Хмельницький національний університет
Факультет інформаційних технологій
Кафедра кібербезпеки

КВАЛІФІКАЦІЙНА РОБОТА

Кулачук Ольги Романівни

на здобуття ступеня вищої освіти Бакалавра

Система виявлення аномалій у журналах безпеки мережевої інфраструктури інтернет провайдера

Галузь знань 12 – Інформаційні технології

Спеціальність 125 – Кібербезпека

Освітня програма Кібербезпека

КРБКБ.220162.22.01.05 ПЗ

Виконала студентка 3 курсу, група КБс-22-1 _____ Ольга КУЛАЧУК
Ініціали, прізвище

Керівник канд. тех. наук, доцент _____ Юрій КЛЬОЦ
Науковий ступінь, вчене звання — Ініціали, прізвище

Нормоконтролер старший викладач _____ Сергій МОСТОВИЙ
Науковий ступінь, вчене звання — Ініціали, прізвище

До захисту допускаю:

Зав. кафедри кібербезпеки _____ Юрій КЛЬОЦ
Ініціали, прізвище

9 06 2025р.

Хмельницький, 2025

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет Інформаційних технологій

Кафедра Кібербезпеки

Рівень вищої освіти Бакалавр

Галузь знань 12 – Інформаційні технології

Спеціальність 125 – Кібербезпека

Освітня програма Кібербезпека

ЗАТВЕРДЖУЮ

Завідувач кафедри кібербезпеки

Юрій КЛЬОЦ _____

____ _____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ
Кулачук Ользі Романівні

1 Тема роботи Система виявлення аномалій у журналах безпеки мережевої інфраструктури інтернет провайдера

Керівник роботи к.т.н, доцент Кльоц Юрій Павлович

Затверджено наказом ректора університету від ___ _____ 2025 № ____

2 Строк подання студентом кваліфікаційної роботи на кафедру _____

3 Вихідні дані до роботи Дослідити архітектуру систем виявлення аномалій у логах комп'ютерних систем. Проаналізувати сучасні підходи до виявлення аномалій за допомогою глибокого навчання, зокрема на основі LSTM-автоенкодерів. Сформувати навчальну вибірку на основі реальних логів (BETH та Cisco Audit). Розробити метод виявлення аномалій на часових рядах логів із використанням автоенкодера на базі LSTM. Провести навчання моделі, налаштування гіперпараметрів і оцінку достовірності класифікації.

4 Зміст пояснювальної записки (перелік питань, які потрібно розробити) Архітектура систем виявлення аномалій. Роль логів та методи їх аналізу. Класифікація IDS-систем. Використання LSTM-автоенкодера. Підхід до формування та обробки наборів даних BETH і Cisco Audit. Архітектура моделі та параметри навчання. Тестування та оцінка достовірності моделі.

5 Перелік графічного матеріалу (із зазначенням обов'язкових креслень)Структура автокодувальника LSTM. Попередня обробка наборів даних. Матриці плутанини.

6 Консультанти розділів кваліфікаційної роботи

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

7 Дата видачі завдання ____ _____ 2025 р.

КАЛЕНДАРНИЙ ПЛАН

Назва етапів (розділів) кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
Вибір і затвердження теми кваліфікаційної роботи	07.02.2025	
Ознайомлення з предметною областю	10.02.2025	
Дослідження існуючих рішень	26.02.2025	
Постановка задачі	06.03.2025	
Визначення загальних принципів рішення задачі	18.03.2025	
Деталізація принципів рішення задачі	14.04.2025	
Розробка проєктних рішень	24.04.2025	
Апробація проєктних рішень	04.05.2025	
Оформлення пояснювальної записки згідно вимог	28.05.2025	
Оформлення графічної частини	31.05.2025	
Захист КР	10.06.2025	

Студентка _____ Ольга КУЛАЧУК

Керівник кваліфікаційної роботи _____ Юрій КЛЬОЦ

АНОТАЦІЯ

Тема кваліфікаційної роботи: «Система виявлення аномалій у журналах безпеки мережевої інфраструктури інтернет провайдера».

Авторка роботи: Кулачук Ольга Романівна.

Керівник роботи: Кльоц Юрій Павлович.

Пояснювальна записка: 62 с., 1 додаток, 16 рис., 41 джерел.

Графічна частина: 3 презентаційних слайди.

Ключові слова: LSTM, автоенкодер, лог-файл, виявлення аномалій, кібербезпека, IDS, лог-аналіз.

У сучасних умовах стрімкого розвитку мережевих технологій і зростання складності ІТ-інфраструктур важливим завданням є забезпечення надійного та своєчасного виявлення потенційно небезпечних подій у системах. У кваліфікаційній роботі розроблено систему виявлення аномалій у журналах безпеки інтернет-провайдера, що використовує глибоке навчання на основі автоенкодера з архітектурою LSTM. Такий підхід дозволяє аналізувати послідовні лог-дані, виявляючи нестандартні шаблони поведінки, характерні для кіберзагроз. У процесі дослідження здійснено аналіз існуючих методів лог-аналізу та систем виявлення вторгнень, обґрунтовано вибір саме LSTM-автоенкодера для задачі безнаглядного виявлення аномалій. Було проведено попередню обробку реальних логів, сформовано навчальні вибірки, реалізовано нейронну модель, налаштовано гіперпараметри та здійснено повноцінне тестування. Оцінка достовірності моделі виконувалася з використанням метрик точності, повноти, F1-міри та аналізу похибки реконструкції. Результати демонструють високу достовірність і стабільність моделі до аномалій навіть у складних багатовимірних лог-файлах. Запропоноване рішення має значний практичний потенціал для застосування в мережах провайдерів, корпоративних системах та інших середовищах, що потребують гнучкого й ефективного моніторингу безпеки.

05.06.2025

ABSTRACT

Topic of the qualification work: «DDoS attack detection system in 5GN networks»

Author: Kulachuk Olha Romanivna

Supervisor: Klots Yurii Pavlovych

Explanatory note: 62 p., 1 appendix, 16 figures, 41 references.

LIST OF KEYWORDS: LSTM, autoencoder, log file, anomaly detection, cybersecurity, IDS, log analysis.

In today's environment of rapid development of network technologies and increasing complexity of IT infrastructures, an important task is to ensure reliable and timely detection of potentially dangerous events in systems. In the qualification work, a system for detecting anomalies in the security logs of an Internet provider using deep learning based on an autoencoder with LSTM architecture was developed. This approach allows analyzing sequential log data, identifying non-standard behavioral patterns typical of cyber threats. In the course of the study, the existing methods of log analysis and intrusion detection systems were analyzed, and the choice of an LSTM auto-encoder for the task of unsupervised anomaly detection was substantiated. We pre-processed real logs, formed training samples, implemented the neural model, tuned hyperparameters, and performed full testing. The model's reliability was evaluated using the metrics of accuracy, completeness, F1-measure, and reconstruction error analysis. The results demonstrate high reliability and stability of the model to anomalies even in complex multivariate log files. The proposed solution has significant practical potential for use in provider networks, corporate systems, and other environments requiring flexible and efficient security monitoring.

05.06.2025

ЗМІСТ

ВСТУП

В умовах постійного зростання складності та масштабності інформаційних систем проблема забезпечення їхньої надійності та безпеки набуває особливої значущості. Кількість інцидентів, пов’язаних із несанкціонованим доступом, шкідливим впливом на функціонування цифрових ресурсів, а також зловживанням легітимними привілеями, неухильно зростає. Серед важливих напрямів дослідження в галузі інформаційної безпеки особливе місце займає завдання виявлення аномальної активності, що потенційно свідчить про загрозу для функціонування IT-інфраструктури. У цьому контексті аналіз лог-файлів, які містять структуровані записи про події у системах, є фундаментальним джерелом емпіричних даних, необхідних для здійснення моніторингу та детектування відхилень від нормативної поведінки.

Традиційні підходи до виявлення порушень, які ґрунтуються на сигнатурному аналізі або ручному формулюванні правил, продемонстрували свою обмеженість у контексті сучасних динамічних загроз. Вони є неефективними при виявленні нових, раніше неописаних атак, а також часто неспроможні забезпечити вчасну реакцію в умовах великого обсягу вхідних даних. У зв’язку з цим, усе більшої актуальності набувають адаптивні методи, що базуються на парадигмах машинного та глибинного навчання, зокрема архітектури, що реалізують механізми обробки послідовностей, такі як рекурентні нейронні мережі з довготривалою короткочасною пам’яттю (LSTM).

Застосування нейронних мереж, орієнтованих на часові ряди, дозволяє будувати моделі, здатні до самонавчання на основі нормальної поведінки системи без потреби в чітко визначених ознаках атак. Завдяки здатності моделювати як короткострокові, так і довгострокові залежності у лог-даних, LSTM-моделі забезпечують високий рівень чутливості до контексту подій, що критично важливо для точного виявлення відхилень. Особливий інтерес у цьому контексті становлять автоенкодери, побудовані на базі LSTM, які можуть бути ефективно використані

для безнаглядного навчання з метою детектування аномалій на основі аналізу похибок реконструкції.

Метою цієї кваліфікаційної роботи є розробка та експериментальна верифікація моделі автоматизованого виявлення аномалій у логах комп'ютерних систем з використанням автоенкодера, реалізованого на основі архітектури LSTM. Наукова новизна дослідження полягає у поєднанні механізмів часової обробки даних з автоматизованою генерацією ознак з лог-файлів, що дозволяє підвищити точність і надійність системи виявлення загроз.

У процесі реалізації поставленої мети передбачалося здійснити комплексний аналіз предметної області, дослідити структурні особливості системних логів, обґрунтувати вибір методів попередньої обробки та нормалізації даних, а також побудувати архітектуру нейронної мережі із відповідними параметрами навчання. Подальша валідація моделі здійснювалася на основі відкритих репрезентативних наборів даних, що містять як нормальні, так і аномальні події, зокрема BETH та Cisco Audit. На підставі порівняльного аналізу ефективності було визначено придатність обраного підходу до практичного використання в системах забезпечення інформаційної безпеки.

Таким чином, тема цієї роботи є актуальною як з прикладної, так і з наукової точки зору, оскільки поєднує інноваційні підходи до обробки великих обсягів даних із сучасними засобами інтелектуального аналізу поведінки в інформаційних системах. Результати дослідження можуть бути використані в подальшому для побудови модулів виявлення вторгнень, інтегрованих у засоби захисту корпоративних мережевих середовищ та платформ з підвищеними вимогами до надійності.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА НАЯВНИХ РІШЕНЬ

1.1 Журнал безпеки та логування подій

Логування є важливим не лише для безпеки, а й для сучасних бізнесів, орієнтованих на інформаційні технології (IT), оскільки воно забезпечує кілька функцій, таких як усунення неполадок, аналіз продуктивності, оптимізація та аудит, що робить збір і зберігання журналів надзвичайно важливими [1-2]. Належне управління журналами та їх аналітика дозволяють організаціям виявляти інциденти безпеки, шахрайські дії та операційні проблеми. Без метрик, які можна отримати з журналів, важливі рішення щодо діяльності організації не можуть бути ухвалені. Метрики дозволяють встановити певні норми для визначення того, чи дотримуються ці норми, адже відхилення від них є показником наявної проблеми. Організації також повинні створити та підтримувати інфраструктуру управління журналами, що відповідає їхнім потребам [3-4]. Наприклад, машини та хости, які не беруть участі в процесі ведення журналів, можуть стати ідеальними векторами для зловмисників. Аналогічно, невідслідковування деяких подій у журналах може приховувати ознаки атаки. Крім того, організації повинні встановити правила використання даних журналів, що включають аналітику, документацію та звітність про аномалії — основні результати, які досягаються завдяки інфраструктурі ведення журналів.

Логування також виступає основним засобом для забезпечення підзвітності в IT, оскільки більшість дій користувачів і систем фіксуються в журналах [5-6]. Однак аналіз журналів часто здійснюється реактивно, тобто після виявлення проблеми, а не проактивно, щоб виявити поточну активність і попередити про можливі проблеми. Традиційно більшість журналів не аналізуються в реальному часі, що значно знижує їхню цінність.

Журнали відіграють важливу роль у фіксації та зберіганні важливої інформації про події, що відбуваються в комп’ютерних системах, мережах, додатках та IT-інфраструктурі. Вони є цінним джерелом даних, яке дозволяє отримати уявлення про поведінку системи, усунення неполадок, моніторинг

продуктивності, забезпечення безпеки та сприяє дотриманню вимог. Журнали, або журнали подій, записують широкий спектр подій та дій, які можуть відбуватися в системі. Загалом існує три основні причини для ведення журналів на різних джерелах даних: операційне логування, безпекове логування та комплаєнс-логування.

Логування операцій допомагає системним адміністраторам отримувати корисну інформацію, яка сповіщає їх про збої або ситуації, що можуть вимагати дій [7]. Окрім того, воно використовується для надання послуг та впливає на фінансові рішення. Безпекове логування орієнтоване на виявлення та реагування на проблеми безпеки, такі як атаки, зараження шкідливими програмами або крадіжка даних [8-9]. Аудиторські журнали часто застосовуються для запису спроб автентифікації користувачів та інших рішень доступу, з метою аналізу, чи має користувач доступ до ресурсу без належної автентифікації. Комплаєнс-логування, яке часто перетинається з безпековим логуванням, охоплює регуляції та вимоги до IT та систем, що стосуються проектування та безпеки систем.

Ці три основні причини ведення журналів визначають існування різних типів журналів, які відповідають певним цілям. Вони використовуються для запису різних аспектів діяльності системи, подій у мережі та інцидентів безпеки. Необхідні журнали можуть змінюватися в залежності від конкретної IT-інфраструктури, додатків та вимог галузі. Різні типи журналів, як-от операційне чи безпекове логування, можуть мати свої особливості. Безпекові журнали фіксують події, пов’язані з безпекою, і можуть містити інформацію про спроби автентифікації, зміни привілеїв, порушення політики безпеки, активність брандмауера та події виявлення вторгнень. Ці журнали грають важливу роль у виявленні та розслідуванні порушень безпеки та забезпеченні відповідності стандартам безпеки. Аудиторські журнали охоплюють широкий спектр подій для забезпечення відповідності, відстеження змін у системах і підтримки аудиторських процесів. Вони містять інформацію про дії користувачів, зміни конфігурацій систем, адміністративні дії та зміни політик, що є важливим для підтримки цілісності системи, підзвітності та відповідності вимогам регулюючих органів.

Кожне повідомлення в журналі має певну структуру, деякі з яких включають елементи людської мови, але основною складовою є різноманітні інформаційні шаблони. Для забезпечення зручності в рамках існуючих регуляцій можна визначити стандартизований набір полів, що повинні бути присутні в кожному записі. Серед найбільш часто використовуваних полів є: дата та час, тип запису журналу, система, що його створила, джерело події, показники успіху чи невдачі, пріоритет або важливість повідомлення та ім’я користувача для журналів, пов’язаних із діяльністю користувачів.

Синтаксис журналів має важливе значення для впровадження будь-якого виду автоматизованого аналізу даних журналів. Деякі програми для аналізу журналів розробили "схеми повідомлень", щоб визначити синтаксис журналів. Ці схеми спрямовані на те, щоб охопити більшість повідомлень, які можуть з’являтися в різних журналах з різних пристроїв, систем і додатків. Для автоматизованої системи, що здійснює аналіз журналів, важливо мати розуміння синтаксису журналів, що підлягають аналізу, яке зазвичай закодовано в шаблоні. Різні системи та постачальники використовують кілька вибраних типів логування та варіантів синтаксису. Наприклад, деякі типи безпекового логування використовують XML для більш зручної інтеграції інформації, а багато операційних журналів ведеться через syslog із неструктурованими або напівструктурованими повідомленнями.

Деякі записи журналів мають структурований формат, що містить інформацію, яка добре підходить для виявлення потенційних спроб вторгнень. Такі мережеві журнали дозволяють визначити, що була спроба атаки на хост, але не можуть підтвердити, чи була вона успішною. Для більш повного розуміння ситуації можна комбінувати журнали з різних систем, таких як Snort і TCP-Wrappers, щоб отримати більш точну картину того, що відбувається в мережі.

1.2 Класифікація систем виявлення вторгнень

Системи виявлення вторгнень (IDS) призначені для моніторингу мережевого трафіку чи систем з метою виявлення ознак несанкціонованого доступу або шкідливої активності [10-12]. Це важливий елемент кібербезпеки, який дозволяє постійно стежити за станом програмного та апаратного забезпечення в мережі, щоб вчасно виявляти загрози, ідентифікувати їх, здійснювати зниження ризиків, дотримуватися стандартів безпеки, проводити судову експертизу та запобігати вторгненням. Незважаючи на десятиліття розвитку цієї галузі кібербезпеки, системи IDS досі стикаються з викликами, такими як покращення точності виявлення, зниження рівня хибних спрацьовувань та виявлення нових типів атак.

Для розв'язання цих проблем дослідники дедалі частіше використовують методи машинного навчання. Вони дозволяють автоматично розрізняти нормальні дані та аномальні із високою точністю навіть у великих масивах інформації. Крім того, такі методи мають високу здатність до узагальнення, що дозволяє їм виявляти раніше невідомі загрози. IDS на основі машинного навчання не потребують глибоких знань у конкретній галузі, що спрощує їх створення [13-15]. Ще більшу увагу привертають методи глибокого навчання, які є підрозділом машинного навчання. Вони показують виняткову ефективність, оскільки здатні автоматично навчатися ознакам на основі необроблених даних і надавати результати у форматі "від початку до кінця", що робить їх практичними для використання.

В контексті IDS вторгненням вважається спроба незаконного або несанкціонованого доступу до інформації про комп'ютерні системи або пошкодження їхньої роботи. Такі системи кібербезпеки спрямовані на виявлення широкого спектру порушень безпеки, починаючи з зовнішніх атак і закінчуючи зловживаннями з боку внутрішніх користувачів. Основні функції IDS включають моніторинг мереж і хостів, аналіз поведінки систем, генерацію сповіщень та реагування на підозрілі дії [16].

IDS поділяються на методи, засновані на способах виявлення, та методи, що базуються на джерелах даних. Способи виявлення поділяються на виявлення

сигнатур та аномалій.

Методи виявлення, засновані на сигнатурах (Signature-based IDSs, або SIDS), базуються на технологіях співставлення шаблонів для ідентифікації відомих атак, тому їх також називають методами, заснованими на знаннях. Коли дані містять сигнатуру вторгнення, яка збігається з сигнатурою у базі даних, система SIDS генерує сповіщення про конкретне вторгнення. Головною перевагою таких систем є низький рівень хибних спрацьовувань, а також можливість детально вказати тип атаки та її потенційні причини [17-19].

Однак зростаюча кількість «нульових» атак значно знижує ефективність SIDS, оскільки для таких атак немає попередньо визначених сигнатур. Крім того, методи зловмисників постійно удосконалюються, щоб робити аномальну активність максимально схожою на нормальну. Навіть найменші адаптації дозволяють обійти встановлені правила.

Епоха Інтернету речей (IoT) та великих даних ускладнює ситуацію, адже для охоплення всіх можливих атак потрібно створювати величезну кількість правил, що суттєво збільшує розмір бази даних сигнатур. Постійне оновлення цієї бази для врахування нових атак вимагає значних зусиль і з часом може призводити до зниження продуктивності системи.

Системи виявлення аномалій (AIDS) можуть вирішувати проблеми, властиві іншим методам, оскільки вони базуються на створенні профілю прийнятної поведінки, а не виявленні аномалій. Такі системи формують модель нормальної поведінки системи або мережі, використовуючи методи машинного навчання, статистичні або знання, засновані на попередньому досвіді. Якщо поведінка значно відхиляється від цієї моделі, вона розглядається як аномалія. Основою цього підходу є припущення, що шкідлива активність відрізняється від типової поведінки користувачів. Однією з ключових переваг AIDS є здатність виявляти нові та «нульові» атаки, оскільки для цього не потрібно мати попередні дані про них, лише розуміння того, що поведінка є нетиповою. Однак цей підхід може спричиняти велику кількість хибних спрацьовувань, адже будь-яке невідоме відхилення автоматично вважається загрозою. Крім того, такі системи не здатні надати

пояснення причин виявленої аномалії [20-22].

Системи виявлення вторгнень також класифікуються за джерелами вхідних даних на хост-орієнтовані (HIDS) та мережеві (NIDS). HIDS аналізують журнали операційної системи, файрволів, додатків та баз даних, що дозволяє точно визначати місце вторгнення й реагувати на нього [23-25]. Проте, такі системи залежать від ресурсів хоста, не здатні виявляти мережеві атаки та залежать від надійності самого хоста. NIDS, навпаки, аналізують мережевий трафік, який збирається за допомогою перехоплення пакетів і розгортається на ключових елементах мережі, таких як маршрутизатори або комутатори. Вони можуть виявляти зовнішні атаки, тоді як HIDS ефективніше працюють із внутрішніми загрозами, які не пов'язані з мережевим трафіком [26-28]. Комбінація NIDS, HIDS та файрволів забезпечує надійний захист як від внутрішніх, так і від зовнішніх атак.

IDS у формі систем виявлення аномалій (ADS), раніше згадані як AIDS, включають методи, засновані на статистиці, часових рядах та машинному навчанні. Існують три різні підходи до навчання моделей на основі машинного навчання: з наглядом, без нагляду та напівнаглядовий. Навчання з наглядом використовує мічені дані для нормальних подій та аномалій для навчання моделі виявлення. Прикладами алгоритмів з наглядом є машини опорних векторів (SVM), байєсівські мережі та штучні нейронні мережі (ANN). Хоча методи з наглядом працюють краще за підходи, засновані на сигнатурах, вони стикаються з труднощами при виявленні нульових атак, оскільки для таких випадків немає мічених даних. Крім того, для ефективного навчання цими методами потрібні добре збалансовані та мічені дані, що є складними для дослідників, а процес маркування даних може бути трудомістким, схильним до помилок і виснажливим. Методи без нагляду, навпаки, не потребують мічених даних, оскільки вони орієнтовані на виявлення внутрішньої структури в даних без явних цільових виходів. До таких підходів належать моделі прихованих марковських процесів, кластеризація k-середніх та ієрархічна кластеризація. Основна мета методів без нагляду — організувати вхідні дані в кластери, групуючи їх за подібністю всередині кожного окремого кластера [29-30]. Однак головним обмеженням цих методів є високий рівень хибнопозитивних

результатів, коли негативні випадки неправомірно ідентифікуються як позитивні, оскільки відсутні мітки для корекції. Методи без нагляду також залежать від експертизи користувача для інтерпретації та маркування класів після класифікації. Напівнаглядове навчання використовує лише нормальні дані для процесу навчання, а потім вводить немічені дані, які містять як нормальні події, так і аномалії, на етапі тестування. На практиці знайти нормальні дані легше, ніж дані з аномаліями.

Також ці методи можна застосовувати до моделей глибинного навчання для ADS. Моделі глибинного навчання складаються з різноманітних глибинних мереж, де до моделей з наглядом відносяться глибинні нейронні мережі (DNN), згорткові нейронні мережі (CNN) та рекурентні нейронні мережі (RNN). Моделі без нагляду включають автоенкодери, обмежені машини Больцмана (RBM) та генеративні змагальні мережі (GAN). Незважаючи на те, що ефективність моделей глибинного навчання перевищує ефективність традиційних моделей машинного навчання, вони мають деякі недоліки. Через високу складність моделей глибинного навчання їх час навчання та тестування значно довший, ніж у моделей машинного навчання. Крім того, кількість навчальних параметрів та гіперпараметрів значно більша, ніж у моделей машинного навчання. Оскільки моделі глибинного навчання є "чорними ящиками", їх результати важко інтерпретувати, що є вагомим аспектом у глибинному навчанні, оскільки їхня складність створює труднощі в розумінні, як вони прийшли до своїх передбачень чи рішень. Проте моделі глибинного навчання мають здатність навчатися з необроблених даних та володіють більшою здатністю до підлаштування завдяки своїй складній структурі та великій кількості параметрів.

Щодо HIDS, ці системи виявлення використовують лог-орієнтоване виявлення атак. Використання логів як джерела даних для ADS має кілька переваг. Логи містять детальну інформацію про контент, що дозволяє виявляти специфічні атаки, такі як SQL-ін'єкції, U2R та R2L-атаки. Крім того, логи часто містять інформацію про користувачів та часові мітки, що дозволяє відслідковувати зловмисників, а також визначати час і тривалість атак. Результати виявлення на основі логів є інтерпретованими, оскільки логи фіксують весь процес вторгнення.

Однак проблемою аналізу логів є залежність від знань у галузі кібербезпеки. Більше того, різні застосунки та системи мають різні формати логів, що призводить до низької масштабованості. Лог-орієнтоване виявлення атак включає в себе гібридні методи, які поєднують правила та машинне навчання, методи на основі текстового аналізу та методи на основі видобутку особливостей з логів. Методи на основі видобутку особливостей з логів передбачають видобуток ознак з логів згідно з галузевими знаннями та виявлення аномальних поведінок за допомогою цих ознак. Поведінка вторгнення може залишати сліди в різних логах, які можна аналізувати та виявляти з використанням алгоритмів класифікації.

Загальна структура виявлення аномалій на основі логів складається з чотирьох етапів: збір логів, парсинг логів, видобуток ознак та виявлення аномалій. Програмні системи та апаратні засоби регулярно генерують логи, які можуть використовуватись для різних цілей, зокрема для виявлення аномалій. Зазвичай кожен лог є рядком напівструктурованого тексту, виведеним за допомогою оператора логування в програмному коді. Лог зазвичай складається з мітки часу та детального повідомлення, яке може описувати симптоми помилки, цільовий компонент та IP-адресу. Збір та зберігання логів є основною частиною і першим кроком в аналізі логів та подальшому виявленні аномалій на їх основі.

Для того, щоб здійснити аналіз логів та застосувати методи виявлення аномалій, такі як машинне навчання, логи повинні бути перетворені на структуровані події з однаковими полями в кожному повідомленні для подальшого аналізу. Парсинг текстових повідомлень логів у структурований формат дозволяє здійснювати ефективний пошук, фільтрацію, групування, підрахунок та складний аналіз логів [31]. Існують різні техніки та інструменти для парсингу логів, зокрема: добування частих шаблонів, кластеризація, ітераційне поділення, найдовша спільна підпослідовність, дерево парсингу, еволюційні алгоритми та інші евристики. Крім того, парсинг логів можна поділити на два основні режими: офлайн та онлайн. Традиційний спосіб парсингу логів ґрунтується на створенні регулярних виразів (Regex) або grok-шаблонів для витягнення шаблонів подій і ключових параметрів, що є типом парсингу на основі правил. Однак регулярні вирази не підходять для

загальних цілей, оскільки кожен тип лог-повідомлення потребує унікального регулярного виразу, що базується на специфічних знаннях у певній галузі, і також потребує постійного оновлення з новими записами логів [32].

Щоб запустити алгоритм виявлення аномалій на даних логів, необхідно витягнути ознаки, які будуть подаватись в алгоритм. Видобуток ознак полягає у визначенні конкретних подій або шаблонів логів, які мають відношення до аномалій, які потрібно виявити. Цей етап аналізу логів може також включати попередню обробку та трансформацію даних логів, щоб зробити їх придатними для аналізу. Крім того, якість та точність видобутку ознак безпосередньо впливають на точність виявлення аномалій за допомогою моделі машинного навчання. Процес видобутку ознак можна здійснювати різними способами, а найкращий підхід залежить від алгоритму виявлення аномалій, набору даних логів та того, які саме аномалії потрібно виявляти.

Логи можна групувати за ідентифікаторами, де кожне сесійне вікно має унікальний ідентифікатор. Для кожної лог-послідовності генерується вектор ознак, що представляє кількість поява кожної події. Усі вектори ознак разом формують матрицю ознак — матрицю підрахунку подій.

Коли дані логів були проаналізовані, а ознаки витягнуті, можна приступати до виявлення аномалій. Виявлення аномалій може включати встановлення порогових значень, визначення правил або використання технік навчання з учителем і без нього для виявлення відхилень або аномалій. У літературі виявлення аномалій може позначатися різними термінами, такими як виявлення подій, виявлення рідкісних подій, виявлення вторгнень або виявлення зловживань, які описують одну й ту ж мету — виявлення рідкісних точок даних, які значно відрізняються від загального розподілу даних. Рівень відхилення зазвичай визначається як міра сили або ймовірність того, що точка є аномалією, що відома як оцінка аномалії.

Виявлення аномалій із використанням цього методу є складним завданням, оскільки аномальні частини набору даних часто складають менше ніж 1%, що означає, що звичайні бінарні класифікатори будуть мати точність понад 99%, якщо

всі точки будуть позначені як нормальні, що ускладнює завдання виявлення аномалій. Однак для досягнення задовільних результатів у виявленні аномалій потрібно обрати правильний метод, що залежить від характеристик набору даних, який використовується. Серед важливих властивостей для вибору підходу до виявлення аномалій можна виділити наявність міток у даних, тип даних (тимчасові чи нетимчасові), а також наявність одновимірних чи багатовимірних даних. Крім того, типи аномалій, що містяться в наборі даних, також впливають на вибір методу виявлення аномалій. Наприклад, точкові аномалії зазвичай виявляються методами рідкісної класифікації, а колективні аномалії вимагають методів, які зосереджуються на незвичних формах даних.

Виявлення аномалій на часових рядах може значно відрізнятися від виявлення аномалій на нетимчасових даних, таких як просторові дані. Наприклад, для просторових даних основні методи виявлення аномалій полягають у вимірюванні відхилення аномальних точок від решти даних або у використанні кластеризації, де всі точки в менш щільних областях позначаються як аномальні. Головне припущення при виявленні аномалій на просторових даних полягає в тому, що точки є незалежними одна від одної, чого немає на часових рядах, де точки залежні одна від одної, і зміни в останніх точках можуть впливати на наступні.

Виявлення аномалій на основі логів можна поділити на два основні категорії: виявлення аномалій у пакетах логів і виявлення аномалій у потоках логів. Пакетне виявлення аномалій не дозволяє миттєво виявляти аномалії і є більш вимогливим до ресурсів, оскільки аналіз логів проводиться після збору значної кількості логів. Виявлення аномалій у потоках, навпаки, дозволяє негайно отримати сповіщення, хоча більшість сучасних методів потокового виявлення базуються на навчанні з учителем.

Зі збільшенням масштабів і складності сучасних програмних та апаратних систем обсяг логів зріс до неймовірних розмірів, і тому традиційні методи перевірки логів стали непрактичними. Наступним логічним кроком для досягнення ефективного виявлення аномалій є використання рішень на основі глибокого навчання.

Аномалії можна визначити як відхилення, що не є частиною нормальної поведінки системи, і їх можна охарактеризувати як аномалії, ненормальності, відхилення або крайні значення. Аномалії виникають практично в усіх сферах і тому досліджуються в широкому колі прикладних областей, таких як мережеву безпеку, Інтернет речей, медицину та виробничі системи. Аномалії можуть бути безпосередньо виміряні або потребувати спостереження за допомогою деяких методів непрямого оцінювання стану.

1.3 Застосування нейронних мереж для виявлення аномалій

RNN є типом нейронної мережі, який підходить для обробки послідовних або часових даних. Основною її особливістю є вектор стану, розташований у прихованих елементах, який зберігає пам’ять про всі попередні елементи послідовності, довжина якої може бути довільною. RNN має зворотний зв’язок, який з’єднує приховані нейрони в різні моменти часу, що дозволяє мережі враховувати попередні входи для впливу на поточні значення входу та виходу. Ще однією характерною особливістю RNN є те, що вона використовує однакові параметри в усіх шарах мережі, наприклад, однакову вагу в межах кожного шару, хоча ці ваги змінюються під час процесу зворотного поширення і градієнтного спуску, щоб забезпечити підкріплене навчання. RNN, однак, часто стикаються з проблемами вибухаючих або зникаючих градієнтів, які можуть або призвести до нестабільності моделі, або зупинити процес навчання через незначність оновлень вагових параметрів [33-35].

Метод довготривалої короткочасної пам’яті (LSTM) є вдосконаленою версією RNN, яка вирішує ці проблеми за допомогою механізму трьох шлюзів: вхідного, вихідного та забуття. Ці шлюзи контролюють, яку інформацію слід зберігати, додавати чи видаляти, що дозволяє моделі розпізнавати шаблони в даних, що зберігаються протягом різних часових проміжків. У LSTM використовується два основних вектори — короткочасна пам’ять та довготривала

пам’ять, які разом допомагають ефективно працювати з послідовними даними. Завдяки цьому LSTM часто застосовується до завдань аналізу часових рядів [36-37].

LSTM нейронні мережі широко використовуються для вирішення завдань прогнозування, класифікації, розпізнавання шаблонів, аналізу та роботи з послідовностями у різних галузях, зокрема в комп’ютерних науках для виявлення аномалій та прогнозування. Залежно від потреб, LSTM може бути однонаправленою, двонаправленою (BiLSTM) або деревоподібною (Tree-LSTM). Також існує ієрархічна архітектура LSTM, яка передбачає використання кількох LSTM-шарів для зберігання більшої кількості інформації.

Завдяки здатності навчатися часовим зв’язкам і зберігати їх у компактних представленнях стану, LSTM мережі добре підходять для виявлення контекстуальних аномалій. Відхилення реальних результатів системи від очікуваних результатів, які генерує LSTM, може використовуватися для цілей виявлення аномалій. Крім того, LSTM можуть бути інтегровані в різноманітні архітектури нейронних мереж, наприклад, енкодер-декодер, гібридні моделі, графові підходи та моделі з трансферним навчанням.

Автоенкодер є різновидом нейронної мережі, яка належить до сімейства прямопередавальних нейронних мереж і використовується в поєднанні з LSTM для виявлення аномалій [38-40]. Це зумовлено здатністю LSTM захоплювати часові залежності, а автоенкодерів — навчатися представленням даних. Автоенкодер створений для того, щоб кодувати вхідні дані в стислу й інформативну репрезентацію, а потім декодувати їх так, щоб відновлені дані максимально відповідали оригіналу. Його основна мета полягає у навчанні, в режимі без нагляду, корисного представлення даних, яке можна застосовувати для таких завдань, як кластеризація та виявлення аномалій. Навчання автоенкодера може здійснюватися як цілком, так і поступово, додаючи шари поетапно, що дозволяє створювати глибші моделі.

Існують різні типи автоенкодерів, зокрема неповний автоенкодер, шумозахищений автоенкодер, розріджений автоенкодер і змагальний автоенкодер.

Ці моделі можуть змінюватися або комбінуватися для створення нових, залежно від завдань, які вони мають вирішувати. Основна ідея роботи автоенкодера полягає в тому, що розподіли нормальних і аномальних даних значно відрізняються. Аномалія в цьому контексті визначається як спостереження або послідовність спостережень, які суттєво відхиляються від загального розподілу даних. Таким чином, коли автоенкодер намагається реконструювати дані, аномальні значення після декодування виявляють значні відмінності. Цей підхід оптимізовано так, щоб на виході мережа видавала дані, максимально подібні до вхідних, зменшуючи й збільшуючи їхній розмір.

Автоенкодери використовують зворотне поширення для отримання вихідного вектора, схожого на вхідний, застосовуючи нелінійні активаційні функції й кілька шарів для навчання нелінійних залежностей у даних. Цей метод належить до категорії навчання без нагляду, оскільки не потребує маркованих даних для тренування. Автоенкодер складається з двох фаз. У фазі кодування зменшується розмірність вхідних даних, тоді як у фазі декодування отримані дані відновлюються до простору, подібного до початкового. Головним завданням є мінімізація помилки реконструкції, яка обчислюється за допомогою різних функцій, таких як сума квадратів похибок (SSE), перехресна ентропія чи середньоквадратична похибка (MSE).

Математично, фаза кодування зменшує вхідні дані до латентного простору за допомогою ваг, векторів зміщення та активаційної функції, тоді як у фазі декодування ваги, зсуви й активаційні функції можуть відрізнятися, щоб відновити дані до початкового вигляду. Повний процес автоенкодера можна описати через функції кодування та декодування, метою яких є зведення помилки реконструкції до мінімуму. Ця помилка може вимірюватися середньоквадратичною похибкою, середньою абсолютною похибкою або коренем середньоквадратичної похибки.

Щоб визначити, чи є дані аномалією, використовується порогове значення для помилки реконструкції. Найпростішим підходом є фіксований поріг, який вибирається вручну на основі характеристик вхідних даних і бажаного рівня чутливості та специфічності. Ідеальний автоенкодер має точно реконструювати

нормальні дані, але при цьому бути стійким до надмірного запам’ятовування чи перенавчання. Це досягається шляхом створення оптимальної функції втрат.

Для визначення оптимальних гіперпараметрів автоенкодера на основі LSTM важливо встановити відповідні діапазони пошуку та методи вибірки для кожного параметра. Діапазон значень має відповідати природі параметра, враховуючи, чи є він дискретним або неперервним, попередній досвід роботи з подібними моделями та результати експериментів. Наприклад, такі гіперпараметри, як швидкість навчання, мають значний вплив на процес оптимізації, тому їхній діапазон повинен бути досить широким для охоплення найбільш продуктивних варіантів. Натомість параметри, що мають менший вплив, можуть мати вужчий діапазон значень. Надто великі або надто малі значення для будь-якого гіперпараметра можуть суттєво знизити ефективність моделі, тому їх необхідно ретельно налаштовувати.

Гіперпараметри, які враховуються при налаштуванні автоенкодера LSTM, включають кількість зразків у пакеті, які модель обробляє перед оновленням параметрів, а також кількість нейронів у шарах LSTM, що використовуються для вилучення ознак. Сюди входять проміжні виміри та латентний простір, оскільки модель використовує два шари LSTM – один для кодувальника, інший для декодувальника. Також враховується кількість попередніх часових кроків, які модель аналізує при реконструкції вхідних послідовностей, та швидкість навчання, яка визначає темп оновлення вагових коефіцієнтів. Надмірно велике значення швидкості навчання може призвести до пропуску оптимального рішення, тоді як надто мале значення збільшить кількість необхідних ітерацій для досягнення бажаного результату.

Крім цього, у процесі регуляризації використовується коефіцієнт, який визначає ймовірність виключення певних нейронів під час тренування, щоб запобігти перенавчанню. Кількість епох задає, скільки разів модель проходитиме через увесь набір даних, і цей параметр потрібно налаштовувати так, щоб уникати як недостатнього, так і надмірного навчання. Активаційна функція використовується для визначення вихідного сигналу нейронів, а функція оптимізації – для коригування ваг і швидкості навчання, щоб мінімізувати втрати

та підвищити точність моделі. Функція втрат служить для оцінки різниці між передбаченими вихідними значеннями та початковими даними, з метою мінімізації цієї розбіжності та забезпечення ефективності моделі.

Сучасні автоенкодери на основі LSTM демонструють високу ефективність у задачах виявлення аномалій. Модель, яку передбачається створити, повинна мати здатність виявляти аномалії у багатовимірних тимчасових даних журналів, де аномалії зумовлені зловмисними діями. Вона повинна розпізнавати як нові, так і раніше відомі атаки, виявляючи патерни, які значно відрізняються від нормальної поведінки. Водночас важливо, щоб модель могла відрізняти нешкідливі дії, які спричиняють відхилення, від потенційно шкідливих. Крім того, вона повинна бути адаптована для роботи з різними типами журналів і різними методами попередньої обробки даних.

Вибір автоенкодера LSTM для аналізу аномалій у багатовимірних тимчасових журналах хоста базується на його здатності розпізнавати складні закономірності в даних і взаємозв'язки між ними. Патерни, які модель не розпізнає під час тренування, розглядаються як аномалії. Завдяки здатності розпізнавати як короткострокові, так і довгострокові закономірності, автоенкодери LSTM можуть виявляти широкий спектр аномалій, включаючи складні набори даних. Вони здатні навчатися складним нелінійним залежностям у послідовностях, що робить їх цінними для задач виявлення аномалій у часових рядах. До того ж, такі моделі використовують неконтрольоване навчання, що є перевагою у випадках, коли доступ до розмічених даних є обмеженим або коштовним.

1.4 Постановка задачі

Розробка систем виявлення аномалій у логах комп'ютерних систем є складним і багатоетапним завданням, що поєднує методи кібербезпеки, машинного навчання, обробки послідовних даних і аналізу поведінкових шаблонів. У межах даної кваліфікаційної роботи передбачається вирішення комплексу

взаємопов’язаних наукових і прикладних задач, спрямованих на створення ефективної інтелектуальної системи виявлення аномалій з використанням глибинної нейронної мережі типу LSTM-автоенкодера. З огляду на мету дослідження, формулюються наступні конкретні задачі:

– Вивчення предметної області та аналіз сучасного стану проблеми.

– Обґрунтування доцільності використання вибраної архітектури нейронної мережі, враховуючи специфіку вхідних даних.

– Розробка методології попередньої обробки лог-даних. Потрібно розробити процедури для збору, очищення, нормалізації та парсингу логів із різних джерел, таких як системні журнали, журнали аудиту та мережеві логи. Здійснюється перетворення неструктурованих або напівструктурованих записів у формалізовану матрицю ознак, придатну для подачі у нейронну мережу.

– Формалізація навчальної вибірки для моделі. Слід здійснити сегментацію даних у часові вікна із фіксованою довжиною, які формують послідовності входів для LSTM-моделі. На цьому етапі визначається структура входу моделі, яка враховує кількість часових кроків, кількість ознак та формат представлення даних.

– Проєктування та реалізація архітектури LSTM-автоенкодера. Необхідно розробити модель глибокої нейронної мережі, що включає шари LSTM-енкодера, латентне представлення, шари декодера та вихідний шар, який генерує реконструйовану послідовність. Підбираються функція втрат, оптимізатор, активаційні функції, кількість нейронів і шарів з урахуванням обчислювальної складності та здатності до узагальнення.

– Налаштування та оптимізація гіперпараметрів моделі. Потрібно визначити оптимальні значення гіперпараметрів, таких як кількість нейронів, розмірність латентного простору, кількість епох, розмір пакету, швидкість навчання, коефіцієнт регуляризації тощо. Оптимізація може здійснюватися із використанням методів типу Hyperband, Grid Search або крос-валідації.

– Навчання моделі на реальних лог-даних. Проводиться навчання автоенкодера на сукупності нормальних логів із репрезентативних наборів даних,

зокрема BETH та Cisco Audit. У ході навчання модель формує уявлення про типову поведінку, що дозволяє використовувати її для виявлення аномалій шляхом аналізу похибки реконструкції.

– Встановлення порогових значень аномалії та обчислення метрик. На основі результатів реконструкції визначається порогове значення, перевищення якого інтерпретується як аномалія. Проводиться розрахунок ключових метрик ефективності моделі: точності (Accuracy), повноти (Recall), точності позитивного передбачення (Precision), F-міри (F1-score), хибнопозитивної (FPR) та хибнонегативної (FNR) частот.

– Оцінка достовірності та стійкості моделі. Модель тестується на відокремленому наборі даних, що містить як нормальні, так і аномальні записи, з метою перевірки її здатності до генералізації. Також перевіряється стабільність результатів на різних підмножинах даних за допомогою k-fold крос-валідації.

– Формулювання висновків та перспектив подальших досліджень. Завершальним етапом є інтерпретація отриманих результатів, формулювання висновків щодо ефективності розробленого підходу, а також виявлення можливих напрямів для подальшої оптимізації та масштабування системи, зокрема в напрямку потокової обробки логів або гібридних архітектур.

2 МЕТОД ВИЯВЛЕННЯ АНОМАЛІЙ ТА НАВЧАННЯ НЕЙРОННОЇ МЕРЕЖІ

2.1 Метод виявлення аномалій

Метод виявлення аномалій складається з двох основних етапів. Спочатку реалізовано та проведено навчання моделі виявлення аномалій у вигляді автоенкодера на основі LSTM, яка вивчає закономірності нормальних багатовимірних часових логів. Після цього відбувається виявлення аномалій у тестових даних шляхом обчислення аномальних оцінок на основі похибок реконструкції та їх порівняння з визначеним пороговим значенням.

Автоенкодер LSTM спеціально розроблений для виявлення та запам’ятовування закономірностей у даних. Його структура складається з кількох шарів (рис.2.1): вхідного, енкодера, латентного простору, декодера та вихідного.

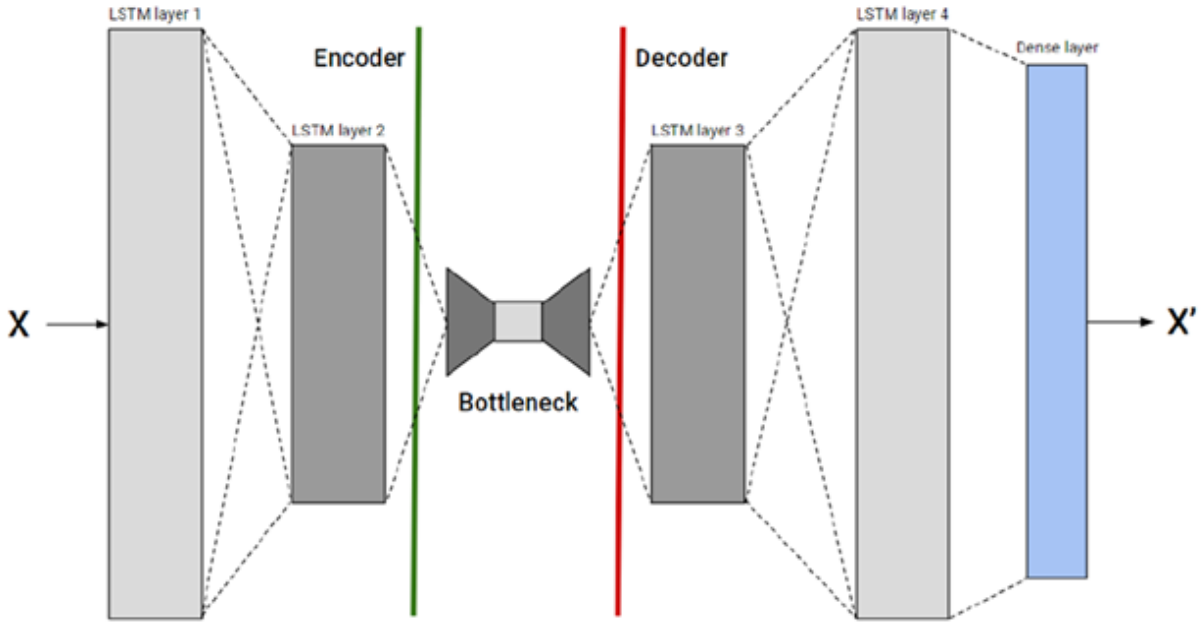


Рисунок 2.1 - Структура автокодувальника LSTM

Вхідний шар це перший LSTM-шар, який отримує багатовимірні часові лог-дані, де кожна точка входу представляє послідовність спостережень за кілька часових кроків. Енкодер стискає та кодує вхідні дані у представлення меншої

розмірності. Нейрони енкодера навчаються виявляти часові залежності у вхідних даних, що дозволяє враховувати вплив минулих спостережень на майбутні. Латентний простір являє собою проміжне представлення, яке містить закодовану інформацію з вхідних даних у зменшеному вигляді. Декодер відтворює вихідні дані з їхнього закодованого представлення. Нейрони декодера працюють у зворотному напрямку, генеруючи вихідні послідовності, які мають максимально наближатися до оригінальних вхідних даних. Вихідний шар формує реконструйовану послідовність, яка за розмірами має відповідати вхідним даним.

Перед навчанням моделі необхідно визначити розмірність шарів, кількість нейронів і коефіцієнти відсіву. Крім того, у процесі навчання автоенкодера LSTM враховуються функція втрат та порогове значення. Функція втрат використовується для мінімізації різниці між початковими даними та реконструйованими значеннями. У ході навчання ця функція оцінює, наскільки точно модель здатна відновлювати початкові дані. Порогове значення для виявлення аномалій визначається на основі даних, які класифікуються як нормальні. Під час тестування послідовності з похибкою реконструкції, що перевищує це порогове значення, вважаються аномальними.

Виявлення аномалії виробляється на основі похибкової реконструкції, яка генерує автокодер LSTM під час тестування даних. Ці похибки використовуються як індикатори аномалій. Встановлений поріг, визначений на основі реконструкційних похибок нормальних навчальних даних, дозволяє класифікувати дані як нормальні або аномальні. Формула відображає цей процес, демонструючи, як обрана функція втрат створення похибки реконструкції шляхом порівняння вихідної бази даних із реконструйованою. Якщо отримане значення вище за встановлений поріг, то дані класифікуються як аномальні, інакше вважаються нормальними.

Автокодер LSTM навчається на основі логів, що не є аномаліями, щоб розпізнавати закономірності нормальних багатовимірних часових даних. Під час навчання відбувається оптимізація гіперпараметрів, які оцінюються за допомогою крос-валідації з використанням методу k-means. Такий підхід гарантує стабільність

результатів моделей на різних наборах даних. Для коректної обробки часових рядів набір даних розділяється по цілісності, зберігаючи їх хронологічний порядок. Гіперпараметри коригуються з підсумкових результатів на різних підмножинах даних.

Формується три набори даних: навчальний, що містить лише нормальні логи та становить (k-1) усіх усіх даних; валідаційний, що також складається лише з нормальних логів і займає 1/k від загального обсягу; та тестовий набір, який включає як нормальні, так і аномальні записи. Тестовий набір має бути достатньо великим, щоб забезпечити репрезентативну оцінку роботи моделі.

На початковому етапі гіперпараметри вилучаються за допомогою алгоритму Hyperband Tuner, який оптимізує вибір параметрів, оцінюючи продуктивність різних конфігурацій під час навчання. Потім створюється крос-валідація на основі k-means з вибором значення k=10, що дозволяє зменшити вплив випадкових факторів і підвищити оцінку продуктивності моделі. Для кожного розбиття модель навчається на вибірці, а решта використовується для контролю втрат і запобігання перенавчанню.

Далі встановлений поріг для виявлення аномалії, який базується на основі похибок реконструкції тестових даних. Для цього використовується метрика, що забезпечує точність і повноту, що дозволяє точніше класифікувати аномалії навіть у нерівномірно розподілених наборах даних. Після визначення оптимальних гіперпараметрів модель навчається на повному навчальному наборі, встановлюється поріг похибки реконструкції, а потім відновлюється тестування на невідомих даних, що містять аномалії.

Оцінка ефективності моделі базується на вибраних метриках, які дозволяють аналізувати її продуктивність на тестових даних. Завдання цього підходу – забезпечити стабільність і точну роботу моделі як на валідаційних, так і на тестових даних. Агреговані результати з різних етапів оцінки допомагають отримати комплексне виявлення здатності моделі виявляти аномалії.

Налаштування гіперпараметрів здійснюється за допомогою Hyperband Tuner з бібліотеки Keras, що дозволяє ефективно досліджувати широкий спектр

конфігураційних параметрів, водночас розподіляючи гіперресурси на найбільш перспективні варіанти. Hyperband збалансовує дослідження нових конфігурацій та використання найкращих знайдених параметрів. Важливими параметрами налаштування є максимальна кількість епох для навчання однієї моделі та кількість ітерацій алгоритму.

Для ефективного виявлення аномалій у часових багатовимірних логах реалізовано механізм look back , який визначає кількість попередніх часових кроків, що враховуються моделлю при аналізі даних. Завдяки цій моделі отримується контекст, необхідний для виявлення закономірностей у часових рядах. Якщо значення look back дорівнює n, то на кожній кроці модель аналізує дані в області від поточного моменту до tn. Це дозволяє створювати перекривання системних логів і навчати модель розпізнавати взаємозв'язок між подіями в часовому контексті.

2.2 Набори даних BETH та Cisco Audit

У даному дослідженні було використано два набори часових логів для навчання та тестування автоенкодера LSTM, спрямованого на виявлення аномалій. Першим набором даних став BETH, що містить реальні аномальні події, отримані з мережі 23 хостів-«пасток», що дозволяє моделі оцінювати аномалії в різних мережевих середовищах. Ці події були зафіксовані протягом п'ятигодинного періоду на великій хмарній платформі, що надає набору реалістичний часовий контекст. Набір містить понад 8 мільйонів записів, що робить його одним із найбільших доступних наборів даних для кібербезпеки. Він охоплює як сучасну активність користувачів, так і атаки, маючи чітке маркування та включаючи структуровані, різнорідні характеристики. У кожному з хостів записані як нормальні події, так і щонайбільше одна атака, що дозволяє детальніше аналізувати поведінкові патерни. Навчальні та валідаційні підмножини містять тільки нормальну активність, а тестовий набір включає аномалії, що дозволяє

використовувати його без додаткової фільтрації.

Атаки в наборі даних BETH включають виконання команд Bash для отримання інформації про пам’ять системи, видалення доступу інших користувачів через SSH та розгортання ботнет-вузла. Окремі зловмисні дії пов’язані з криптомайнінгом і переміщенням між серверами. BETH є одним з небагатьох наборів даних, який включає журнали процесів ядра та мережі, що дає можливість отримати цілісне уявлення про зловмисну поведінку. Приклад функцій відображено у таблиці 2.1.

Таблиця 2.1 - Опис і тип кожної функції в журналах процесів на рівні ядра в наборі даних BETH

Характеристика	Тип	Опис
timestamp	float	секунди з моменту завантаження системи
processid*	int	ідентифікатор процесу, що створив цей запис
threadid	int	ідентифікатор потоку, що створив цей запис
parentprocessid*	int	ідентифікатор батьківського процесу
userid*	int	ідентифікатор користувача, що виконав процес
mountnamespace*	int (long)	обмеження монтування, в яких працює цей процес
processname	string	виконуваний командний рядок
hostname	string	ім'я сервера
eventid*	int	ідентифікатор події, що згенерувала цей запис
eventname	string	назва події, що згенерувала цей запис
argsnum*	int	довжина аргументів
returnvalue*	int	значення, повернене цією подією (зазвичай 0)
stackaddresses	list of int	пам’ятні значення, пов’язані з процесом
args	list of dictionaries	список аргументів, переданих цьому процесу
sus	int (0 або 1)	бінарна мітка підозрілої події (1 – підозріла, 0 – ні)
evil	int (0 або 1)	бінарна мітка відомої шкідливої події (0 – безпечна, 1 – ні)

У порівнянні з іншими наборами даних кібербезпеки, такими як KDD Cup 1999, DARPA 1998, ISCX IDS 2012 та NSL-KDD, BETH має перевагу завдяки актуальності даних і включенню реальних активностей. Приклад повідомлення журналу, яке можна знайти в наборі даних, відображено в таблиці 2.2. Зібрані дані містять дві основні категорії: виклики процесів на рівні ядра і мережевий трафік. Для порівняльного аналізу використовувалася підмножина журналів процесів, яка була поділена на піднабори для навчання, перевірки та тестування. Розподіл складав 60% для навчання, 20% для перевірки та 20% для тестування, причому тестовий набір містить атаки.

Таблиця 2.2 - Приклад повідомлення журналу із набору даних BETH

BETH log features	
Timestamp	414.445453
ProcessId	7414
ThreadId	7414
ParentProcessId	7411
UserId	1001
MountNameSpace	4026531840
ProcessName	head
HostName	Ip-10-100-1-217
EventId	1005
EventName	security_file_open
StackAddresses	[]
ArgsNum	4
ReturnValue	0
Args	”[{’name’: ’pathname’, ’type’: ’const char*’, ’value’: ’/usr/lib/locale/C.UTF-8/LC_MEASUREMENT’}, {’name’: ’flags’, ’type’: ’int’, ’value’: ’O_RDONLY*j*O_LARGEFILE’}, {’name’: ’dev’, ’type’: ’dev_t’, ’value’: 211812353}, {’name’: ’inode’, ’type’: ’unsigned long’, ’value’: 5890}]”
Sus	1
Evil	1

Другий набір даних, Cisco Audit, був наданий компанією та містив журнали

з усіх пристроїв Cisco, зафіксовані протягом 3,5 години у звичайний робочий день. Загалом у ньому налічувалося понад мільйон логів, з яких 223 085 були пов’язані з аудитом і виділені для аналізу. Cisco Audit охоплює логи про події, імена користувачів, IP-адреси та спроби аутентифікації. Для експериментів додано поле Sus, яке встановлюється в 1 у разі невдалої аутентифікації, а також поле Evil, що позначає атаки. Приклад повідомлення журналу з набору даних аудиту Cisco показано в таблиці 2.3.

Таблиця 2.3 - Приклад повідомлення журналу з набору даних аудиту Cisco

	1
	1

Для створення тестового набору в Cisco Audit було змодельовано загрозу з боку «інсайдера». Було створено обліковий запис користувача з обмеженими правами, після чого за допомогою скрипта на Python цей користувач намагався отримати доступ до об’єктів мережі Cisco, до яких він не мав повноважень. Усі ці спроби були зареєстровані як невдалі спроби входу. Таким чином, зловмисник намагався дослідити мережу, щоб потенційно здійснити подальші зловмисні дії, такі як збір інформації або порушення роботи системи.

Оскільки три підмножини BETH (навчальна, валідаційна та тестова) містять понад 8 мільйонів точок даних, їх обсяг був суттєво зменшений для забезпечення

ефективного виконання моделі. Основний час займало навчання, тому розмір навчального набору підбирався з урахуванням продуктивності моделі, витрат часу та кількості епох. Для налаштування гіперпараметрів використано перші 20 000 записів BETH, розподілені у співвідношенні 90% для навчання та 10% для валідації. Надалі для крос-валідації методом k-means використано наступні 10 000 записів, поділених на 10 частин, де кожного разу навчання виконувалося на 9 000 точках, а 1 000 використовувалися для валідації. Фінальний навчальний набір містив ще 20 000 точок даних. Порогове значення аномалій визначалося на основі 10 000 точок із валідаційного набору, а тестування виконувалося на повному тестовому наборі BETH, що містив значну кількість даних, позначених як Evil.

У Cisco Audit після обробки залишилося 223 085 записів. Атака була зафіксована між записами з 70 165 до 107 067, тому вони включені до тестового набору, а не використовувалися для навчання. У підсумку навчальний набір містив 50 000 перших записів, тестовий набір – 100 000 записів, а решта 73 085 використовувалися для валідації. Як і в BETH, дані поділили для налаштування гіперпараметрів і крос-валідації, проте для Cisco Audit знадобилося більше навчальних даних для досягнення стабільної продуктивності моделі. У тестуванні використовувався весь тестовий набір для отримання максимально надійного та точного результату.

Загалом, розміри навчальних, валідаційних та навчальних наборів даних, які були використані, вказано у таблиці 2.4.

Таблиця 2.4 – Розміри наборів даних, що було використано

Розмір набору даних		
набір даних		
навчальний		
валідаційний		
тестовий		

2.3 Навчання та тестування нейронної мережі

Для реалізації моделі використовувалося обладнання у вигляді ноутбука з процесором Intel Core i5-8265U (1.60GHz–1.80GHz) під керуванням операційної системи Windows 10 Pro. Розробка та тестування проводилися в середовищі Microsoft Visual Studio Community 2022 (64-bit).

Модель була створена за допомогою мови програмування Python та бібліотеки Keras, яка є API для глибокого навчання і працює на основі TensorFlow. Keras була обрана завдяки своїй простоті, можливості швидкого прототипування та підтримці різних рівнів абстракції, включаючи шари, активаційні функції, функції втрат і оптимізатори. Вона також містить вбудовану реалізацію LSTM-шарів, що забезпечило необхідну функціональність для дослідження.

Щоб забезпечити точність, узгодженість та відтворюваність результатів, було проведено комплексну оцінку вибраного методу та використаних даних. Висока якість логів гарантувала відсутність помилкових висновків, а використання двох різних наборів даних підвищило надійність валідації та загальну застосовність моделі.

Оптимізація гіперпараметрів здійснювалася за допомогою алгоритму Hyperband, який ефективно знаходив найбільш перспективні конфігурації параметрів. Крос-валідація за методом k-means дозволяла оцінити ефективність моделі на різних підмножинах даних, що забезпечило її стабільність незалежно від конкретного розподілу вибірки. Для оцінки використовувалися загальноприйняті метрики машинного навчання, що дало змогу порівнювати результати з іншими дослідженнями у сфері виявлення аномалій.

Щоб забезпечити надійність підходу та відтворюваність результатів, використовувалися окремі валідаційні та тестові вибірки, а також метод *k*-means крос-валідації. Надійні результати повинні були залишатися стабільними як у межах навчального процесу, так і в різних підмножинах даних. Для відтворення результатів важливо було також обрати відповідні метрики оцінки та забезпечити стійкість архітектури нейронної мережі.

Для оцінки створеної моделі застосовувався метод k-means крос-валідації, оскільки він є загальновизнаним у машинному навчанні та допомагає зменшити вплив варіацій у наборі даних. Він також використовувався для порівняння ефективності різних гіперпараметрів. Модель навчалася і тестувалася для кожної можливої конфігурації параметрів, а отримані результати узагальнювалися за всіма k ітераціями, що дозволяло отримати об'єктивний підсумковий показник продуктивності. Оптимальний набір гіперпараметрів визначався як той, що давав найкращий результат.

Для всебічної оцінки ефективності моделі застосовувалися кілька показників: Accuracy, Precision, Recall, F-Measure, False Negative Rate (FNR) і False Positive Rate (FPR)**.** Ці метрики базувалися на поняттях:

– TP (True Positive) – кількість випадків, коли атака правильно визначена як атака;

– TN (True Negative) – кількість випадків, коли нормальні дані правильно класифіковані як нормальні;

– FP (False Positive) – кількість нормальних подій, помилково визначених як атаки;

– FN (False Negative) – кількість атак, які модель неправильно класифікувала як нормальні події.

Метрики розраховувалися за наступними формулами.

Accuracy – загальна точність моделі, яка вимірює частку правильних передбачень відносно всіх випадків:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{2.1}$$

Precision – відображає, яка частина передбачених атак була визначена правильно:

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

Recall – показує, яку частку реальних атак модель змогла виявити:

$$Recall = \frac{TP}{TP + FN} \tag{2.3}$$

Fscore – середнє гармонічне Precision і Recall, яке узагальнює загальну продуктивність моделі:

$$Fscore = \frac{Recall + Precision}{2} \tag{2.4}$$

FNR (False Negative Rate) – частка пропущених атак (невірно класифікованих як нормальні події):

$$FNR = \frac{FN}{TP + FN} \tag{2.5}$$

FPR (False Positive Rate) – частка помилкових спрацьовувань, коли нормальні події були помилково позначені як атаки:

$$FPR = \frac{FP}{TP + FP} \tag{2.6}$$

Accuracy була використана як базовий показник, але в умовах незбалансованих даних (коли нормальні події значно переважають атаки) її інформативність може бути обмеженою. Precision допомагала оцінити впевненість моделі у виявленні атак, тоді як Recall показувала, наскільки повно модель знаходить атаки. Fscore надавала загальну оцінку, поєднуючи Precision і Recall. FNR вказувала на частку атак, які система не змогла виявити, а FPR – на частку помилкових попереджень. Ці метрики дозволяли отримати детальне уявлення про продуктивність моделі та її здатність виявляти аномалії в реальних умовах.

На початковому етапі лог-дані у вихідному форматі були завантажені та

оброблені. Для цього з вибраної кількості записів було виділено необхідні характеристики логів та перетворено їх у формат DataFrame за допомогою бібліотеки Pandas. Оскільки всі лог-характеристики мали текстовий формат, вони були закодовані відповідним чином. Далі виконано тимчасову трансформацію даних відповідно до значення look back, що дозволило моделі враховувати часовий контекст. Двовимірний масив даних перетворено у тривимірний, що забезпечило збереження послідовності подій.

Для уніфікації шкали значень виконано стандартизацію, яка привела всі характеристики до середнього значення 0 та стандартного відхилення 1. Це запобігло домінуванню окремих характеристик із більшими числовими значеннями, що могло б уповільнити процес навчання моделі. Алгоритм попередньої обробки набору даних зображено на рисунку 2.2.

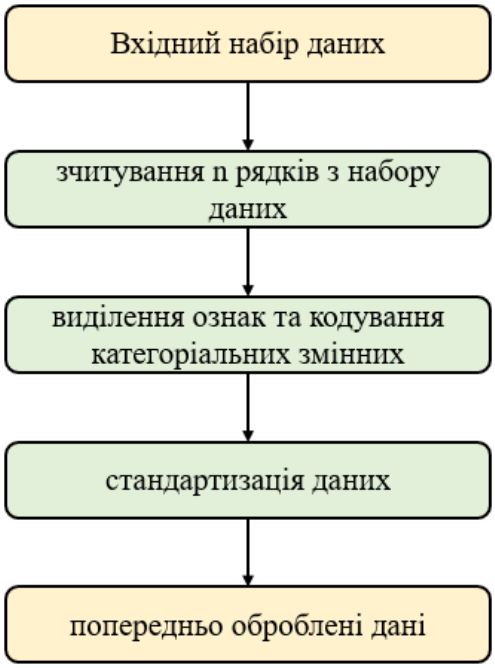


Рисунок 2.2 – Попередня обробка наборів даних

Для набору Cisco Audit довелося виконати додаткові кроки через нерівномірну структуру логів. Було розроблено спеціальні шаблони для виділення ключової інформації та впорядкування лог-характеристик. Окрім цього, для аналізу можливостей моделі та її оцінки до кожного логового запису додано поля Sus і Evil,

подібно до того, як це було зроблено у BETH. Логи про невдалі спроби автентифікації були позначені як підозрілі (Sus), тоді як логи, пов’язані з атакою, були відзначені як підозрілі та зловмисні (Evil). Оскільки невдалі спроби входу до мережевої інфраструктури були рідкісними, у навчальному наборі Cisco Audit виявлено лише п’ять таких випадків, а у валідаційному наборі – два. У тестовому наборі всі події, позначені як Sus, стосувалися змодельованої атаки. Поле Evil використовувалося як "істинна розмітка" для обох наборів.

Процес навчання складався з трьох основних етапів. Спочатку було знайдено оптимальні гіперпараметри шляхом тестування різних конфігурацій. Потім відібрані конфігурації були перевірені за допомогою крос-валідації, а далі визначено порогове значення для класифікації аномалій.

Оптимальні гіперпараметри визначалися на основі попередніх досліджень, експериментальних результатів і специфіки завдання. Було розглянуто діапазони параметрів, використані в інших роботах, присвячених виявленню аномалій у часових рядах. Проте межі цих діапазонів коригувалися відповідно до використаних у дослідженні наборів даних і поставлених цілей.

У багатьох дослідженнях використовувалося поняття sliding window, яке допомогло визначити відповідні значення batch size та look back. Вікна створювалися шляхом поділу часових рядів на менші підпослідовності, що дозволяло зберігати часову залежність між подіями.

Для активаційної, оптимізаційної та функції втрат вибрано початкові значення, що використовуються у більшості досліджень: ReLU – як функція активації, Adam – як оптимізатор, MSE – як функція втрат. Однак експерименти показали, що MAE давала стабільніші результати та була менш чутливою до викидів, тому вона була використана в остаточній версії моделі.

Налаштування гіперпараметрів здійснювалося за допомогою алгоритму Hyperband із параметрами: max_epochs = 5, epochs = 2 та hyperband_iterations = 2. У процесі пошуку оптимальних значень було виконано загалом 20 тестових запусків. Цей процес повторювався шість разів, щоб вручну встановити значення look back та розміри шарів. П’ять найкращих конфігурацій гіперпараметрів було відібрано на

основі мінімальних значень функції втрат у валідаційних даних. Для перевірки стабільності їхньої продуктивності всі п’ять конфігурацій пройшли додаткову крос-валідацію.

Для кожного з п’яти найкращих варіантів гіперпараметрів була виконана *k*-means крос-валідація (k=10), щоб оцінити їхню стабільність. Навчальні набори, що містили по 20 000 записів для BETH та Cisco Audit, були поділені на 10 частин. Модель навчалася на дев’яти підмножинах, а десяте використовувалося для валідації. Процес повторювався 10 разів для кожної конфігурації.

Модель навчалася виключно на нормальних даних, щоб навчитися відтворювати закономірності нормальної поведінки з мінімальними похибками реконструкції. Щоб ідентифікувати аномалії в тестових даних, було встановлено порогове значення, яке базувалося на рівні реконструкційної похибки.

Завданням було вибрати значення порогу, яке дозволяло відфільтровувати незначні коливання нормальних даних, але водночас правильно розпізнавати аномалії. Для цього побудовано графік залежності між точністю (Precision) та повнотою (Recall) для різних значень порогу (рис. 2.3 та таблиця 2.5). Оптимальне значення визначалося на основі компромісу між цими показниками.

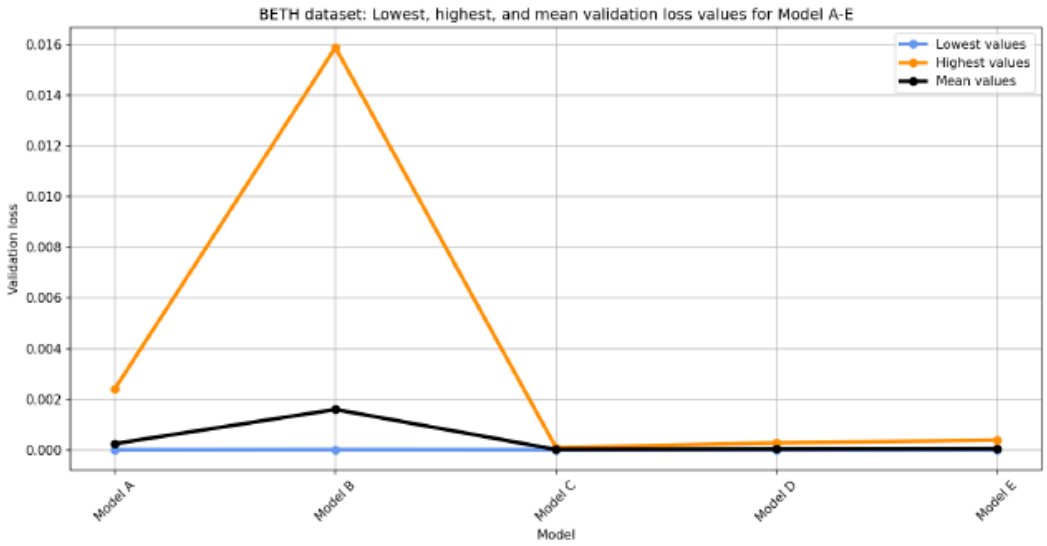


Рисунок 2.3 - Найнижча, найвища та середня втрата валідності при k-кратній перехресній перевірці, виконаній для конфігурацій гіперпараметрів на наборі даних BETH

Таблиця 2.5 - Найнижчі, найвищі та середні значення втрат валідації для моделі F-J, отримані на основі набору даних аудиту Cisco

Аудит Cisco: Найнижче, найвище та середнє значення втрат при перевірці для моделей F-J				
	Найнижче значення	Найвище значення	Середнє значення	Час, що минув
				16 хв 40 с
				8 хв 47 с
				11 хв 26 с
				24 хв 39 с
				9 хв 39 с

Для обох наборів даних встановлені порогові значення були однаковими. У випадку Cisco Audit (рис. 2.4) спостерігався ефект, коли при підвищенні порогу менше точок даних перевищувало його, що призводило до ситуацій, коли точність (Precision) дорівнювала 0 або 1. Якщо жодна реальна аномалія не була визначена, Precision знижувалася до 0, а якщо було знайдено лише одну аномалію та вона була правильно ідентифікована, то Precision сягала 100%.

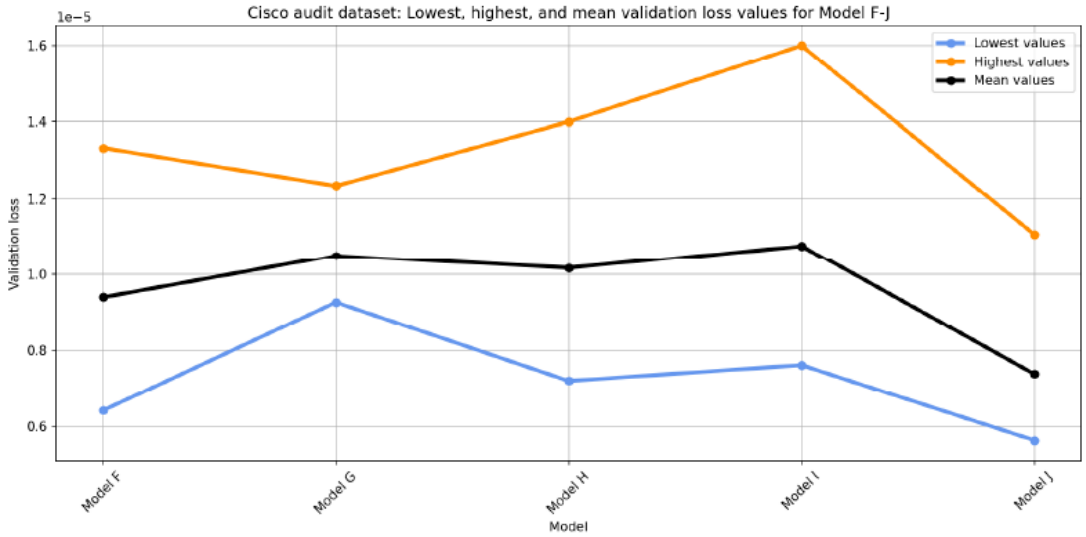


Рисунок 2.4 - Найнижча, найвища та середня втрата валідації при k-кратній перехресній перевірці, виконаній на конфігураціях з гіперпараметрами з набору даних аудиту Cisco

Таким чином, оптимізована модель отримала налаштовані параметри та порогові значення, що дозволяли ефективно виявляти аномалії у двох різних наборах даних.

На етапі тестування модель була навчена з використанням оптимальних гіперпараметрів і навчального набору даних. Кількість навчальних точок даних безпосередньо впливала на тривалість навчання, тому було важливо обрати достатньо великий обсяг даних, який не збільшував би час обчислень без значного покращення результатів. Результати представлено на рис. 2.5 та 2.6.

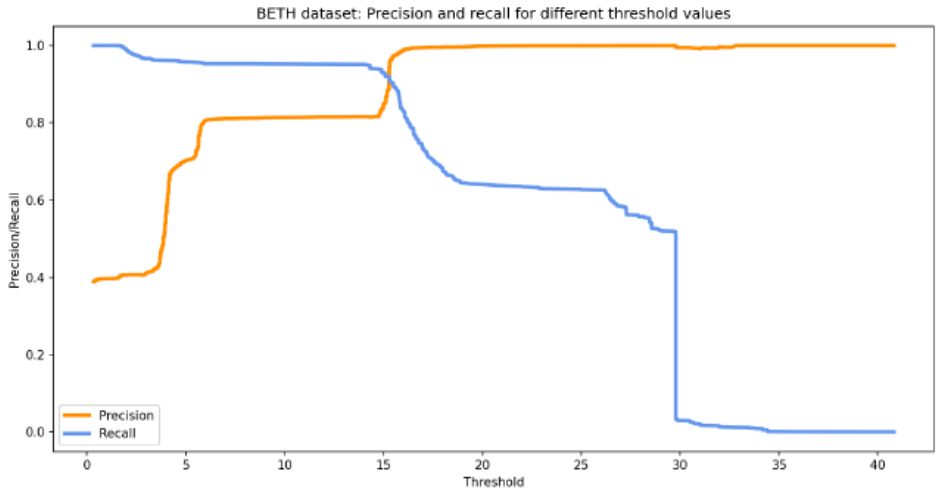


Рисунок 2.5 - Компроміс між точністю та відгуком для різних порогових значень для набору даних BETH

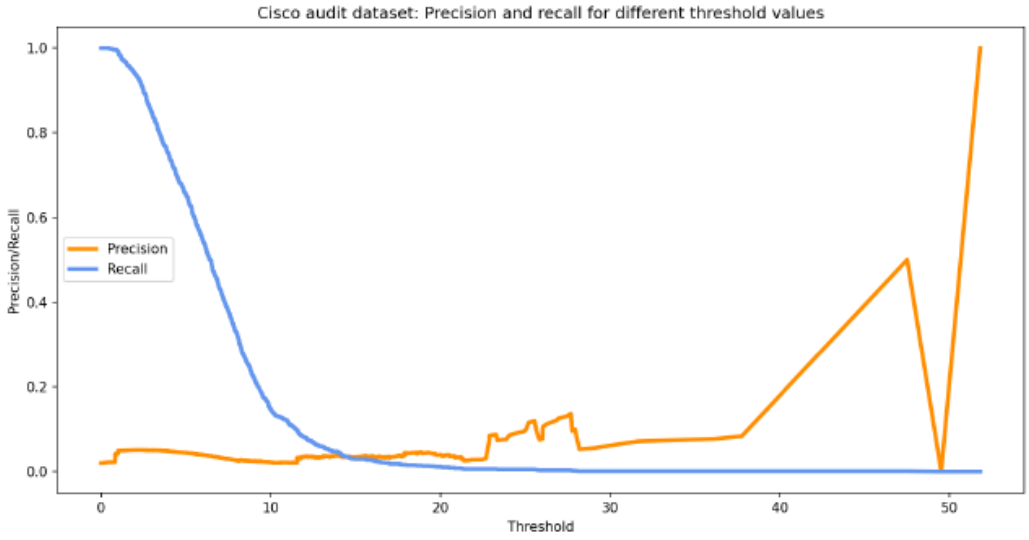


Рисунок 2.6 - Компроміс між точністю та відгуком для різних порогових значень для набору даних аудиту Cisco

Кількість епох була встановлена на рівні 200 із застосуванням механізму Early Stopping, який контролював функцію втрат на валідаційному наборі з паузою в 50 епох. Остаточна модель для набору BETH пройшла всі 200 епох. Графік тренувальних і валідаційних втрат (рис. 2.7) продемонстрував, що валідаційна втрата виявилася нижчою за тренувальну. Це могло бути наслідком використання регуляризації dropout у навчальних даних, що не застосовувалася у валідаційних, викликаючи підвищену втрату під час навчання.

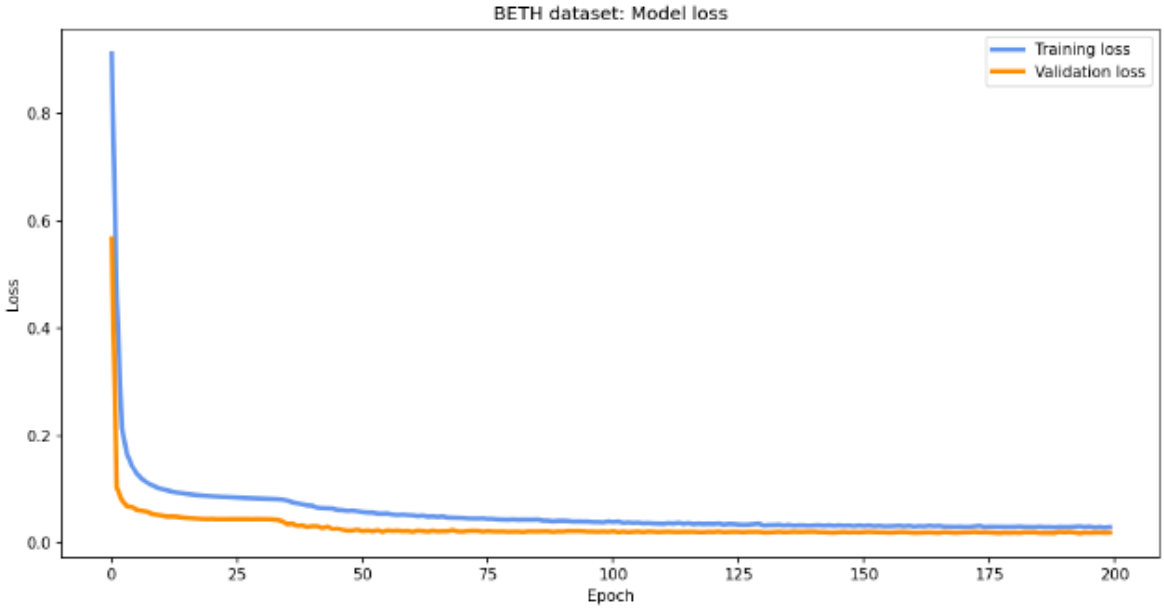


Рисунок 2.7 - Втрати на навчання та валідацію під час останнього навчання перед тестуванням моделі на наборі даних BETH

Остаточна модель для набору Cisco Audit також завершила 200 епох навчання та досягла нижчого рівня втрат як у навчанні, так і у валідації порівняно з моделлю для BETH. Водночас у цьому випадку валідаційна втрата виявилася вищою за тренувальну, що є типовим явищем у процесі навчання нейромереж (рис. 2.8).

Тестування моделі проводилося на повному тестовому наборі BETH, що складався з 188 967 записів, а також на 100 000 записах із набору Cisco Audit. Спосіб поділу даних і вибір підмножин для тестування мали значний вплив на кінцеві результати. Оскільки у вибірках містився або дуже високий відсоток аномалій, або навпаки, мала їх кількість, зосереджена у певні часові періоди,

використання різних підмножин могло суттєво змінювати результати, тому було вирішено застосовувати повні тестові набори.

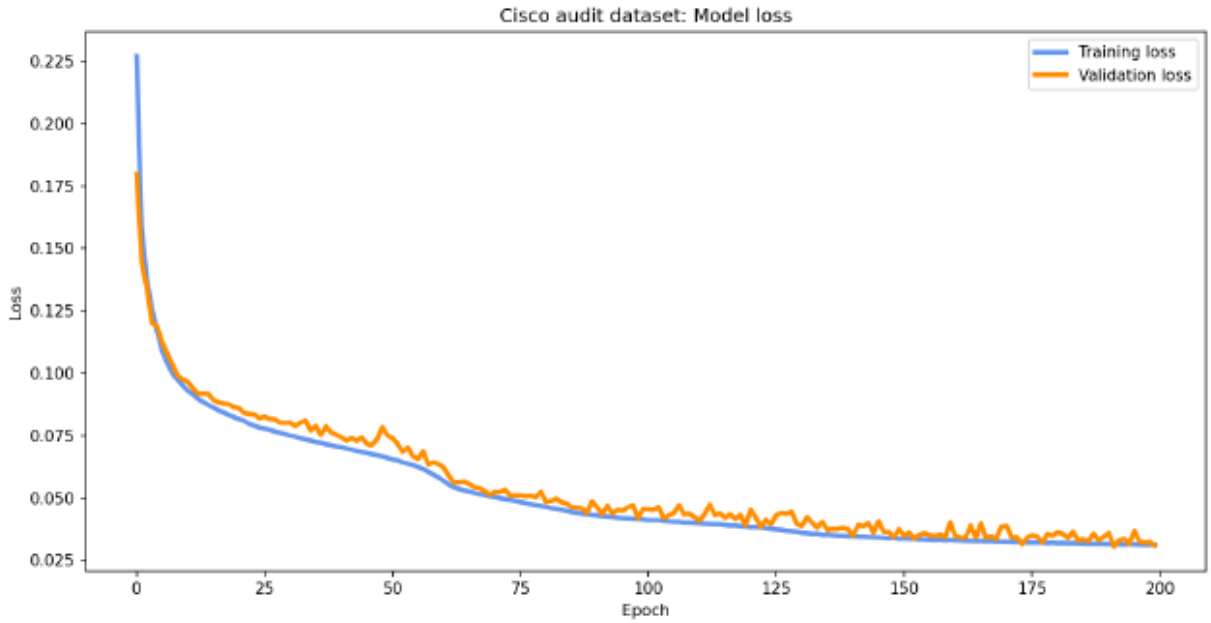


Рисунок 2.8 - Втрати на навчання та валідацію під час останнього навчання перед тестуванням моделі на наборі даних аудиту Cisco

У тестовому наборі BETH аномальні події становили 83,84% загальної кількості, що було враховано при фінальному аналізі. У наборі Cisco Audit цей показник був значно нижчим – лише 1,38%. Для кожної тестової точки було розраховано втрату реконструкції за допомогою метрики MAE. Отримані значення втрат, які визначали аномальність кожної точки, були візуалізовані разом із встановленим порогом аномалій. Графік для BETH продемонстрував, що після 50 000 записів аномальні значення були зосереджені навколо оцінки 30, що дозволило легко їх ідентифікувати. У Cisco Audit аномальні значення поступово переходили в нормальні, що ускладнювало їхнє виявлення.

Візуалізовані результати тестування містили всі точки даних, позначені як Normal (Evil = 0) або *Evil* (Evil = 1), що дало змогу оцінити ефективність виявлення аномалій у кожному з наборів даних.

2.4 Висновки до розділу

У другому розділі було розглянуто методологію розробки та реалізації системи виявлення аномалій у логах з використанням архітектури глибокого навчання на основі LSTM-автоенкодера. Основним завданням, яке вирішувалося в межах цього етапу дослідження, стало створення надійної та чутливої моделі, здатної виявляти відхилення у поведінці логів у реальному або наближеному до реального часовому середовищі.

Було обґрунтувано обраний підхід до виявлення аномалій. У межах цього етапу було проаналізовано особливості нейронних мереж типу LSTM, що здатні працювати з послідовними часовими рядами, та переваги автоенкодерів як механізму самонавчання. Структура запропонованої моделі включала вхідний LSTM-шар, енкодер, латентне представлення, декодер та вихідний шар, що забезпечувало здатність моделі фіксувати довгострокові залежності у даних та точно відтворювати шаблони нормальної поведінки. Конструкція автоенкодера дозволила використовувати похибку реконструкції як основу для обчислення аномальної оцінки, що у свою чергу стало критерієм виявлення відхилень.

Ретельна увага була приділена формуванню та обробці вхідних даних. Набори логів, обрані для навчання, включали дані BETH та Cisco Audit, що суттєво відрізнялися за структурою, змістом і контекстом. У процесі обробки здійснювалося кодування категоріальних ознак, масштабування числових значень та трансформація даних у тривимірні масиви для забезпечення повноцінної роботи LSTM-моделі. Було впроваджено концепцію sliding window для збереження хронологічної послідовності подій, що стало важливим чинником при навчанні на часових рядах.

Для обробки даних Cisco Audit було розроблено спеціальні шаблони, які дозволили виділити ключові атрибути подій та забезпечити уніфіковану структуру логів. Це особливо важливо з огляду на те, що дані Cisco представляли собою реальні корпоративні журнали, що проходили попередню фільтрацію, на відміну від більш узагальненого і симульованого набору BETH. Було також впроваджено

допоміжні поля Sus і Evil, які сприяли маркуванню аномальних і підозрілих записів, що надало змогу точніше оцінити продуктивність моделі.

Наступним кроком стало формування навчального, валідаційного та тестового наборів даних. Було визначено оптимальні пропорції для розподілу даних, які забезпечили як можливість налаштування гіперпараметрів, так і перевірку узагальнювальної здатності моделі. При цьому особливий акцент робився на уникненні «перетікання» інформації між наборами, щоб не допустити некоректного завищення результатів.

Складовою частиною розділу став також процес налаштування гіперпараметрів. Була реалізована процедура оптимізації з використанням алгоритму Hyperband, яка дозволила відібрати найбільш ефективні комбінації параметрів, таких як розмір вікна look back, розмір пакету batch size, кількість епох та інші. У межах цієї оптимізації проводилися багаторазові тестові запуску моделі, а п’ять найкращих конфігурацій пройшли крос-валідацію, що дозволило забезпечити стабільність обраних параметрів на різних підмножинах даних.

Розділ також включав експериментальну оцінку ефективності навчання. Під час навчання модель мала змогу адаптуватися виключно на нормальних логах, що відповідало концепції напівнаглядового підходу — коли модель вивчає лише звичні патерни поведінки, а аномалії виявляє як відхилення від цих патернів. На цьому етапі формувався поріг аномальної оцінки, перевищення якого інтерпретувалося як сигнал про відхилення від норми.

У дослідженні були обґрунтовано вибрані метрики ефективності: точність, повнота, F1-міра, а також хибнопозитивні та хибнонегативні частоти. Саме ці показники дозволили надалі здійснити порівняльний аналіз між різними наборами даних і забезпечити об’єктивну оцінку здатності моделі до виявлення відхилень.

Таким чином, другий розділ демонструє послідовний підхід до обробки даних, побудови та навчання моделі, вибору метрик і підходів до тестування. Застосовані методи забезпечили гнучкість та адаптивність моделі до даних різної природи, а також відкрили перспективи для подальшого вдосконалення системи, зокрема через впровадження потокової обробки логів або гібридних архітектур.

3 ОЦІНКА ДОСТОВІРНОСТІ СИСТЕМИ

3.1 Результат тестування моделі

Оцінювання ефективності моделей базувалося на їхній здатності точно виявляти аномалії в тестових наборах даних, уникаючи помилкової класифікації нормальних точок як аномальні. Основними критеріями оцінки були показники True Positive (TP) та True Negative (TN), які визначали рівень успішності виявлення аномальних і нормальних записів відповідно.

Результати тестування моделі на наборі BETH (рис. 3.1) показали True Positive Rate (TPR) на рівні 83.11% та True Negative Rate (TNR) у 14.45%. False Negative Rate (FNR) склав 0.73%, що означає, що 1 383 аномальні точки не були виявлені. Водночас, False Positive Rate (FPR) досяг 1.71%, що відповідає 3 223 нормальним точкам, помилково класифікованим як аномалії.

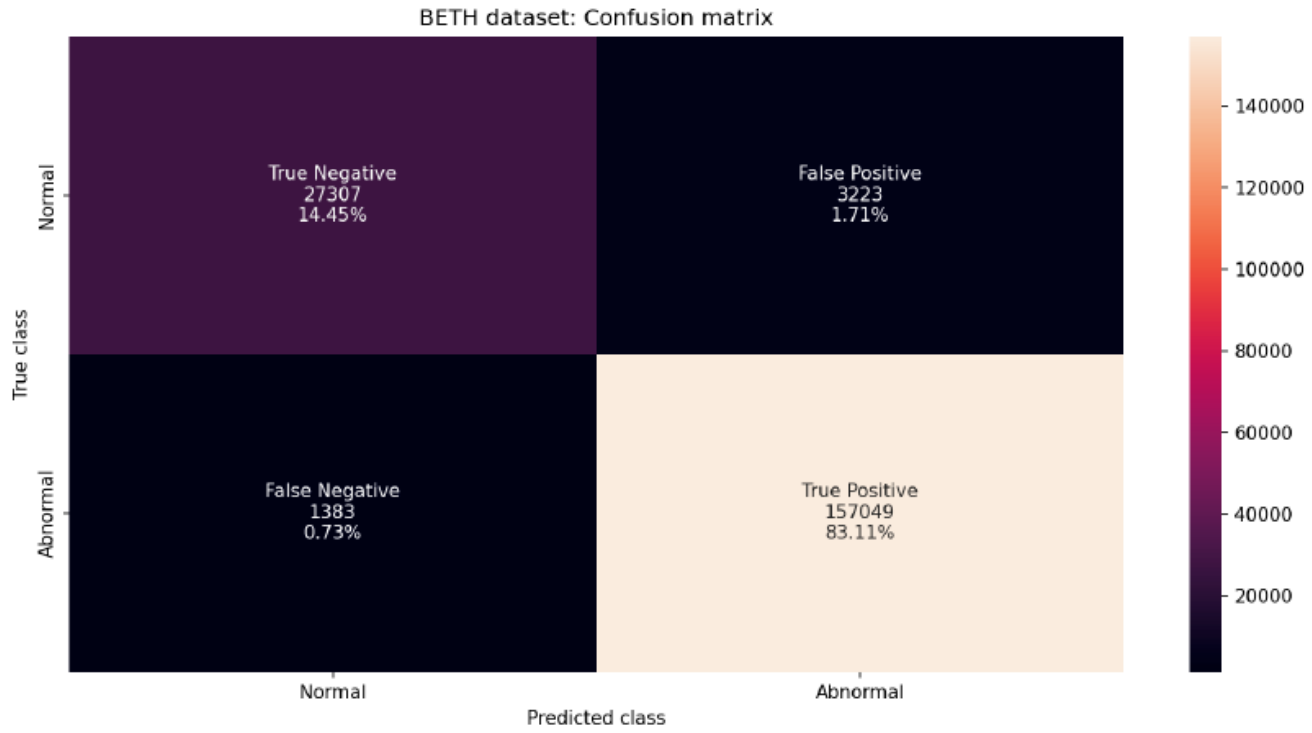


Рисунок 3.1 - Результуюча матриця плутанини, отримана в результаті тестування на тестовому наборі даних BETH

При аналізі результатів тестування моделі на наборі Cisco Audit (рис. 3.2)

виявлено значно нижчі показники ефективності: True Positive Rate склав лише 0.16%, а True Negative Rate – 92.36%. False Negative Rate досяг 1.22%, оскільки більшість аномалій не перевищили встановлений поріг. False Positive Rate становив 6.26%, що означало помилкову класифікацію 6 258 нормальних точок як аномалій.

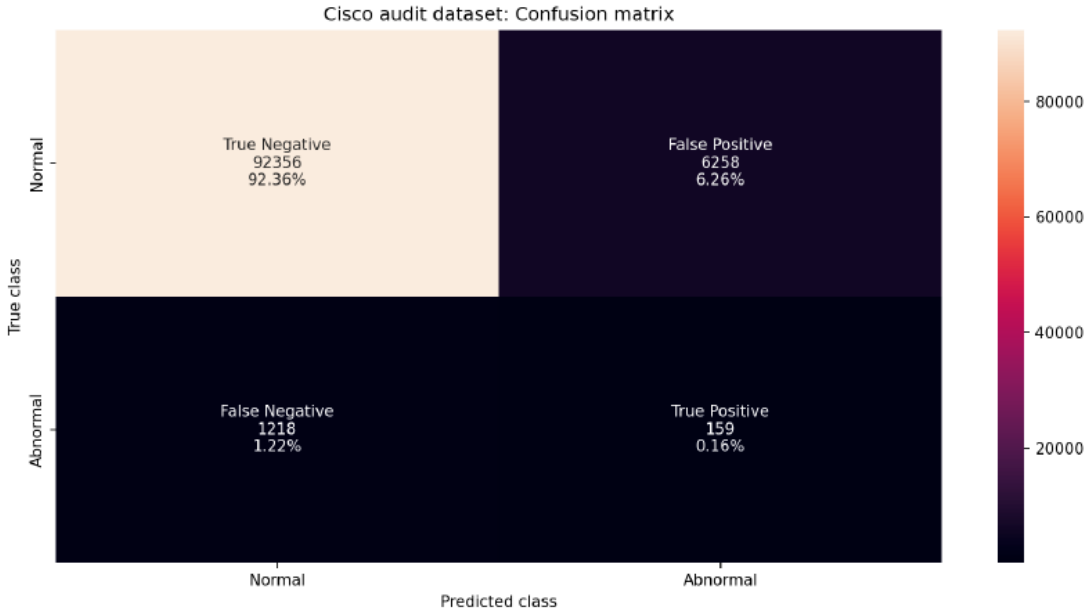


Рисунок 3.2 – Матриця плутанини, отримана в результаті тестування, проведеного на наборі тестових даних для аудиту Cisco

Для набору BETH підсумкові метрики оцінки були такими:

$$Precision_{BETH} = \frac{157049}{157049 + 3223} * 100\% = 97.99\%$$

$$Recall_{BETH} = \frac{157049}{157049 + 1383} * 100\% = 99.13\%$$

$$F - measure_{BETH} = \frac{2 * 0.9799 * 0.9913}{0.9799 + 0.9913} * 100\% = 98.56\%$$

$$FNR_{BETH} = \frac{1383}{157049 + 1383} * 100\% = 0.87\%$$

$$FPR_{BETH} = \frac{3223}{157049 + 3223} * 100\% = 2.01\%$$

Висока точність демонструє, що модель загалом добре розрізняє нормальні

та аномальні події, однак у випадках незбалансованих даних цей показник може бути оманливим. Висока прецизійність (97.99%) свідчить про те, що більшість передбачених аномалій були правильними. Високий показник повноти (99.13%) означає, що модель ефективно виявляла реальні аномалії. Високе значення F-міри (98.56%) підтверджує баланс між точністю та повнотою, а низькі значення FNR і FPR свідчать про високу якість класифікації.

Для Cisco Audit модель показала значно гірші результати:

$$Precision_{Cisco} = \frac{159}{159 + 6258} * 100\% = 2.48\%$$

$$Recall_{Cisco} = \frac{159}{159 + 1218} * 100\% = 11.55\%$$

$$F - measure_{Cisco} = \frac{2 * 0.0248 * 0.1155}{0.0248 + 0.1155} * 100\% = 4.1\%$$

$$FNR_{Cisco} = \frac{1218}{159 + 1218} * 100\% = 88.45\%$$

$$FPR_{Cisco} = \frac{6258}{159 + 6258} * 100\% = 97.52\%$$

Ці результати свідчать про те, що модель значно частіше помилково класифікувала нормальні точки як аномалії, а також пропускала значну кількість реальних аномалій.

Аналіз кривих ROC (рис. 3.3) показав, що для набору BETH площа під кривою (AUC) склала 99.5%, що демонструє здатність моделі ефективно розрізняти нормальні події та аномалії. У випадку Cisco Audit AUC досяг лише 76.6%, що свідчить про значно гіршу якість класифікації аномалій.

Оскільки тестовий набір BETH містив 83.84% аномалій, було вирішено протестувати модель на валідаційному наборі, що складався виключно з нормальних записів. Модель навчалася з порогом аномалії 15, а під час тестування на валідаційному наборі вона правильно класифікувала 99.61% нормальних точок, а 0.39% було помилково позначено як аномальні. Це підтвердило стабільність моделі при роботі як із наборами, що містять велику кількість аномалій, так і з наборами, де їх немає.

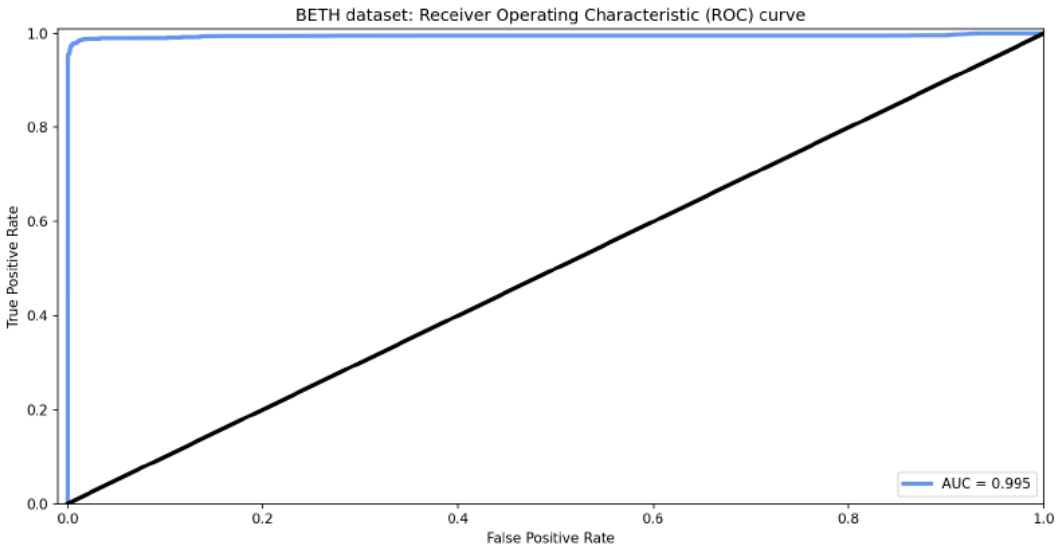


Рисунок 3.3 - ROC-крива з набору даних BETH

Додатковий експеримент був проведений із набором Cisco Audit, де було використано додаткове поле Sus, що відзначало підозрілі події, такі як помилки автентифікації. Це дозволило моделі краще диференціювати аномальні точки. Після повторного визначення порогу було виявлено, що включення Sus дозволило підвищити точність і знизити кількість хибних спрацьовувань. Значення AUC при цьому досягло 99.6%, що перевищило навіть результат моделі для BETH (99.5%).

3.2 Порівняльний аналіз

Для оцінки ефективності моделей було проведено порівняння з дослідженням представленими у іншим роботах, де також використовувався автоенкодер LSTM для виявлення аномалій у логах. У їхньому дослідженні використовувалися дані CERT Insider Threat, які містили лише 0.03% аномалій, що робило їх суттєво незбалансованими.

Модель, розроблена в цьому дослідженні, працювала з двома наборами: BETH (рис. 3.4), що містив 83.84% аномалій, і Cisco Audit, де цей показник складав 1.38%. Результати показали, що модель для BETH продемонструвала стабільно високі показники. Водночас модель для Cisco Audit мала низьку повноту (recall) і

високий коефіцієнт хибних позитивів (FPR = 97.52%), що означало значну кількість неправильно класифікованих нормальних точок.

Додавання допоміжного поля Sus у Cisco Audit покращило результати (рис. 3.5-3.8), однак навіть після цього модель працювала гірше, ніж для BETH та CERT.

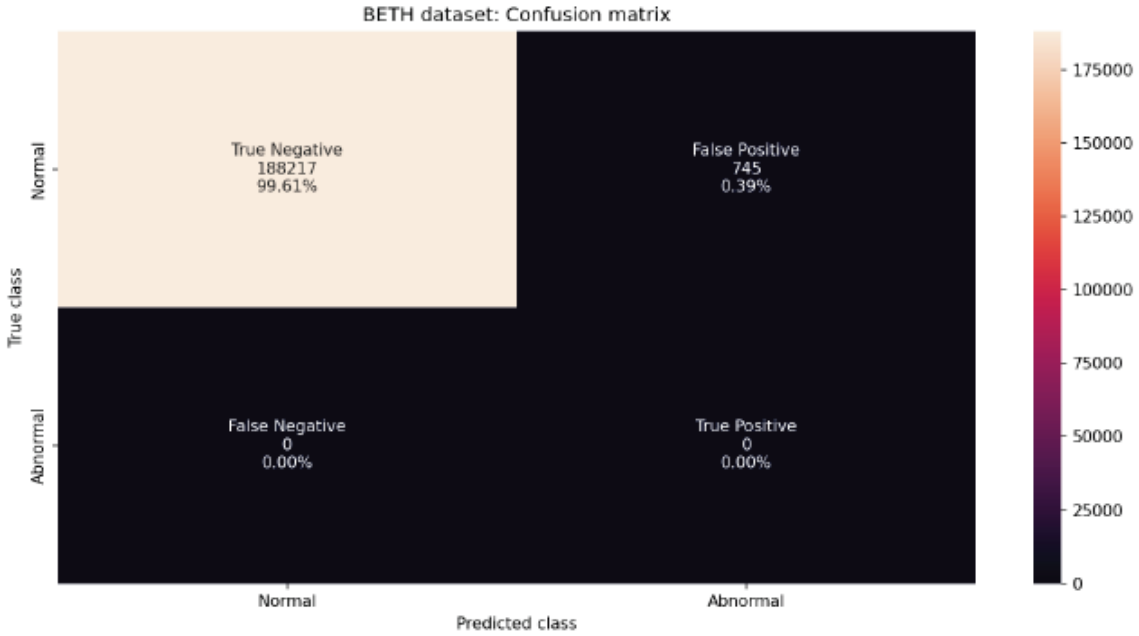


Рисунок 3.4 - Матриця плутанини в результаті тестування, проведеного на наборі даних валідації BETH

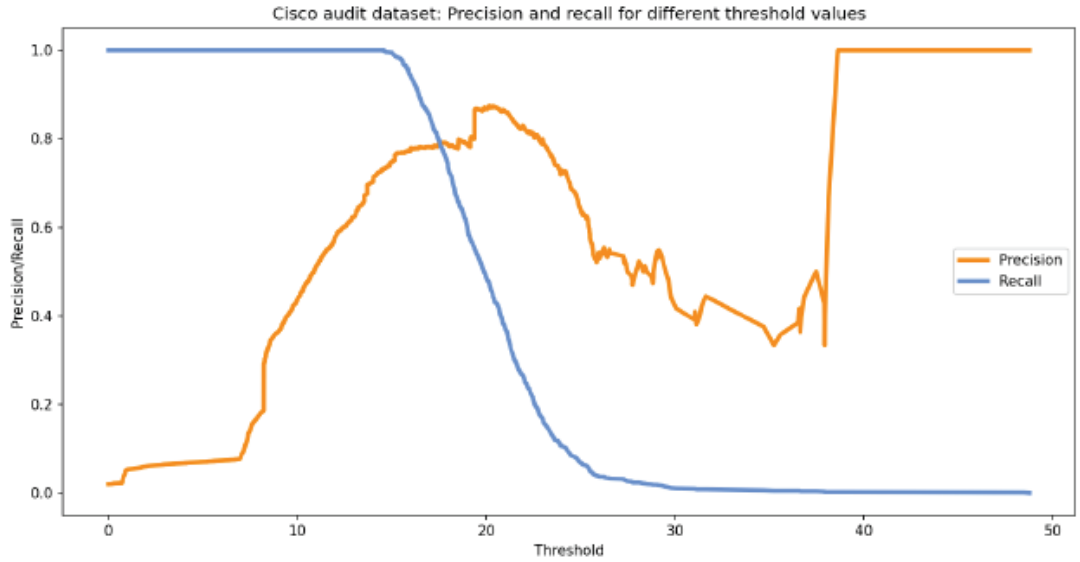


Рисунок 3.5 - Помилки реконструкції для всіх точок даних у наборі даних для аудиту Cisco, включаючи поле «Sus»

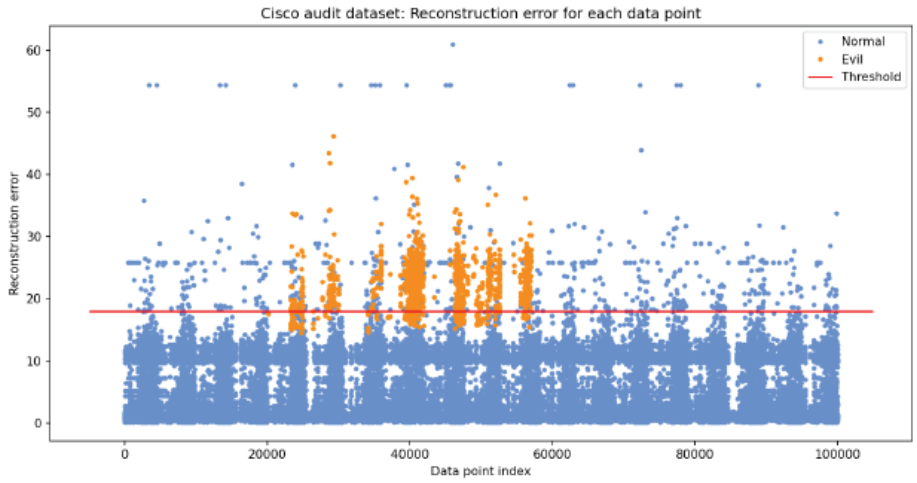


Рисунок 3.6 - Помилки реконструкції для всіх точок даних у наборі даних для аудиту Cisco, включаючи поле «Sus»

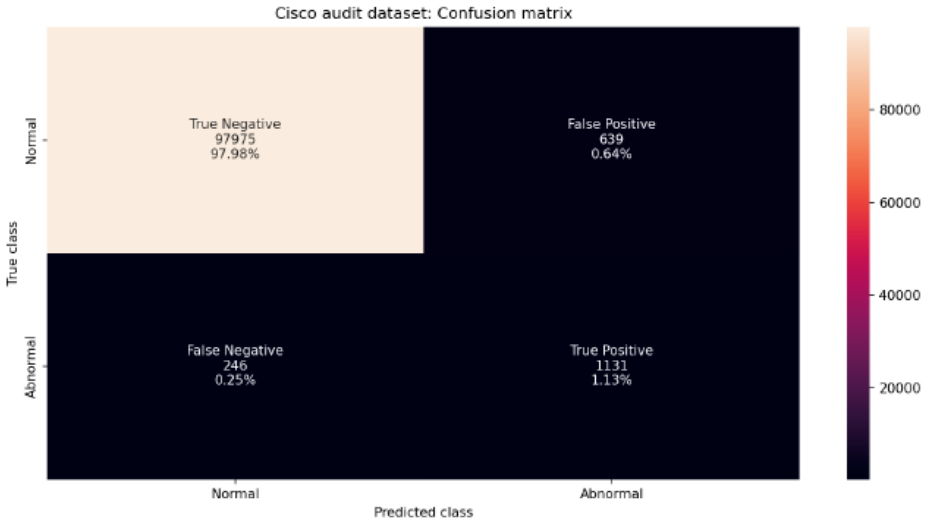


Рисунок 3.7 - Матриця плутанини для набору даних для аудиторського тестування Cisco, включаючи поле «Sus»

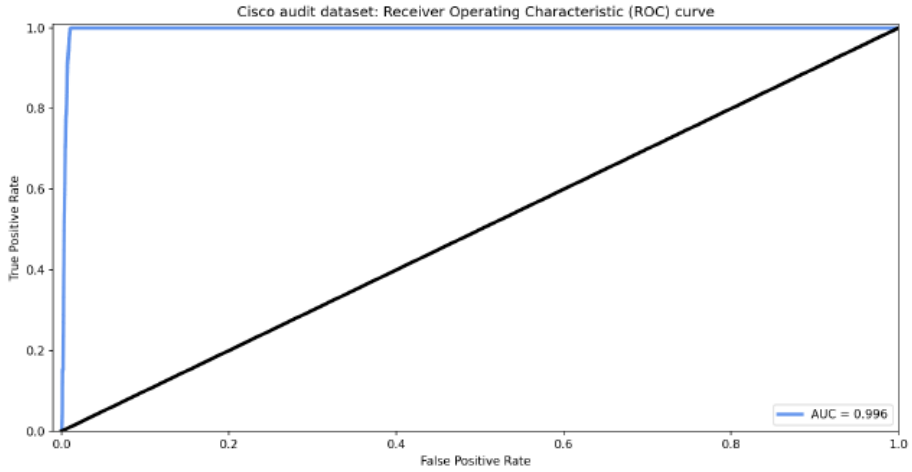


Рисунок 3.8 - ROC-крива для набору даних аудиторського тестування Cisco, включаючи поле «Sus»

Висновки показали, що різниця в продуктивності значною мірою залежала від особливостей набору даних. Набір даних BETH складався з журналів процесів на рівні ядра, які включали дії користувачів на певних хост-серверах, а набір даних аудиту Cisco складався з дій з входу до системи на певних мережевих хостах. Тобто набір BETH був спеціально створений для навчання та тестування моделей виявлення аномалій, тоді як Cisco Audit містив реальні корпоративні логи, які проходили попередню обробку. Відмінності у структурі логів, характері атак та методах попередньої обробки могли суттєво вплинути на кінцеві результати.

Кількість аномальних точок не була ключовим чинником ефективності, оскільки модель для BETH працювала краще, незважаючи на високу частку аномалій, тоді як модель для Cisco Audit мала проблеми навіть при значно меншій кількості аномалій у тестовому наборі.

3.3 Порівняльний аналіз ефективності LSTM-автоенкодера та класичних методів виявлення аномалій із перспективою інтеграції

Для забезпечення об’єктивної оцінки ефективності розробленої моделі автоенкодера на основі рекурентної нейронної мережі з довгою короткочасною пам’яттю (LSTM) було проведено ґрунтовний порівняльний аналіз з класичними методами виявлення аномалій. Такий підхід дозволяє визначити, наскільки нова модель перевершує традиційні алгоритми, що часто застосовуються у сфері кібербезпеки та аналізу лог-файлів. Для базових моделей були обрані три відомі алгоритми без нагляду: One-Class Support Vector Machine (One-Class SVM), Isolation Forest (IF) та метод локальної щільності Local Outlier Factor (LOF). Ці методи широко використовуються у задачах виявлення аномалій завдяки їхній простоті, ефективності та широкій підтримці у різних програмних платформах.

One-Class SVM є класичним підходом, що намагається знайти межу, яка максимально відокремлює нормальні дані від потенційних аномалій, виходячи з теорії опорних векторів. Isolation Forest, навпаки, базується на ідеї ізоляції точок у

випадкових деревах, завдяки чому ефективно виявляє рідкісні випадки навіть у великих наборах даних. LOF застосовує локальну оцінку щільності, дозволяючи ідентифікувати точки, що суттєво відрізняються від своїх сусідів у багатовимірному просторі. Кожен з цих методів має свої переваги і недоліки, що робить їх ідеальними для порівняння з більш складною нейронною моделлю.

Для збереження однорідності експерименту всі моделі було навчено на однаковому наборі логів, що містив різноманітні події системного та мережевого характеру. Перед навчанням проведено детальний етап попередньої обробки даних, зокрема витяг ознак, нормалізація, усунення пропущених значень, а також конвертація часових позначок у відповідний формат. Це дозволило гарантувати, що результати тестування відображають лише різницю в алгоритмах, а не у вхідних даних.

За підсумками тестування автоенкодер LSTM продемонстрував найвищу F1-міру — 0.91, що свідчить про збалансованість між точністю та повнотою виявлення аномалій. Цей результат суттєво перевищує показники класичних методів: Isolation Forest досяг F1-міри на рівні 0.78, а One-Class SVM — 0.74. Така різниця в ефективності пояснюється здатністю нейронної мережі моделювати часові залежності в послідовностях логів. Завдяки цьому вона виявляє контекстуальні аномалії, що можуть проявлятися не у вигляді окремих підозрілих записів, а як відхилення в часовому ряді подій. Класичні методи, що розглядають кожен запис незалежно, не можуть врахувати цей контекст, тому пропускають складні випадки або генерують багато хибнопозитивних спрацьовувань.

Особливу увагу варто звернути на стійкість LSTM-моделі до шуму у даних — явища, що характерне для лог-файлів, які часто містять велику кількість несуттєвих або випадкових подій. Автоенкодер завдяки своїй архітектурі ефективно ігнорує поодинокі випадкові “сплески”, що не впливають на загальний патерн послідовності. Навпаки, метод LOF, що орієнтований на локальні зміни щільності, часто дає багато хибних спрацьовувань, що значно ускладнює роботу аналітиків безпеки та збільшує навантаження на системи реагування. Крім того, класичні методи навчання працюють значно швидше і менш ресурсомісткі, проте

втрачають у точності та здатності адаптуватися до складних і динамічних даних.

Важливо підкреслити, що автоенкодер на основі LSTM не лише показує найкращу загальну ефективність, але й демонструє високу здатність до узагальнення на нових, раніше не бачених даних. Це робить його особливо цінним для застосування у реальних системах моніторингу безпеки, де постійно з'являються нові типи атак і помилок.

Що стосується можливості інтеграції розробленої LSTM-моделі з іншими системами, цей аспект має особливе значення для практичної реалізації. Автоенкодер може бути легко включений як модуль у більші системи комплексного моніторингу безпеки, де він служить інструментом глибокого аналізу подій, доповнюючи традиційні механізми фільтрації. Застосування LSTM-автоенкодера у складі комплексних рішень дозволяє використовувати переваги як класичних моделей (швидкість та простота інтерпретації), так і глибинного навчання (висока точність та здатність виявляти складні патерни). Такий підхід значно підвищує загальну надійність системи.

Важливою складовою є можливість інтегрувати модель у існуючі SIEM-системи (Security Information and Event Management), які вже збирають, агрегують та аналізують велику кількість подій у режимі реального часу. Включення автоенкодера LSTM у ці платформи дає змогу автоматично виявляти аномалії на основі часових закономірностей, які не помітні традиційним засобам. Крім того, модель може виступати як фільтр або попередній етап обробки, знижуючи навантаження на аналітиків, що працюють з підозрілими подіями.

Ще одна перспектива полягає у можливості об'єднання LSTM-автоенкодера з іншими методами в ансамблеву систему, де результати різних моделей обробляються спільно для досягнення кращої збалансованості між точністю та швидкодією. Такі ансамблі можуть комбінувати переваги, наприклад, швидкість Isolation Forest, локальну чутливість LOF і контекстуальне розпізнавання LSTM, що відкриває нові горизонти для побудови високоефективних систем виявлення аномалій. Крім того, сучасні розробки дозволяють впроваджувати модель у хмарні сервіси або використовувати у вигляді мікросервісів із гнучким масштабуванням,

що особливо важливо для великих корпоративних мереж. Автоматизація процесів навчання та оновлення моделі на основі нових даних забезпечує безперервну актуалізацію знань про поведінку системи і адаптацію до нових загроз.

Отже, розроблена LSTM-модель автоенкодера не лише демонструє високі показники ефективності у лабораторних умовах, а й має широкий потенціал для практичного застосування у складних багаторівневих системах кібербезпеки. Її інтеграція з існуючими рішеннями значно підвищує здатність організацій до швидкого виявлення і реагування на різноманітні аномалії та інциденти, що є критично важливим у сучасних умовах зростаючих кіберзагроз.

3.4 Висновки до розділу

Процес дослідження розпочався з аналізу літератури у сфері виявлення аномалій у логах, а також вивчення відповідних нормативних документів і стандартів, особливо в галузі телекомунікацій. Було визначено та узагальнено основні загрози, які такі стандарти мають запобігати. Окрім цього, розглянуто попередні дослідження, присвячені використанню методів глибокого навчання для виявлення аномалій у логах, а також проаналізовано процес обробки логів із цією метою. На основі цього здійснено вибір відповідної моделі глибокого навчання, що потребувало додаткового дослідження. Також було зібрано окремий набір даних для навчання системи виявлення аномалій.

Наступним етапом стало планування процесу впровадження та тестування моделі для виявлення аномалій у логах. Важливим аспектом було визначення способу обробки наборів даних, необхідних для навчання, валідації та тестування моделі. Вибір характеристик логів відбувався з урахуванням експертних думок. Далі набори даних були розібрані, розділені, впорядковані у часовому контексті та масштабовані. Водночас здійснювалось налаштування гіперпараметрів обраної моделі глибокого навчання. Після цього модель була побудована, навчена, валідована та протестована на підготовлених наборах даних, а отримані результати агреговані для подальшої оцінки.

ВИСНОВКИ

У межах цієї кваліфікаційної роботи було здійснено дослідження, присвячене розробці системи виявлення аномалій у логах комп'ютерних систем з використанням глибинної нейронної мережі на основі LSTM-автоенкодера. У роботі послідовно вирішено наукові, технічні й прикладні завдання, що охоплюють аналіз предметної області, створення методу виявлення аномалій, реалізацію моделі нейронної мережі та перевірку її ефективності на реальних наборах даних.

У першому розділі було проведено комплексний аналіз предметної області, структури журналів подій, а також принципи функціонування систем виявлення вторгнень. Увага була зосереджена на класифікації IDS за методами виявлення, а також за джерелами даних. Розглянуто переваги використання логів як джерела даних для виявлення атак, оскільки саме журнали подій містять багату інформацію про поведінку системи. Особливий акцент було зроблено на перевагах глибинного навчання, зокрема на здатності нейронних мереж LSTM виявляти аномалії у часових послідовностях. Обґрунтовано вибір автоенкодера на основі LSTM як архітектури, здатної навчатись на нормальних логах без попереднього маркування аномалій.

У другому розділі докладно описано метод виявлення аномалій, реалізований на основі LSTM-автоенкодера. Модель побудовано так, щоб вона навчалася на послідовностях нормальної поведінки комп'ютерної системи, а аномалії ідентифікувала як відхилення у вигляді значної похибки реконструкції. Було описано процес формування вхідних даних, зокрема розбиття логів на часові вікна за допомогою техніки sliding window. Описано архітектуру автоенкодера: вхідний шар, енкодер, латентне представлення, декодер і вихід. Ретельно опрацьовано гіперпараметри моделі, їх оптимізацію за допомогою Hyperband, та застосовано крос-валідацію. Особливу увагу приділено процесу обробки логів та їх трансформації у формат, придатний для подачі до LSTM-моделі. У цьому ж розділі розглянуто реальні набори даних — BETH та Cisco Audit. Було здійснено підготовку навчальних, валідаційних і тестових підмножин. Набір BETH містить реальні атаки в умовах великомасштабного середовища з кількома хостами-«пастками». Натомість Cisco Audit

відображає типову внутрішню корпоративну активність із моделюванням загроз з боку інсайдерів. Було розроблено методику маркування підозрілих та шкідливих записів за допомогою полів Sus та Evil, що дозволило провести кількісний аналіз ефективності моделі.

У третьому розділі здійснено експериментальну оцінку достовірності побудованої системи виявлення аномалій. Визначено порогові значення для класифікації подій як нормальних або аномальних. Проведено тестування моделі на ізольованому наборі даних із наявністю як нормальних, так і аномальних записів. Було застосовано класичні метрики оцінки ефективності: точність, повнота, F1-міра, а також рівні хибнопозитивних і хибнонегативних спрацювань. Результати показали високу здатність моделі до генералізації та чутливість до нехарактерних подій навіть за відсутності явних ознак атак. Аналіз результатів свідчить про ефективність використання LSTM-автоенкодера для задач виявлення аномалій у логах комп'ютерних систем, особливо в умовах відсутності маркованих даних про атаки. Модель здатна самостійно навчатися на нормальній поведінці та виявляти відхилення без потреби в заздалегідь відомих сигнатурах загроз. Вона може застосовуватись у хост-орієнтованих та мережевих середовищах і має потенціал до інтеграції у системи моніторингу безпеки в режимі наближеного до реального часу.

Серед перспектив подальших досліджень варто виділити розширення моделі для потокового аналізу логів, що дозволить здійснювати моніторинг в реальному часі; впровадження гібридних архітектур на основі поєднання LSTM з трансформерами; а також використання механізмів пояснюваного ШІ для підвищення інтерпретованості результатів. Модель також може масштабуватись для роботи з даними з різною структурою, що важливо для корпоративних мереж з багатокомпонентною IT-архітектурою. Отже, результати кваліфікаційної роботи підтверджують доцільність і практичну значущість застосування глибокого навчання для автоматизованого виявлення аномалій у логах, а також демонструють високий потенціал нейронних мереж на основі LSTM для забезпечення адаптивного моніторингу інформаційної безпеки в умовах сучасних загроз.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Цирканюк Д., Соколов В. Методика розслідування інцидентів інформаційної безпеки. Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка». 2024. Т. 2, № 26. С. 140–154. DOI: 10.28925/2663-4023.2024.26.675

2. Колесник А. М., Ткач М. В. Пошук аномалій в системах автентифікацій за допомогою методів машинного навчання. XXI Всеукраїнська науково-практична конференція студентів, аспірантів та молодих вчених «Теоретичні і прикладні проблеми фізики, математики та інформатики» (Київ, 11–12 травня 2023 р.): матеріали конференції. Київ: КПІ ім. Ігоря Сікорського, 2023. С. 387–391.

3. Gnatyuk S., Berdibayev R., Sydorenko V., Zhyharevych O., Smirnova T. Система корелювання подій та управління інцидентами кібербезпеки на об’єктах критичної інфраструктури. Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка». 2023. Т. 3, № 19. С. 176–196. DOI: 10.28925/2663-4023.2023.19.176196

4. Lakhno V., Blozva A., Husiev B., Osypova T., Matus Y. Інтегрування та захист IoT пристроїв у наявній інфраструктурі комп’ютерної мережі закладу освіти. Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка». 2021. Т. 3, № 11. С. 85–99. DOI: 10.28925/2663-4023.2021.11.8599

5. Що таке логи та як з ними працювати? URL: https://training.qatestlab.com/blog/technical-articles/logging-in-concepts-requirements-levels/ (дата звернення: 05.04.2025).

6. Що таке log file – Терміни та визначення у сфері кібербезпеки. URL: https://www.vpnunlimited.com/ua/help/cybersecurity/log-file (дата звернення: 05.04.2025).

7. Григор’єв Д., Антипенко В. Актуальні тенденції оптимізації процесів логування та моніторингу в хмарних системах. Information Technology: Computer Science, Software Engineering and Cyber Security. 2024. № 2. С. 17–24. DOI: 10.32782/IT/2024-2-3

8. Pynbianglut Hadem, Saikia D. K., Moulik S. An SDN-based Intrusion Detection

System using SVM with Selective Logging for IP Traceback. Computer Networks. 2021. Vol. 191. DOI: 10.1016/j.comnet.2021.108015

9. Zhao Z., Xu C., Li B. A LSTM-Based Anomaly Detection Model for Log Analysis. Journal of Signal Processing Systems. 2021. Vol. 93. Pp. 745–751. DOI: 10.1007/s11265-021-01644-4

10. Abdulganiyu O. H., Ait Tchakoucht T., Saheed Y. K. A systematic literature review for network intrusion detection system (IDS). International Journal of Information Security. 2023. Vol. 22. Pp. 1125–1162. DOI: 10.1007/s10207-023-00682-2

11. Ahmad Z., Shahid Khan A., Wai Shiang C., Abdullah J., Ahmad F. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Transactions on Emerging Telecommunications Technologies. 2021. Vol. 32. Article e4150. DOI: 10.1002/ett.4150

12. Qazi E.-u.-H., Imran M., Haider N., Shoaib M., Razzak I. An intelligent and efficient network intrusion detection system using deep learning. Computers and Electrical Engineering. 2022. Vol. 99. DOI: 10.1016/j.compeleceng.2022.107764

13. Alzahrani A. O., Alenazi M. J. F. Designing a Network Intrusion Detection System Based on Machine Learning for Software Defined Networks. Future Internet. 2021. Vol. 13, No. 5. Article 111. DOI: 10.3390/fi13050111

14. Hidayat I., Ali M. Z., Arshad A. Machine Learning-Based Intrusion Detection System: An Experimental Comparison. Journal of Computational and Cognitive Engineering. 2022. Vol. 2, No. 2. Pp. 88–97.

15. Muhammad A. R., Sukarno P., Wardana A. A. Integrated Security Information and Event Management (SIEM) with Intrusion Detection System (IDS) for Live Analysis based on Machine Learning. Procedia Computer Science. 2023. Vol. 217. Pp. 1406–1415. DOI: 10.1016/j.procs.2022.12.339

16. What Is An Intrusion Detection System (IDS)? URL: https://www.fortinet.com/resources/cyberglossary/intrusion-detection-system (дата звернення: 11.03.2025).

17. Agoramoorthy M., Ali A., Sujatha D., F. M. R. T., Ramesh G. An Analysis of Signature-Based Components in Hybrid Intrusion Detection Systems. 2023 Intelligent

Computing and Control for Engineering and Business Systems (ICCEBS). Chennai, India. 2023. Pp. 1–5. DOI: 10.1109/ICCEBS58601.2023.10449209.

18. Rehman F., Mushtaq F., Zaman H. A Host-based Intrusion Detection: Using Signature-based and AI-driven Anomaly Detection for Enhanced Cybersecurity. 2024 4th International Conference on Digital Futures and Transformative Technologies (ICoDT2). Islamabad, Pakistan. 2024. Pp. 1–7. DOI: 10.1109/ICoDT262145.2024.10740248.

19. Ahmed U., Nazir M., Sarwar A., et al. Signature-based intrusion detection using machine learning and deep learning approaches empowered with fuzzy clustering. Scientific Reports. 2025. Vol. 15. Article 1726. DOI: 10.1038/s41598-025-85866-7

20. Nguyen V. Q., Ngo T. L., Nguyen L. M., Nguyen V. H., Shone N. Deep Nested Clustering Auto-Encoder for Anomaly-Based Network Intrusion Detection. 2023 RIVF International Conference on Computing and Communication Technologies (RIVF). Hanoi, Vietnam. 2023. Pp. 289–294. DOI: 10.1109/RIVF60135.2023.10471853.

21. Acharya T., Annamalai A., Chouikha M. F. Efficacy of CNN-Bidirectional LSTM Hybrid Model for Network-Based Anomaly Detection. 2023 IEEE 13th Symposium on Computer Applications & Industrial Electronics (ISCAIE). Penang, Malaysia. 2023. Pp. 348–353. DOI: 10.1109/ISCAIE57739.2023.10165088.

22. Ayad A. G., Sakr N. A., Hikal N. A. A hybrid approach for efficient feature selection in anomaly intrusion detection for IoT networks. Journal of Supercomputing. 2024. Vol. 80. Pp. 26942–26984. DOI: 10.1007/s11227-024-06409-x

23. Karnan L., Mahalakshmi S. B., T. V. Hybrid Deep Learning Cloud Intrusion Detection. 2024 Third International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS). Krishnankoil, India. 2024. Pp. 1–6. DOI: 10.1109/INCOS59338.2024.10527729.

24. Satilmiş H., Akleylek S., Tok Z. Y. A Systematic Literature Review on Host-Based Intrusion Detection Systems. IEEE Access. 2024. Vol. 12. Pp. 27237–27266. DOI: 10.1109/ACCESS.2024.3367004.

25. Rastogi R., Yadav G., Sharma J., Singhwall J., Gupta M. Statistical Surveillance for Host-Based Intrusion Detection System (HIDS): An Intelligent System for Automation. In: Devi V. A. (eds) Sustainable IoT and Data Analytics Enabled

Machine Learning Techniques and Applications. Springer, Singapore. 2024. DOI: 10.1007/978-981-97-5365-9_5.

26. Du J., Yang K., Hu Y., Jiang L. NIDS-CNNLSTM: Network Intrusion Detection Classification Model Based on Deep Learning. IEEE Access. 2023. Vol. 11. Pp. 24808–24821. DOI: 10.1109/ACCESS.2023.3254915.

27. Nalini N., Chaudhary A., Surendran S., Muthuraja M., Ahmed I., N. T. J. Network Intrusion Detection System for Feature Extraction Based on Machine Learning Techniques. 2023 5th International Conference on Inventive Research in Computing Applications (ICIRCA). Coimbatore, India. 2023. Pp. 440–445. DOI: 10.1109/ICIRCA57980.2023.10220789.

28. Pillai S. E. V. S., Vallabhaneni R., Pareek P. K., Dontu S. Strengthening Cybersecurity using a Hybrid Classification Model with SCO Optimization for Enhanced Network Intrusion Detection System. 2024 International Conference on Distributed Computing and Optimization Techniques (ICDCOT). Bengaluru, India. 2024. Pp. 1–9. DOI: 10.1109/ICDCOT61034.2024.10516247.

29. Li W., Varakantham P. Unsupervised Training Sequence Design: Efficient and Generalizable Agent Training. Proceedings of the AAAI Conference on Artificial Intelligence. 2024. Vol. 38, No. 12. Pp. 13637–13645. DOI: 10.1609/aaai.v38i12.29268

30. Liang J., Zhang S., Zhao R., Wu Y., Liu Y., Pan S. Omni-Frequency Channel-Selection Representations for Unsupervised Anomaly Detection. IEEE Transactions on Image Processing. 2023. Vol. 32. Pp. 4327–4340. DOI: 10.1109/TIP.2023.3293772.

31. Zhang T., Qiu H., Castellano G., Rifai M., Chen C. S., Pianese F. System Log Parsing: A Survey. IEEE Transactions on Knowledge and Data Engineering. 2023. Vol. 35, No. 8. Pp. 8596–8614. DOI: 10.1109/TKDE.2022.3222417.

32. Ma J., Liu Y., Wan H., Sun G. Automatic Parsing and Utilization of System Log Features in Log Analysis: A Survey. Applied Sciences. 2023. Vol. 13, No. 8. Article 4930. DOI: 10.3390/app13084930.

33. Das S., Tariq A., Santos T., Kantareddy S. S., Banerjee I. Recurrent Neural Networks (RNNs): Architectures, Training Tricks, and Introduction to Influential

Research. In: Colliot O. (eds) Machine Learning for Brain Disorders. Neuromethods. 2023. Vol. 197. Humana, New York. DOI:10.1007/978-1-0716-3195-9_4.

34. Mienye I. D., Swart T. G., Obaido G. Recurrent Neural Networks: A Comprehensive Review of Architectures, Variants, and Applications. Information. 2024. Vol. 15, No. 9. Article 517. DOI:10.3390/info15090517.

35. Saravanan V., Madiajagan M., Rafee S. M., et al. IoT-based blockchain intrusion detection using optimized recurrent neural network. Multimedia Tools and Applications. 2024. Vol. 83. Pp. 31505–31526. DOI: 10.1007/s11042-023-16662-6.

36. Malashin I., Tynchenko V., Gantimurov A., Nelyub V., Borodulin A. Applications of Long Short-Term Memory (LSTM) Networks in Polymeric Sciences: A Review. Polymers. 2024. Vol. 16, No. 18. Article 2607. DOI: 10.3390/polym16182607.

37. Zhou H., Kang L., Pan H., et al. An intrusion detection approach based on incremental long short-term memory. International Journal of Information Security. 2023. Vol. 22. Pp. 433–446. DOI: 10.1007/s10207-022-00632-4.

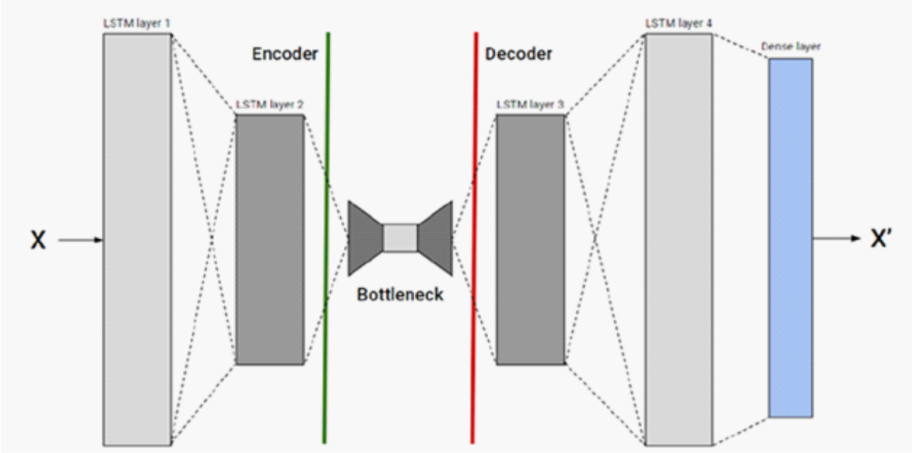
38. Bahlali A. R., Bachir A., Cheriet A. Malicious Encrypted Network Traffic Detection Using Deep Auto-Encoder with a Custom Reconstruction Loss. 2023 International Symposium on Networks, Computers and Communications (ISNCC). Doha, Qatar. 2023. Pp. 1–7. DOI: 10.1109/ISNCC58260.2023.10323710.

39. Kan D., Fang X. Event log anomaly detection method based on auto-encoder and control flow. Multimedia Systems. 2024. Vol. 30. Article 29. DOI: 10.1007/s00530-023-01199-3.

40. Li Z., Huang C., Qiu W. An intrusion detection method combining variational auto-encoder and generative adversarial networks. Computer Networks. 2024. Vol. 253. DOI: 10.1016/j.comnet.2024.110724.

41. СОУ 207.01:2017. Текстові документи. Загальні вимоги. Хмельницький: ХНУ, 2017. 46 с. URL: https://msn.khnu.km.ua/pluginfile.php/466522/mod_resource/content/1/132_C%20T%20A%20H%20Д%20A%20P%20T%20чист%20.pdf

Структура автокодувальника LSTM



Таблиця 2.1 - Опис і тип кожної функції в журналах процесів на рівні ядра в наборі даних BETH

Характеристика	Тип	Опис
timestamp	float	секунди з моменту завантаження системи
processid*	int	ідентифікатор процесу, що створив цей запис
threadid	int	ідентифікатор потоку, що створив цей запис
parentprocessid*	int	ідентифікатор батьківського процесу
userid*	int	ідентифікатор користувача, що виконав процес
mountnamespace*	int (long)	обмеження монтування, в яких працює цей процес
processname	string	виконуваний командний рядок
hostname	string	ім'я сервера
eventid*	int	ідентифікатор події, що згенерувала цей запис
eventname	string	назва події, що згенерувала цей запис
argsnum*	int	довжина аргументів
returnvalue*	int	значення, повернене цією подією (зазвичай 0)
stackaddresses	list of int	пам'ятні значення, пов'язані з процесом
args	list of dictionaries	список аргументів, переданих цьому процесу
sus	int (0 або 1)	бінарна мітка підозрілої події (1 – підозріла, 0 – ні)
evil	int (0 або 1)	бінарна мітка відомої шкідливої події (0 – безпечна, 1 – ні)

Таблиця 2.4 – Розміри наборів даних, що було використано

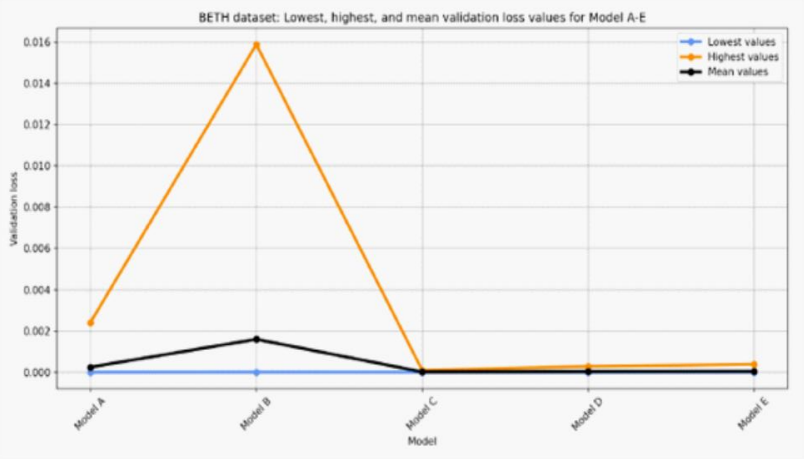
Розмір набору даних		
набір даних	BETH	Cisco audit
навчальний	20 000	50 000
валідаційний	2 000	5 000
тестовий	188 967	100 000

					КРКБ.220162.22.01.05 Е8		
Зм.	Арк.	№ докум.	Підпис	Дата	Система виявлення аномалій у журналах безпеки мережевої інфраструктури інтернет провайдера	Літера	Маса	Масштаб
Розроб.		Купачук О.Р.				Н		
Перевір.		Кльоц Ю.П.			Структура автокодувальника LSTM			
Т. контр.						Аркуш	Аркушів 1	
Н. контр.		Мостовий С.В.				ХНУ, КБс-22-1		
Затв.		Кльоц Ю.П.						

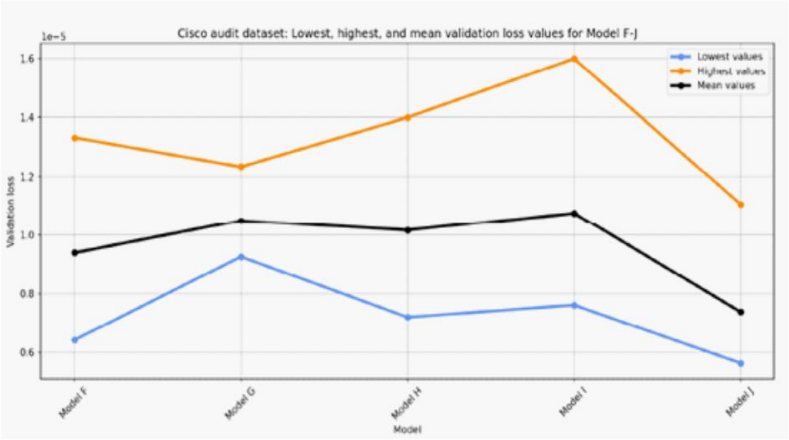
ДОДАТОК А

Копії графічної частини

Найнижча, найвища та середня втрата валідності при k-кратній перехресній перевірці, виконаній для конфігурацій гіперпараметрів на наборі даних BETH



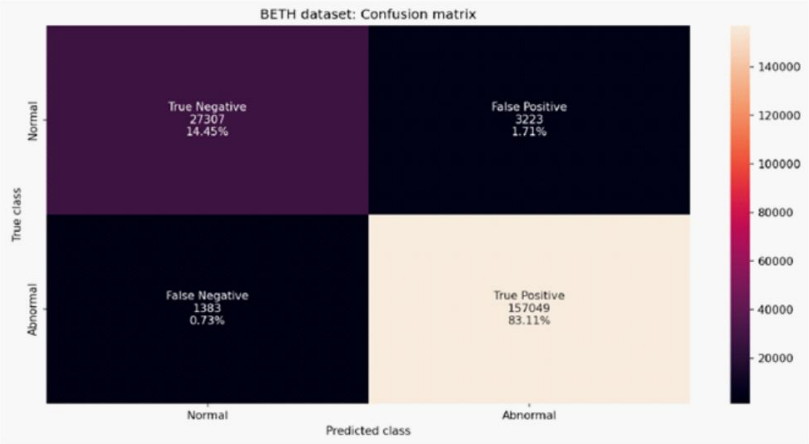
Найнижча, найвища та середня втрата валідації при k-кратній перехресній перевірці, виконаній на конфігураціях з гіперпараметрами з набору даних аудиту Cisco



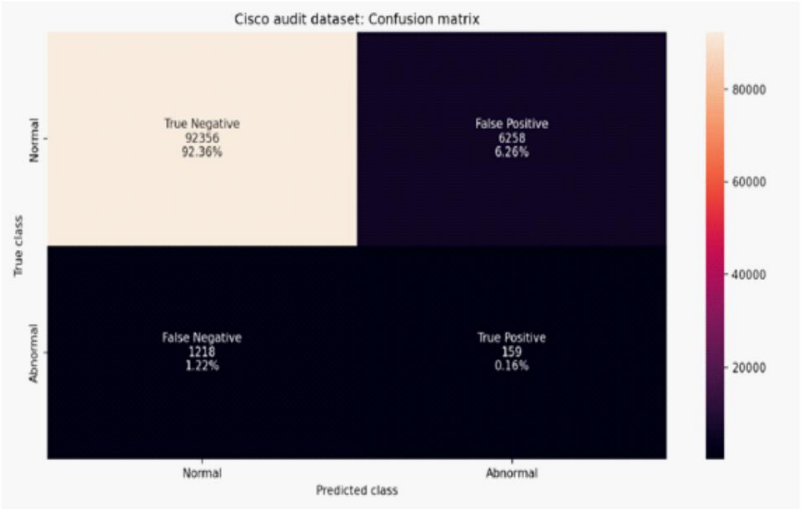
Попередня обробка наборів даних

					КРКБ.220162.22.01.05 Е8		
Зм.	Арк.	№ докум.	Підпис	Дата	Система виявлення аномалій у журналах безпеки мережевої інфраструктури інтернет провайдера	Літера	Маса	Масштаб
Розроб.		Купачук О.Р.				Н		
Перевір.		Кльоц Ю.П.			Попередня обробка наборів даних			
Т. контр.						Аркуш	Аркушів 1	
Н. контр.		Мостовий С.В.				ХНУ, КБс-22-1		
Затв.		Кльоц Ю.П.						

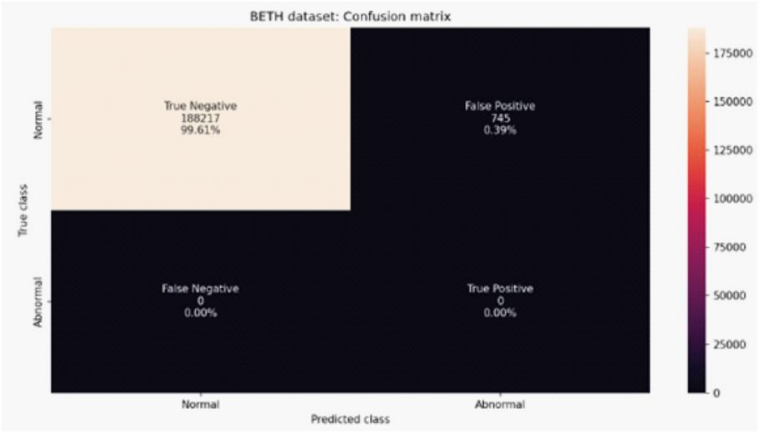
Результуюча матриця плутанини, отримана в результаті тестування на тестовому наборі даних BETH



Матриця плутанини, отримана в результаті тестування, проведеного на наборі тестових даних для аудиту Cisco



Матриця плутанини в результаті тестування, проведеного на наборі даних валідації BETH



					КРКБ.220162.22.01.05 Е8		
					Система виявлення аномалій у журналах безпеки мережевої інфраструктури інтернет провайдера	Літера	Маса	Масштаб
Зм.	Арк.	№ докум.	Підпис	Дата		Н		
Розроб.		Купачук О.Р.			Матриці плутанини	Аркуш	Аркушів 1	
Перевір.		Кльоц Ю.П.						
Т. контр.								
Н. контр.		Мостовий С.В.				ХНУ, КБс-22-1		
Затв.		Кльоц Ю.П.						

Завідувачу кафедри кібербезпеки
к.т.н., доц. Кльоцу Ю.П.
Кулачук Ольги Романівни
Студентки ФІТ, 3 курсу, групи КБс-22-1

ЗАЯВА

З правилами чинного Положення про систему забезпечення академічної доброчесності у Хмельницькому національному університеті, згідно з яким виявлення академічного плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту і застосування заходів дисциплінарної та академічної відповідальності, ознайомлена. Про використання програмно-технічних засобів для перевірки кваліфікаційних робіт здобувачів вищої освіти на наявність академічного плагіату оповіщена та надаю свою згоду на обробку й збереження університетом моєї роботи в інституційному репозитарії Хмельницького національного університету.

Також надаю університету право на передачу моєї роботи для обробки та збереження в базах даних програмно-обчислювального комплексу StrikePlagiarism та/або програмнотехнічного засобу Anti-Plagiarism) і використання роботи для виявлення академічного плагіату в інших роботах, які перевіряються програмно-технічними засобами та користувачами, що мають доступ до цих програмно-технічних засобів, виключно в обмежених цілях для виявлення текстових збігів в роботах.

Робота надається для перевірки в електронному варіанті. Електронна версія моєї роботи збігається (ідентична) з друкованою.

05.06.2025
дата

підпис

Anti-Plagiarism (UA) v-15.281 Educational

The maximum coincidence with one document 1.0%

Dictionaries check: en_US, ru_RU, ua_UA. **Errors in the documents: 11%**

ID: 243861 Title: Система виявлення аномалій у журналах безпеки мережевої інфраструктури інтернет провайдера Added in a DB: 2025-06-06 Authors: Кулачук Ольга Романівна Heads: Кльоц Ю.П. Consultants: Opponents:	Document		Sum coincidence on the DB	
	Symbols	Lexemes	Symbols	Lexemes
	89708	682	1277 (1%)	19 (3%)

Plagiarism sources

ID	Description	Plagiarism presence in the document	
		Symbols	Lexemes

Протокол аналізу звіту подібності науковим керівником

Заявляю, що я ознайомився (-лась) з Повним звітом подібності, який був згенерований Системою виявлення і запобігання плагіату щодо роботи:

Автор: Кулачук Ольга Романівна

Співавтор:

Назва: Система виявлення аномалій у журналах безпеки мережевої інфраструктури інтернет провайдера

Науковий керівник:

Підрозділ: Кафедра кібербезпеки

Коефіцієнт подібності 1:2.3%

Коефіцієнт подібності 2:0.6%

Мікропробіли: 0

Заміна букв: 0

Інтервали: 0

Білі знаки: 0

Дата створення звіту: 2025-06-06 11:23:23.0

Після аналізу Звіту подібності констатую наступне:

☑ **Запозичення, виявлені в роботі є законними і не є плагіатом. Рівень подібності не перевищує допустимої межі. Таким чином робота незалежна і приймається.**

☐ **Запозичення не є плагіатом, але перевищено граничне значення рівня подібностей. Таким чином робота повертається на доопрацювання.**

☐ **Виявлено запозичення і плагіат або навмисні текстові спотворення (маніпуляції), як передбачувані спроби укриття плагіату, які роблять роботу невідповідною вимогам законодавства (Ст. 32. ЗУ Про вищу освіту, пункт 3.1, Ст. 42. ЗУ Про освіту) та вимог НАЗЯВО (Критерій 5), а також кодексу етики і процедурам. Таким чином робота не приймається.**

Обгрунтування:

07.06.2025р.

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 виявлення аномалій у журналах безпеки мережевої інфраструктури інтернет провайдера

Автор: Кулачук Ольга Романівна

Спеціальність: 125 – Кібербезпека

Освітня програма:Кібербезпека

Науковий керівник:Юрій Кльоц, канд. техн. наук, доцент

Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом (далі – зазначаються підстави віднесення запозичень до правомірних). 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 (далі – зазначаються детальні та аргументовані підстави віднесення запозичень до правомірних). 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.	
5	Інше:	

Підтвердження:

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

1) запозичення розміщені в розділах аналізу існуючих методів та технологій, які не описують безпосередньо авторське дослідження і не стосуються результатів роботи;
2) усі запозичення фрагментарні, або мають належним чином оформленні посилання;
3) всі зафіксовані системою ознаки модифікації тексту відносяться до комбінування латинських символів зі україномовними скороченнями індексів в формулах, що не є модифікацією тексту.

Сумарний обсяг всіх запозичень, визначений системою виявлення збігів/ ідентичності/схожості, складає 2.3%, що, з урахуванням наведених обґрунтувань, відповідає характеру наукового дослідження і свідчить на користь кваліфікаційної роботи.

Керівник роботи _____ Юрій КЛЬОЦ

Завідувач кафедри Кб _____ Юрій КЛЬОЦ

Гарант ОП _____ Віктор ЧЕШУН

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

освітнього ступеня «бакалавр»

Студент Кулачук Ольга Романівна

Тема Система виявлення аномалій у журналах безпеки мережевої інфраструктури інтернет-провайдера

Спеціальність 125 – Кібербезпека

Обсяг кваліфікаційної роботи освітньо-кваліфікаційного рівня «бакалавр»:

кількість листів креслень 3; кількість сторінок записки 62.

1. Короткий зміст роботи та прийнятих рішень У роботі розроблено систему виявлення аномалій у логах безпеки мережевої інфраструктури з використанням LSTM-автоенкодера. Авторка проаналізувала існуючі IDS-рішення, дослідила журнали BETH та Cisco Audit, здійснила формування наборів даних, навчання та тестування моделі. Застосовано сучасні підходи глибинного навчання для аналізу часових рядів та визначення аномалій через похибку реконструкції. Проведено оцінку моделі за метриками точності, повноти, F1-міри та похибки.

2. Висновок про відповідність кваліфікаційної роботи завданню Кваліфікаційна робота повністю відповідає завданню. Поставлені цілі виконані, як у теоретичному, так і в практичному аспектах, з дотриманням вимог до оформлення та структури пояснювальної записки.

3. Характеристика виконання кожного розділу роботи, ступінь використання останніх досягнень науки і техніки і передових методів роботи: У вступі обгрунтовано актуальність теми, визначила мету, завдання. Перший розділ присвячено глибокому аналізу предметної області. Розглянуто сутність логів як джерела інформації для виявлення інцидентів безпеки, надала повну класифікацію систем виявлення вторгнень, а також проведено огляд сучасних підходів до виявлення аномалій. У другому розділі запропоновано власну архітектуру LSTM-автоенкодера для виявлення аномалій, описано етапи формування навчальних вибірок на основі реальних логів, визначено методи попередньої обробки даних, та детально викладено процес налаштування гіперпараметрів моделі. Третій розділ містить результати тестування системи: аналіз точності, повноти, F1-міри, а також порівняння з традиційними підходами.

4. Позитивні сторони Робота містить практичну реалізацію моделі, що є перевагою порівняно з суто теоретичними дослідженнями. Програмна частина проєкту може бути адаптована для реального використання в мережах інтернет-провайдерів або корпоративних інформаційних системах. Виклад матеріалу є послідовним, чітким та логічно впорядкованим, що полегшує сприйняття навіть складних технічних концепцій.

5. Негативні сторони роботи У роботі подано обмежену кількість графічних матеріалів, що ускладнює візуальне сприйняття складних технічних процесів, описаних у тексті.

6. Оцінка графічного оформлення та пояснювальної записки роботи Графічне оформлення кваліфікаційної роботи відповідає темі роботи та виконане з дотриманням стандартів. В цілому, графічне оформлення є якісним, а пояснювальна записка відповідає нормам оформлення.

7. Відгук про роботу в цілому Кваліфікаційна робота заслуговує позитивної оцінки, оскільки весь матеріал роботи є структурованим, чітким та послідовним. Усі розділи роботи мають логічну послідовність, що сприяє зрозумінню викладеного матеріалу в рамках теми роботи.

8. Інші зауваження

9. Оцінка кваліфікаційної роботи Враховуючи всі позитивні та негативні сторони представленої кваліфікаційної роботи, можна зробити висновок, що вона заслуговує оцінки «задовільно».

РЕЦЕНЗЕНТ (прізвище, ім'я, по батькові, посада, місце роботи)
Бойко Юлій Миколайович,
доктор технічних наук, професор, професор кафедри телекомунікацій, медійних та інтелектуальних технологій

« 06 » червня 2025