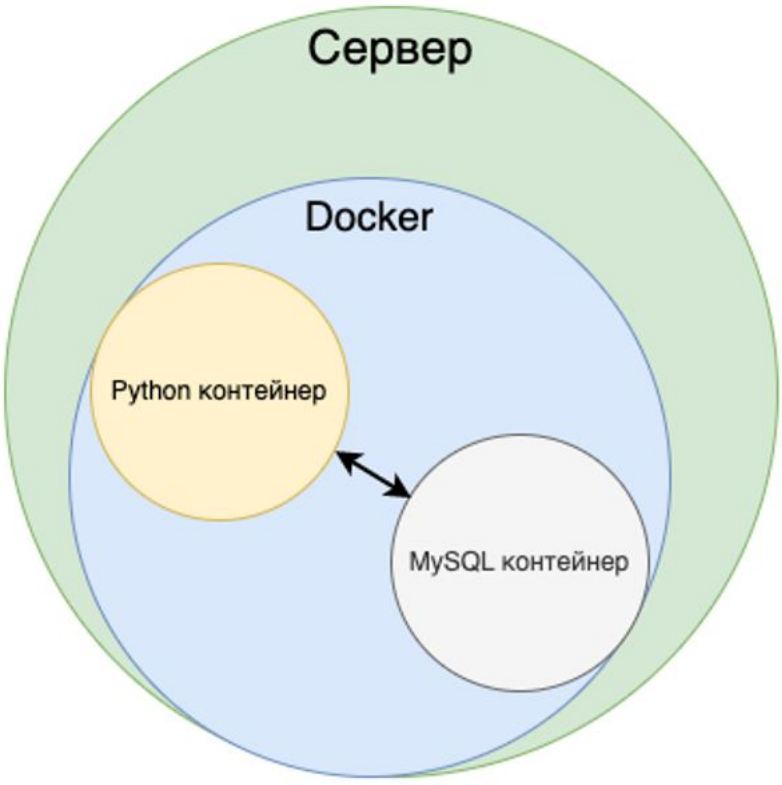


Рисунок 3.1 – Структура системи з MySQL як docker контейнер

Контейнери обмінюються інформацією про логи в режимі реального часу. Це дозволяє аналізувати логи неперервним потоком.

Ця структура системи дозволяє розподілити функціональність та ресурси між різними компонентами, забезпечуючи гнучкість, масштабованість та ефективність роботи.

За замовчуванням, MySQL визначений, як docker контейнер, проте можливі різні варіації розташування СУБД.

Конфігурація з СУБД, яка встановлена на сервері, зображено на рисунку 3.2.

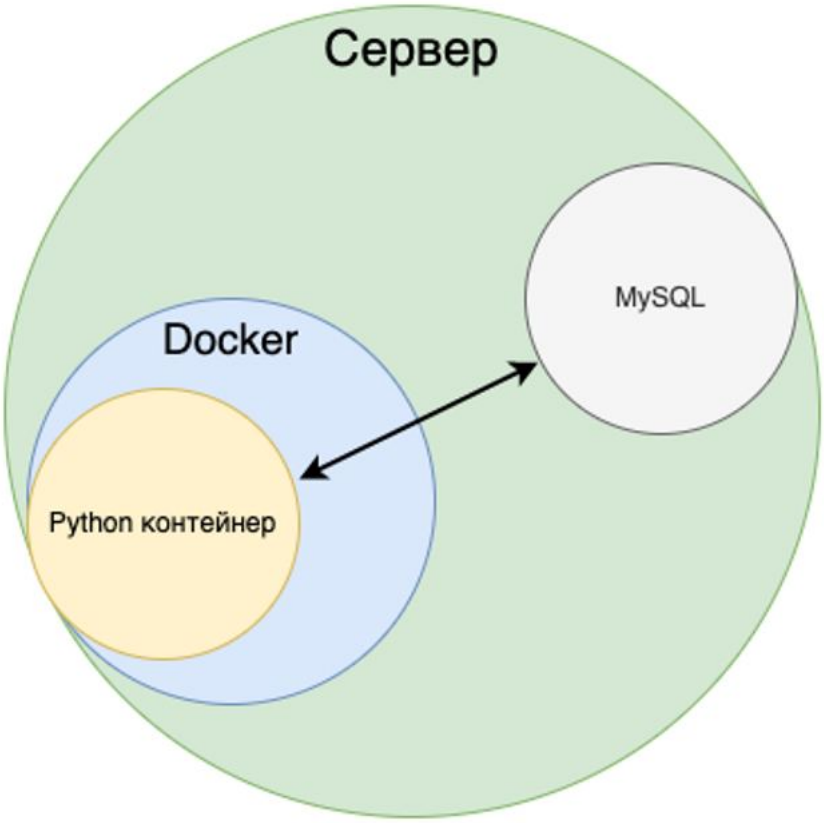


Рисунок 3.2 – Структура системи з MySQL встановленим на сервері

Для налаштування цієї структури, потрібно тільки застосувати конфігураційний файл, який зображений на рисунку 2.1, до СУБД.

Ця конфігурація дозволяє інтегрувати проект в системи, які вже працюють певний час і відрізняються від початкової конфігурації проекта.

Наприклад, існує магазин з продажу різних побутових товарів. Він використовує MySQL як основну базу даних для сайту. Вона встановлена на сервері. Для того, щоб інтегрувати захист від витоку даних не потрібно робити

додаткових маніпуляцій з СУБД і даними, які в ній зберігаються. Достатньо, як було вказано вище, додати конфігураційний файл до вже існуючого MySQL.

MySQL може бути встановлений не тільки на сервері або засобами докера. Існують рішення, при яких СУБД знаходиться на віддаленому сервері, зображено на рисунку 3.3. Для цієї конфігурації системи потрібно виконати додаткові кроки.

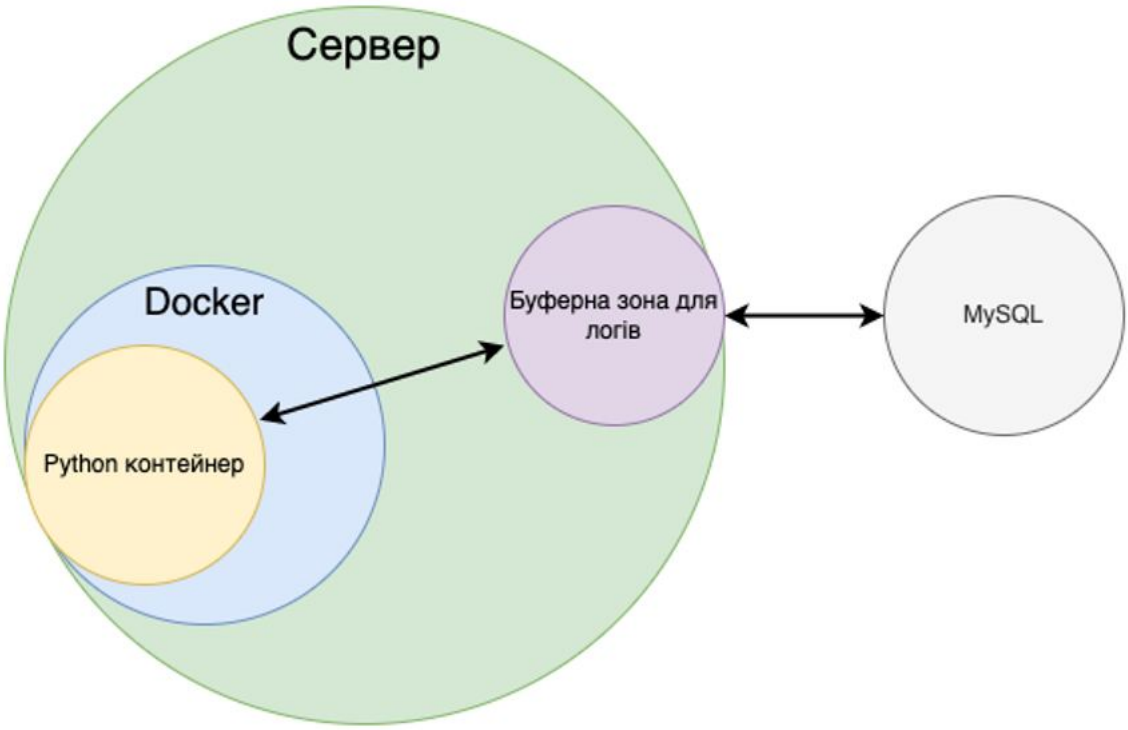


Рисунок 3.3 – Структура системи з MySQL, встановленим на віддаленому сервері

Якщо взяти попередній приклад роботи сайту з MySQL, то можна побачити, що для коректної роботи доведеться додати функціонал, який буде синхронізувати логи з віддаленого сервера на локальний.

Наразі, системою не передбачені рішень, які б могли допомогти в цій ситуації через те, що віддалені сервера – різні. Кожен сервер може по різному налаштовуватися і це є неможливим реалізувати як частину проекту. Тому цю роботу перекладено на спеціалістів клієнта.

Клієнтом, в цьому контексті, є суб’єкт, який інтегрує систему захисту від витоку даних на своїх проектах.

Ще одним невід'ємним компонентом структури є cron. Cron – це стандартний планувальник завдань в операційних системах Unix та Unix-подібних системах. Він дозволяє автоматично запускати програми або скрипти на заданих інтервалах часу або з певною регулярністю [40].

Крон працює на основі конфігураційного файлу, який містить список завдань та їх параметри. Приклад конфігураційного файлу зображений на рисунку 3.4. Кожне завдання описується за допомогою спеціального синтаксису, який визначає час виконання, команду або скрипт, який має бути виконаний.

```
# 1.0.0 - Start of the leak detector crontab config
* * * * * /usr/bin/env python3 /var/www/html/main.py >> /var/www/html/var/cron.log 2>&1
# 1.0.0 - End of the leak detector crontab config
```

Рисунок 3.4 – Конфігурація для cron

Ця утиліта забезпечить безперервну роботу ШІ. За замовчуванням, виконання встановлено на 5 зірок.

Кожна зірка означає певну одиницю часу: хвилини, години, день місяця, місяць та рік. Зірка ж означає, що команда буде запускатися в усі проміжки часу, тому 5 зірок означає, що ШІ буде запускатися кожної хвилини. Ця конфігурація дозволить працювати системі безперебійно в режимі реального часу і не потребує виконання вручну. Також, результат записується в cron.log файл, в якому можна відслідковувати роботу ШІ та його рішення. Це є звичайним рішенням під час використання крона адже він не виводить ніякої інформації в консоль.

3.2 Аналіз результатів та оцінка ефективності системи

В цьому підпункті будуть представлені результати роботи програми та їх аналіз. Для цього розглянемо отримані дані та виконаємо їх візуалізацію для кращого розуміння.

Перш за все, переглянемо основні числові показники результатів, такі як точність, чутливість, специфічність тощо. Порівняємо ці показники зі сподіваними значеннями та розглянемо, наскільки успішно програма впоралася з поставленими завданнями.

Далі, для більш детального аналізу, побудуємо графіки, що відображатимуть залежності між вхідними даними та результатами програми. Наприклад, можемо побудувати графік залежності точності від кількості ітерацій або графік, де порівняємо різні алгоритми або параметри програми.

За результатами навчання модель була навчена на штучно згенерованому датасеті. Датасет містить в собі по 2500 позитивних та негативних випадків.

Дивлячись на рисунок 3.5 з таблицею, можна зрозуміти, що модель складається з трьох шарів та має певну кількість параметрів.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 200, 32)           976448

 lstm (LSTM)                 (None, 64)                24832

 dense (Dense)               (None, 1)                 65

=================================================================
Total params: 1,001,345
Trainable params: 1,001,345
Non-trainable params: 0
_________________________________________________________________
```

Рисунок 3.5 – Таблиця з інформацією про структуру моделі

Кожен шар має свій розмір вихідних даних. Варто відзначити, що позначення None тут означає довільну кількість прикладів.

Протягом навчання модель показувала різні метрики для кожної епохи, які зображені на рисунку 3.6.

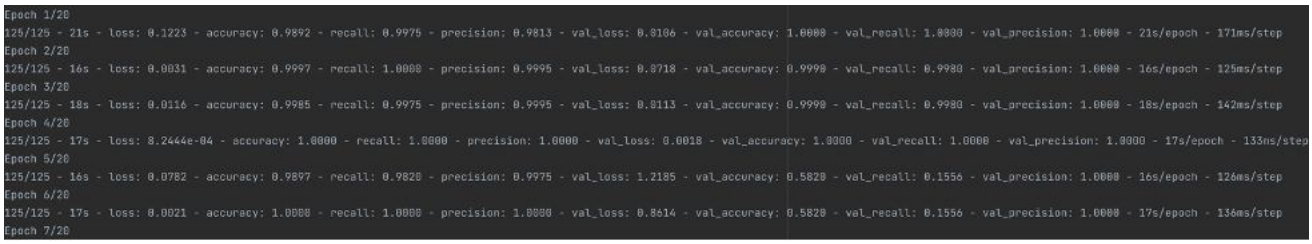


Рисунок 3.6 – Результати навчання моделі відображені в метриках

До цих метрик належать:

– loss (втрата) – це числова величина, яка вимірює, наскільки добре модель прогнозує значення вихідної змінної порівняно з правильними значеннями. Оптимізація цієї величини під час навчання моделі є головною метою;

– accuracy (точність) – це відсоток правильних прогнозів моделі відносно загальної кількості прикладів у наборі даних. Вона вимірює, наскільки добре модель класифікує дані. Чим вища точність, тим краще модель виконує свою задачу;

– val_loss (втрата на валідаційному наборі) – це значення втрати, обчислене на валідаційному наборі даних. Використовується для оцінки продуктивності моделі на незалежних від тренувального наборі даних. Це допомагає виявити перенавчання або недонавчання моделі;

– val_accuracy (точність на валідаційному наборі) – це значення точності, обчислене на валідаційному наборі даних. Воно показує, наскільки добре модель класифікує дані на валідаційному наборі. Використовується для оцінки загальної продуктивності моделі;

– recall, також відомий як чутливість або true positive rate, є метрикою, яка вимірює відношення правильно класифікованих позитивних прикладів до всіх справжніх позитивних прикладів;

– precision є метрикою, яка вимірює точність класифікації моделі на позитивних прикладах. Вона показує, який відсоток з прикладів, визначених моделлю як позитивні, є дійсно правильними [41].

Хоч і наведені на рисунку 3.6 значення були видані, як результат навчання, програмою, їх можна перевірити за допомогою формули.

Спершу розглянемо функцію витрат.

Для навчання моделі використовувалась функція BinaryCrossentropy, зображена у формулі 3.1.

$$loss = -(y \cdot \log(y_{hat}) + (1 - y) \cdot \log(1 - y_{hat})), \quad (3.1)$$

де y – правильне значення (0 або 1), y_{hat} – передбачене значення моделі, що лежить в діапазоні [0, 1].

Так, як y має 2 значення: 0 та 1, функцію можна розділити на 2.

При $y = 1$:

$$loss = -\log(y_{hat}).$$

При $y = 0$:

$$loss = -\log(1 - y_{hat}).$$

Беручи до уваги, що правильно відповіддю в датасеті є 1, а неправильною – 0. То Використаєм першу формулу. Зробимо розрахунок для першої епохи. Беручи до уваги, що значення y_{hat} еквівалентне 0.8848, результатом розрахунків буде $loss = 0.1223$. Також, зробимо розрахунок точності (accuracy) за допомогою формули 3.2. Формула виглядає наступним чином:

$$accuracy = \frac{n}{N}, \quad (3.2)$$

де n – кількість правильно вказаних відповідей, N – загальна кількість спроб.

Розрахуємо точність для 1 епохи тренувальної вибірки. Всього в датасеті 5000 позитивних та негативних сценаріїв, 80% з них – це тренувальна вибірка. Тобто,

$$N = 5000 \cdot \frac{80}{100} = 4000.$$

Усього правильних відповідей модель зробила 3957, тому:

$$accuracy = \frac{3957}{4000} = 0.9892.$$

Як результат, можна побачити, що перерахувавши, значення співпадають з наведеними на рисунку 3.6.

Аналогічно можна порахувати значення для val_loss та val_accuracy. Формули для них не відрізняються від попередніх, лише відмінним є вибірка.

Для цих значень використовується валідаційна вибірка, де

$$N = 5000 \cdot \frac{10}{100} = 500.$$

Тому,

$$val\_loss = -\log(0.9894) = 0.0106.$$

Для точності,

$$val\_accuracy = \frac{500}{500} = 1.$$

Так, як модель вказала всі відповіді правильно для валідаційної вибірки, точність еквівалентна 1.

Precision обчислюється як співвідношення кількості правильно класифікованих позитивних об'єктів до загальної кількості об'єктів, які модель визначила як позитивні, за формулою 3.3.

Формула для precision виглядає наступним чином:

$$precision = \frac{TP}{(TP + FP)}, \quad (3.3)$$

де TP – це кількість правильно класифікованих позитивних прикладів, FP – кількість негативних прикладів.

Для першої епохи $TP = 3925$, а $FP = 75$, то

$$precision = \frac{3925}{3925+75} = 0.98125.$$

Значення val_precision розраховується за тією ж формулою.

$$\text{val\_}precision = \frac{500}{500+0} = 1.$$

Recall обчислюється як співвідношення кількості правильно класифікованих позитивних об'єктів до загальної кількості дійсно позитивних об'єктів у датасеті, описано в формулі 3.4. Формула для recall має вигляд:

$$\text{recall} = \frac{\text{TP}}{(\text{TP} + \text{FN})}, \quad (3.4)$$

де TP – кількість правильно класифікованих позитивних прикладів, а FN – кількість неправильно класифікованих негативних прикладів, які були помилково визнані моделлю як позитивні

Використовуючи цю формулу, розрахуємо значення для recall та val_recall. Для recall значення TP еквівалентне 3990, а FN = 10, тому

$$recall = \frac{3990}{3990+10} = 0.9975.$$

Для val_recall: TP = 4000, а FN = 0, тому

$$\text{val}\_recall = \frac{4000}{4000+0} = 1.$$

Аналізуючи результати, які були отримані в результаті обчислення рівнянь, можна зробити висновок, що вони збігаються з метриками, які були розраховані моделлю під час навчання. Порівняння значень моделі та розраховані за формулами представлені в таблицях 3.1 та 3.2.

Таблиця 3.1 – Аналіз результатів для тренувальної вибірки

Джерело	loss	accuracy	recall	precision
З моделі	0.1223	0.9892	0.9975	0.9813
З розрахунків	0.1223	0.9892	0.9975	0.9813

Таблиця 3.2 – Аналіз результатів для валідаційної вибірки

Джерело	val_loss	val_accuracy	val_recall	val_precision
З моделі	0.0106	1	1	1
З розрахунків	0.0106	1	1	1

Беручи до уваги, що задача моделі – визначати чи текст належить до позитивного або негативного сценарію, важливим є значення точності. Подивившись на графік можна зробити висновок, що модель була навчена до майже ідеального стану точності. Це означає, що модель дуже точно класифікує дані.

Точність 1 означає, що всі прогнози моделі є правильними, відповідаючи справжнім міткам класу для всіх прикладів даних. Це є найвищим рівнем точності, який може бути досягнутий моделлю.

Графік порівняння accuracy та val_accuracy поданий на рисунку 3.7.

Таке високе значення accuracy та val_accuracy свідчить про дуже ефективну та точну модель, яка добре розуміє взаємозв'язки в даних та правильно класифікує їх.

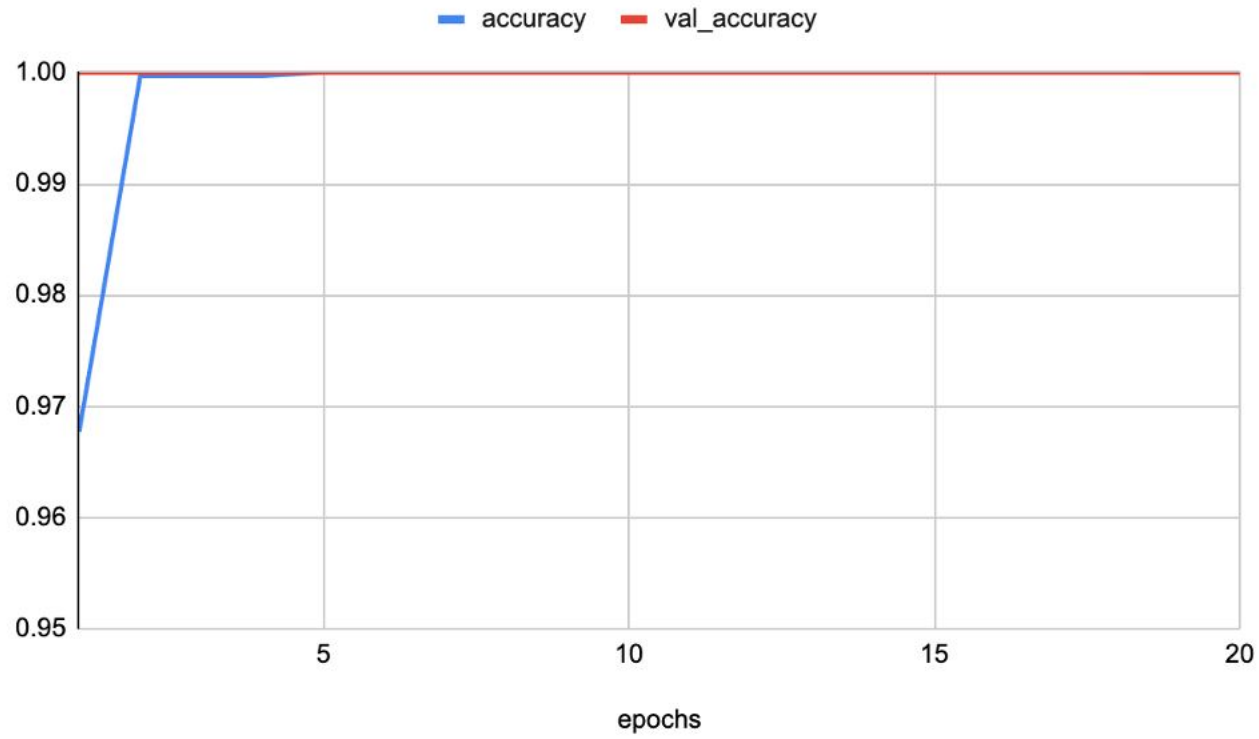


Рисунок 3.7 – Графік співвідношення accuracy та val_accuracy до епох

Порівнюючи значення loss і val_loss, можна отримати важливу інформацію про ефективність моделі під час тренування і валідації. loss відображає точність моделі на тренувальних даних. З іншого боку, val_loss вимірює точність моделі на валідаційних даних, які не використовуються під час тренування.

Подивившись на рисунок 3.8 з графіком можна побачити, що значення помилок є дуже малими. Це вказує на те, що модель досягла низького рівня помилки або втрати під час навчання та валідації. Це означає, що модель добре узгоджується з тренувальними та валідаційними даними, і її прогнози акуратно підходять до справжніх значень.

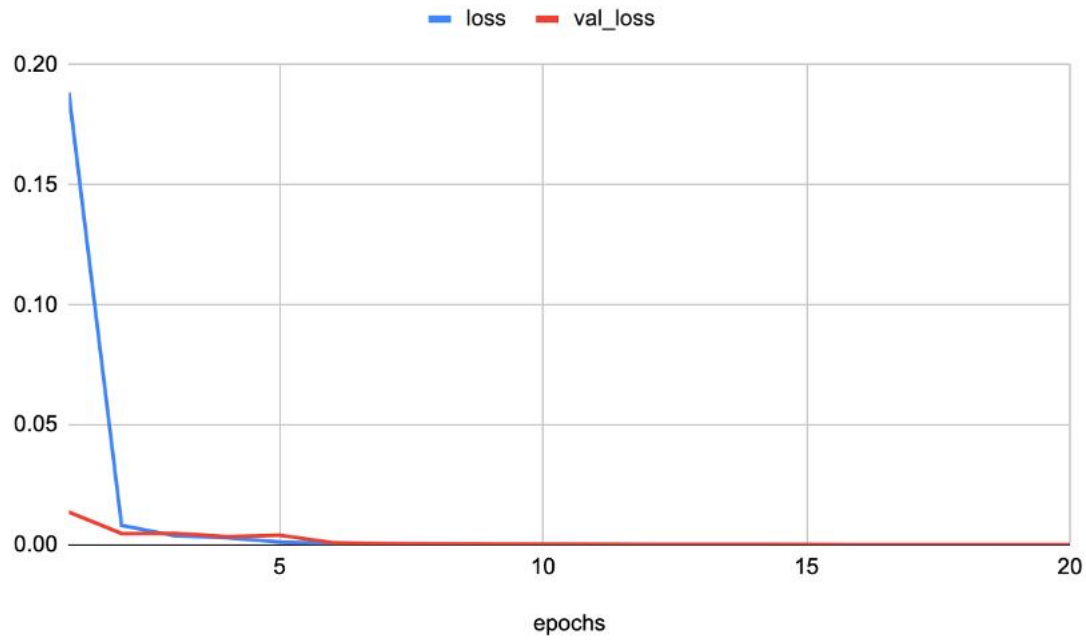


Рисунок 3.8 – Графік співвідношення loss та val_loss до епох

Малі значення loss та val_loss свідчать про те, що модель ефективно вчиться знаходити корисні закономірності та шаблони в даних. Вона здатна точно прогнозувати або класифікувати дані, з мінімальною помилкою або втратою.

3.3 Тестування та валідація результатів

Беручи до уваги, що система призначена для захисту від витоку даних, потрібно симулювати спершу атаку. Ця атака має бути націлена на MySQL та дані, які там знаходяться.

Для тестового випадку, було створено простий скрипт, який робить дамп однієї або багатьох баз даних, наведений на рисунку 3.9.

Цей код є доволі простий та слугує тільки для тестування.

В файлі знаходиться 3 рядка команд:

1. Підключення до сервера за допомогою ssh;

2. Створення дампу;

3. Викачка дампу на інший сервер.

```
#!/bin/sh
# connect to server
ssh ubuntu@89.207.132.170
# make dump
docker exec -i first_ai_mysql mysqldump -uroot -pmagento2 --all-databases > dump.sql
# download dump
scp dump.sql ubuntu@89.207.132.170:/home/data
```

Рисунок 3.9 – Код скрипта

Важливо підкреслити, що цей скрипт використовувався тільки в контрольованому локальному середовищі.

Варто відзначити, що в цей скрипт адаптований на систему, яка використовує Docker для роботи MySQL. Схема цієї структури зображена на рисунку 3.1, яка є за замовчуванням та використовувалась для розробки проекту.

Наступна схема, на рисунку 3.10, відображає покрокові дії тестового інциденту.

Рисунок 3.10 – Схема виконання тесту

Вона включає в себе 4 кроки:

1. Застосування скрипту;

2. Аналіз логів системою та визначення передбачення за допомогою ШІ;

3. Відправлення електронного листа на пошту адміністратора для його реагування на інцидент;

4. Власне, певні дії адміністратора, які націлені на захист даних. До цих дій можна віднести: відключення несанкціонованого підключення до сервера, переривання роботи процесу скрипта або виключення сервера для збереження даних.

Результатом роботи ШІ є виведення в консоль інформації:

– про дані, які були проаналізовано, їхню кількість та час затрачений на визначення передбачення;

– про те, чи був відправлений лист на пошту.

На рисунку 3.11 зображена консоль після роботи системи з інформацією визначеною в попередньому списку.

```
root@5ce878194501:/var/www/html# python main.py
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
Start analysis
Data size: 2
1/1 [==============================] - 0s 437ms/step
1/1 [==============================] - 0s 37ms/step
Email sent with error
Finished
root@5ce878194501:/var/www/html#
```

Рисунок 3.11 – Результат роботи ШІ

Також варто відзначити швидку роботу ШІ з визначенням відповіді. В загальному на один аналіз йде від 30 до 50 мілісекунд, проте бувають випадки,

коли час може збільшуватися до 500 мілісекунд. Це збільшення пов’язане здебільшого з великою кількістю тексту, яку потрібно обробити.

На рисунку 3.12 можна побачити швидкість виконання кожної ітерації при аналізі великої кількості тексту.

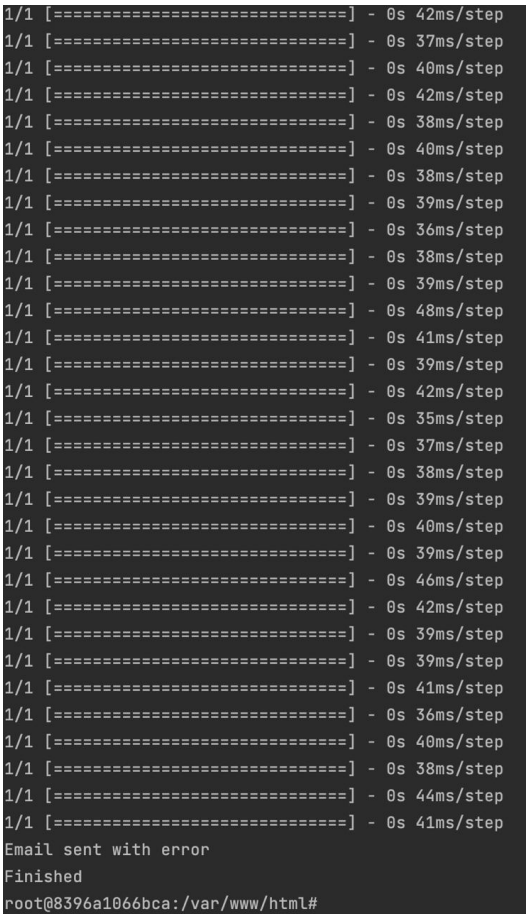


Рисунок 3.12 – Швидкість аналізу кожного запису

Відразу після виконання шкідливого скрипту, ШІ проаналізував логи в режимі реального часу за допомогою cron. Як результат виконання аналізу, був надісланий email на пошту адміністратору.

Вміст листа представлений на рисунку 3.13. Лист включає повідомлення про те, що ШІ виявив стороннє втручання до СУБД MySQL та логи, які аналізувалися для прийняття подальших дій адміністратором сервера.

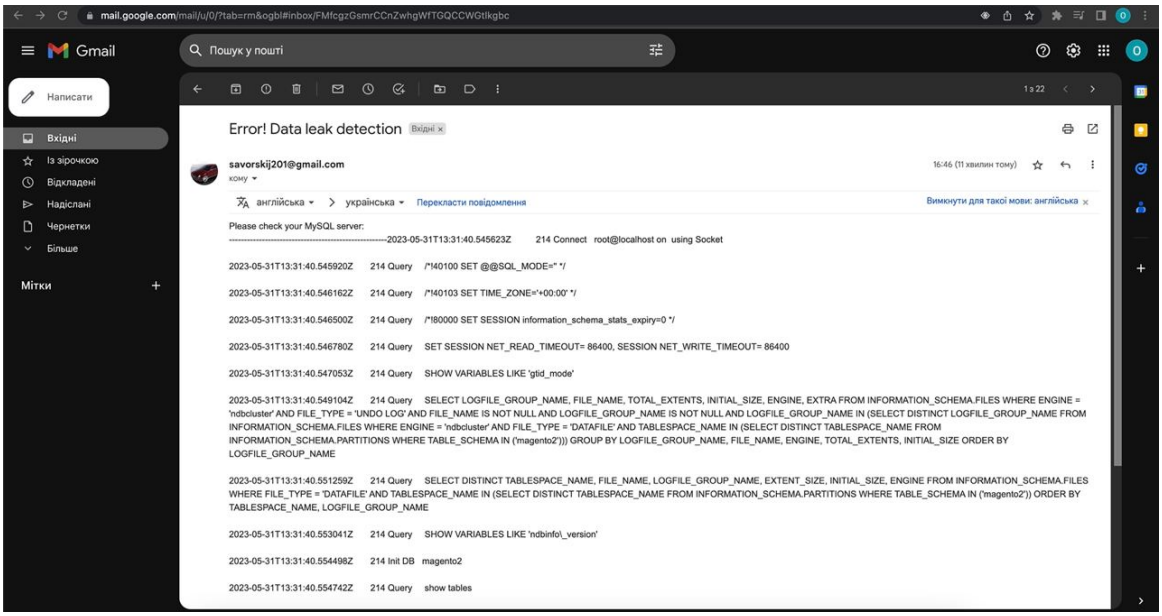


Рисунок 3.13 – Зображення листа, який прийшов на електронну пошту

3.4 Висновки

У цьому розділі було проведено аналіз результатів навчання ШІ та протестовано функціонал системи з захисту MySQL від витоку даних.

Перш за все, навчання ШІ за допомогою бібліотеки Keras дало можливість створити моделі нейронних мереж, які виявили високу точність в завданнях класифікації. Було проаналізовано результати навчання та валідації моделей, оцінено їх метрики точності, втрати та швидкодії. Це підтвердило ефективність використання ШІ в системі для розпізнавання та аналізу даних.

Метрики, такі як loss, accuracy, recall і precision, є важливими інструментами для оцінки ефективності моделей машинного навчання і класифікації.

Кожна з цих метрик відображає різні аспекти ефективності моделі. При оцінці моделі важливо звертати увагу на кожну з цих метрик, оскільки вони можуть мати різні значення в залежності від контексту задачі.

Розрахунки цих метрик здійснюються на основі кількості правильно класифікованих та помилкових результатів, використовуючи формули, що враховують розподіл прикладів по категоріям.

Для повної оцінки ефективності моделі рекомендується розглядати кілька метрик одночасно, а також здійснювати порівняння з іншими моделями і базовими рівнями ефективності для отримання об'єктивного уявлення про її продуктивність.

Далі, було перевірено функціонал системи з захисту MySQL від витоку даних. Тестування системи є важливою частиною розробки програмного забезпечення, включаючи системи кібербезпеки. Його метою є перевірка функціональності, надійності, безпеки та продуктивності системи перед її впровадженням в реальні умови.

Застосування шкідливих скриптів для емуляції витоку даних дозволило оцінити ефективність захисних механізмів та виявити можливі вразливості. Систему було протестовано за допомогою скрипта та оцінено її здатність виявляти незаконний доступ до даних. Отримані результати дозволили визначити сильні та слабкі сторони системи з захисту та прийняти відповідні заходи для поліпшення безпеки даних.

Загалом, використання ШІ в системі баз даних MySQL може покращити її функціональність та безпеку. Аналіз результатів навчання ШІ та тестування функціоналу системи з захисту підтвердив ефективність використання цих технологій у проекті.

Використання штучного інтелекту в системах захисту від витоку даних дозволяє підвищити рівень безпеки, забезпечити швидку відповідь на потенційні загрози та знизити ризик витоку чутливої інформації. Враховуючи постійний розвиток технологій ШІ, його використання стає все більш важливим у сучасних системах захисту даних.

ВИСНОВКИ

У даній роботі метою було створити систему запобігання витоку даних за допомогою ШІ та оцінити її ефективність. Для досягнення цієї мети було проведено детальне дослідження витоків даних, існуючих рішень та доступних датасетів. Далі, було розроблено процес генерації власного датасету, навчено модель ШІ та використане відповідне середовище розробки для реалізації цього процесу. Нарешті, було досліджено структуру проекту, включаючи роль і розташування MySQL, проаналізовано метрики результатів та протестовано систему на ефективність її захисту.

В розділі 1, було досліджено витоки даних, класифікацію та захист від них. Було проведено дослідження різних датасетів, які можуть бути використані для тренування моделі, і виявлено їх переваги та недоліки. Аналізуючи існуючі рішення та доступні датасети, з'ясувалося, що наявні системи та підходи не гарантують повної безпеки від витоку даних, тому мета проекту полягала в створенні системи, яка могла б це забезпечити.

У розділі 2, було описано процес генерації власного датасету, навчання моделі та використання Python для розробки та експериментів. Використання ШІ за допомогою бібліотеки Keras дозволило створити потужну модель, яка показала високу точність та надійність в класифікації даних. Було розглянуто різні аспекти побудови датасету, включаючи збір та обробку даних. Також, було проведено навчання моделі на згенерованому датасеті та оцінено її результати.

В розділі 3, було розглянуто структуру проекту, зосередившись на ролі та розташуванні MySQL. Аналізуючи метрики результатів, такі як точність, відтворюваність та швидкодія, було оцінено ефективність системи та виявлено можливі вразливості. Було проведено тестування функціоналу системи з захисту MySQL від витоку даних. Були використані реальні дані та сценарії, щоб перевірити ефективність системи. Було оцінено, наскільки система виявляє можливі витоки даних, а також наскільки точно вона класифікує ці витоки.

Загальний результат роботи свідчить про високий потенціал системи у запобіганні витоку даних та забезпеченні безпеки бази даних MySQL. Впровадження цього проекту може мати значний вплив на сучасні сценарії захисту даних, сприяючи забезпеченню конфіденційності та цілісності інформації.

Система може використовуватися в різних областях, де безпека даних є важливим аспектом, таких як банківський сектор, медична сфера, електронна комерція та багато інших. Цей проект виступає важливою основою для подальшого розвитку та вдосконалення систем захисту даних.

Враховуючи швидкий розвиток технологій та зростання кількості загроз в сфері кібербезпеки, система може бути ефективним інструментом для запобігання витоку даних та підвищення рівня безпеки. Це дозволить організаціям максимально захистити свою конфіденційну інформацію та зберегти довіру своїх клієнтів.

Отже, висновки роботи підтверджують, що розроблена система має великий потенціал для ефективного запобігання витоку даних і забезпечення безпеки бази даних MySQL. Впровадження цього проекту в реальних сценаріях може стати важливим кроком у покращенні захисту даних і сприяти збереженню конфіденційності в інформаційній епосі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Davidoff S. Data Breaches: Crisis and Opportunity. Pearson Education, Limited, 2019. 464 p.

2. Information technology – Security techniques – Storage security. URL: https://www.iso.org/obp/ui/es/#iso:std:iso-iec:27040:ed-1:v1:en (дата звернення: 05.03.2023).

3. Що таке кібератака?. URL: https://www.microsoft.com/uk-ua/security/business/security-101/what-is-a-cyberattack (дата звернення: 15.03.2023).

4. Kontrolle und Disziplinen або як позбавитися інсайдерських загроз. URL: https://it-solutions.ua/blog/kontrolle-und-disziplinen-abo-yak-pozbavitisya-insajderskih-zagroz/ (дата звернення: 11.03.2023).

5. Витік даних чи злам – в чому різниця. URL: https://10guards.com/ua/articles/whats-the-difference-between-a-data-leak-and-a-data-breach/ (дата звернення: 12.03.2023).

6. Access management system. URL: https://www.solarwinds.com/access-rights-manager/access-management-system (date of access: 14.03.2023).

7. Computer security resource center. URL: https://csrc.nist.gov/glossary/term/security_audit (date of access: 16.03.2023).

8. Newman R. C. Computer security: protecting digital resources. Sudbury, Mass: Jones and Bartlett Publishers, 2009. 122 p.

9. EC-Council. Computer forensics: investigating network intrusions and cybercrime , 2nd edition. Cengage Learning, 2016. P. 87-88.

10. What is data loss prevention?. URL: https://www.digitalguardian.com/blog/what-data-loss-prevention-dlp-definition-data-loss-prevention (date of access: 18.03.2023).

11. Josang A. A consistent definition of authorization, proceedings of the 13th international workshop on security and trust management. 2019. 56 p.

12. Hoy M. B. Alexa, siri, cortana, and more: an introduction to voice assistants. medical reference services quarterly. 2018. 12p.

13. Moroney L. AI and Machine Learning for Coders: A Programmer's Guide to Artificial Intelligence. O'Reilly Media, Incorporated, 2020. 300 p.

14. What is artificial intelligence in medicine?. URL: https://www.ibm.com/topics/artificial-intelligence-medicine (date of access: 20.03.2023).

15. What is natural language processing?. URL: https://www.ibm.com/topics/natural-language-processing (date of access: 23.03.2023).

16. Darktrace. Darktrace cyber-AI analyst: autonomous investigations. 2022. 5p.

17. IBM watson. URL: https://www.ibm.com/watson (date of access: 26.03.2023).

18. Oltsik J. FireEye myth and reality. 2018. 112 p.

19. Collins M. Network Security Through Data Analysis: From Data to Action. O'Reilly Media, 2017. 428 p.

20. Bishop C. M. Pattern recognition and machine learning. Springer, 2016. 72 p.

21. What is the difference between test and validation datasets?. URL: https://machinelearningmastery.com/difference-test-validation-datasets/ (date of access: 29.03.2023).

22. Matthias K., Kane S. P. Docker: Up & Running: Shipping Reliable Containers in Production. O'Reilly Media, 2018. 352 p.

23. Docker compose docs. URL: https://docs.docker.com/compose/ (date of access: 1.04.2023).

24. Grippa V. M., Kuzmichev S. Learning MySQL: Get a Handle on Your Data. O'Reilly Media, Incorporated, 2021. 550 p.

25. Docker hub. URL: https://www.docker.com/products/docker-hub/ (date of access: 5.04.2023).

26. The general query log. URL: https://dev.mysql.com/doc/refman/8.0/en/query-log.html (date of access: 5.04.2023).

27. Log Monitoring: not the ugly sister. URL: https://pandorafms.com/blog/log-monitoring/ (date of access: 7.04.2023).

28. Introduction to TensorFlow. URL: https://www.tensorflow.org/learn (date of

access: 8.04.2023).

29. The Sequential model. URL: https://keras.io/guides/sequential_model/ (date of access: 09.04.2023).

30. Aggarwal C. C. Neural Networks and Deep Learning. Cham : Springer International Publishing, 2018. URL: https://doi.org/10.1007/978-3-319-94463-0 (date of access: 04.06.2023).

31. Müller A. C., Guido S. Introduction to Machine Learning with Python: A Guide for Data Scientists. O'Reilly Media, Incorporated, 2018. 394 p.

32. Embedding. URL: https://www.tensorflow.org/api_docs/python/tf/keras/layers/Embedding (date of access: 11.04.2023).

33. LSTM. URL: https://www.tensorflow.org/api_docs/python/tf/keras/layers/LSTM (date of access: 13.04.2023).

34. Dense. URL: https://www.tensorflow.org/api_docs/python/tf/keras/layers/Dense (date of access: 15.04.2023).

35. Buduma N. Fundamentals of Deep Learning: Designing Next-Generation Machine Intelligence Algorithms. O'Reilly Media, 2017. 298 p.

36. Intuition of adam optimizer. URL: https://www.geeksforgeeks.org/intuition-of-adam-optimizer/ (date of access: 19.04.2023).

37. HDF5. URL: https://www.loc.gov/preservation/digital/formats/fdd/fdd000229.shtml (date of access: 20.04.2023).

38. Tokenizer. URL: https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/text/Tokenizer (date of access: 21.04.2023).

39. Keras pad_sequences. URL: https://www.educba.com/keras-pad-sequences/ (date of access: 21.04.2023).

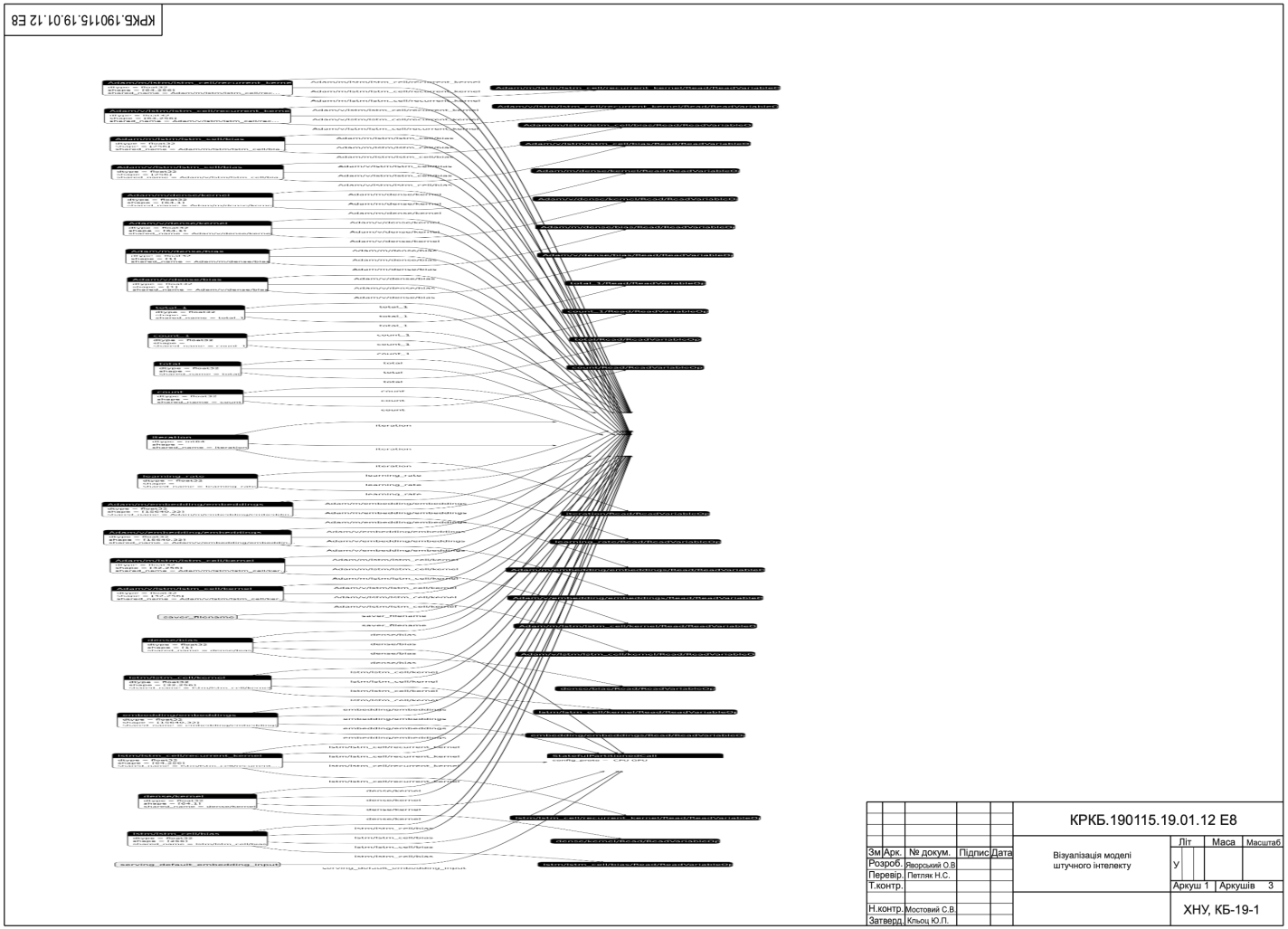
40. Cron job. URL: https://www.hostinger.com/tutorials/cron-job (date of access:

23.04.2023).

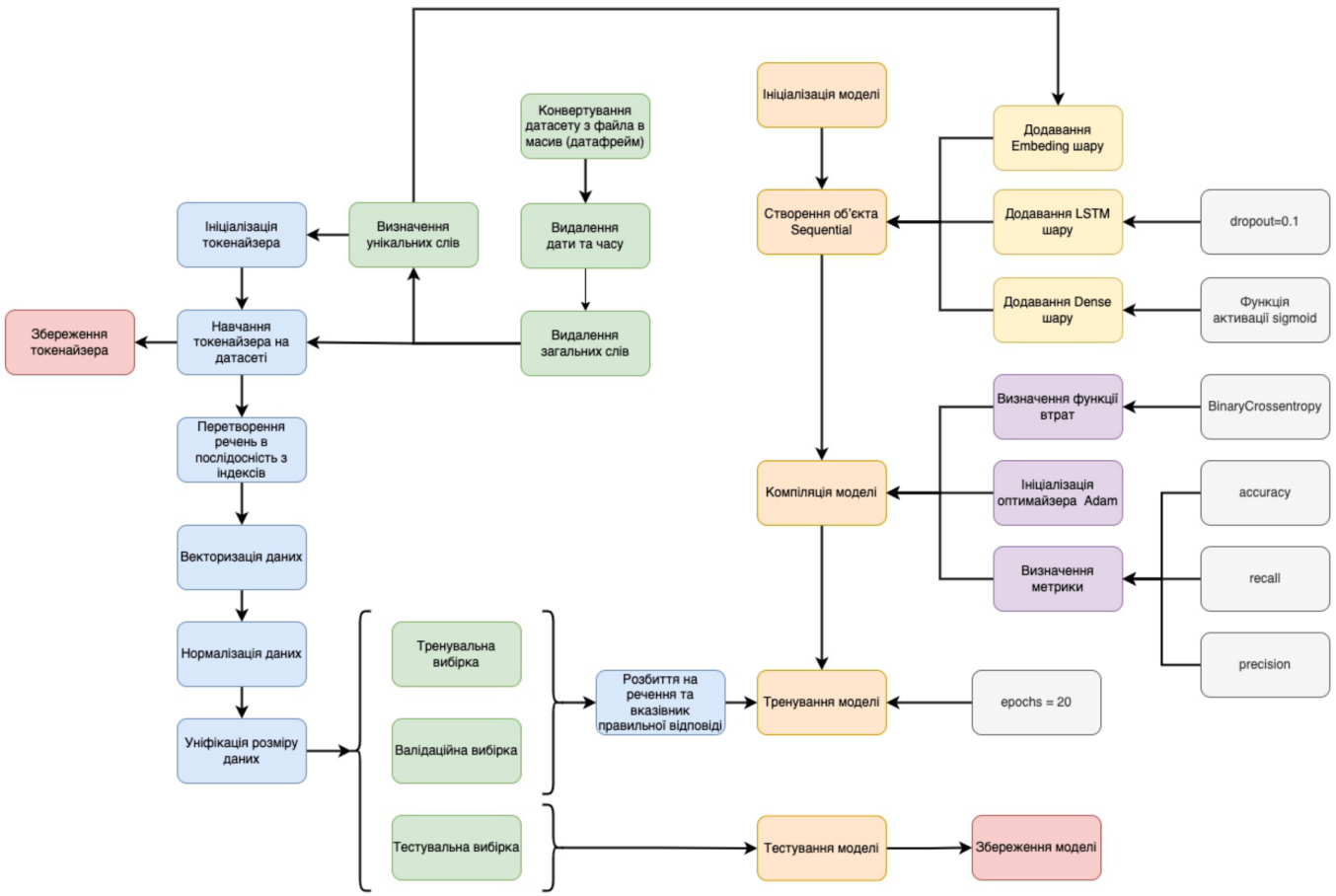
41. Precision and recall | essential metrics for data analysis. URL: https://www.analyticsvidhya.com/blog/2020/09/precision-recall-machine-learning/ (date of access: 28.04.2023).

ДОДАТОК А

Копія графічної частини

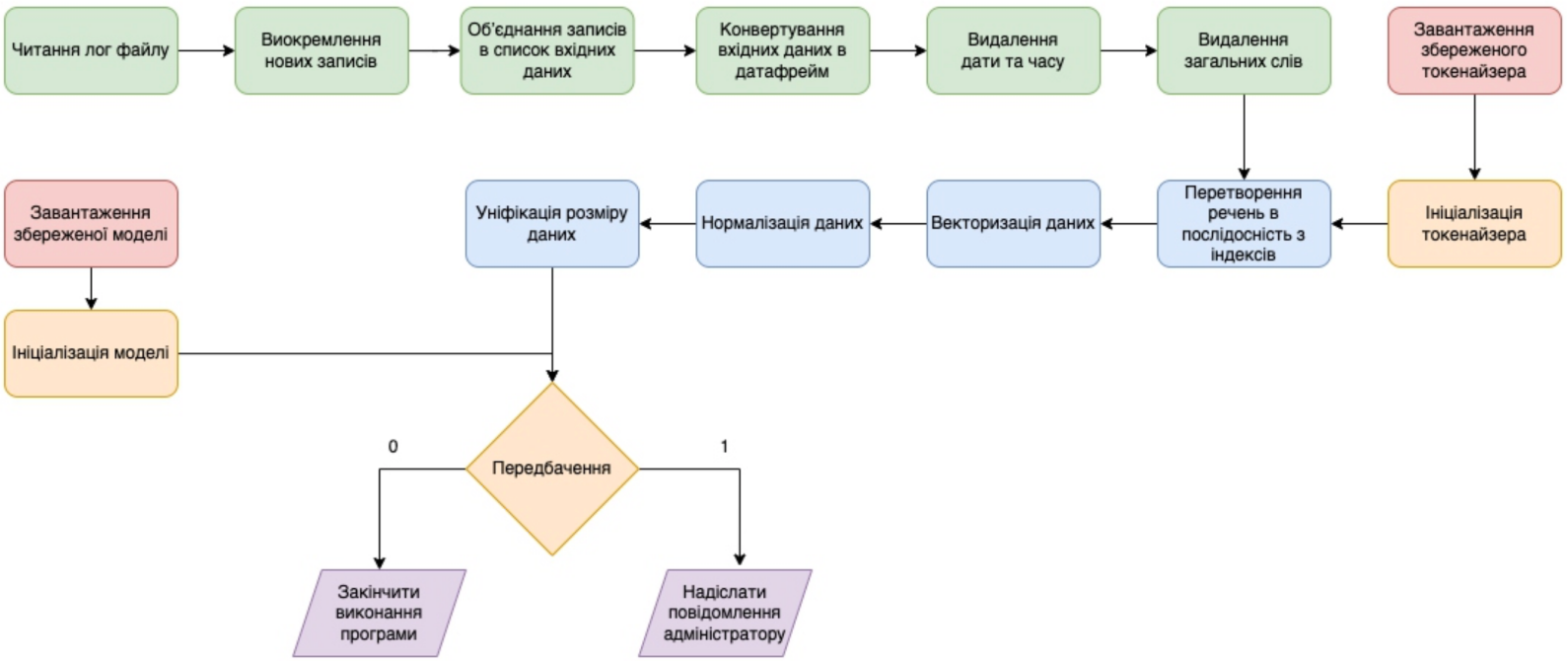


КРКБ.190115.19.01.12 Е8



						КРКБ.190115.19.01.12 Е8			
						Схема ініціалізації та навчання моделі та токенайзера	Літ	Маса	Масштаб
Зм	Арк.	№ докум.	Підпис	Дата			У		
Розроб.		Яворський О.В.							
Перевір.		Петляк Н.С.					Аркуш 2	Аркушів 3	
Т.контр.									
Н.контр.		Мостовий С.В.					ХНУ, КБ-19-1		
Затверд.		Кльоц Ю.П.							

КРКБ.190115.19.01.12 Е8



					КРКБ.190115.19.01.12 Е8		
Зм	Арк.	№ докум.	Підпис	Дата	Алгоритм роботи системи	Літ	Маса / Масштаб
Розроб.		Яворський О.В.				У	
Перевір.		Петляк Н.С.				Аркуш 3	Аркушів 3
Т.контр.							
Н.контр.		Мостовий С.В.				ХНУ, КБ-19-1	
Затверд.		Кльоц Ю.П.					

ДОДАТОК Б

Лістинг програмного коду

```
import sys
sys.path.append('train')
import os
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '1'

from train.transformer import Transformer
from train.tokenizer import LogTokenizer
from train.trainer import LogModel
from train.cleaner import Cleaner
from app.loader import Log
from app.mail_provider import Sender


def main():
    tokenizer = LogTokenizer().get()
    cleaner = Cleaner()
    model = LogModel(cleaner).get_model()
    transformer = Transformer(tokenizer)
    sender = Sender()
    log = Log()

    print("Start analysis")
    additional_text = ''
    data = log.get_data()
    print(f"Data size: {len(data)}")
    for text in data:
        data_to_analyze = transformer.transform(text)
        prediction = model.predict(data_to_analyze)
        if prediction > 0.9:
            additional_text += '-----------------------------------------------------' + text

    if additional_text != '':
        sender.send_error_email(additional_text)
        print("Email sent with error")

    print("Finished")

main()

class LogModel:
    def __init__(self, cleaner: Cleaner):
        # TODO file name could be with timestamp, change to parameter
        self.df = pd.read_csv('/var/www/html/var/mysql/train/merged/merged_logs.csv')
        self.cleaner = cleaner
        self.train_size = int(self.df.shape[0] * 0.8)
        self.max_length = 200
        self.model = None
```

```
def clean_df(self):
    self.df["text"] = self.df.text.map(self.cleaner.trim_datetime)
    self.df["text"] = self.df.text.map(self.cleaner.remove_stopwords)

def get_num_unique_words(self):
    count = Counter()
    for text in self.df.text.values:
        for word in text.split():
            count[word] += 1
    return len(count)

def get_train(self):
    # return tuple with sentences and labels
    train_df = self.df[:self.train_size]
    return train_df.text.to_numpy(), train_df.target.to_numpy()

def get_val(self):
    # return tuple with sentences and labels
    val_df = self.df[self.train_size:]
    return val_df.text.to_numpy(), val_df.target.to_numpy()

def get_model(self):
    if self.model is None:
        model = tf.keras.models.load_model('/var/www/html/var/model')
        if isinstance(model, keras.models.Sequential):
            self.model = model
        else:
            raise RuntimeError(
                'Model is not Sequential. Something went wrong when loading model. Please, re-train model'
            )

    return self.model

def init_model(self):
    # differentiate between train and validation
    train_sentences, train_labels = self.get_train()
    val_sentences, val_labels = self.get_val()

    # tokenizer training
    log_tokenizer = LogTokenizer(self.get_num_unique_words()).set_default_tokenizer()
    log_tokenizer.fit(train_sentences)

    # preprocessing data for training
    train_padded = log_tokenizer.get_padded(train_sentences, self.max_length)
    val_padded = log_tokenizer.get_padded(val_sentences, self.max_length)

    # model initialization
    model = keras.models.Sequential()
    model.add(layers.Embedding(self.get_num_unique_words(), 32, input_length=self.max_length))

    model.add(layers.LSTM(64, dropout=0.1))
    model.add(layers.Dense(1, activation="sigmoid"))
```

```
        loss = keras.losses.BinaryCrossentropy(from_logits=False)
        optim = keras.optimizers.Adam(lr=0.001)
        metrics = ["accuracy", tf.keras.metrics.Recall(), tf.keras.metrics.Precision()]

        model.compile(loss=loss, optimizer=optim, metrics=metrics)

        # model training
        model.fit(train_padded, train_labels, epochs=20, validation_data=(val_padded, val_labels), verbose=2)

        # model saving
        model.save('/var/www/html/var/model')

        return log_tokenizer, model

class Generator:
    def __init__(
            self,
            mysql_adaptor: adaptor.Mysql,
            mysql_logs: logs.MysqlLogs,
            faker: Faker,
            path_handler: path.PathHandler
    ):
        self.connection = mysql_adaptor
        self.logs_manager = mysql_logs
        self.faker = faker
        self.path_handler = path_handler

    def generate_positive_case(self, count: int):
        for i in range(count):
            database_name = self.faker.word() + '_' + datetime.now().strftime("%Y_%m_%d_%H_%M_%S")
            self.connection.create_database(database_name)

            last_line = self.logs_manager.get_line_count('/var/www/html/var/mysql/logs/mysql.log')

            # TODO remove or archive dump
            self.connection.create_dump(database_name, self.path_handler.dumps)

            self.logs_manager.create_train_log_from_line(last_line, 'positive/' + database_name + '.log')
            self.connection.drop_database(database_name)

        self.connection.disconnect()

    def generate_negative_case(self, count: int):
        lines = []
        files = os.listdir(self.path_handler.log)
        for file in files:
            with open(os.path.join(self.path_handler.log, file), 'r') as log:
                lines += log.readlines()

        file_max_lines = 13
        min = 0
        max = len(lines) - 1 - file_max_lines
        # TODO add continuous selection because random can select same line
```

```
        for i in range(count):
            start_line = random.randint(min, max)
            self.logs_manager.create_train_file(
                'negative/' + str(i) + '.log',
                lines[start_line:start_line + file_max_lines]
            )

    def convert_to_csv(self):
        positive_files = self.merge_array(self.path_handler.train_positive, 1)
        negative_files = self.merge_array(self.path_handler.train_negative, 0)
        data = positive_files + negative_files

        random.shuffle(data)
        now = datetime.now().strftime("%Y_%m_%d")
        with open(os.path.join(self.path_handler.train_merged, 'merged_logs' + '.csv'), 'w') as csvfile:
            csv_writer = csv.writer(csvfile)
            csv_writer.writerow(['target', 'text'])
            for row in data:
                csv_writer.writerow([row['target'], row['text']])

    def merge_array(self, path_to_files: str, target: int):
        result = []
        files = os.listdir(path_to_files)
        for file in files:
            with open(os.path.join(path_to_files, file), 'r') as log:
                lines = log.readlines()
            result += [{"text": ' '.join(lines), "target": target}]

        return result

class Sender:
    def __init__(self):
        self.email = os.environ.get('GMAIL_ADDRESS')
        self.password = os.environ.get('GMAIL_PASSWORD')
        self.admin_email = os.environ.get('ADMIN_EMAIL')

    def send_email(self, subject, body):
        with smtplib.SMTP('smtp.gmail.com', 587) as smtp:
            smtp.ehlo()
            smtp.starttls()
            smtp.ehlo()
            smtp.login(self.email, self.password)
            msg = f'Subject: {subject}\n\n{body}'
            smtp.sendmail(self.email, self.admin_email, msg)

    def send_error_email(self, additional_text: str):
        subject = 'Error! Data leak detection'
        body = 'Please check your MySQL server:\n' + additional_text
        self.send_email(subject, body)


        class LogTokenizer:

            def __init__(self, num_words=0):
```

```
    self.tokenizer = None
    self.num_words = num_words
    if os.path.exists('/var/www/html/var/model/tokenizer/') is False:
        os.makedirs('/var/www/html/var/model/tokenizer')
    self.tokenizer_path = '/var/www/html/var/model/tokenizer/tokenizer.pickle'

def fit(self, texts):
    self.tokenizer.fit_on_texts(texts)
    self.save_tokenizer()

def get_sequence(self, texts):
    return self.tokenizer.texts_to_sequences(texts)

def get_padded(self, texts, max_length=100):
    self.check()
    sequences = self.get_sequence(texts)
    return pad_sequences(
        sequences,
        maxlen=max_length,
        padding="post",
        truncating="post"
    )

def save_tokenizer(self):
    with open(self.tokenizer_path, 'wb') as handle:
        pickle.dump(self.tokenizer, handle, protocol=pickle.HIGHEST_PROTOCOL)

def load_tokenizer(self):
    if os.path.exists(self.tokenizer_path) is False:
        return False

    with open(self.tokenizer_path, 'rb') as handle:
        tokenizer = pickle.load(handle)

    if isinstance(tokenizer, Tokenizer) is False:
        return False
```

```
            return tokenizer

        def set_default_tokenizer(self):
            self.tokenizer = Tokenizer(self.num_words)
            return self

        def set_tokenizer(self):
            if self.tokenizer is None:
                self.tokenizer = self.load_tokenizer()

            if self.tokenizer is False:
                raise RuntimeError('Tokenizer not found in ' + self.tokenizer_path + ". Please, run model
training")
            return self.tokenizer

        def get(self):
            self.set_tokenizer()
            self.check()
            return self

        def check(self):
            if self.tokenizer is None:
    raise RuntimeError('Tokenizer is not defined. Try set_default_tokenizer() or set_tokenizer()')
```

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

освітньо-кваліфікаційного рівня «бакалавр»

Студент Яворський Олександр Віталійович
Тема: «Система запобігання витоку даних на основі штучного інтелекту»
Галузь знань 12 «*Інформаційні технології*» Спеціальність 125 «*Кібербезпека*» Освітня програма «Кібербезпека»

Обсяг дипломної роботи освітньо-кваліфікаційного рівня «бакалавр»: кількість листів креслень 3; кількість сторінок записки 63;

1. Короткий зміст КР та прийнятих рішень Кваліфікаційна робота присвячена дослідженню питань, пов'язаних з розробкою та впровадженням системи запобігання витоку динах на основі штучного інтелекту. Для досягнення цієї мети було проведено дослідження існуючих систем з витоку даних та моделей штучного інтелекту. Створено і розроблену таку систему, яка дозволяє виявляти витоки даних, використовуючи спроможність штучного інтелекту до навчання та передбачення. Робота має на меті допомогти підприємствам забезпечити безпеку своїх даних та інформації.

2. Висновок про відповідність КР завданню Кваліфікаційна робота у повній мірі відповідає поставленому завданню як в теоретичній так і у практичній частині роботи.

3. Характеристика виконання кожного розділу роботи, ступінь використання останніх досягнень науки і техніки і передових методів роботи: У вступі обґрунтовується актуальність теми роботи, її зв'язок з галуззю знань «Інформаційні технології» та спеціальністю «Кібербезпека», формулюється мета та основні завдання кваліфікаційної роботи. У першому розділі було проведено аналіз існуючих систем запобігання витоку даних, що дозволило виявити проблеми та завдання, що потребують вирішення, а також досліджено датасети, їх використання та види. У другому розділі було проаналізовано та вибрано інструменти для середовища розробки, створено алгоритм генерації датасету та навчання моделі штучного інтелекту. У третьому розділі наведено аналіз багатокомпонентної структури розробленої системи, розглянуто результати навчання моделі штучного інтелекту за допомогою згенерованого датасета, зроблено порівняльні розрахунки метрик моделі, проведено тестування розробленої системи.

4. Позитивні сторони кваліфікаційної роботи полягають у тому що, застосування розробленої системи запобігання витоку даних дозволило покращити безпеку даних, що зберігаються в базі даних. Гнучкість структури розробленої системи дозволяє інтегрувати її в проекти різної будови та складності. Штучний інтелект використовується для виявлення нестандартної поведінки, яка може бути визнана як витік даних, що дозволяє забезпечити більш ефективний механізм захисту. Використання штучного інтелекту дозволяє масштабувати функціонал системи за допомогою перенавчання моделі на доповненому датасеті. Узагальнюючи, можна зазначити, що реалізація системи запобігання витоку даних, заснованої на штучному інтелекті, була вдалою, і вона проявила свою ефективність у поліпшенні захисту даних в проектах різного рівня складності. Застосування відповідних технологій та процедур дозволило забезпечити підвищену інформаційну безпеку і зменшити ризики витоку даних. Реалізована система є надійним інструментом для захисту цінної інформації.

5. Негативні сторони проекту: -

6. Оцінка графічного оформлення та пояснювальної записки роботи. Графічне оформлення виконане відповідно до теми кваліфікаційної роботи із дотриманням усіх стандартів. У загальному графічне оформлення виконане на достатньому технічному рівні. Пояснювальна записка відповідає нормам для її оформлення та вимогам

7. Відгук про роботу в цілому В загальному кваліфікаційна робота заслуговує позитивної оцінки. Весь матеріал кваліфікаційної роботи структурований, чіткий та послідовний. Усі розділи роботи послідовні та логічні, що дозволяє чітко розуміти викладений матеріал в рамках тематики кваліфікаційної роботи. У пояснювальній записці багато наглядних пояснень. Графічний матеріал дозволяє наочно побачити доцільність та ефективність рішень, які були прийняті за основу для досягнення поставленої задачі проектування.

8. Інші зауваження -

9. Оцінка дипломної роботи Розглянувши позитивні та негативні сторони представленої кваліфікаційної роботи, можна зробити висновок, що робота заслуговує оцінки «відмінно/ А (4,75)».

РЕЦЕНЗЕНТ (прізвище, ім'я, по батькові, посада, місце роботи) завідувач кафедри автоматизації, комп'ютерно-інтегрованих технологій та робототехніки, д.т.н., професор Мартинюк Валерій Володимирович

« 5 » червня 2023 . _____ (підпис)

РІШЕННЯ ЕКСПЕРНОЇ КОМІСІЇ

КАФЕДРИ КІБЕРБЕЗПЕКИ

ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:
Назва: Система запобігання витоку даних на основі штучного інтелекту

Автор: Яворський Олександр Віталійович
Спеціальність: 125 – Кібербезпека
Освітня програма: освітньо-професійна
Науковий керівник: Петляк Наталія Сергіївна
Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом. Робота приймається до захисту.	відповідає
2	Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи	
3	Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	

Підтвердження:

Оригінальність тексту роботи за результатами перевірки системою Unichek складає 95,77%, оригінальність тексту роботи за результатами перевірки системою Anti-Plagiarism v-15.257 складає 100%.

Згідно з Положенням про дотримання академічної доброчесності в Хмельницькому національному університеті (https://khmnu.edu.ua/wp-content/uploads/normatyvni-dokumenty/polozhennya/pro-systemu-zabezpechennya-akademichnoyi-dobrochesnosti.pdf) така авторська робота, обсяг оригінального тексту у відсотках до загального обсягу матеріалу в якій складає 90-100 %, визнається роботою з високою унікальністю тексту.

Запозичення, виявлені в роботі, є законними і не є плагіатом, оскільки:

1. Сумарний обсяг всіх запозичень, визначений системою виявлення збігів/ ідентичності/схожості, складає 4.23%, з яких 1% є збігами з одним джерелом, зумовленими наявністю типових фразеологічних виразів предметної області, а також формулюваннями, які утворюють загальновживані фрази.
2. Інші збіги є збігами в назвах використаних друкованих видань, розміщених в переліку джерел посилань

Керівник роботи _____ Н. С. Петляк

Завідувач кафедри кібербезпеки _____ Ю. П. Кльоц

UNICHECK

Ім'я користувача:
Кафедра кібербезпеки

ID перевірки:
1015417655

Дата перевірки:
04.06.2023 22:16:21 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
04.06.2023 22:17:05 EEST

ID користувача:
100008300

Назва документа: **Яворський**

Кількість сторінок: **77** Кількість слів: **14197** Кількість символів: **108380** Розмір файлу: **8.27 MB** ID файлу: **1015080335**

4.23%
Схожість

Найбільша схожість: **1.54%** з джерелом з Бібліотеки (ID файлу: **1015079141**)

3.16% Джерела з Інтернету	497	Сторінка 79
2.66% Джерела з Бібліотеки	200	Сторінка 82

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи	8

Anti-Plagiarism v-15.257

Максимальне співпадіння з одним документом 1.0%

Словники перевірки: en_US, ru_RU, ua_UA. **Помилок в документах: 11%**

	Документ		Сумарний збіг по Базі Даних	
ID: 114654 Назва: Система запобігання витоку даних на основі штучного інтелекту Додано в БД: 2023-06-04 Автора: Яворський О.В. Керівники: Петляк Н.С, Консультанти: Опоненти:	Символи	Лексеми	Символи	Лексеми
	84268	1217	1613 (2%)	30 (2%)

Джерело плагіату

ID	Опис	Наявність плагіату в документі	
		Символи	Лексеми