Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук (або центр післядипломної освіти, або навчально-науковий центр заочної форми навчання)
(повна назва)

Кафедра програмної інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження методів нейронних мереж для виявлення шахрайства
(тема)

Виконав:
студент (ка) 2 курсу, групи ІПЗм-22-4

Скримінський Н. О.
(прізвище, ініціали)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва спеціальності)
Тип програми освітньо-наукова

Керівник доц. каф. ПІ, Афанасьєва І.В.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри _____ З.В.Дудар
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук (або центр післядипломної освіти, або навчально-науковий центр заочної форми навчання)
Кафедра програмної інженерії
Рівень вищої освіти другий (магістерський)
Спеціальність 121 – Інженерія програмного забезпечення
Тип програми освітньо-наукова програма
Освітня програма Інженерія програмного забезпечення
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«___» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Скримінському Нікіті Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи «Дослідження методів нейронних мереж для виявлення шахрайства»

Затверджена наказом по університету від 29.03. 2024р. № 250 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 17.06.2024

3. Вихідні дані до роботи дослідити методи нейронних мереж для виявлення шахрайства, середовище розробки VS Code 2023, мова програмування Python, бібліотеки: Numpy, Sklearn, PyTorch, Adam, HOA, DGL

4. Перелік питань, що потрібно опрацювати в роботі
мета роботи, аналіз предметної галузі і постановка задачі, огляд та аналіз літературних джерел з дослідження, дослідження теоретичне, дослідження практичне.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз предметної галузі	23.01.2024 – 14.02.2024	*Виконано*
2	Розробка постановки задачі	14.02.2024 – 21.03.2024	*Виконано*
3	Розробка ПЗ	21.03.2024 – 02.04.2024	*Виконано*
4	Проведення експерименту	02.04.2024 – 23.04.2024	*Виконано*
5	Оформлення пояснювальної записки	01.05.2024 – 26.05.2024	*Виконано*
6	Підготовка презентації та доповіді	26.04.2024 – 02.05.2024	*Виконано*
7	Нормоконтроль, рецензування	02.06.2024 – 12.06.2024	*Виконано*
8	Попередній захист	12.06.2024	*Виконано*
9	Здача роботи у електрониий архів	15.06.2024	*Виконано*
10	Допуск до захисту у зав. кафедри	17.06.2024	*Виконано*

Дата видачі завдання 30.03.2024 р.

Студент _____ Скримінський Н. О.
(підпис)

Керівник роботи _____ доц. каф. ПІ, Афанасьєва І.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ ABSTRACT

Робота містить: 69 сторінок, 19 рисунків, 7 таблиць, 20 джерел, 6 додатків.

БАНКІВСЬКА ГАЛУЗЬ, ЕФЕКТИВНІСТЬ, НЕЙРОННІ МЕРЕЖІ, РИЗИКИ ШАХРАЙСТВА, ФІНАНСОВА БЕЗПЕКА.

Об'єктом дослідження є використання нейронних мереж для виявлення шахрайства в банківській сфері. Робота присвячена огляду та аналізу методів і технологій, що використовуються для виявлення аномалій у фінансових операціях з метою запобігання шахрайству та підтримки фінансової безпеки клієнтів і банківських установ.

Метою роботи є аналіз різних типів нейронних мереж, їх архітектур та методів навчання з метою визначення оптимального підходу до виявлення шахрайства. Основними завданнями є дослідження та порівняння різних підходів, визначення їх ефективності та розробка рекомендацій щодо використання нейронних мереж у практичному застосуванні для виявлення шахрайства.

Методи розробки базуються на таких технологіях, Python, бібліотеки: Numpy, Sklearn, PyTorch.

У роботі представлено детальний аналіз та дослідження методів виявлення шахрайства, в тому числі з використанням нейронних мереж, у банківській сфері. Дослідження охоплює розвиток цифрових фінансових послуг, аналіз викликів та ризиків шахрайства з банківськими картками, а також розглядає технологічні вразливості та кіберзагрози, які можуть загрожувати фінансовій безпеці.

Дослідження та аналіз показують, що використання нейронних мереж є ефективним засобом виявлення шахрайства в банківській сфері. Глибоке навчання дозволяє моделям адаптуватися до різних шаблонів і виявляти аномалії, що робить їх важливим інструментом у сучасному фінансовому середовищі.

BANKING INDUSTRY, EFFICIENCY, NEURAL NETWORKS, FRAUD RISKS, FINANCIAL SECURITY.

The object of research is the use of neural networks to detect fraud in the banking sector. The work is devoted to the review and analysis of methods and technologies used to detect anomalies in financial transactions in order to prevent fraud and maintain financial security of customers and banking institutions.

The aim of the work is to analyse different types of neural networks, their architectures and training methods in order to determine the optimal approach to fraud detection. The main tasks are to study and compare different approaches, determine their effectiveness and develop recommendations for the use of neural networks in practical applications for fraud detection.

The development methods are based on the following technologies, Python, libraries: Numpy, Sklearn, PyTorch.

The paper presents a detailed analysis and research of fraud detection methods, including those using neural networks, in the banking sector. The study covers the development of digital financial services, analysis of challenges and risks of bank card fraud, and also considers technological vulnerabilities and cyber threats that may threaten financial security.

Research and analysis show that the use of neural networks is an effective means of detecting fraud in the banking sector. Deep learning allows the models to adapt to different patterns and detect anomalies, making them an important tool in today's financial environment.

Я, Скримінський Нікіта Олександрович студент групи ІПЗм-22-4, здобувач освіти на другому (магістерському) рівні кафедра програмної інженерії, заявляю: моя кваліфікаційна робота на тему «Дослідження методів нейронних мереж для виявлення шахрайства», що буде представлена в екзаменаційну комісію для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу

ElArKhNURE. Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

ВСТУП

Виявлення шахрайства – це набір процесів та аналізів, які дозволяють компаніям виявляти та запобігати несанкціонованій діяльності. Це стало одним із головних викликів для більшості організацій, особливо в банківській, фінансовій, роздрібній торгівлі та електронній комерції.

Шахрайська діяльність включає відмивання грошей, кібератаки, шахрайські банківські претензії, підроблені банківські чеки, крадіжка особистих даних і багато подібних незаконних дій. У результаті організації впроваджують сучасні технології виявлення та запобігання шахрайству та стратегії управління ризиками для боротьби зі зростаючими шахрайськими транзакціями на різних платформах. Ці методи застосовують адаптивну та прогнозну аналітику (тобто машинне навчання) для створення оцінки ризику шахрайства разом із моніторингом шахрайських подій у реальному часі. Це дозволяє постійно контролювати транзакції та злочини в режимі реального часу. Це також допомагає розшифрувати нові та складні профілактичні заходи за допомогою автоматизації [1].

Будь-який вид шахрайства негативно впливає на фінансові результати та ринкову репутацію організації, а також відлякує як майбутніх потенційних, так і нинішніх клієнтів. Враховуючи масштаб і охоплення цих вразливих організацій, для них стало критично важливим запобігати шахрайству і навіть передбачати підозрілі дії в режимі реального часу [2].

Є кілька ключових речей, які слід мати на увазі щодо можливостей виявлення. По-перше, важливо мати систему, яка може ідентифікувати потенційні загрози. Ця система повинна мати можливість відстежувати дані та активність на всіх каналах, включаючи онлайн, особисто та по телефону [3].

За останні кілька років спостерігається тенденція до зростання попиту на застосування моделей виявлення аномалій у різних галузях. Але найбільший попит має банківська галузь. В наш час ця тенденція зумовлена зростанням кількості транзакцій та підвищенням попиту фінансових послуг через Інтернет. У банківському секторі боротьба з картковим шахрайством має особливе значення,

оскільки шахраї можуть використовувати різні хитрощі та технологічні інновації, щоб обійти традиційні методи безпеки банків. Шахрайство з банківськими картками може нанести велику шкоду для людей. Це може принести втрату зарплатні або велику суму яку людина відкладала на якусь цінну покупку.

Існує кілька способів запобігання шахрайству, наприклад використання аналітики для ідентифікації ризик факторів, налаштування систем, виявлення або навчання працівників пошуку ознак шахрайства. Вживаючи превентивних заходів, бізнес може захистити себе від фінансових втрат і шкоди своїй репутації.

Виявлення шахрайства має на меті виявити шахраїв і шахрайську діяльність, одночасно запобігаючи втраті грошей та майна. Існують різноманітні системи та інструменти виявлення шахрайства, які використовуються, щоб не відставати від цифрових загроз, що постійно розвиваються, особливо у фінансовій, медичній, державній, страховій та роздрібній галузях. Машинне навчання є життєво важливими компонентами виявлення шахрайства, які використовують методи аналізу даних для виявлення шахрайства [4].

Виявлення шахрайства є складною проблемою. Справа в тому, що шахрайські операції трапляються рідко; вони представляють дуже невелику частину діяльності всередині організації. Проблема полягає в тому, що невеликий відсоток активності може швидко перетворитися на великі втрати, без відповідних інструментів і систем. Оскільки традиційні схеми шахрайства не окупаються, шахраї навчилися змінювати свою тактику. Хороша новина полягає в тому, що завдяки прогресу в аналітиці шахрайства системи можуть вивчати, адаптувати та виявляти нові моделі запобігання шахрайству [5].

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Аналіз предметної галузі

Використання кредитних карток і поява онлайн-покупок значно полегшили життя як споживачам, так і ритейлерам. На жаль, з приходом цифрової революції значно зросла кількість випадків крадіжок кредитних карток. Шахрайство з кредитними картками є серйозною проблемою для банків. З появою онлайн покупок банки почали розробляти все більше механізмів для захисту своїх клієнтів та нові підходи використання банківських послуг.

Хоча технологічні інновації у фінансах не є новим поняттям, інвестиції в нові технології значно зросли за останні роки, а темпи інновацій є досить швидкими. Зараз ми взаємодіємо з нашим банком за допомогою мобільних технологій. Ми здійснюємо платежі, переказуємо гроші та робимо інвестиції, використовуючи різноманітні нові інструменти, які були недоступні ще кілька років тому. Штучний інтелект, соціальні мережі, машинне навчання, мобільні додатки, технологія розподілених реєстрів, хмарні обчислення та аналітика великих даних призвели до появи нових послуг і бізнес-моделей як відомих фінансових установ, так і нових учасників ринку[6].

Однією з ключових переваг цифрових фінансових послуг є їхня ефективність та зручність для клієнтів. Впровадження безпаперових та онлайн-сервісів спрощує процедури та полегшує взаємодію з фінансовими установами.

Мобільні платежі та електронні гаманці стають все більш популярними, прискорюючи та спрощуючи процеси транзакцій. Вони інтегрують технології NFC та QR-кодів, що дозволяє здійснювати ефективні платежі.

Але розвиток цифрових фінансових послуг також сприяє фінансовій інклюзії, забезпечуючи доступ до фінансових послуг для менш розвинених регіонів за допомогою цифрових технологій.

Однак, зі зростанням використання цифрових технологій зростають і ризики, зокрема кіберзлочинності та шахрайства. Забезпечення кібербезпеки та

захисту персональних даних стає нагальним питанням для фінансових установ, які прагнуть забезпечити високий рівень довіри клієнтів.

Злочин шахрайства з кредитними картками починається тоді, коли хтось викрадає кредитну або дебетову картку, або шахрайським шляхом отримує номер картки та іншу інформацію про рахунок, необхідну для успішного використання картки. Хоча фізична крадіжка кредитних карток трапляється, сучасні технології призвели до різкого зростання кількості випадків перехоплення інформації про рахунок в електронному вигляді. Власник рахунку, продавець, у якого було викрадено або перехоплено інформацію про картку, і навіть емітент картки можуть не знати про компрометацію доти, доки ця інформація не буде використана для здійснення покупок.

Оскільки популярність онлайн-покупок та оплати рахунків стрімко зросла, для здійснення покупок більше не потрібно мати фізичну кредитну чи дебетову картку, а можна навіть відкрити фінансовий рахунок і отримати кредитну картку виключно за допомогою онлайн-транзакцій. Через це злочинці, які можуть отримати достатньо особистої інформації про інших осіб, можуть використовувати цю інформацію для шахрайства з кредитними картками, відкриваючи нові рахунки або отримуючи нові картки на вже існуючі рахунки.

Підробка та клонування фізичних карток залишається головною загрозою. Зловмисники використовують технології для отримання доступу до карткових даних, що може призвести до несанкціонованих транзакцій та збитків для клієнтів.

Фішингові та ідентифікаційні атаки стають все більш витонченими. Зловмисники використовують різноманітні методи для викрадення конфіденційної інформації, яка потім може бути використана для онлайн-шахрайства. Наприклад зловмисники можуть створити повну копію сайту банку або магазину, яким ви користуєтесь, і після входу в її систему ваші дані будуть викрадені. Це відбувається, коли зловмисник, маскуючись під довірену особу, обманом змушує жертву самостійно відкрити електронний лист або текстове повідомлення. Потім одержувача обманом змушують перейти за посиланням, що може призвести до

встановлення шкідливого програмного забезпечення, зависання системи або розкриття конфіденційної інформації. Атака може мати руйнівні наслідки. Для приватних осіб вони включають несанкціоновані покупки, крадіжку коштів або викрадення особистих даних.

Крім того, фішинг часто використовується для проникнення в корпоративні або державні мережі, а також як частина більш масштабної атаки для обходу периметрів безпеки, поширення шкідливого програмного забезпечення в закритому середовищі або отримання привілейованого доступу до захищених даних.

Організація, яка зазнає такої атаки, зазвичай зазнає значних фінансових збитків, а також втрачає частку ринку, репутацію та довіру споживачів. Залежно від масштабу, спроба фішингу може перерости в інцидент безпеки, після якого компанії може бути важко оговтатися [7].

Також банківські термінали та банкомати піддаються ризику компрометації через введення шкідливого програмного забезпечення. Це може призвести до втрат грошей або витоку особистих даних. Бувають такі види махінацій з банківськими терміналами:

- зняття готівки за допомогою "білого" пластику;
- використання скімінгових інструментів, тобто копіювання даних платіжних карт, у тому числі з магнітної смуги, запис ПІН-коду та інше;
- зняття коштів у банкоматі без відображення цієї операції на рахунку – Transaction Reversal Fraud;
- зняття готівки власником картки без її фізичного отримання;
- cash trapping.

Одним з найпопулярніших видів шахрайства є скімінг.

Скіммінг – один з видів шахрайства з банківськими картами, що полягає в зчитуванні і копіюванні інформації з магнітного чіпа. Для цього використовується складне спеціалізоване обладнання, яке копіює дані. Але щоб скористатися ними необхідно ще знати ПІН-код. Для його зчитування застосовуються спеціальні накладні клавіатури або мініатюрні приховані камери. Все це дозволяє

зловмисникам виготовляти повний дублікат картки і надалі використовувати його для списання коштів з рахунку справжнього власника [8].

Шахраї намагаються зробити його максимально малопомітним, а тому виконують зазвичай у вигляді тонкої пластини вставленої всередину картоприймача або накладки, що закріплюється на картрідер банкомату.

Непідготовленому користувачеві виявити цей модуль досить складно, адже зловмисники намагаються максимально точно скопіювати оригінальну форму елемента банкомату і підібрати колір. На щастя користувачів, це вдається далеко не завжди, що і робить чіп помітним.

Також існують ручні портативні скімери, які можуть використовуватися офіціантами або продавцями для зчитування інформації з картки, переданої клієнтом. Такі пристрої немає сенсу під щось маскувати, тому виглядають вони як звичайний картрідер банкомату або платіжного терміналу.

Важлива відмінність скіммінгу від інших способів шахрайства з банківськими картами в тому, що технічно (і на законодавчому рівні) для банку операція по зняттю грошей з використанням копії карти і вкраденого ПІН-коду нічим не відрізняється від звичайної операції. Тобто довести, що гроші були зняті саме шахраєм дуже складно. Тому рекомендується використовувати інтернет-банкінг і смс-оповіщення, щоб оперативно отримувати інформацію про всі операції по карті або рахунку. Це дозволить своєчасно звернутися в банк для блокування карти і написання претензії про підозрілі операції. Так як повернути гроші, вкрадені зловмисниками, при зверненні в банк через добу після злочину буде дуже непросто.

Розслідування шахрайства з кредитними картками найчастіше починається з того, що споживач звертається до місцевої поліції із заявою про крадіжку або несанкціоноване використання його кредитної чи дебетової картки.

Повідомлення про злочин або підозру, що рахунок особи був скомпрометований, може вимагати від власника картки підписання заяви під присягою, в якій детально описуються спірні транзакції, а також заяви під страхом

покарання за неправдиві свідчення про те, що він не здійснював оскаржуваних ним платежів.

Банківські та кредитні установи, а також торгова комісія та правоохоронні органи наполегливо рекомендують негайно повідомляти про підозру в шахрайстві з кредитними картками [9].

Професійні слідчі у справах про шахрайство з кредитними картками навчаються "електронному відстеженню" електронних та онлайн-транзакцій. Ці фахівці з інформаційних технологій, систем фінансової безпеки та судово-бухгалтерської експертизи використовують високотехнологічні методи, щоб визначити, чи використовувалася викрадена інформація з картки для здійснення покупок або відкриття нових рахунків.

Такі фахівці з розслідування шахрайства часто можуть зупинити відкриття та використання вторинних рахунків, мінімізуючи збитки для власника картки та її емітента.

Крім того всі країни світу створюють свої механізми для неможливість обходу безпеки в банківській сфері. Наприклад, виконуючи функцію нагляду за платіжними картками, що діють в єврозоні, Євросистема уважно стежить за тенденціями у сфері карткового шахрайства. У січні 2008 року Рада керуючих ЄЦБ затвердила систему нагляду за платіжними картками. Статистична інформація про ці схеми збирається в рамках гармонізованого впровадження цієї системи.

Кожній платіжній системі необхідно надати загальні бізнес-дані, а також обсяги та вартість карткових транзакцій і відповідне шахрайство з розбивкою за країнами для всіх країн Єдиного платіжного простору євро (SEPA) та в сукупності для всіх країн, що не входять до SEPA.

Загальна вартість транзакцій з використанням карток, емітованих у SEPA, у 2019 році склала 5,16 трлн євро, з яких 1,87 млрд євро були шахрайськими (див. рис. 1.1).

Вартість карткового шахрайства зросла на 3,4% порівняно з 2018 роком, тоді як вартість загальних карткових транзакцій зросла на 6,5%. Таким чином, вартість

загальних карткових транзакцій зростала швидше, ніж шахрайство, що призвело до незначного зниження частки шахрайства у загальній вартості транзакцій з 0,037% у 2018 році до 0,036% у 2019 році.

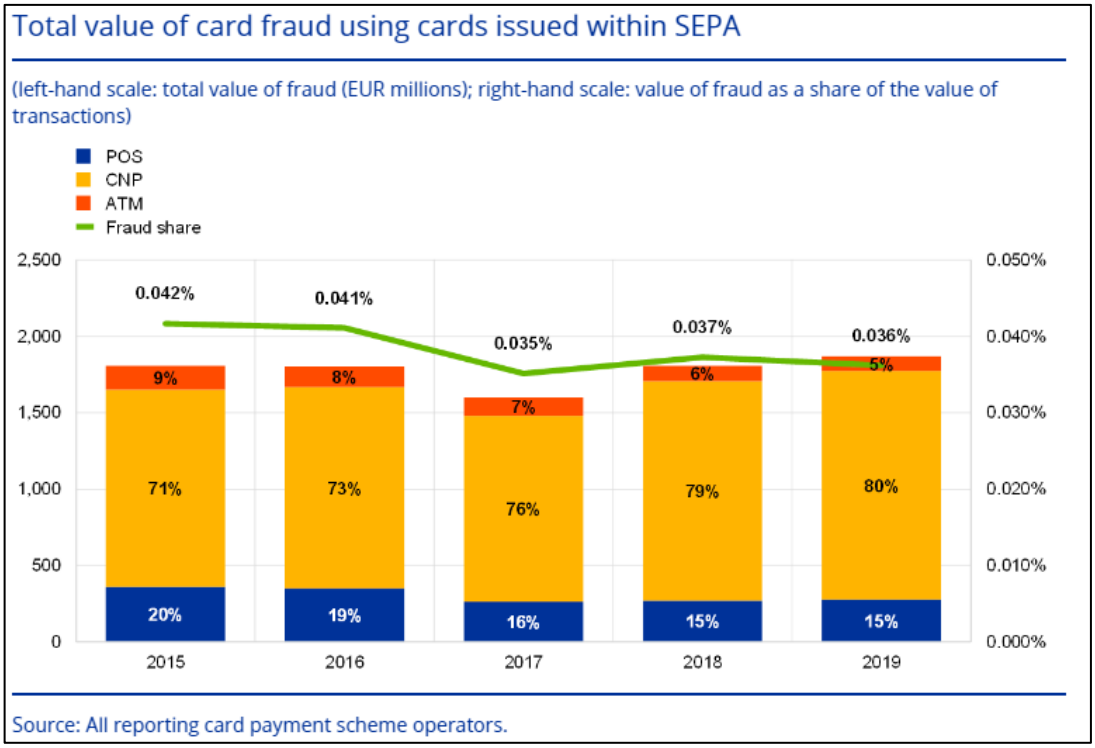

Рисунок 1.1 - Загальна кількість карткового шахрайства з використанням карток, випущених в межах SEPA

Це відбулося після незначного зростання частки шахрайства у 2018 році з найнижчого зафіксованого рівня у 2017 році (0,035%). Як у 2018, так і в 2019 році показники залишаються значно нижчими за п'ятирічний максимум, зафіксований у 2015 році (0,042%). Якщо взяти ширшу перспективу за останні десять років, то до середини десятиліття рівень карткового шахрайства зростав як в абсолютному, так і у відносному вираженні, але в останні роки він суттєво покращився у відносному вираженні. Натомість частка шахрайства через банкомати та POS-термінали зменшилася до 5% та 15% від загального обсягу шахрайств відповідно [10].

1.2 Виявлення проблеми

Існує безліч моделей та алгоритмів для виявлення шахрайства в різних сферах, включаючи фінансові операції, електронну комерцію, медичні послуги та інше. Для досягнення цілі можна використати нейроні мережі. Нейронні мережі доводять свою ефективність у виявленні складних і нелінійних взаємозв'язків серед даних. Вони можуть автоматично навчатися та адаптуватися до різноманітних шахрайських схем, включно з тими, які важко формалізувати за допомогою традиційних алгоритмів.

Але задача виявлення шахрайства за допомогою нейронних мереж стикається з декількома проблемами та викликами. Наприклад, для ефективного навчання нейронних мереж необхідні великі та різноманітні набори даних. Однак у виявленні шахрайства може бути важко накопичити достатню кількість реальних випадків шахрайства для ефективного навчання. Кількість шахрайських транзакцій часто набагато менша, ніж кількість легальних транзакцій. Це може призвести до проблеми дисбалансу класів, коли нейромережева модель може бути надмірно сфокусована на класі легальних транзакцій і неефективно виявляти шахрайські. Також моделі шахрайства можуть змінюватися, а шахраї можуть використовувати нові стратегії. Це ускладнює розробку моделей, які можуть ефективно виявляти нові форми шахрайства.

Врахування різних умов і контекстів може бути важливим для точного виявлення шахрайства. Наприклад, поведінка користувачів може відрізнятися вдень і вночі, і моделі повинні вміти адаптуватися до цих умов.

Крім того, невизначеність умов, що змінюються, є ще одним значним викликом. Щоб точно виявляти шахрайство, моделі повинні враховувати обставини та контекст, що змінюються. Наприклад, врахування географічного розташування, типу пристрою або звичних моделей покупок може значно підвищити ефективність виявлення шахрайства.

Іншою проблемою є великий обсяг даних і потреба в обчислювальних ресурсах. Використання глибоких нейронних мереж може вимагати значних обчислювальних потужностей для навчання та висновків. Важливо знайти баланс

між складністю моделі та доступними обчислювальними ресурсами, а також провести оптимізацію для зменшення впливу на навколишнє середовище.

Вищезазначені проблеми підкреслюють важливість використання гнучких та адаптивних моделей, а також постійного моніторингу та оновлення для виявлення нових видів шахрайства. Лише завдяки поєднанню експертних знань, аналізу даних та сучасних методів машинного навчання можна досягти високої точності виявлення шахрайства та забезпечити надійний захист фінансових операцій та інших сфер від шахрайських дій.

Розглянемо методи вирішення поставлених задач.

Для вирішення проблеми недостатньої кількості реальних випадків шахрайства ми можемо використати метод синтезу даних. Цей підхід створює штучні випадки шахрайства, що дозволяє розширити кількість навчальних даних та покращити продуктивність моделі.

Вибір глибоких мереж з архітектурою, яка може адаптуватися до нових шаблонів, дозволяє будувати гнучкі та ефективні моделі. Це особливо важливо для виявлення шахрайства, оскільки стратегії шахраїв можуть постійно змінюватися. Також глибокі нейронні мережі можуть ефективно вивчати деталі та складні дані, що робить їх особливо корисними для виявлення нових форм шахрайства. Зміна стратегій шахраїв може вимагати постійного навчання моделі, що робить глибокі нейронні мережі інструментом вибору.

Нейронні мережі можуть обробляти різні типи даних, такі як текстова інформація, зображення, голосові дані тощо. Це робить їх універсальними для використання в різних галузях, включаючи фінансовий сектор.

Нарешті, важливо зазначити, що виявлення шахрайства є динамічним завданням, а використання глибокого навчання дозволяє системам ефективно адаптуватися до постійно мінливих варіацій шахрайських схем. Для більш наочного прикладу було наведено таблицю 1.1.

У таблиці показані основні проблеми, які виникають, і відповідні методи їх вирішення за допомогою нейронних мереж.

Таблиця 1.1 – Проблеми та методи вирішення

Проблема	Метод вирішення
Нестача реальних прикладів шахрайства	Використання синтетичних даних та створення нових спроб шахрайства
Врахування контексту та умов	Додавання параметрів, за допомогою яких модель зможе враховувати різні умови
Постійне оновлення моделі	Регулярне оновлення, підтримка актуальності виявлення шахрайства
Адаптація до нових методів шахрайства	Використання нейроних мереж або глибокого навчання для ефективнішого виявлення шахрайства

1.3 Постановка задачі

Провівши аналіз предметної галузі та виявивши проблеми, виникає потреба в більш швидших та досконалих методах для виявлення шахрайства на базі нейромереж. Тому ця сфера є дуже актуальною на сьогодні. В рамках дослідження необхідно виконати наступні цілі:

- розглянути і проаналізувати існуючі методи виявлення шахрайста за допомогою нейроних мереж;
- визначення переваг і обмежень нейромереж;
- вибір типу та конфігурації нейронної мережі, яка найбільш підходить для вирішення поставленої задачі;
- відбір та підготовка набору даних для навчання та тестування;
- навчання нейронної мережі на підготовленому наборі даних для розпізнавання шаблонів, пов'язаних з шахрайством;

- аналіз результатів виявлення шахрайства, включаючи точність, чутливість, специфічність та інші метрики, що відображають ефективність моделі;
- порівняння з іншими рішеннями.

Дослідження повинно надати повну інформацію щодо ефективності та можливостей різних нейронних мереж для виявлення шахрайства.

2 АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ ТА АЛГОРИТМІВ

2.1 Огляд методів та алгоритмів машинного навчання

Машинне навчання (ML) – це спосіб реалізації штучного інтелекту; фактично, це спеціалізована галузь в рамках широкої сфери ШІ, яка, в свою чергу, є галуззю комп'ютерних наук. За допомогою машинного навчання ми можемо розробляти алгоритми, які здатні навчатися без явного програмування. До таких алгоритмів відносяться:

- дерева рішень;
- наївний Байєс;
- випадковий ліс;
- машина опорних векторів;
- K-найближчий сусід;
- кластеризація K-середніх;
- модель гауссової суміші;
- прихована модель Маркова.

Ключ - це дані. Ви надаєте дані, що характеризуються різними атрибутами або ознаками, для аналізу та розуміння алгоритмами. Ці алгоритми створюють межі прийняття рішень на основі наданих даних, що дозволяє їм робити прогнози або класифікації. Після того, як алгоритм обробив і зрозумів дані – по суті, навчився сам – можна переходити до етапу тестування. Тут ви вводите в алгоритм нові дані, і він видає результати без будь-якого подальшого програмування.

Глибоке навчання - це підкатегорія машинного навчання, яка фокусується на структуруванні процесу навчання для комп'ютерів, щоб вони могли розпізнавати закономірності та приймати рішення, подібно до того, як це робить людина. Наприклад, якщо ми навчаємо комп'ютер розрізняти різних тварин, ми починаємо з простих, базових понять, таких як кількість ніг, і поступово вводимо більш складні, такі як середовище проживання та поведінка. Глибоке навчання – це, по суті, тип складного багатошарового фільтра. Ми вводимо сирі, неорганізовані дані у верхній частині, і вони проходять через різні шари нейронної мережі,

уточнюючись і аналізуючись на кожному рівні. Зрештою, те, що з'являється внизу, є послідовною, структурованою інформацією або точним "прогнозом". Цей процес чудово розшифровує дані з декількома рівнями абстракції, що робить його незамінним для таких завдань, як розпізнавання зображень і мови, обробка природної мови та розробка ігрових стратегій. Він лежить в основі багатьох інновацій, від віртуальних асистентів на наших телефонах до розробки автономних транспортних засобів [11]. У сфері машинного навчання глибоке навчання відрізняється використанням нейронних мереж з трьома або більше шарами (див рис. 2.1). Ці багатошарові нейронні мережі намагаються відтворити моделі навчання людського мозку, дозволяючи комп'ютеру аналізувати і навчатися на основі величезних обсягів даних. Одношарова мережа може робити елементарні прогнози, але коли ми додаємо більше шарів, мережа стає здатною розуміти складні закономірності та взаємозв'язки, підвищуючи точність своїх прогнозів.

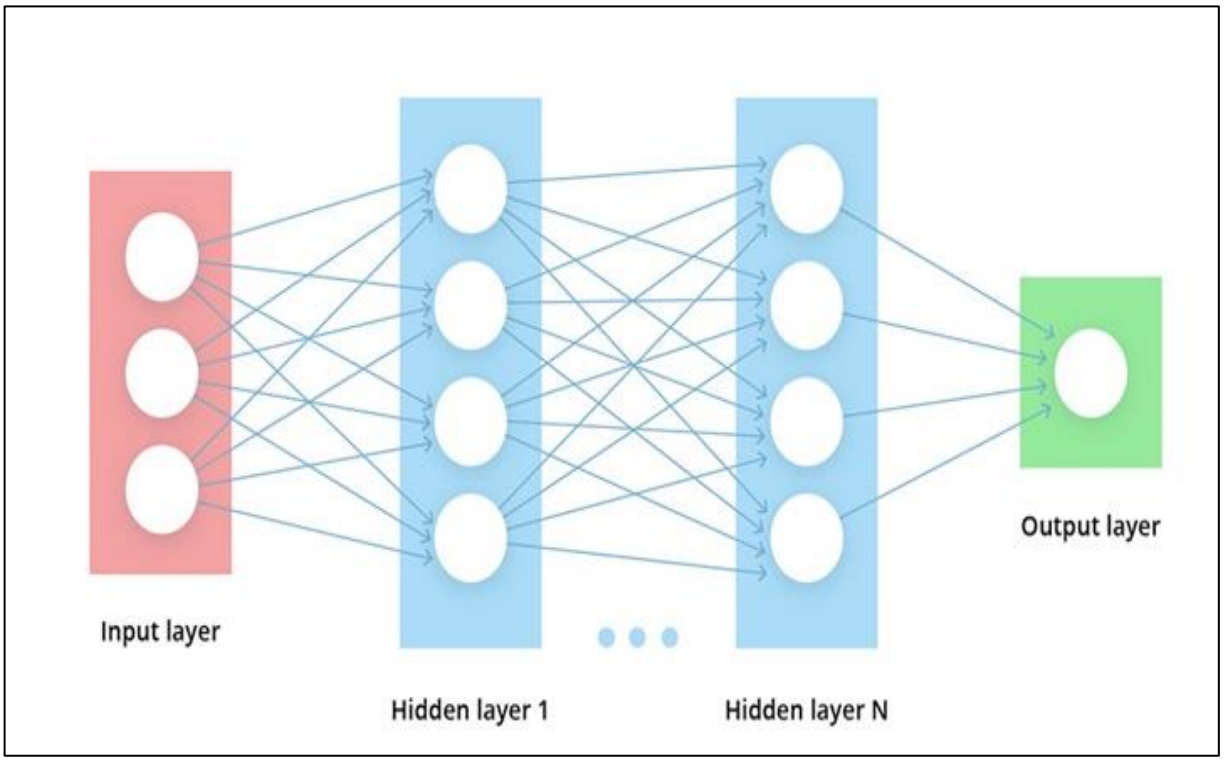

Рисунок 2.1 - Глибока нейронна мережа

Як ви можете бачити на рисунку 2.1 є вхідний шар, приховані шари та вихідний шар. Вхідними даними можуть бути грошові дані або діапазон часових

рядів. Прихований шар - відомий як ваги, які дізнаються під час навчання нейронної мережі. Вихідний шар дає прогноз вхідних даних, які подали у мережу.

Також, бувають такі види нейроних мереж.

Пряма нейронна мережа: Нейронна мережа прямого поширення є однією з найпростіших штучних нейронних мереж. У цій ANN дані або вхідні дані рухаються в одному напрямку. Вони входять в ANN через вхідний шар і виходять через вихідний шар, тоді як приховані шари можуть існувати, а можуть і не існувати. Таким чином, нейронна мережа прямого поширення має тільки хвилю, що поширюється вперед, і зазвичай не має зворотного поширення.

Згорткова нейронна мережа: Згорткова нейронна мережа має деяку схожість з нейронною мережею прямого поширення, де зв'язки між елементами мають вагу, яка визначає вплив одного елемента на інший. Але ANN має один або більше одного згорткового шару, який використовує операцію згортки на вході, а потім передає отриманий результат у вигляді вихідних даних наступному шару. CNN має застосування в обробці мови та зображень, що особливо корисно для комп'ютерного зору. Взагалі, згорткова нейронна мережа є досить унікальним типом нейронних мереж. Ця унікальність буде проявлятися в декількох особливостях. Вони мають три виміри: глибину, висоту і ширину. Вузли в кожному з шарів пов'язані лише з невеликою частиною, а не з усіма, а результат згортається в набір імовірнісних оцінок, згрупованих уздовж осі глибини [12].

Рекурентна нейронна мережа: Рекурентна нейронна мережа зберігає вихід шару і подає цей вихід назад на вхід, щоб краще передбачити результат шару. Перший шар в RNN дуже схожий на нейронну мережу прямого поширення, і рекурентна нейронна мережа запускається після того, як обчислюється вихід першого шару. Після цього шару кожна одиниця запам'ятовує деяку інформацію з попереднього кроку, щоб вона могла діяти як комірка пам'яті при виконанні обчислень. LSTM (Long Short-Term Memory) – це тип рекурентної нейронної мережі (RNN), спеціально розроблений для вирішення проблеми довготривалої залежності, яку часто спостерігають у послідовних даних.

Модульні нейронні мережі складаються з декількох нейромережевих

модулів, кожен з яких відповідає за вирішення певної підзадачі. Ці модулі взаємодіють і співпрацюють для вирішення більш складних завдань. Такий модульний підхід забезпечує гнучкість і масштабованість для вирішення великомасштабних і різноманітних проблем. Модульні нейронні мережі використовуються в таких галузях, як робототехніка, де різні модулі відповідають за сприйняття, планування та контроль. Розділяючи проблему на менші, керовані частини, ці мережі можуть ефективно вирішувати складні завдання. Наприклад, архітектура нейронної мережі може складатися з взаємопов'язаних шарів: embedding, bidirectional LSTM, pooling, dropout та два dense шари [13]. Цей підхід дозволяє розділити складні задачі на менші підзадачі, що робить процес навчання більш ефективним.

2.2 Популярність методів для виявлення шахрайства

У цьому розділі висвітлено зміст кваліфікаційної роботи, який включає популярні методи виявлення фінансового шахрайства та методи машинного навчання, що використовуються в методах виявлення шахрайства(див. рис. 2.2).

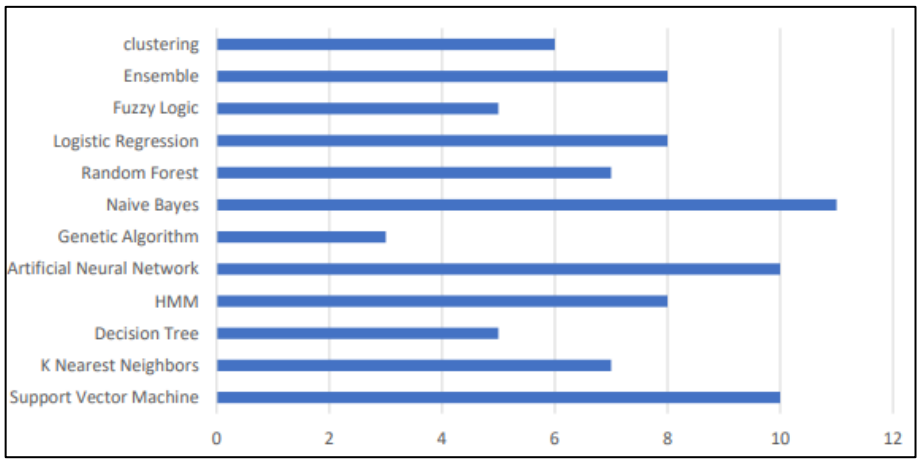

Рисунок 2.2 - Частота використання методів машинного навчання для виявлення шахрайства.

Рисунок 2.2 висвітлює частоту використання методів машинного навчання для виявлення шахрайства.

В результаті можна побачити, що алгоритм NB є найпопулярнішим методом, який використовується для виявлення шахрайських дій у фінансовому секторі, за ним йдуть SVM та ANN. Це свідчить про те, що NB, SVM та ANN є найбільш найпопулярнішими методами машинного навчання, які використовуються для виявлення фінансового шахрайства на основі проаналізованої літератури. На рисунку 3 показано частотний розподіл методів машинного навчання, що використовуються для виявлення шахрайства. методів машинного навчання, що використовуються для виявлення шахрайства, а також розподіл частоти випадків фінансового
шахрайства [14].

2.3 Аналіз та порівняння рішень та конкурентів

Таблиця 2.1 надає узагальнену оцінку кожного типу нейронної мережі з точки зору їхніх функцій, властивостей, залежності від даних, обчислювальної потужності, часу тренування, а також характеристик виводу.

Таблиця 2.1 - Порівняння та оцінка методів

Критерій	Пряма нейронна мережа	Згорткова нейронна мережа	Рекурентна нейронна мережа	Машинне навчання	Глибоке навчання
Функціонування	Простий та швидкий	Здатна виявляти просторові патерни	Здатна моделювати часові залежності та виявляти аномалії в послідовностях транзакцій.	Різні, включаючи SVM, Decision Trees	Використовує багатошарові нейроні мережі
Виділення особливостей	Виявлення статичних ознак	Комбінує просторові та часові ознаки	Спеціалізовані у виявленні просторових патернів	Залежить від конкретного методу	Залежить від конкретного методу

Залежність від даних	Незалежний від часу	Залежний від часу та просторових	Залежний від	Залежить від	Залежить від конкретно

Кінець таблиці 2.1

Критерій	Пряма нейронна мережа	Згорткова нейронна мережа	Рекурентна нейронна мережа	Машинне навчання	Глибоке навчання
		залежностей	х залежностей	ного методу	го методу
Обчислювальна потужність	Низький	Високий	Високий	Залежить від конкретного методу	Високий
Час на навчання	Швидкий	Високий	Зазвичай високий	Залежить від конкретного методу	Залежить від конкретного методу
Вихідні дані	Класифікація, прогнозування	Аналіз часових рядів, обробка послідовностей	Комбінація обробки Часових просторових ознак	Залежить від конкретного методу	Залежить від конкретного методу

На даний момент відомі компанії, що працюють у сфері виявлення шахрайства за допомогою нейронних мереж:

- Feedzai використовує штучний інтелект і машинне навчання для виявлення шахрайства в платіжних операціях;
- Forter спеціалізується на рішеннях для електронної комерції, використовуючи технологію нейронних мереж для виявлення аномальної поведінки та шахрайства;
- Sift використовує машинне навчання для виявлення шахрайства в різних онлайн-середовищах, включаючи електронну комерцію;
- Kount зосереджується на аналізі шахрайської активності та ризиків у режимі реального часу за допомогою різноманітних алгоритмів;

– Riskified використовує передову аналітику даних і машинне навчання для виявлення шахрайства в електронній комерції.

2.4 Інформаційна підготовка прийняття рішень

В минудому розділі було описано декілька рішень які можуть використовуватися в виявлені шахрайства. Щоб максимілізувати ефективність виявлення шахрайства, скористаємося лінійною адаптивною згорткою з ваговими коефіцієнтами.

Вибір буде проводитись серед наступних методів.

Метод опорних векторів (SVM) виявлення шахрайства: використання SVM для розділення правдивих та підозрілих транзакцій на основі векторів ознак.

Переваги: Ефективний при роботі з великими обсягами даних, може працювати в умовах високої розмірності.

Недоліки: Може вимагати великої кількості обчислювальних ресурсів, не завжди ефективний при виявленні складних неструктурованих шахрайських сценаріїв.

Виявлення шахрайства на основі правил: розробка правил та порігових значень для ідентифікації підозрілих транзакцій.

Переваги: Простий у реалізації, може бути ефективним для виявлення відомих шахрайських сценаріїв.

Недоліки: Може бути неефективним при виявленні нових та неочікуваних видів шахрайства.

Глибоке навчання для виявлення аномалій: використання глибоких нейронних мереж для виявлення аномалій та винятків у звичайних патернах транзакцій.

Переваги: Здатний виявляти складні та нові види шахрайства, може адаптуватися до змінюючихся умов.

Недоліки: Вимагає великої кількості даних для навчання, складний у налаштуванні та інтерпретації результатів.

Використання рекурентних нейронних мереж (RNN): застосування RNN для аналізу послідовностей транзакцій та виявлення аномальних змін у часі.

Переваги: Здатний враховувати динаміку та залежності між транзакціями в часі, ефективний при виявленні шахрайства з певними шаблонами. Недоліки: Може вимагати більше обчислювальних ресурсів та даних для ефективного навчання.

Використання генетичних алгоритмів: застосування генетичних алгоритмів для оптимізації параметрів та структури моделі виявлення шахрайства.

Переваги: Здатність автоматично підлаштовувати параметри моделі для максимізації ефективності, ефективність при оптимізації складних задач. Недоліки: Може вимагати значних обчислювальних ресурсів, потребує уважного підбору параметрів для уникнення перетренування. У якості критеріїв для порівняння методів, було обрано наступні параметри:

- точність моделі: Інтервальна шкала від 0 до 100%. Може набувати таких значень як 92%, 95%, 98%. Вищі значення вказують на більш високу точність виявлення шахрайства. Важливий критерій для системи виявлення шахрайства, оскільки точність визначає, наскільки ефективно модель класифікує транзакції на правдиві та шахрайські;
- пропускна здатність: Інтервальна шкала в мілісекундах. Може набувати таких значень як 50 ms, 40 ms, 35 ms.. Нижчі значення вказують на більш ефективний час обробки. Забезпечення реального часу виявлення шахрайства вимагає мінімізації часу, необхідного для аналізу та класифікації транзакцій;
- вартість: Відсоткова шкала від 0 до 100% від загального бюджету. Може набувати таких значень як 15%, 10%, 8%. Нижчі значення вказують на економічну ефективність системи. Мінімізація витрат дозволяє оптимізувати використання ресурсів та забезпечити ефективність виявлення шахрайства без зайвих витрат;
- адаптивність: Ординальна шкала від 1 до 5, де 5 - висока здатність до адаптації, а 1 - низька. Вищі значення вказують на більшу здатність до

адаптації. Оскільки типи шахрайства змінюються, система повинна мати можливість адаптуватися до нових моделей і мінливих умов;

– масштабованість: Логарифмічна шкала від 1 до 1000 (кількість транзакцій). Значення: 100, 300, 1000. Вищі значення вказують на більшу ефективність при великому обсязі даних. Оскільки обсяг транзакцій зростає, ефективність системи з великим обсягом даних є ключовим фактором для забезпечення швидкості і точності виявлення.

Наведемо векторний опис альтернатив за обраними критеріями у вигляді таблиці 2.2.

Таблиця 2.2 - Оцінка методів

Методи	Критерії				
	Точність моделі (%)	Пропускна здатність (ms)	Вартість	Адаптивність	Масштабованість (транзакцій)
Нейронна мережа	98	30	10	4	1000
SVM	95	40	12	3	500
Правила	92	50	8	5	300
Глибоке навчання	97	38	15	4	800
Генетичні алгоритми	96	37	11	5	700

Приведемо усі шкали до принципу оптимальності «за максимумом». На таблиці 2.3 показано принцип оптимальності.

Таблиця 2.3 – Приведена таблиця оптимальності за максимумом

Методи	Критерії				
	Точність моделі (%)	Виграш пропускної здатності (ms)	Виграш вартості	Адаптивність	Масштабованість (транзакцій)
Нейронна мережа	98	20	5	4	1000
SVM	95	10	3	3	500
Правила	92	0	7	5	300
Глибоке навчання	97	12	0	4	800
Генетичні алгоритми	96	13	4	5	700

Потім використаємо принцип Парето (див. таб. 2.4). Оскільки SVM та правила мають гірші характеристики у порівння з нейронною мережою, то їх можна виключити.

Таблиця 2.4 - Порівняння та оцінка методів

Методи	Критерії				
	Точність моделі (%)	Виграш пропускної здатності (ms)	Виграш вартості	Адаптивність	Масштабованість (транзакцій)
Нейронна мережа	98	20	5	4	1000

Кінець таблиці 2.4

Методи	Критерії				
	Точність моделі (%)	Виграш пропускної здатності (ms)	Виграш вартості	Адаптивність	Масштабованість (транзакцій)
Генетичні алгоритми	96	13	4	5	700

Перейдемо до нормування критеріїв. Оскільки після використання принципу Парето залишилося лише два варіанти вибору, то нормувати критерії можна за наступним принципом. Значення характеристики, того варіанту, яке буде більше, буде унормовано до значення 1, а того варіанту, яке меньше, буде унормовано до значення 0. Зміни було внесено до таблиці 2.5.

Таблиця 2.5 – Нормалізована таблиця

Методи	Критерії				
	Точність моделі (%)	Виграш пропускної здатності (ms)	Виграш вартості	Адаптивність	Масштабованість (транзакцій)
Нейронна мережа	1	1	1	0	1
Генетичні алгоритми	0	0	0	1	0

У записаній нами задачі вибору критерії мають різну важливість. Тому для вирішення задачі вибору, використаємо лінійну адаптивну згортку з ваговими коефіцієнтами. Порядок критеріїв, за важливістю:

– адаптивність;

– точність моделі;

– виграш пропускної здатності;

– масштабованість;

– виграш вартості.

Визначення коефіцієнтів:

– адаптивність: $\frac{5}{1+2+3+4+5} = \frac{5}{15} = \frac{1}{3}$.

– точність моделі: $\frac{4}{1+2+3+4+5} = \frac{4}{15}$.

– виграш пропускної здатності: $\frac{3}{1+2+3+4+5} = \frac{3}{15} = \frac{1}{5}$.

– масштабованість: $\frac{2}{1+2+3+4+5} = \frac{2}{15}$.

– виграш вартості: $\frac{1}{1+2+3+4+5} = \frac{1}{15}$.

Розрахунки корисності:

– нейронна мережа: $\frac{4}{15} * 1 + \frac{1}{5} * 1 + \frac{1}{15} * 1 + \frac{1}{3} * 0 + \frac{2}{15} * 1 = 0{,}67.$

– генетичні алгоритми: $\frac{4}{15} * 0 + \frac{1}{5} * 0 + \frac{1}{15} * 0 + \frac{1}{3} * 1 + \frac{2}{15} * 0 = 0{,}33.$

Фінальний результат вирішення задачі вибору знаходиться в таблиці 2.6.

Таблиця 2.6 – Результати перетворення

Методи	Точність моделі (%)	Виграш пропускної здатності	Виграш вартості	Адаптивність	Масштабованість	Корисність
Нейронна мережа	1	1	1	0	1	0,67
Генетичні алгоритми	0	0	0	1	0	0,33

Отже за результатами вирішення задачі вибору, для проведення дослідження методів виявлення шахрайства за допомогою нейронних мереж, краще використовувати метод нейронної мережі. Даний метод є найбільш точним, має значний виграш пропускної здатності, гарну масштабованість та найменшу вартість. Розраховане значення корисності для даного методу дорівнює 0,67.

3 ПІДГОТОВКА ДО ЕКСПЕРИМЕНТУ

3.1 Вимоги до нейроної мережі та реалізація дослідження

Нейромережа може бути реалізована на будь-якій мові програмування. Важливим критерієм є дослідження ефективності використання різних методів оптимізації та архітектур нейромереж. Можна спробувати згорткові, рекурентні або комбіновані нейронні мережі з різними функціями активації та шарами. Або використати різдні методи оптимізації. Наприклад генетичний алгоритм, метод рою часток, Harris Hawks, На вході нейромережа для виявлення шахрайства повинна отримувати характеристики транзакцій або іншу інформацію про транзакції, яка може бути корисною для визначення їхньої автентичності. Ці характеристики можуть включати такі дані, як сума транзакції, тип транзакції, географічне розташування, історія клієнта тощо.

Результатом роботи нейронної мережі має бути прогноз щодо того, чи є дана транзакція шахрайською чи ні. Це може бути виражено у вигляді ймовірності (наприклад, у відсотках) або бінарної класифікації (шахрайська чи нешахрайська транзакція).

Загалом завдання можна розділити на дві частини - дослідницьку та практичну. З точки зору апаратного забезпечення, система потребує комп'ютера або сервера з достатньою обчислювальною потужністю для запуску алгоритмів виявлення. Для швидкої обробки даних можно використовувати бібліотеку Deep Graph Library. Для швидкої побудови нейромереж можливо використовувати бібліотеки TensorFlow або Keras.

3.2 Обрання датасету, метрик, методів та вид нейромережі

Для навчання нейронної системи потрібно взяти набір даних. Збір великої кількості даних про транзакції з різних джерел, включаючи як справжні, так і шахрайські транзакції. Це дозволить створити набір даних для навчання та тестування моделей. За основу було взято набір даних з Kaggle.Назва датасету

IEEE-CIS Fraud Detection 2019. Для оцінки моделей ми можемо використати наступні метрики оцінки.

Точність(Accuracy). Відсоток правильно класифікованих прикладів (шахрайських і нешахрайських транзакцій) від загальної кількості прикладів у тестовому наборі даних (див. рис. 3.1). Точність корисна для загальної оцінки моделі, але може бути недостатньою у випадках дисбалансу класів.

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{All Samples}}$$

Рисунок 3.1 – Формула точності

Чутливість(Recall або True Positive Rate). Відсоток шахрайських транзакцій, які були правильно виявлені, від усіх справжніх шахрайських транзакцій (див. рис. 3.2). Чутливість важлива, коли нам потрібно максимізувати виявлення шахрайства, уникаючи при цьому пропуску справжніх шахрайських транзакцій.

$$\text{чутливість} = \frac{\text{число істинно позитивних}}{\text{число істинно позитивних + число хибно негативних}}$$
$$= \frac{\text{число істинно позитивних}}{\text{загальне число хворих осіб у генеральній сукупності}}$$
$$= \text{ймовірність позитивного тесту за умови, що пацієнт має захворювання}$$

Рисунок 3.2 – Формула чутливості

Специфічність(True Negative Rate). Відсоток правильно виявлених нешахрайських транзакцій серед усіх дійсно нешахрайських транзакцій(див. рис. 3.3). Специфічність важлива, коли нам потрібно мінімізувати кількість хибнопозитивних спрацьовувань.

$$\text{специфічність} = \frac{\text{число істинно негативних}}{\text{число істинно негативних + число хибно позитивних}}$$
$$= \frac{\text{число істинно негативних}}{\text{загальне число нехворих осіб у генеральній сукупності}}$$
$$= \text{імовірність негативного тесту за умови, що пацієнт не є хворим}$$

Рисунок 3.3 – Формула специфічності

F1-показник (F1-score). Середнє гармонійне значення між чутливістю та точністю. Показник F1 корисний у випадках незбалансованих класів, оскільки він враховує як точність, так і чутливість(див. рис. 3.4).

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{\text{TP}}{\text{TP} + \frac{1}{2}(\text{FP} + \text{FN})}$$

TP = number of true positives
FP = number of false positives
FN = number of false negatives

Рисунок 3.4 – Формула F1

AUC-ROC (Area Under the Receiver Operating Characteristic curve). Площа під ROC-кривою, яка дозволяє оцінити загальну продуктивність моделі без урахування конкретного порогу відсікання(див рис. 3.5). Чим вище значення AUC-ROC, тим краще модель розділяє класи.

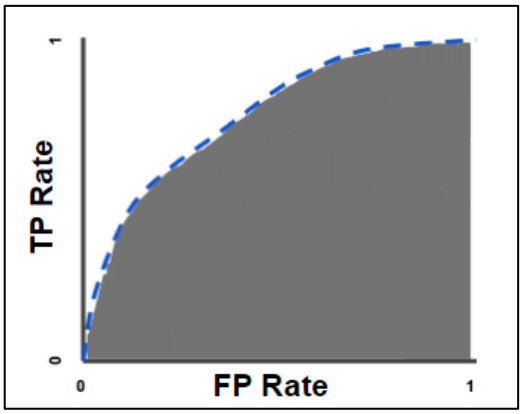

Рисунок 3.5 – Графік AUC-ROC

В експеременті для порівння було взято декілька конкурентів. Двоє з них було реалізовано за допомогою графова нейроної мережі та бібліотекою deep

graph library. Графові нейронні мережі (GNN) – це вид нейронних мереж, призначених для обробки графової структурованої інформації. У порівнянні з традиційними нейронними мережами, які опрацьовують вектори або послідовності даних, GNN можуть працювати безпосередньо з графами, що робить їх дуже потужними для моделювання взаємозв'язків та властивостей великих складних систем, таких як соціальні мережі, біологічні молекули або фінансові транзакції. Deep Graph Library (DGL) - це високорівнева бібліотека для глибокого навчання на графах. Вона надає інтерфейс для створення, навчання та використання графових нейронних мереж (GNN) для різних завдань, таких як класифікація графів, передбачення властивостей вершин та зв'язків, генерація графів тощо [15].

Ще дві нейронні мережі використовували Whale Optimization Algorithm (WOA) та Hawk optimization algorithm (HOA). WOA – метаевристичний алгоритм оптимізації, натхненний соціальною поведінкою горбатих китів. Він був запропонований Сеєдалі Мірджалілі, який представив його у 2016 році як нову техніку оптимізації для вирішення оптимізаційних задач [16]. HOA – популярний алгоритм безградієнтної оптимізації на основі рою з кількома активними та змінними в часі фазами розвідки та експлуатації [17].

3.3 Опис експереметну

Експеримент для виявлення шахрайства може бути проведений за допомогою певних методів та технік, спрямованих на виявлення недостовірної поведінки або дій осіб. Нижче наведено загальний опис такого експерименту:

- збір даних: для початку потрібно мати дані про шахрайство. Це може бути дані з сайту Kaggle, які містять інформацію, необхідну для аналізу шахрайства в обраній сфері. або власноруч зібрані данні;
- оцінка та підготовка даних: оцінка обраного набору даних на Kaggle щодо достовірності та релевантності для подальшого аналізу. При необхідності проведення підготовки даних для подальшого використання;

- вибір моделі: вибір відповідної архітектури моделі або допоміжних алгоритмів для виявлення шахрайства. Це можуть бути готові бібліотеки чи власноруч написані алгоритми;
- навчання: навчання моделі на підготовлених даних, використання оптимізатора для оновлення параметрів моделі та визначення втрат для оцінки точності прогнозів;
- порівняння результатів: порівняння результатів, отриманих від власного експерименту, з результатами, які можуть бути опубліковані або доступні від конкурентів;
- висновки та рекомендації: формування висновків щодо ефективності власних методів в порівнянні з методами конкурентів. На основі цього можуть бути розроблені рекомендації щодо вдосконалення власних практик з виявлення шахрайства.

4 ПРОВЕДЕННЯ ЕКСПЕРИМЕНТУ

4.1 Проектування прототипу

Для створення прототипу було використано такі бібліотеки: Pandas, NumPy, Matplotlib, Scikit-learn.

PyTorch – це науковий обчислювальний інструментарій та платформа для глибокого навчання. Тензорна бібліотека PyTorch та супровідні операції з тензорами в основному відповідають за наукові обчислювальні можливості PyTorch. У тензори PyTorch вбудовано підтримку графічних процесорів. Якщо в нашій системі встановлений графічний процесор, переміщення тензорів на нього та з нього є досить простим за допомогою PyTorch. Тензори мають вирішальне значення для глибокого навчання та нейронних мереж, оскільки вони є структурою даних, яку ми використовуємо для побудови та навчання наших нейронних мереж. PyTorch може запропонувати набагато більше з точки зору розробки та навчання нейронних мереж, ніж просто бібліотека тензорів[18].

На першому етапі було вибрано набір даних та виконано попередню обробку даних . Для цього було використано бібліотеку Scikit-learn (див. рис. 4.1). Вона допомагає яка надає функціональність для створення та тренування різноманітних алгоритмів класифікації, регресії та кластеризації, таких як лінійна регресія, random forest, і працює у зв'язці з бібліотеками NumPy.

```
# Select numerical columns
numerical_cols = [cname for cname in train_df.columns if
train_df[cname].dtype in ['int8', 'int16', 'int32', 'float32']]
# Preprocessing for numerical data
numerical_transformer = Pipeline(steps=[('imputer', SimpleImputer(strategy='constant')), ('scale', StandardScaler())])


# Preprocessing for categorical data
categorical_transformer = Pipeline(steps=[('imputer', SimpleImputer(strategy='constant')),
('onehot', OneHotEncoder(dtype=np.int8, handle_unknown='ignore'))])

# Bundle preprocessing for numerical and categorical data
preprocessor = ColumnTransformer(
    transformers=[
        ('num', numerical_transformer, numerical_cols),
        ('cat', categorical_transformer, categorical_cols)
    ])
train_df = preprocessor.fit_transform(train_df)
print('DONE')
```

Рисунок 4.1 – Попередня обробка даних

Датасет має таблицю транзакцій(див. рис. 4.2).

TransactionAmt	ProductCD	card1	card2	card3	card4	card5	...	V330	V331	V332	V333	V334	V335
68.5	W	13926	NaN	150.0	discover	142.0	...	NaN	NaN	NaN	NaN	NaN	NaN
29.0	W	2755	404.0	150.0	mastercard	102.0	...	NaN	NaN	NaN	NaN	NaN	NaN
59.0	W	4663	490.0	150.0	visa	166.0	...	NaN	NaN	NaN	NaN	NaN	NaN
50.0	W	18132	567.0	150.0	mastercard	117.0	...	NaN	NaN	NaN	NaN	NaN	NaN
50.0	H	4497	514.0	150.0	mastercard	102.0	...	0.0	0.0	0.0	0.0	0.0	0.0

Рисунок 4.2 – Датасет транзакцій

Таблиця транзакцій має вигляд:

– TransactionAMT: сума платежу за транзакцію в USD;
– ProductCD: код продукту, продукт для кожної транзакції;
– card1 – card6: інформація про платіжну картку, така як тип картки, категорія картки, банк-емітент, країна тощо;
– addr: адреса;
– dist: відстань;
– P_ та (R__) emaildomain: домен електронної пошти покупця та одержувача;
– C1-C14: підрахунок, наприклад, скільки адрес пов'язано з платіжною карткою тощо. Фактичне значення маскується;
– D1-D15: часовий інтервал, наприклад, кількість днів між попередніми транзакціями тощо;
– M1-M9: збіг, наприклад, імена на картці та адреси тощо.

Датасет було розділено за 2 типами на категоріальні та числові. Також в цьому датасеті були пропущені дані і для її відтворення було використано Scikit. Для цього було використано класс SimpleImputer. SimpleImputer використовується для заповнення пропущених значень в числових та категоріальних даних перед подальшою обробкою.

Наступним кроком було перетворення значень у бінарні змінні. Для цього було використано клас OneHotEncoder. Він створює нові бінарні змінні для кожної

категорії у вихідній змінній і заповнює їх значеннями 0 або 1, в залежності від того, чи містить кожен запис певну категорію.

Останнім кроком було об’єднання даних за допомогою ColumnTransformer. Це робиться шляхом вказання, який попередній обробник застосовується до яких стовпців даних. Цей підхід автоматизує процес підготовки даних перед їх подальшим використанням у моделі машинного навчання [19].

Після цього за допомогою PyTorch було створено нейронну мережу з чотирма повнозв’язними шарами(див. рис. 4.3).

```
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.fc1 = nn.Linear(543, 256)
        self.fc2 = nn.Linear(256,128)
        self.fc3 = nn.Linear(128,64)
        self.fc4 = nn.Linear(64,1)

        self.dropout = nn.Dropout(p= 0.6, inplace = True)

    def forward(self, x):
        x = self.dropout(F.relu(self.fc1(x)))
        x = self.dropout(F.relu(self.fc2(x)))
        x = self.dropout(F.relu(self.fc3(x)))
        x = F.sigmoid(self.fc4(x))
        return x
```

Рисунок 4.3 – Реалізація нейромережі

Це базовий тип нейронної мережі, який складається з послідовності повнозв’язних шарів, де кожен нейрон у шарі з’єднаний з усіма нейронами попереднього та наступного шарів. У нейромережі є чотири повнозв’язних шари (fc1, fc2, fc3, fc4), і вона використовує регуляризацію за допомогою методу відкидання (dropout) для запобігання перенавчанню. Функції активації relu застосовуються до перших трьох шарів, а sigmoid використовується на виході. Останній шар має один нейрон, що використовується для бінарної класифікації. Кожен повнозв'язний шар має свій власний об'єкт “nn.Linear”, який відповідає за

лінійне перетворення вхідних даних. Для оновлення ваг мережі було застосовано оптимізатор Adam та функцію втрат BCELoss (див. рис. 4.4)

```
criterion = BCELoss()
optimizer = Adam(net.parameters())
```

Рисунок 4.4 – Налаштування тренування та валідації

Оптимізатор Adam (скорочено від Adaptive Moment Estimation) є популярним алгоритмом оптимізації, який використовується при тренуванні нейронних мереж. Він поєднує переваги двох інших розширень стохастичного градієнтного спуску: AdaGrad та RMSProp. Ось основні особливості та принцип роботи Adam:

- адаптивні коефіцієнти навчання: Adam динамічно налаштовує коефіцієнт навчання для кожного параметра, що дозволяє ефективніше тренувати модель;
- оцінка моментів: обчислює експоненціально згладжене середнє попередніх градієнтів (подібно до імпульсу);
- корекція зміщення: Adam включає терміни корекції зміщення для нейтралізації зміщення, яке виникає на початкових етапах тренування, коли оцінки моментів зміщені до нуля.

Перевагою цього медота є те що Adam Поєднує переваги як AdaGrad (який добре працює з розрідженими градієнтами), так і RMSProp (який добре працює в нестаціонарних умовах). Також добре працює для великих задач і з даними, які містять багато шуму.

BCELoss (Binary Cross-Entropy Loss) – це функція втрат, яка використовується для задач бінарної класифікації. Вона вимірює різницю між прогнозованими ймовірностями класів і фактичними бінарними мітками, що допомагає моделі налаштувати свої параметри для точніших прогнозів. BCELoss є диференційованою функцією, що дозволяє використовувати її у градієнтному спуску для налаштування параметрів нейронної мережі. Використовує

логарифміні шкали. Використання логарифмів у формулі забезпечує більші втрати для прогнозів, які далекі від фактичних міток, що стимулює модель бути більш точною. Однією з проблем BCELoss є числова нестабільність, яка може виникати при обчисленні логарифмів дуже малих чисел, що може призвести до значення втрати NaN (Not a Number). Щоб уникнути цього, в практичних реалізаціях додають невелику константу (епсилон) до прогнозованих ймовірностей перед обчисленням логарифмів. BCELoss є важливою функцією втрат для задач бінарної класифікації, яка допомагає налаштовувати нейронні мережі для точніших прогнозів, враховуючи різницю між прогнозованими ймовірностями та фактичними мітками.

Наступним кроком було проведення тренування моделі(див рис. 4.5). Було використовано оцінку AUCROC для тренувального та валідаційного наборів даних.

```
train_auroc_list, valid_auroc_list = [], []
epochs = 20
print('CALCULATION STARTED....')
for epoch in range(1, epochs+1):
    train_hat_list, valid_hat_list, train_true_list, valid_true_list = [], [], [], []
    train_loss_avg = 0
    valid_loss_avg = 0
    for (train_samples, train_targets) in train_loader:
        optimizer.zero_grad()
        net_out = net(train_samples)
        train_loss = criterion(net_out, train_targets)
        train_loss_avg += train_loss
        train_loss.backward()
        optimizer.step()
        for iterrr,ittter in zip(net_out, train_targets):
            train_hat_list.append(iterrr.item())
            train_true_list.append(ittter.item())
    train_loss_avg = train_loss_avg/len(train_loader)
    with torch.no_grad():
        net.eval()
        for (valid_samples, valid_targets) in valid_loader:
            val_out = net(valid_samples)
            valid_loss = criterion(val_out,valid_targets)
            valid_loss_avg += valid_loss

            for iiiter,iteeer in zip(val_out, valid_targets):
                valid_hat_list.append(iiiter.item())
                valid_true_list.append(iteeer.item())
        valid_loss_avg = valid_loss_avg/len(valid_loader)
    net.train()
    train_accuracy_auroc = roc_auc_score(train_true_list, train_hat_list)
    train_auroc_list.append(train_accuracy_auroc)
    valid_accuracy_auroc = roc_auc_score(valid_true_list, valid_hat_list)
    valid_auroc_list.append(valid_accuracy_auroc)
    print("Epoch: {} of {}".format(epoch,epochs))
    print("TRAIN loss: {} | VALID loss: {}".format(train_loss_avg, valid_loss_avg))
    print("TRAIN roc auc score: {} | VALID roc auc score: {}".format(train_accuracy_auroc, valid_accuracy_auroc))
    early_stopping(valid_accuracy_auroc, net)

    if early_stopping.early_stop:
        print("Early stopping")
        break
```

Рисунок 4.5 – Реалізація методу навчання нейромережі

На останньому кроці було відображено графік ROCAUC (див. рис. 4.6). Графік, на якому відображаються значення метрики під час тренування і валідації моделі протягом кожної епохи. Графік допомагає візуалізувати, як змінюється точність моделі під час навчання. Якщо криві точності зростають з епохами, це означає, що модель покращується. Якщо точність валідації перестає покращуватися або навіть зменшується, це може бути ознакою надмірного пристосування, і може бути необхідно використовувати методи регуляризації або зупиняти навчання раніше за допомогою механізму ранньої зупинки (early stopping).

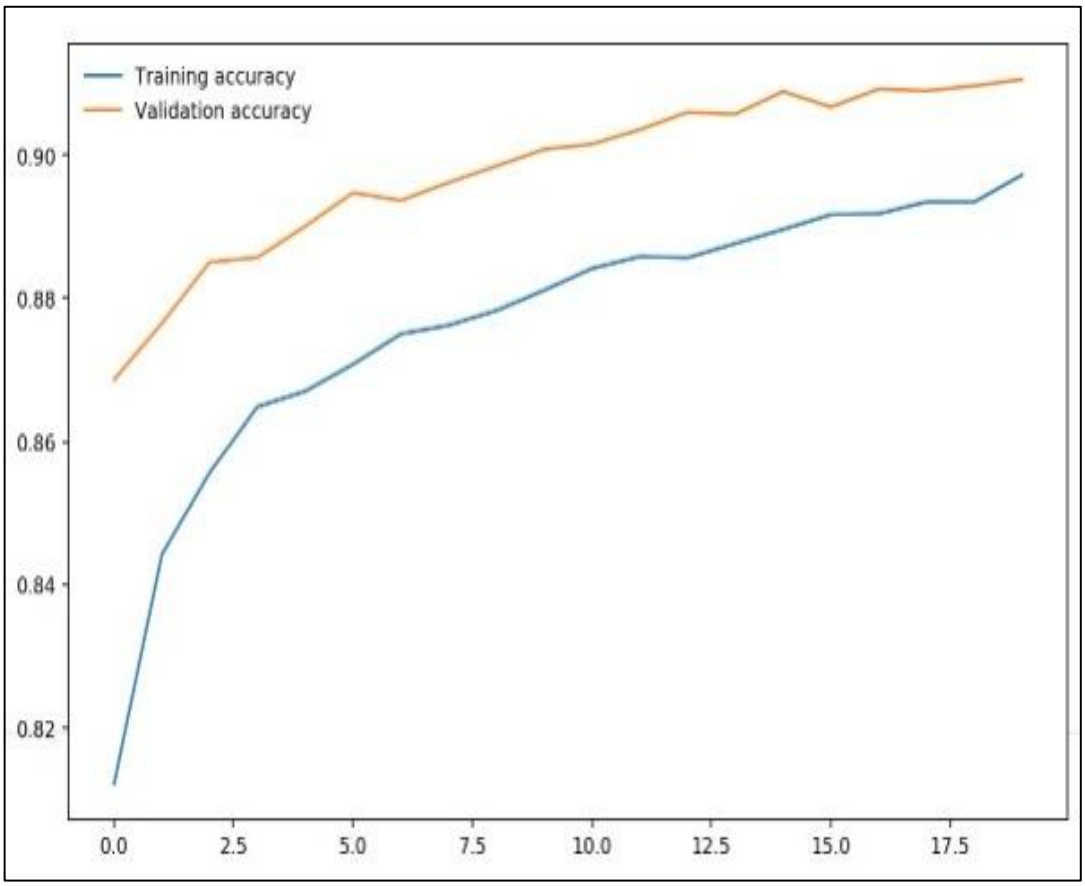

Рисунок 4.6 – графік ROCAUC

Значення для тренувального набору даних позначені лінією "Training accuracy", а для валідаційного – лінією "Validation accuracy.

4.2 Нейромережа з Hawk optimization algorithm

За основу було взято код прототипу і реалізовано алгоритм Hawk Optimization Algorithm (HOA). На рисунку 4.7 зображено реалізацію HOA оптимізатору.

```python
class HawkOptimizer:
    def __init__(self, net, pop_size=30, max_iter=100):  # Increased pop_size and max_iter
        self.pop_size = pop_size
        self.max_iter = max_iter
        self.dim = sum(p.numel() for p in net.parameters())
        self.pop = [self.initialize_solution() for _ in range(pop_size)]
        self.best_solution = None
        self.best_fitness = float('inf')
        self.train_auroc_list = []
        self.valid_auroc_list = []

    def initialize_solution(self):
        solution = []
        for p in self.net.parameters():
            solution.append(p.data.clone().view(-1) + torch.randn_like(p.data.clone().view(-1)) * 0.1)  # Adding randomness
        return torch.cat(solution).to(device)

    def evaluate_fitness(self, solution, loader, criterion):
        idx = 0
        for p in self.net.parameters():
            p.data = solution[idx:idx + p.numel()].view(p.size()).clone()
            idx += p.numel()
        self.net.eval()
        loss = 0
        true_labels = []
        pred_labels = []
        with torch.no_grad():
            for data, target in loader:
                output = self.net(data)
                loss += criterion(output, target).item()
                true_labels.extend(target.cpu().numpy())
                pred_labels.extend(output.cpu().numpy())
        roc_auc = roc_auc_score(true_labels, pred_labels)
        return loss / len(loader), roc_auc

    def update_solution(self, solution, best_solution):
        r1, r2, r3 = np.random.rand(3)
        q = 2 * r3 - 1
        q_tensor = torch.tensor(q, dtype=torch.float).to(device)
        d = torch.abs(best_solution - solution)
        new_solution = best_solution - r1 * d * torch.sign(q_tensor)

        # Adding more variability by introducing noise
        noise = torch.randn_like(new_solution) * 0.01
        new_solution += noise

        return new_solution

    def optimize(self, train_loader, valid_loader, criterion, patience=10):
        early_stopping = EarlyStopping(patience=patience, verbose=True)
        for iter in range(self.max_iter):
            fitnesses = [self.evaluate_fitness(sol, valid_loader, criterion) for sol in self.pop]
            losses, roc_aucs = zip(*fitnesses)
            best_idx = np.argmin(losses)
            if losses[best_idx] < self.best_fitness:
                self.best_fitness = losses[best_idx]
                self.best_solution = self.pop[best_idx].clone()

            new_pop = []
            for sol in self.pop:
                new_sol = self.update_solution(sol, self.best_solution)
                new_pop.append(new_sol)
            self.pop = new_pop

            train_loss, train_roc_auc = self.evaluate_fitness(self.best_solution, train_loader, criterion)
            valid_loss, valid_roc_auc = self.evaluate_fitness(self.best_solution, valid_loader, criterion)

            self.train_auroc_list.append(train_roc_auc)
            self.valid_auroc_list.append(valid_roc_auc)

            print(f"Iteration {iter + 1}, Train ROC AUC: {train_roc_auc}, Valid ROC AUC: {valid_roc_auc}, Best Fitness: {self.best_fitness}")

            early_stopping(valid_roc_auc, self.net)
            if early_stopping.early_stop:
                print("Early stopping")
```

Рисунок 4.7 – Реалізація HOA

HOA – це алгоритм оптимізації, натхненний поведінкою яструбів під час полювання. Він належить до групи метаевристичних алгоритмів і використовує популяційний підхід для пошуку глобального мінімуму або максимуму цільової функції. У контексті машинного навчання HOA може використовуватися для налаштування параметрів нейронних мереж. HOA базується на двох основних фазах:

– фаза пошуку (Exploration Phase): Яструби активно шукають здобич (оптимальні рішення) в різних частинах простору пошуку;
– фаза вилову (Exploitation Phase): Після знаходження здобичі, яструби фокусуються на деталях, щоб зловити її (поліпшити знайдені рішення).

Стратегія хижих птахів реалізується в методі update_solution (див. рис. 4.8), де використовуються випадкові числа r1, r2 і r3, а також змінні q та d, щоб оновити поточні рішення з урахуванням найкращого рішення.

```
def update_solution(self, solution, best_solution):
    r1, r2, r3 = np.random.rand(3)
    q = 2 * r3 - 1
    q_tensor = torch.tensor(q, dtype=torch.float).to(device)
    d = torch.abs(best_solution - solution)
    new_solution = best_solution - r1 * d * torch.sign(q_tensor)

    # Adding more variability by introducing noise
    noise = torch.randn_like(new_solution) * 0.01
    new_solution += noise

    return new_solution
```

Рисунок 4.8 – реалізація стратегії хижих птахів

У цьому методі реалізовано наступні елементи стратегії хижих птахів:

– випадкові числа використовуються для визначення різних стратегій оновлення солюцій;
– напрямок q. Використання випадкового значення для визначення напряму руху;

– поточне рішення оновлюється на основі найкращого рішення з додаванням випадковості. В кінці реалізації програми було виведо графік ROCAUC (див. рис. 4.9).

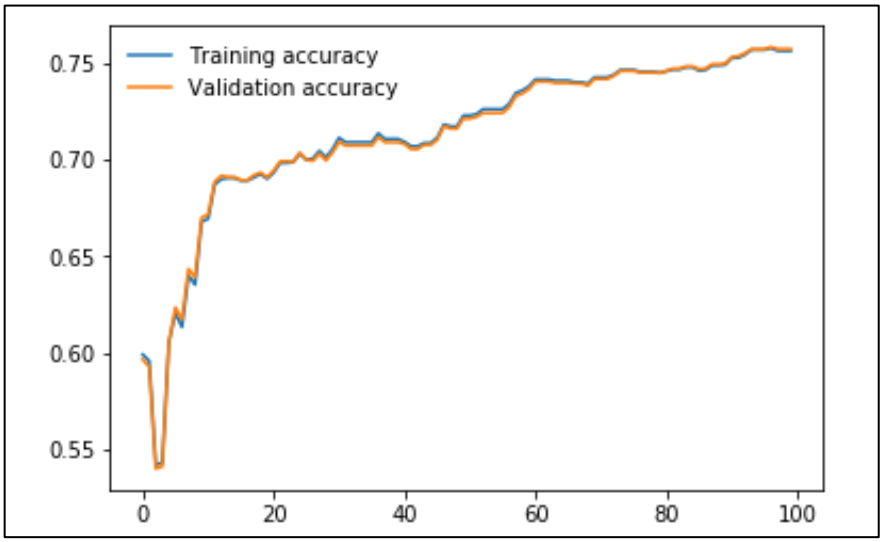

Рисунок 4.9 – ROCAUC для HOA

На рисунку видно що перші кілька ітерацій точність навчання та валідації швидко зростає. Це означає, що модель швидко знаходить основні патерни в даних і починає краще їх передбачати Приблизно після 20-30 ітерацій точність валідації стабілізується і починає зростати повільніше. Це може свідчити про те, що модель досягла певного рівня оптимізації і більше не знаходить значних покращень. В кінці навчання точність валідації досягає значення близько 0.75, що свідчить про прийнятний рівень продуктивності моделі на валідаційному наборі даних. Але щоб покращити результат потрібно додаткові налаштування гіперпараметрів, аугментації даних або додавання більш складних архітектур (наприклад, залишкових блоків).

4.3 GNN з DGL

Останнім проектом з яким було порівнянно 2 інших прототипа було графова нейромережа з використанням бібліотеки DGL [20]. Проект призначений для виявлення шахрайських транзакцій з використанням графових нейронних мереж (GNN) за допомогою Amazon SageMaker та Deep Graph Library (DGL). DGL – це

бібліотека для обробки графів з використанням глибокого навчання. Вона була розроблена для спрощення створення, навчання та застосування моделей глибокого навчання на графах. Проект формулює виявлення шахрайства як задачу класифікації, де GNN навчається розрізняти шахрайські та легітимні транзакції на основі даних взаємодій.

На рисунку 4.10 зображенно тренування моделі.

```
def train(model, trainer, loss, features, labels, train_loader, test_loader, train_g, test_g, train_mask, test_mask,
          ctx, n_epochs, batch_size, output_dir, thresh, scale_pos_weight, compute_metrics=True, mini_batch=True):
    duration = []
    for epoch in range(n_epochs):
        tic = time.time()
        loss_val = 0.

        for n, batch in enumerate(train_loader):
            node_flow, batch_nids = train_g.sample_block(nd.array(batch).astype('int64'))
            batch_indices = nd.array(batch, ctx=ctx)
            with autograd.record():
                pred = model(node_flow, features[batch_nids.as_in_context(ctx)])
                l = loss(pred, labels[batch_indices], mx.nd.expand_dims(scale_pos_weight*train_mask, 1)[batch_indices])
                l = l.sum()/len(batch)

            l.backward()
            trainer.step(batch_size=1, ignore_stale_grad=True)

            loss_val += l.asscalar()

        duration.append(time.time() - tic)
        metric = evaluate(model, train_g, features, labels, train_mask, ctx, batch_size, mini_batch)
        logging.info("Epoch {:05d} | Time(s) {:.4f} | Loss {:.4f} | F1 {:.4f}".format(
                epoch, np.mean(duration), loss_val/(n+1), metric))

    class_preds, pred_proba = get_model_class_predictions(model, test_g, test_loader, features, ctx, threshold=thresh)

    if compute_metrics:
        acc, f1, p, r, roc, pr, ap, cm = get_metrics(class_preds, pred_proba, labels, test_mask, output_dir)
        logging.info("Metrics")
        logging.info("""Confusion Matrix:
                        {}
                        f1: {:.4f}, precision: {:.4f}, recall: {:.4f}, acc: {:.4f}, roc: {:.4f}, pr: {:.4f}, ap: {:.4f}
                    """.format(cm, f1, p, r, acc, roc, pr, ap))

    return model, class_preds, pred_proba
```

Рисунок 4.10 – тренування нейромережі

На цьому рисунку зображено такі аспекти, як:

– нормалізація даних: дану частину виконано у функції normalize, щоб уникнути впливу різних масштабів значень на процес навчання;

– семплювання: використовується для вибірки підграфів з графа під час навчання, що дозволяє ефективніше використовувати пам'ять та обчислювальні ресурси;

– обчислення градієнтів: виконується автоматично за допомогою функції autograd.record();
– оновлення ваг: використовується оптимізатор, який оновлює ваги на основі обчислених градієнтів;
– оцінка продуктивності: регулярна оцінка на валідаційних даних допомагає відстежувати прогрес моделі та уникати переобучення.

На рисунку 4.11 зображено графік ROC-крива для GNN з використанням бібліотеки DGL.

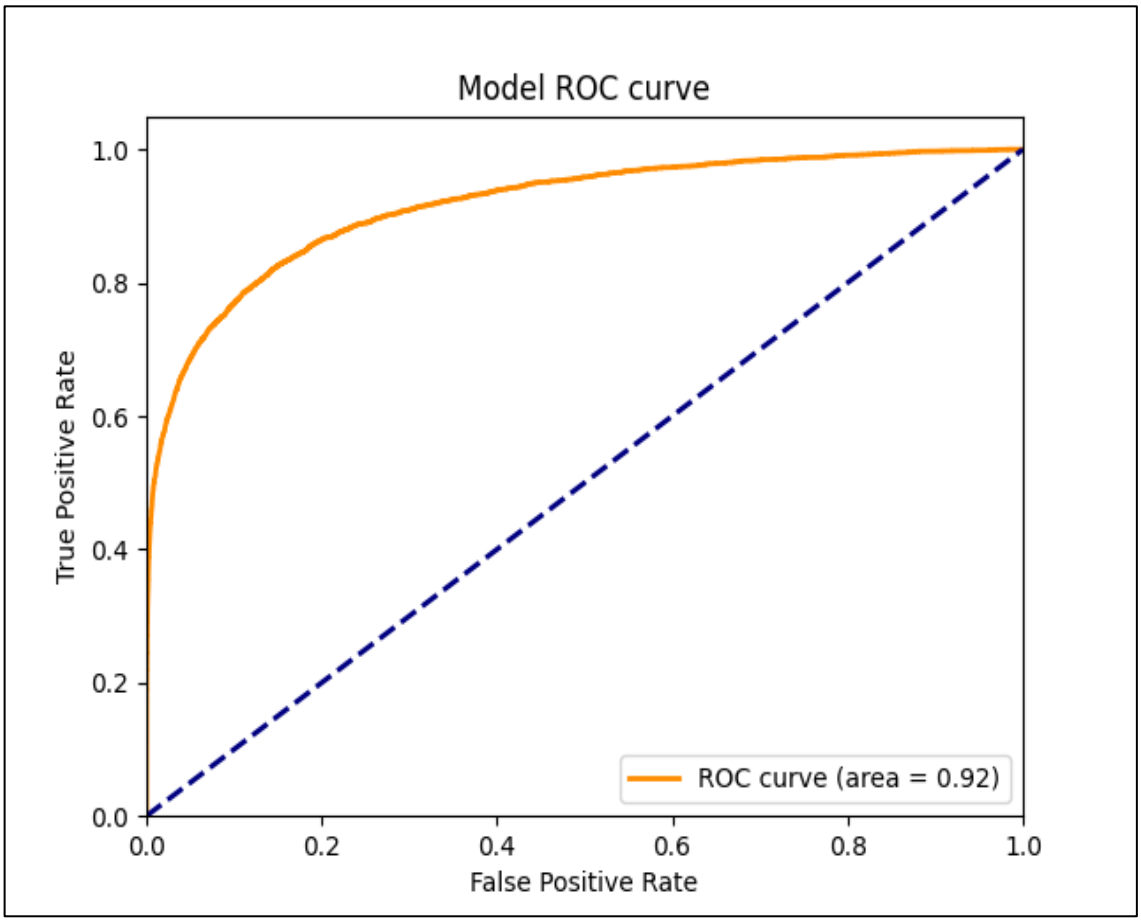

Рисунок 4.11 – ROC крива для GNN з DGL

На графіку зображена ROC-крива, яка показує продуктивність моделі класифікації. AUC – рівна 0.92, що вказує на високу здатність моделі правильно класифікувати позитивні та негативні випадки. Графік демонструє, що модель добре розрізняє позитивні та негативні класи з високою точністю.

4.4 Порівняння методів

Враховуючи, що дані дуже незбалансовані, нам потрібно знайти компроміс між Recall і Precision. Враховуючи, що помилкова класифікація нешахрайських транзакцій як шахрайських серйозно вплине на досвід користувачів, пріоритетом є точність. Тому точність для кожного методу є:

- прототип з Adam: забезпечує високу точність як на навчальних, так і на валідаційних даних (приблизно 0.90);
- GNN з DGL: відмінно працює для задач виявлення шахрайства з високою точністю (AUC = 0.92);
- прототип з HOA: показує стабільне зростання точності, досягаючи близько 0.75 як на навчальних, так і на валідаційних даних.

На основі цих результатів можна зробити висновок, що GNN з DGL є найбільш перспективною моделлю для цього конкретного завдання, оскільки вона досягає найвищої точності. Таким чином, якщо важлива висока точність і ресурсні обмеження не є критичними, GNN з DGL буде найкращим вибором. Adam також показує високу точність і швидке сходження, підходить для багатьох загальних задач. HOA може бути корисним у випадках, коли потрібна стабільність і уникнення перенавчання, хоча його точність трохи нижча.

ВИСНОВКИ

У сучасному світі, де фінансові операції стають все більш цифровими, важливо використовувати ефективні заходи для виявлення шахрайства. Використання нейронних мереж для цього стає все більш популярним та ефективним.

В останні роки зростає попит на використання нейромережевих моделей для виявлення аномалій, особливо в банківській сфері. Банківська галузь особливо схильна до ризиків шахрайства, зокрема, шахрайства з банківськими картками, яке може завдати великої шкоди клієнтам.

Виявлення шахрайства за допомогою нейронних мереж є перспективним напрямком у світі фінансових технологій. Технології штучного інтелекту з використанням глибокого навчання можуть забезпечити більш точні та швидкі результати, ніж традиційні методи. Важливо ретельно підходити до вибору рішення, враховуючи специфіку бізнесу та галузі, в якій воно використовується.

Метою цієї роботи було дослідження різних методів виявлення шахрайства. Основною проблемою було виявлення та аналіз даних, та розробка правильної архітектури. Ці проблеми можуть призвести до нечіткої моделі та збільшення часу для виявлення шахрайства.

У цій роботі ми проаналізували існуючі методи виявлення шахрайства за допомогою нейронних мереж та визначили їхні переваги та обмеження. Нейронні мережі виявляються потужним інструментом для виявлення шахрайства завдяки своїй здатності автоматично виявляти складні закономірності у великих обсягах даних. Однак вони також мають свої обмеження, такі як потреба у великій кількості даних для ефективного навчання та необхідність ретельного підбору конфігурації моделі для досягнення найкращої продуктивності.

В ході роботи ми обрали тип та конфігурацію нейронної мережі, яка найбільше підходить для поставленої задачі, а також підібрали та підготували набір даних для навчання та тестування моделі. Ми навчили нейромережу розпізнавати шахрайські патерни на підготовленому наборі даних.

Після навчання ми проаналізували результати виявлення шахрайства, включаючи точність, чутливість, специфічність та інші метрики, які відображають продуктивність моделі.

Останнім кроком було порівняння рішення з іншими методами виявлення шахрайства. Це дозволило зрозуміти переваги та недоліки нашого підходу порівняно з іншими рішеннями та визначити шляхи подальшого вдосконалення.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. What Is Fraud Detection? Definition, Types, Applications, and Best Practices URL: https://www.spiceworks.com/it-security/vulnerability-management/articles/what-is-fraud-detection/ (дата звернення 09.06.2024)

2. Optimizing Fraud Detection in Financial Services with Graph Neural Networks and NVIDIA GPUs. URL: https://developer.nvidia.com/blog/optimizing-fraud-detection-in-financial-services-with-graph-neural-networks-and-nvidia-gpus/ (дата звернення: 27.12.2023)

3. Fraud Prevention: Definition & How It Works URL: https://www.okta.com/identity-101/fraud-detection/ (дата звернення: 09.06.2024)

4. What is fraud detection and why is it needed? URL: https://www.fraud.com/post/fraud-detection URL: (дата звернення 09.06.2024)

5. Fraud detection and machine learning: What you need to know URL: https://www.sas.com/en_us/insights/articles/risk-fraud/fraud-detection-machine-learning.html (дата звернення 09.06.2024)

6. Цифрові фінанси як інноваційний підхід до вдосконалення сектору фінансових послуг. URL: https://www.researchgate.net/publication/356983363_CIFROVI_FINANSI_AK_INNOVACIJNIJ_PIDHID_DO_VDOSKONALENNA_SEKTORU_FINANSOVIH_POSLUG (дата звернення: 27.12.2023)

7. Що таке фішинг (фішингова атака)? URL: https://futurenow.com.ua/shho-take-fishyng/ (дата звернення: 27.12.2023)

8. Що таке скімінг? URL: https://mywallet.ua/ua/blog/moshennicheskie-shemy-i-afery/chto-takoe-skimming/ (дата звернення: 27.12.2023)

9. Credit card fraud URL: https://legaldictionary.net/credit-card-fraud/ дата звернення: 27.12.2023)

10. Seventh report on card fraud. URL: https://www.ecb.europa.eu/pub/cardfraud/html/ecb.cardfraudreport202110~cac4c418e8.en.html (дата звернення: 28.12.2023).

11. Machine Learning (ML) vs Deep Learning (DL): A Comparative Guide. URL: https://www.datacamp.com/tutorial/machine-deep-learning (дата звернення: 28.12.2023).

12. Afanasieva, I., Golian, N., Golian, V., Khovrat, A., & Onyshchenko, K. (2023). Application of Neural Networks to Identify of Fake News. CEUR Workshop Proceedings, 3396, 346-358.

13. Afanasieva I.V., et al. Neural network approach for emotional recognition in text / I.V. Afanasieva, D.S. Nazarenko, N.V. Golian // Біоніка інтелекту, Харків: ХНУРЕ, 2019. – 1(92). – С. 9-14.

14. Financial Fraud Detection Based on Machine Learning: A Systematic Literature Review. URL: https://www.mdpi.com/2076-3417/12/19/9637 (дата звернення: 28.12.2023).

15. Deep Graph Library URL: https://www.dgl.ai (дата звернення 26.03.2024).

16. The Whale Optimization Algorithm URL: https://www.sciencedirect.com/science/article/abs/pii/S0965997816300163 (дата звернення 26.03.2024).

17. Harris Hawks optimization URL: https://paperswithcode.com/method/hho (дата звернення 26.03.2024).

18. PyTorch URL: https://itwiki.dev/data-science/ml-reference/ml-glossary/pytorch (дата звернення 26.03.2024).

19. Nazarenko, D., Afanasieva, I., Golian, N., & Golian, V. (2021). Investigation of the deep learning approaches to classify emotions in texts. CEUR Workshop Proceedings, 2870, 206-224.

20. Sagemaker-graph-fraud-detection URL: https://github.com/awslabs/sagemaker-graph-fraud-detection/tree/master (дата звернення 01.06.2024)