

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки**

До захисту допущено
Завідувач кафедри

_____ Дмитро ЛАНДЕ
(підпис)

«_____» _____ 2025 р.

**Дипломна робота
на здобуття ступеня бакалавра
за освітньо-професійною програмою «Системи, технології та математичні
методи кібербезпеки»
спеціальності 125 «Кібербезпека»**

на тему: Аналіз ефективності методів виявлення фішингових атак за допомогою
трансформерних моделей

Виконав (-ла): здобувач вищої освіти **IV** курсу, групи **ФБ-11**
(шифр групи)

Яцентюк Андрій Олександрович
(прізвище, ім'я, по батькові) (підпис)

Керівник кандидат фізико-математичних наук **Рибак О.В.**
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові) (підпис)

Рецензент зав.каф. ММЗІ, к.т.н. **Яковлєв С. В.**
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище, ім'я, по батькові) (підпис)

Засвідчую, що у цій дипломній роботі немає
запозичень з праць інших авторів без відповідних
посилань.

Здобувач вищої освіти _____
(підпис)

Київ – 2025 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

Рівень вищої освіти – перший (бакалаврський)
Спеціальність – 125 «Кібербезпека»
Освітньо-професійна програма «Системи, технології та математичні методи кібербезпеки»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Дмитро ЛАНДЕ
(підпис)
«_____» _____ 2025 р.

ЗАВДАННЯ
на дипломну роботу здобувачу вищої освіти

Яцентюк Андрій Олександрович

(прізвище, ім'я, по батькові)

1. Тема роботи: Аналіз ефективності методів виявлення фішингових атак за допомогою трансформерних моделей

науковий керівник роботи _____ ,
кандидит фізико-математичних наук Рибак О.В. _____ ,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «26» травня 2025 р. № 1761-с

2. Термін подання здобувачем дипломної роботи 11 червня 2025 р.

3. Вихідні дані: Наукові роботи про методи виявлення фішингових атак, аналітичні звіти про актуальні тенденції фішингу та кіберінцидентів, технічні статті про архітектуру трансформерних моделей, оригінальні наукові публікації про моделі BERT, RoBERTa, Flan-T5, OPT, Qwen2 та їх тренування, освітні ресурси з реалізації трансформерних моделей у практичних завданнях, професійна література з кібербезпеки.

4. Зміст роботи: огляд основ фішингових атак та методів їх виявлення, аналіз сучасних трансформерних моделей для обробки природньої мови, вибір датасетів і критеріїв оцінки ефективності, донавчання моделей RoBERTa, Qwen2, Flan-T5, експериментальне порівняння результатів, інтерпретація результатів точності та

матриць помилок, формулювання висновків щодо придатності моделей для практичного використання у сфері кібербезпеки.

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо): презентація до захисту дипломної роботи.

6. Дата видачі завдання 28 листопада 2024

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів дипломної роботи	Примітка
1	Визначення теми роботи та її узгодження з науковим керівником	28 листопада 2024	+
2	Формулювання мети дослідження, постановка завдань	листопад 2024	+
3	Огляд наукової літератури по темі дипломної роботи	листопад 2024 – квітень 2025	+
4	Огляд методів виявлення фішингових атак і принципів роботи трансформерних моделей	14 квітня 2025 – 27 квітня 2025	+
5	Огляд популярних трансформерних моделей	28 квітня 2025 – 4 травня 2025	+
6	Аналіз моделей – RoBERTa, Qwen2, T5 та їх можливостей застосування як детектори	5 травня 2025 – 12 травня 2025	+
7	Експериментальне дослідження RoBERTa	12 травня 2025 – 18 травня 2025	+
8	Експериментальне дослідження Qwen2-0.5B Instruct	18 травня 2025 – 22 травня 2025	+
9	Експериментальне дослідження Flan-T5	23 травня 2025 – 31 травня 2025	+
10	Оформлення висновків, списку використаних джерел, переліку умовних позначень, символів, одиниць, скорочень і термінів	1 червня 2025 – 8 червня 2025	+
11	Консультації з науковим керівником	1 червня 2025 – 8 червня 2025	+
12	Написання та переклад реферату	1 червня 2025 – 8 червня 2025	+

Здобувач вищої освіти

(підпис)

Андрій ЯЦЕНТЮК
(власне ім'я, ПРІЗВИЩЕ)

Науковий керівник

(підпис)

Олександр РИБАК
(власне ім'я, ПРІЗВИЩЕ)

РЕФЕРАТ

Дипломна робота містить: 52 сторінок, 13 таблиць, 5 ілюстрацій, 0 додаток, 25 джерел літератури.

Об'єкт дослідження - Методи виявлення фішингових атак в кібербезпеці.

Предмет дослідження – Ефективність трансформерних моделей у виявленні фішингових атак на основі текстових даних.

Мета роботи – тренування та аналіз ефективності трансформерних моделей як інструментів для виявлення фішингових атак на основі текстових даних.

Робота містить опис методів виявлення фішингових атак, а також процес тренування та аналізу ефективності трансформерних моделей.

Результати роботи включають – тренування таких трансформерних моделей для виявлення фішингових атак: RoBERTa, Qwen2 0.5B Instruct, Flan-T5.

Ключові слова - фішингові атаки, трансформерні моделі ,обробка тексту, RoBERTa, Qwen2, Flan-T5.

ABSTRACT

The thesis contains: 52 pages, 13 tables, 5 illustrations, 0 appendix, 25 references.

Object of the research – Methods for detecting phishing attacks in cybersecurity.

Subject of the research – The effectiveness of transformer models in detecting phishing attacks based on textual data.

The aim of the work – To train and analyze the effectiveness of transformer models as tools for detecting phishing attacks based on textual information.

The work includes a description of phishing attack detection methods, as well as the process of training and evaluating the performance of transformer-based models.

The results of the work include – training of the following transformer models for phishing attack detection: RoBERTa, Qwen2 0.5B Instruct, Flan-T5.

Keywords – phishing attacks, transformer models, text processing, RoBERTa, Qwen2, Flan-T5.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів	7
ВСТУП.....	8
1 Аналіз класичних методів виявлення фішингових атак	11
1.1 Основи фішингових атак.....	11
1.2 Класичні методи виявлення фішингових атак	14
Висновки до розділу 1	16
2 Трансформені моделі	17
2.1 Принципи роботи трансформерних моделей.....	17
2.2 Огляд популярних трансформерних моделей.....	24
Висновки до розділу 2	34
3 Експериментальне дослідження.....	35
3.1 Вибір датасету та критерії оцінки ефективності	35
3.2 Експериментальне дослідження ефективності моделі RoBERTa	36
3.3 Експериментальне дослідження ефективності моделі Qwen2-0.5B Instruct.....	39
3.4 Експериментальне дослідження ефективності моделі Flan-T5.....	41
3.5 Експериментальне дослідження ефективності моделі GPT-4.....	44
Висновки до розділу 3	45
Висновки	47
Перелік джерел посилань	50

Перелік умовних позначень, символів, одиниць, скорочень і термінів

AI (Artificial Intelligence) – Штучний інтелект

DNS (Domain Name System) – система доменних імен

GLUE (General Language Understanding Evaluation) – набір тестів для оцінки розуміння мови трансформерними моделями.

Kaggle – онлайн платформа для публікації датасетів

MLM (Masked Language Modeling) – метод попереднього навчання моделей, що передбачає маскуванню частини вхідного тексту для подальшого прогнозування.

NLP (Natural Language Processing) – Обробка природної мови

NSP (Next Sentence Prediction) – задача навчання моделі, яка полягає у прогнозуванні, чи є друге речення продовженням першого.

Векторизація – процес перетворення текстових або інших даних у числову форму для обробки машинного навчання

Токен (Token) – базова одиниця обробки тексту у трансформерних моделях, яка може бути словом або частиною слова.

ВСТУП

У сучасному світі, через стрімкий розвиток інформаційних технологій та цифрових комунікацій, тема кібербезпеки стає більш актуальною з кожним днем. Разом з швидким зростанням кількості користувачів інтернету, електронних платежів та веб-сервісів загалом, відбувається стрімкий зріст кількості кіберзлочинів з використанням соціальної інженерії, а саме фішингових атак. Вони спрямовані на отримання чутливої інформації такої як логіни, паролі, банківські реквізити або інші персональні дані. Через надзвичайну простоту реалізації, психологічний вплив на жертву атаки та здатність проходити крізь досить велику кількість традиційних методів захисту, фішинг становить значну загрозу для окремих користувачів та організацій. Ця загроза є особливо актуальною у корпоративному середовищі, де наслідки успішної фішингової атаки можуть призвести до витоку конфіденційної інформації, компрометацію доступу до захищених ресурсів, порушення бізнес процесів, значні фінансові та репутаційні втрати організації. Окрім того, фішингові атаки стають складнішими з часом, збільшується використання персоналізації, використання соціального контексту та легітимно виглядаючих шаблонів повідомлень, що значно ускладнює їх виявлення.

Актуальність роботи полягає в тому, що стрімкий розвиток трансформених мовних моделей, зокрема BERT, DeepSeek, GPT та інших, дозволяє використовувати ШІ до широкого спектру задач обробки інформації. У зв'язку з цим, більшої уваги набуває використання штучного інтелекту, а саме методів глибинного навчання пов'язаних з використанням трансформерних моделей спрямованих на оброблення природної мови, що дозволяє аналізувати потенційно шкідливий текст з урахуванням контексту. Ці моделі здатні виявляти складні синтаксичні та семантичні залежності в тексті, що відкриває нові можливості для створення детекторів фішингових листів.

Основною метою роботи є дослідження ефективності використання генеративних трансформених моделей у якості детектора фішингових листів. Для

проведення цього дослідження необхідно, виконати дослідження ефективності виявлення фішингових листів без тренування використовуючи стандартну модель, виконати дослідження ефективності виявлення фішингових листів з додатковим тренуванням, виконати дослідження ефективності виявлення фішингових листів з використанням додаткової допоміжної інформації, виконати дослідження отриманих результатів та порівняти отримані результати їх роботи один з одним та з раніше розробленими методами з виявлення фішингових листів. Таке багатоступеневе дослідження дозволить оцінити точність трансформерів, виявити їх обмеження та можливий потенціал для впровадження в корпоративних та персональних системах електронної пошти.

Об'єктом дослідження є методи виявлення фішингових атак у кібербезпеці.

Предметом дослідження є ефективність трансформерних моделей у виявленні фішингових атак на основі виключно текстових даних.

Методами дослідження є використання аналітичного методу, що в себе включає використання наукових публікацій, технічної документації, відкритих матеріалів досліджень та аналіз сучасних практик виявлення фішингових атак. Експериментального методу, що в себе включає донавчання трансформерних моделей на спеціально підготовлених даних. Метод порівняльного аналізу, що буде використаний для оцінки ефективності моделей за різними метриками такими як точність, F1-міра у різних конфігураціях, а саме без навчання, з донавчанням, та з залученням додаткової контекстної інформації. А також метод практичного тестування, що полягає у практичному тестуванні на реальних або спеціально створених прикладах фішингових листів.

Практичне значення роботи полягає у визначенні можливостей ефективного використання трансформерних мовних моделей як інструменту виявлення фішингових атак. Крім того, отримані результати дослідження можуть бути використані як основа для подальшого інтегрування трансформерних моделей у

системи фільтрації електронної пошти, у якості окремих модулів або інструментів у системах захисту.

1 Аналіз класичних методів виявлення фішингових атак

1.1 Основи фішингових атак

З кожним роком інтернет технології набувають все більшу важливість у життєдіяльності організацій та житті окремих користувачів.

Разом з стрімким розвитком цифрових сервісів зростає і кількість кіберзагроз, серед яких особливе місце посідає фішинг.[3][4]

Фішинг – це один з найстаріших методів кібершахрайства, що полягає у використанні соціальної інженерії для отримання конфіденційної інформації добровільно від користувача[2]. Серед поширених прикладів цільової інформації для викрадення можна назвати логіни, паролі, банківські реквізити, персональні дані. Ціллю такої фішингової атаки може бути як окрема людина так і ціла організація.

Поняття фішингу (від англ. *phishing*) виникло ще в середині 1990х років і походить від слова *fishing* – що означає “рибальство”, де замість вилову риби, зловмисники займаються виловом конфіденційної інформації користувачів. Початкові фішингові атаки мали досить примітивний характер, однак на ранніх етапах навіть вони виявились ефективними.

Однією з перших організацій, що стала масовою жертвою фішингової атаки, була “America Online” (AOL). У 1995-1997 роках зловмисники розсилали користувачам повідомлення від імені служби підтримки AOL з проханням підтвердити облікові записи або надати паролі чи платіжні дані. Ці повідомлення були досить прості змістовно, та сучасний користувач дуже легко розпізнав би їх як фішингові, проте на той час користувачі не мали жодного досвіду з розпізнавання таких загроз, тому атака дуже часто була успішною. В результаті фішингової атаки на AOL було викрадено значну кількість облікових записів, що далі були використані для поширення ще більшої кількості фішингових повідомлень.

У 2000х роках фішинг почав набувати масового явища. Особливої популярності набули фішингові листи, що імітували повідомлення від банків або платіжних систем. Наприклад eBay, PayPal, тощо. Ці листи як правило містили попередження про “блокування рахунку” або іншу підозрілу активність що спонукала користувачів перейти на посилання, що було надане разом з повідомленням. Ці посилання вели на підроблені веб-сайти що були візуально схожі на їх легітимні варіанти, що ускладнювало розпізнавання спроби використання соціальної інженерії.

Разом з активним розвитком інтернет технологій та соціальних мереж, фішингові атаки еволюціонували разом з популярними технологіями. Почали набувати популярності більш персоналізовані підходи. Кіберзлочинці почали використовувати методи розвідки з відкритих джерел, а саме аналіз відкритої інформації з профілів користувачів для створення персоналізованого підходу до жертви. Такі атаки стали відомі як *spear-phishing* або цільова фішинг атака. Більш розвиненою версією цільового фішингу стала *whaling* (від англ *whale* - кит) – фішингова атака спрямована на верхню частину керівництва організацій таких як топ-менеджери, фінансові директори або адміністратори мережей, що мали доступ до дуже важливих інформації та систем.

Ці види атак залишаються досить успішним і дотепер, через поєднання соціальної інженерії та технічних засобів, які зловмисники постійно вдосконалюють.

Як приклади успішних сучасних фішингових атак можна привести атаку на американські магазини “Target”. У 2013 році стався виток конфіденційної інформації мільйонів постійних покупців мережі магазинів “Target”. Витік почався з іншої організації “Fazio Mechanical”, організація що поставляла холодильники у магазини. Один робітник компанії “Fazio Mechanical” не помітив фішингової атаки та дозволив зловмисникам встановити шкідливе програмне забезпечення на робочу станцію що дало можливість кіберзлочинцям розповсюдитись по мережі комп’ютерів що належали “Target”.

Іншим більш сучасним прикладом успішної кібератаки з використанням фішингу була кібератака у липні 2020 року на організацію “Twitter”. Зловмисники виконали цілеспрямовану фішингову атаку на працівників організації та переконали декілька працівників перейти по посиланню на підроблену версію внутрішнього корпоративного веб-сайту, що дозволило кіберзлочинцям отримати доступ до адміністративних акаунтів. Завдяки цьому, шахраї захопили близько сотні акаунтів з яких 40 використовувались для публікацій шахрайських публікацій, що пропонували махінації з ерипто-валютою. Більше ніж 100 тисяч долларів було втрачено за лічені години. [6]

Багато користувачів, особливо в умовах перевантаженості інформаційного середовища, не здатні розпізнати фальшиві повідомлення або підробленні вебсайти, особливо якщо ті відзеркалюють відомі веб-сервіси. Крім того, фішинг часто орієнтований на експлуатацію емоційної реакції користувача – наприклад страху перед авторитетом, терміновості деякої дії, загальної цікавості

Загалом, усі фішингові атаки можна класифікувати за масштабом та спрямованістю на дві основні категорії – нецільовий, та цільовий фішинг.[24]

Нецільовий фішинг – це форма фішингової атаки, яка надсилається одночасно великій кількості користувачів без попереднього вибору конкретної жертви. Головною метою такої атаки є максимально можливе охоплення аудиторії з надією, що хоча б частина отримувачів не побачить в фішинговому листі нічого підозрілого. Зазвичай злочинці використовують шаблонні електронні листи або повідомлення відомих сервісів (поштові сервіси, банки, платіжні системи, стрімінгові сервіси та інші), у якому повідомляється про підозрілу активність або про щось незвичайне пов’язане саме з жертвою атаки, спонукаючи її виконати деякі необхідні для зловмисника дії. Цей вид фішингу залишається ефективним через високий рівень довіри до авторитетних брендів і сервісів, а також через недостатнє приділення уваги користувачів до кібергігієни.

Цільовий фішинг – це протилежна форма нецільового фішингу. Цей вид фішингу зосереджений на отримання конфіденційної інформації від конкретної особи чи групи осіб. На відміну від нецільового фішингу, його характерною рисою є попереднє дослідження своєї потенційної жертви зловмисником за допомогою відкритих джерел, наприклад соціальних або корпоративних мереж, та за допомогою витоків (наприклад документації щодо правильної форми запитів, тощо). Зловмисник створює персоніфіковане повідомлення яке виглядає правдоподібно та буде більш цікавим та адаптоване до очікувань саме для конкретного отримувача. Саме такий рівень персоналізації робить цільовий фішинг надзвичайно небезпечним і складним для виявлення стандартними методами виявлення фішингових атак.

У межах цієї роботи основну увагу буде приділено електронним листам як головному каналу доставки фішингових повідомлень, а також аналізу сучасних популярних трансформених моделей для їх автоматизованого виявлення.

1.2 Класичні методи виявлення фішингових атак

На сьогоднішній день існує велика кількість антивірусного програмного забезпечення, які окрім загального захисту системи пропонують послуги попередження фішинговим атакам. Проте ці програмні засоби не гарантують сто відсоткового захисту.

Можливо виділити два основних підходи до виявлення фішингових атак: навчання користувачів та використання програмних засобів.[1][7]

Навчання користувачів – будь якого користувача можна навчити краще розуміти методи фішингових атак. Мета такого навчання є надати користувачу необхідні знання для самостійного виявлення спроб фішингу.

Існує декілька можливих варіантів впровадження такого навчання у організацію а саме – Інтерактивні симуляції, навчальні курси.

Інтерактивні симуляції фішингових атак - користувачі отримують умовно шкідливі листи, що поивнні симулювати реальну фішингову атаку. При успішному виявленні фішингової спроби або при невдалому виявленні система може надати зворотній зв'язок з користувачем.

Навчальні курси – онлайн або офлайн курси що охоплюють теоретичні основи кібергігієни, практичні вказівки та поради що полегшать виявлення фішингової атаки та популярні шаблони атак.

Використання програмних засобів має на меті допомогти користувачу під час роботи. Дуже часто фішингові атаки, особливо цілові, виглядають надзвичайно легітимними, навідь заздалегідь навчений користувач може не зрозуміти своєї помилки, або користувачів занадто багато і було б занадто дорого навчати кожного розпізнавати методи соціальної інженерії. Саме цю прогалину і намагаються вирішити програмні засоби виявлення фішингу.

Програмні методи виявлення фішингових повідомлень розділяються на 3 головні методи – чорні списки, евристика, машинне навчання.

Чорні списки - це доступні усім, постійно оновлювальні списки які містять інформацію щодо раніше виявлених фішингових атаках. Великим недоліком цього методу є значна затримка між часом виявлення фішингової атаки, її аналізом, та оновленням чорного списку. Цієї затримки може бути достатньо для проведення фішингових атак на декілька різних цілей перед оновленням чорного списку.

Евристичні методи виявлення фішингових атак базуються на використанні визначених правил, які дозволяються виявити потенційно шкідливі повідомлення. Цей метод є досить простим в реалізації та не потребує великого об'єму навчальних даних. Ось декілька прикладів евристичних правил – використання підозрілих вкладень (.exe, .js), розбіжність між назвою посилання та самим посиланням, наявність небезпечних символів, використання слів для емоційного тиску (негайно, як можна скоріше), відсутність персоналізації (Дорогий

користувач, ...), використання схожих на відомі домени (paypal.com / paypal.com). На практиці евристичні фільтри впроваджуються у більшості поштових клієнтів. Проте цей метод не є ідеальним через велику кількість помилкових виявлень та низьку стійкість до складних та цільових атак (spear-phishing).

Машинне навчання на теперішній час є одним з найефективніших методів виявлення фішингових повідомлень. Основна перевага цього підходу полягає в тому що модель здатна навчитись розпізнавати велику кількість фішингових повідомлень за наявності достатньої кількості прикладів. Завдяки цьому моделі машинного навчання можуть швидко адаптуватись до нових паттернів фішингових атак, які раніше не спостерігались. Така гнучкість надзвичайно важлива у сфері кібербезпеки в цілому як і в сфері фішингу. Зловмисники постійно модифікують свої методи та паттерни атаки намагаючись обійти стандартні методи захисту. Навчання моделі зазвичай виконується на основі деяких ознак повідомлення таких як наявність підозрілих слів, загальна структура повідомлення, стиль написання, тощо.

До найбільш поширених алгоритмів, які використовуються для виявлення фішингових атак є дерева рішень, метод опорних векторів та нейронні мережі.

Висновки до розділу 1

В результаті проведеного аналізу фішингових атак, було визначено що фішинг все ще залишається одним із найбільш поширених та небезпечних видів кібрзлочинства, що з часом тільки активно розвивається разом з розвитком цифрових технологій.

Фішингові атаки базуються на методах соціальної інженерії, маніпулюючи емоціями та реакціями жертв. Особливу небезпеку становлять цільові атаки типу spear-phishing, що використовують детальну інформацію про потенційну жертву для підвищення ефективності. Фішингові атаки досі залишаються ефективними через людський фактор, недостатній рівень кібер-гігієни та довіру до сервісів.

2 Трансформені моделі

2.1 Принципи роботи трансформерних моделей

У попередньому розділі були розглянуті більш класичні методи виявлення фішингових атак. Хоча вони все ще залишаються досить ефективними, класичні методи часто виявляються обмеженими у виявленні нових фішингових атак.

У зв'язку з цим методи що базуються на машинному навчанні набувають все більшої актуальності – а зокрема на трансформерних моделях, які є найпотужнішим інструментом для оброблення людської мови. Вони здатні аналізувати тексти з урахуванням контексту, семантичних та структурних особливостей, що робить їх надзвичайно привабливими для задач класифікації – зокрема, виявлення фішингових листів.

У цьому розділі буде розглянуто основні принципи роботи трансформерних моделей, описані найбільш популярні моделі, що використовуються для обробки текстових даних, а також проаналізовано їх доцільність у контексті кібербезпеки.

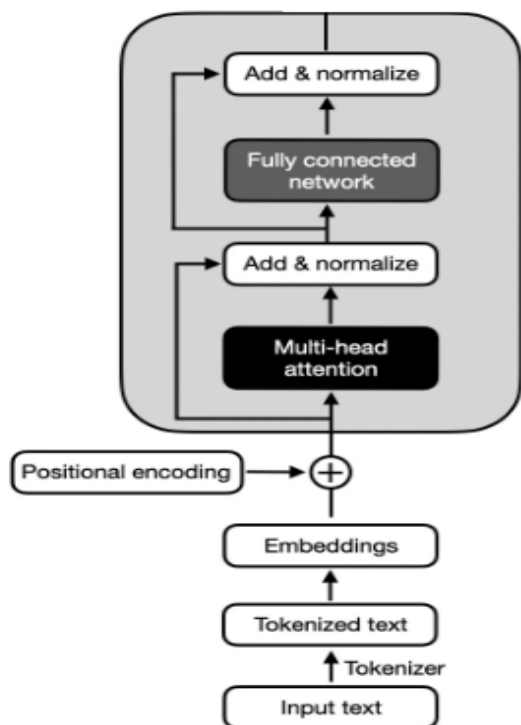
Що таке трансформена модель?

Трансформена модель – це тип нейронної мережі, яка найбільше підходить для обробки послідовних даних, і найчастіше асоціюється з великими мовними моделями. Трансформена модель була вперше описана у науковій статті “Attention is All You Need”[13] (2017 рік публікації), і з того часу стала основою більшості сучасних великих мовних моделей (GPT, BERT, RoBERTa, та інші).

Головним нововведенням до моделей став механізм уваги, який дозволяє моделі звертати найбільшу увагу на найбільш важливі частини вхідних даних, незалежно від послідовності. Це дозволяє уникнути обмежень характерних для традиційних рекурентних мереж - запам'ятовування довготривалих залежностей.

Архітектура енкодер/декодер (Encoder/Decoder)

Encoder



Архітектура енкодер/декодер є ключовою в нейронних мережах що спеціалізуються на обробці послідовностей даних. Ця архітектура дозволяє досить ефективно перетворювати вхідні послідовності будь-якої довжини у відповідні вихідні послідовності враховуючи контекст та складну структуру даних.

Перед тим як інформація поступає до кодера, вона проходить попередню обробку. Перший крок -токенізація, другий - перетворення в векторну форму, третій крок - додається позиційне кодування.

Рисунок 2.1 – Структура енкодера

Процес токенізації розділяє потік вхідних даних на токени. Токен – це набір символів які мають семантичну значущість для моделі. Токенами можуть бути як цілі слова так і їх частини.[17]

Після токенізації, вже токен перетворюється в у векторну форму фіксованого розміру, так щоб результуючий вектор зберігав семантичні властивості слів. Як результат, токени зі схожими значеннями матимуть схожі векторні представлення.

Оскільки трансформена архітектура не є рекурентною, та не має архітектурно вбудованого уявлення про порядок токенів, до кожного перетвореного вектора додається позиційна інформація. Цей процес дозволяє враховувати порядок токенів у послідовності, що є важливою частиною розуміння вхідної послідовності, особливо тексту.

Після перед-підготовки вхідної послідовності, дані передаються до блоку енкодера. У типових кодерах присутні декілька послідовних блоків кодера, що дозволяє поступово поглиблювати розуміння контексту для моделі. Перші блоки звертають увагу на прості або локальні зв'язки в даних, тоді як наступні звертають увагу на більш глобальні та складні зв'язки.

Енкодер відповідає за аналіз вхідної послідовності та побудову її багатовимірного представлення. Архітектура енкодера складається з однотипних блоків котрі включають: механізм багатоголової-уваги (Multi-head attention), механізм залишкового зв'язку та нормалізації (Add & normalize), повнозв'язну нейрона мережа (Fully connected network).

Механізм уваги є першим кроком обробки. Він відповідає за встановленням залежностей між токенами незалежно від їх відстані один від одного. Основою цього процесу є обчислення трьох матриць для кожного токена : запиту, ключа, значення (Query – Q, Key - K, Value - V), шляхом множення на отриманні в результаті тренування матриці вагів W_q , W_k , W_v відповідно. [10]

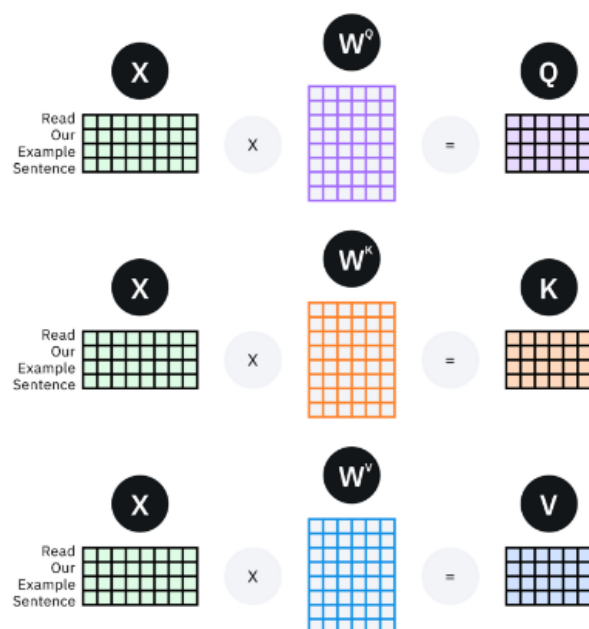


Рисунок 2.2 – Отримання матриць Q, K, V

Далі, виконується обчислення важливості кожного токена по відношенню до кожного іншого токена використовуючи цю функцію:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \quad (2.1)$$

Де Q, K, V – матриці запитів, ключів та значень відповідно, d_k – розмірність вектора ключа (використовується щоб уникнути великих значень при скалярному добутку)

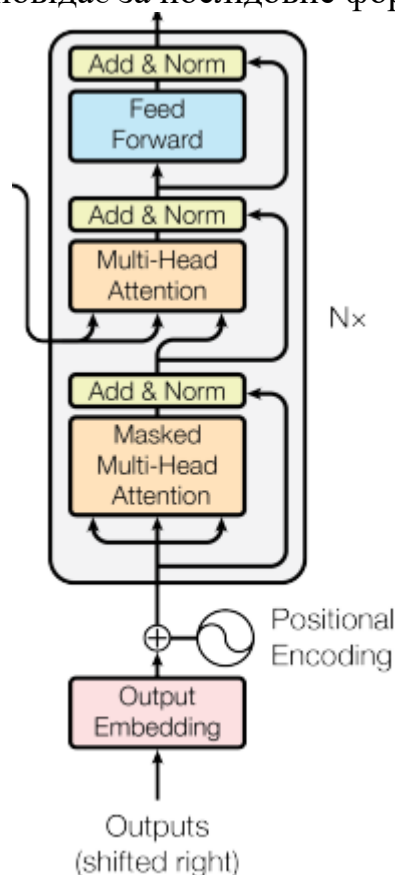
Щоб покращити здатність моделей враховувати різні типи залежностей, використовується механізм багатоголової-уваги (multi-head attention). У ньому використовуються декілька незалежних “голів”, у кожній з них є власні ваги W_q , W_k , W_v . Результати кожної “голови” конкатенуються та лінійно перетворюються в матрицю з контекстно зваженим набором векторів, розмірно сумісною з іншими шарами.[11][13][15]

Наступним кроком обробки інформації є додавання залишкових зв’язків та подальша нормалізація. Результат багатоголової-уваги додається поелементно до її початкового вхідного вектора, що дозволяє зберегти початкову інформацію, після чого виконується нормалізація для стабілізації вихідних даних цього етапу.

Останнім етапом кожного блоку енкодера є обробка інформації у двошаровій повнозв’язній нейронній мережі та подальша нормалізація вихідних даних.

Вихідні дані кожного блоку передаються як вхідні дані на наступний блок енкодера. Лише після обробки даних усіма шарами енкодера, вихід останнього блоку передається на вхід до декодера.

Декодер є другою важливою частиною архітектури енкодер/декодер і відповідає за послідовне формування вихідної послідовності даних на основі



попереднє оброблених даних (дані що пройшли передобробку для використання в кодері) та результатів роботи енкодера.

Структура декодера складається з однакових послідовно з'єднаних блоків схожих за структурою з блоками енкодера. Внутрішня структура декодера може відрізнитись від моделі до моделі в залежності від її задачі. Проте є декілька загальних компонентів: Маскований механізм багатоголової-уваги(Masked multi-head Attention), Механізм уваги до виходу енкодера (Encoder-Decoder Attention), повнозв'язна нейрона мережа(Feed-Forward Network).[9][16]

Рисунок 2.3 – Структура декодера

Першим та надзвичайно важливим етапом обробки у кожному блоці декодера є використання механізму маскованої багатоголової-уваги. Його основна функція – забезпечення генерації кожного нового токена використовуючи лише ті елементи, що були сгенеровані раніше або були передані на вхід енкодера раніше(авторегресивна генерація). Це досягається шляхом маскування частини входу під час обчислення уваги.

Обчислення маскованої багатоголової-уваги базується на вище зазначеній формулі обчислення звичайної багатоголової-уваги, проте з додаванням маски у вигляді верхньотрикутної матриці до матриці схожості (результат перемноження матриць Q та K). Маска обмежує вплив наступних токенів на минулі, за допомогою заповнення відповідних елементів маски значеннями в $-\infty$ перед нормалізацією функцією *softmax*.

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}} + M)V \quad (2.2)$$

Де M – маска.

Механізм маскованої-уваги дозволяє кожному токеноу у послідовності отримувати інформацію лише від попередніх tokenів що дозволяє генерувати наступний token зважаючи тільки на попередню інформацію.

Другим компонентом кожного блоку декодера є Механізм уваги до виходу енкодера (Encoder-Decoder Attention). Його мета – врахування інформації з вхідної послідовності, що була оброблена енкодером.

У цьому блоці поточні query-вектори з декодера порівнюються з векторами ключів та значень з енкодера. Цей етап функціонує аналогічно звичайного блоку багатоголової-уваги, проте з деякими важливими відмінностями: query-вектори формуються на основі вже сгенерованої декодером послідовності, вектори ключів та значень беруться з останнього шару енкодера.

Цей механізм дозволяє моделі пов'язувати кожен вихідний token з релевантною частиною вхідної послідовності. Також, цей блок включає в себе додавання залишкових зв'язків та нормалізацію.

Останнім етапом обробки інформації декодером є використання двошарової-повнозв'язної нейронної мережі до кожного токена незалежно від інших, що дозволяє моделі змінювати представлення кожного токена вихідної послідовності так щоб воно краще відповідало зв'язкам, які були виявленні на попередніх етапах застосування механізмів уваги. Цей блок також застосовує етап додавання залишкових зв'язків та нормкалізацію.

Після огляду основ функціонування трансформених моделей та її ключових компонентів, а саме механізму самоуваги (Self-Attention mechanism), є доцільним перейти до розгляду практичного застосування таких моделей у практичних завданнях у сфері кібербезпеки. Однією з найактуальніших сфер є виявлення

фішингових електронних листів. Ця проблема має критичне значення для захисту корпоративної кофіденціальної інформації та для кібербезпеки загалом.

Традиційні методи виявлення фішингових листів та боротьби з фішингом в більшості базуються на використанні фіксованих правил, сигнатурному аналізі, використанню чорних та білих списків або класичних алгоритмах машинного навчання. Проте ефективність таких підходів часто є обмеженою, через їхню неспроможність швидко адаптуватись до стрімкої еволюції методів соціальної інженерії та змін стилів написання фішингових листів. У цьому випадку трансформені моделі становлять дуже перспективну альтернативу, оскільки моделі враховують складні контекстні залежності в тексті незалежно від порядку слів, або довжини самого тексту.

Архітектури на основі трансформерів, такі як BERT, GPT, DeepSeek та їх модифікації, показують широкі можливості у створенні багатих змістовно представлень тексту. На етапі попереднього навчання (pre-training) моделі здобувають необхідні знання про структуру мови, а завдяки подальшому спеціалізованому навчанню (fine-tuning) моделі можуть бути спеціально адаптовані до специфічних задач, наприклад класифікація електронних листів за ознаками фішингу.

Суттєвою перевагою трансформерних моделей є їх гнучкість до змін у форматуванні та стилістиці тексту. Завдяки здатності трансформерних моделей визначати як поверхневі прості залежності, так і глибокі мовні шаблони, такі моделі потенційно здатні виявляти спроби фішингу навіть з навмисним маскуванням зловмисних дій під дозволені типи обміну повідомлень. Це може сприяти на зменшення кількості помилок першого роду, так і на зменшення кількості помилок другого роду, що є надзвичайно важливим для практичного використання у корпоративній середі.

Крім того, трансформені моделі зазвичай підтримують механізми інтерпретації результатів, а саме аналіз вагів уваги моделі. Це дозволяє оцінити яка частина тексту вплинула найбільше на рішення моделі.

Окремої уваги заслуговує ще можливість трансформеної моделі до адаптації. У середовищі інформаційних загроз, що постійно змінюється, є критично важливим мати можливість проводити швидкі зміни до методів виявлення фішингу. Трансформенні моделі мають можливість бути донавченими на нових прикладах фішингу на льоту, без необхідності перенавчати всю модель. Це робить трансформені моделі потенційно корисним як інструмент для виявлення фішингових листів, що здатен досить швидко адаптуватись до нових загроз.

У підсумку цього розділу, після огляду загальної архітектури трансформених моделей, можу зробити висновок, що трансформені моделі є потенційною альтернативою стандартним підходам до виявлення фішингових електронних листів. Архітектурні особливості трансформерних моделей, а саме їх здатність до швидкої адаптації та глибокого розуміння тексту повідомлення, роблять їх перспективними в якості інструменту виявлення фішингових атак, а саме атак з використанням електронних листів.

2.2 Огляд популярних трансформерних моделей

Після розгляду основних принципів функціонування трансформених моделей, необхідно перейти до аналізу конкретних прикладів трансформених моделей, що активно використовуються для вирішення великого спектру задач, пов'язаних з обробленням природної мови. За останні роки трансформені моделі та їх архітектури значно розвинулись з їх початкових версій, що призвело до появи досить широкого спектру моделей, що були оптимізовані під великий спектр завдань – починаючи з дуже популярної сьогодні генерації тексту та

машинного перекладу, закінчуючи класифікацією та пошуком інформації в мережі.

Велике різноманіття трансформених моделей з'явилося не лише через відмінності у різних типах архітектур таких як encoder-only, decoder-only або encoder-decoder (тільки-кодер, тільки-декодер, кодер-декодер), а й у відмінностях у методах переднавчання, масштабах навчальних вибірок (корпусів), різниці у використаних токенизаторах, підходах до fine-tuning (донавчання) та промпт-інженерії (prompt-engineering). Через це, показники успішності у вирішенні прикладних задач кожної трансформеної моделі відрізняються одна від одної.

У цьому підрозділі буде розглянуто три популярні трансформені моделі які є досить новими та актуальними представниками різних архітектур трансформених моделей. Зокрема, “RoBERTa” як представник класичних тільки-енкодер моделей орієнтованих на глибоке розуміння природної мови, “Phi-2” як представник тільки-декодер архітектури що адаптована до великого спектру можливих задач, “Flan-T5” що є представником енкодер-декодер архітектури.

Аналіз кожної з вище-зазначених моделей дозволить оцінити їх доцільність у задачах класифікації тексту, а саме виявлення фішингових електронних листів на основі тільки основного тексту у тілі листа.

2.2.1 RoBERTa

RoBERTa – Robustly optimized BERT approach (з англ. – Надійно оптимізований підхід на основі BERT). У 2019 році, було виконано дослідження дослідниками Facebook AI щодо можливості покращення початкового тренування моделі BERT, моделі що на той час вважалась проривною у сфері штучного інтелекту. Основна ідея описана у роботі “RoBERTa: A Robustly Optimized BERT Pretraining Approach” полягає не у зміні самієї архітектури моделі, а у покращенні самого процесу переднавчання, що як показало дослідження має великий вплив на успішність моделі. Для подальшого розуміння покращень що

були представлені у роботі, буде доцільно повернутись до роботи “ BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding ”.

BERT стала однією з ключових моделей, що започаткувала нову еру в обробці природної мови. Ця модель була представлена у 2018 році дослідниками з Google AI, та стала основою для багатьох сучасних підходів у задачах пошуку інформації, класифікації тексту та багатьох інших.[18][20]

Головним нововведенням нової трансформеної моделі було введення двонаправленого контексту – можливість аналізу вхідної послідовності водночас з початку до кінця, так і з кінця до початку. Це досягається завдяки використанню архітектури трансформера, що була запозичена з роботи “Attention is All You Need” ті адаптоване під задачу розуміння природної мови.

Архітектурно, BERT є представником тільки-енкодер архітектури (encoder-only), що складається з багатошарового стеку блоків самоуваги (Self-Attention). У роботі було представлено дві основні версії моделі BERT :

BERT_{BASE} – складається з 12 шарів (Transformer blocks/L), прихований шар розмірністю 768 (H), 12 голів уваги (Self-Attention heads/A), з загальною кількістю параметрів – 110 млн. параметрів.

BERT_{LARGE} – складається з 24 шарів, прихований шар розмірністю 1024, 16 голів уваги, з загальною кількістю параметрів – 340 млн. параметрів.

Як вхідні дані модель отримує токенізований текст у вигляді послідовностей, що проходять три етапи передобробки:

Token embedding – представлення кожного вхідного токена у вигляді вектора з словника;

Segment embedding – представлення для визначення належності кожного токена до речення А або речення Б;

Position embedding – представлення позиції кожного токена;

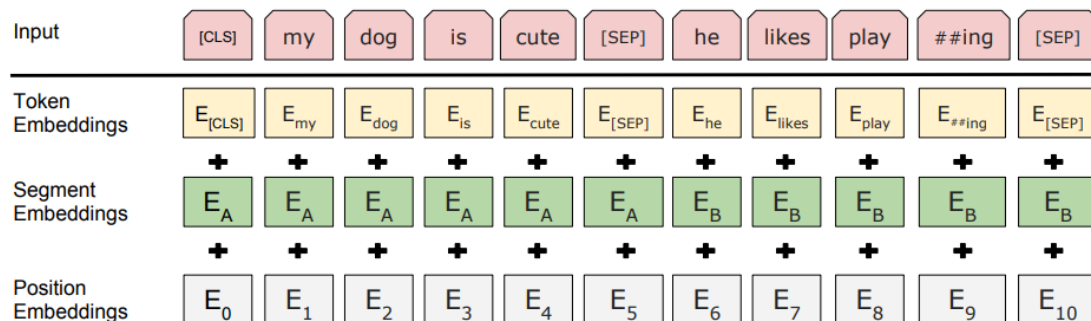


Рисунок 2.4 – Формування представлення

Кожне представлення формується шляхом поелементного додавання словникового представлення (Token embedding), сегментного представлення (Segment embedding) та позиційного представлення (Position embedding).

Трансформена модель BERT замість звичних нам слів, оперує токенами отриманими за допомогою токенизатора “WordPiece”[21] з розміром словника в 30 000 токенів. Це дозволяє моделі обробляти як відомі та часто використовувані слова, так і більш рідкісні або зовсім нові слова шляхом розбиття слова на менші суб-частини.

Процес попереднього навчання моделі BERT, відрізняється від стандартних на той час методів навчання моделей. BERT використовує двонаправлене попереднє навчання на великих корпусах тексту із подальшим донавчанням для конкретних задач.

Попереднє навчання складалось з двох процесів:

Masked Language Modeling (з англ. – Масковане Мовне Моделювання, MLM). Суть цього процесу полягає в тому що модель випадковим чином маскує 15% токенів у вхідній послідовності з яких:

- у 80% випадків маскований токен замінюється на спеціальний токен “[MASK]”

- у 10% випадків маскований токен замінюється на випадковий токен зі словника

- у 10% випадків маскований токен не змінюється

Завданням моделі під час цього процесу є передбачення цих маскованих токенів, маючи доступ до повного контексту вхідної послідовності, звертаючи увагу як на ліву частину від маскованого токена, так і на праву частину, що забезпечує краще навчання та подальше розуміння.

Next Sentence Prediction (з англ. – Прогнозування наступного речення, NSP). Суть цього процесу полягає в тому щоб навчити модель залежностям між окремими реченнями. Трансформена модель отримує 2 речення. Речення В (друге речення) у 50% випадків є наступним реченням до речення А (перше речення), у інших 50% відсотках речення В є випадковим реченням та не є наступним до речення А.

Завдання моделі під час цього процесу полягає в тому щоб визначити, чи є речення В логічним продовженням речення А.

BERT був навчений на двох навчальних корпусах – BookCorpus (~800 млн. слів. Набір художніх творів у відкритому доступі) та English Wikipedia (2.5 млрд слів. Статті з англomовної вікіпедії)

Навчання моделі BERT тривало 1 млн. кроків, з розміром батчу – 256 та довжиною послідовності до 512 токенів, з використанням оптимізатора ADAM.

Модель BERT показала свою ефективність у розумінні природної мови, пошуку відповідей на запитання, моделювання логіки та контексту ситуації. Ця робота доказала що попередньо-навчена трансформена модель може бути адаптована для розв’язання широкого спектру задач з мінімальним донавчанням.

Це стало початковим кроком для створення подальших модифікацій трансформеної моделі BERT, а саме RoBERTa.

Повертаючись до обраної моделі RoBERTa, тепер доцільно розглянути що саме було змінено, порівнюючи з оригінальною моделлю BERT.

Перша важлива зміна, це повне вилучення з процесу навчання процесу тренування за допомогою задачі прогнозування наступного речення (NSP). У роботі було вказано, що емпіричні дослідження показали, що цей метод тренування не тільки не покращує результати але й може знижувати загальну оцінку якості розуміння природної мови. Замість цього, головна увага була приділена до процесу тренування за допомогою динамічного маскування токенів, що забезпечує більшу варіативність вхідних прикладів, і показало схожі або кращі результати за статичне маскування токенів.[19]

Друга важлива зміна, це збільшення корпусу навчальних текстів, а саме збільшення майже в 10 разів від тих що був використаний для навчання моделі BERT. Обсяг нового корпусу навчальних даних сягає 160 ГБ і включає в себе оригінальні навчальні дані моделі BERT – BookCorpus + English Wikipedia, а також CC-News (63 млн. новин, зібраних між 2016 та 2019 роками, та займають приблизно 76 ГБ), OpenWebText (контент з соціальної мережі Reddit що мала більше ніж 3 вподобайки. Займає приблизно 38 ГБ), STORIES (31 ГБ контенту схожих на історії). Збільшення об'єму навчальних даних мало на меті змогу покращити узагальнення інформації про природну мову.

Третя зміна, це збільшення розміру міні-батчів до кількох тисяч прикладів, а сам процес тренування займає 500 тис. кроків, що дозволило покращити перплексію для завдання мовного моделювання та загальне підвищення показнику точності.

Також було використано новий токенизатор – байтовий BPE-токенизатор зі словником в 50 тис. токенів, що покращує роботу з нетиповими словами.

Ці зміни зробили трансформерну модель RoBERTa однією з найефективніших моделей для задач оброблення природної мови, а саме для задач класифікації тексту. Модель стабільно перевершує свою минулу версію BERT на численних тестах таких як GLUE (General Language Understanding Evaluation), SQuAD (Stanford Question Answering Dataset), RACE (ReAding Comprehension dataset from Examinations), що підтверджує доцільність використання цієї моделі у практичних задачах, а саме у якості класифікатора фішингових електронних листів.

2.2.2 Qwen2 0.5B Instruct

Модель Qwen2 0.5B Instruct є частиною лінійки великих мовних моделей, розроблених організацією Alibaba Cloud у рамках проєкту “Qwen2”. Ця серія моделей була представлена у 2024 році як наступна версія попередньої серії Qwen1.5. Основною метою створення Qwen2 було покращення продуктивності моделей у задачах розуміння та подальшої генерації мови, багатомовності, програмування та логічного мислення.

Модель Qwen 0.5B Instruct є найменшою у серії, та містить 500 млн. параметрів і призначена для використання виконання інструкцій користувача з обмеженими обчислювальними потужностями. Вона була спеціально налаштована для виконання завдань що описані у вигляді інструкцій, що робить цю модель придатною для деяких прикладних задач.

Модель Qwen 0.5B Instruct це представник тільки-декодер архітектури з довжиною контексту в 2048 токенів. Значним плюсом цієї моделі є дуже широка підтримка різних мов, а саме англійська, китайська, іспанська, французька, німецька, арабська, російська, корейська, японська, тайська, в'єтнамська.

Мовна модель була навчена на великому корпусі даних, що були зібрані з відкритих джерел. Хоча конкретний список джерел не надавався, було зазначено що для навчання моделі були використані джерела з відкритого коду, технічна

документація, наукові статті, записані діалоги, та просто тексти що охоплюють вище зазначені мови. Довжина конексту моделі в 2048 токенів, була обрана як компроміс між якістю відповіді та швидкістю обробки.

Після базового навчання, модель була донавчена за допомогою інструкційного донавчання, що мала на меті перетворити генеративну мовну модель на помічника що може слідувати інструкціям. Інструкції охоплювали такі завдання як переформулювання тексту, логічне мислення та пояснення, переклад, математичні розрахунки, генерація програмного коду, відповіді на запитання.

Попри те, що модель не була навчена для задач класифікації тексту, вона потенційно може бути використана завдяки своїм загальним можливостям до розуміння природної мови.

2.2.3 Flan-T5

Модель T5 (“Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer”[26]) була представлена Google Research у 2019 році і стала важливим кроком у розвитку великих мовних моделей, а саме у напрямку уніфікації підходів до завдань обробки природної мови. Подальший розвиток трансформерної моделі T5 привів до створення Flan-T5[22], що демонструє покращені можливості завдяки інструкційному донавчанню.

Для подальшого розуміння покращень що були представлені у роботі, буде доцільно повернутись до роботи “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer”. [26]

Трансформена модель T5 базується на стандартній трансформеній архітектурі типу енодер-декодер, що було описано в роботі “Attention is All You Need”. [15]

Ключовою особливістю мовної моделі T5 є уніфікований підхід до обробки інформації та виконання завдань пов'язані з природною мовою. Замість, розробки

моделей для кожного окремого завдання, модель T5 перетворює всі завдання у формат “з тексту в текст”.

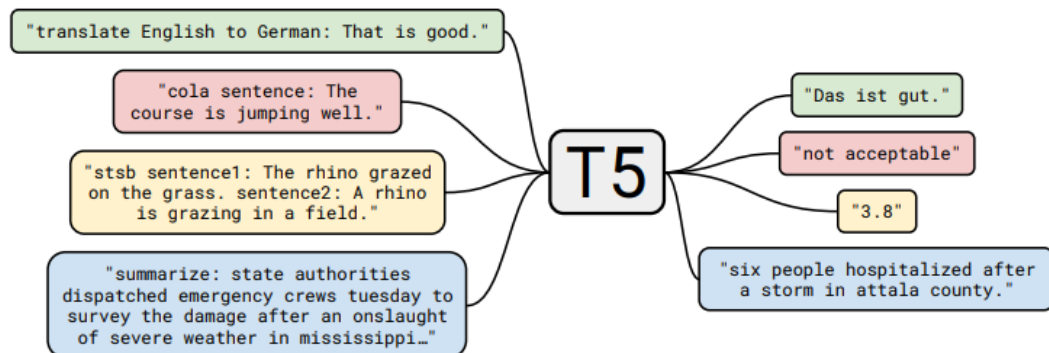


Рисунок 2.5 – Візуалізація роботи моделі Flan-T5

Як приклади стандартних завдань можна зазначити переклад з однієї мови в іншу, узагальнення вхідного тексту, відповідь на питання або класифікація тексту.

Модель T5 була реалізована у багатьох варіантах, що відрізняються розміром та кількістю параметрів.

Таблиця 2.1 – Варіанти моделі Flan-T5

Назва моделі	Загальна к-ть параметрів	К-ть шарів	Розмірність векторів	К-ть голів уваги
T5-Small	~60 млн.	6	512	8
T5-Base	~220 млн.	12	768	12
T5-Large	~770 млн.	24	1024	16
T5-3B	~3 млрд.	24	1024	32
T5-11B	~11 млрд.	24	1024	128

Навчання моделі T5 було виконано на колосальному великому корпусі C4 (Colossal Clean Crawled Corpus), датасет автоматично зібраний з сторінок

інтернету як навчальний для великих мовних моделей. З датасету були видалені усі частини з великою кількістю неанглійського тексту, частини що містять вульгарну або небажану лексику, автоматичні повідомлення та частини де є слово “JavaScript”.

Хоча модель T5 показала хороші результати, її продуктивність у задачах та сценаріях з досить обмеженою кількістю прикладів була досить обмеженою. Для покращення результативності модулі в цих умовах була розроблена модель Flan-T5 (Scaling Instruction-Finetuned Language Models), що є результатом спеціального інструкційного донавчання минулої мовної моделі T5 на ширшому наборі завдань.

Трансформена модель Flan-T5 була донавчена на групі з приблизно 1800 завдань, сформульованих як текстові інструкції, що використовуються у різноманітних областях обробки природної мови включаючи :

- Відповіді на запитання : модель вчиться відповідати на запитання на основі наданого контексту або без його використання, що в себе може включати як фактичні запитання так і логічні.

- Переклад: модель вчиться перекладати тексти між різними мовами, що дозволяє моделі працювати з різними мовами.

- Узагальнення тексту: модель вчиться генерувати стислий зміст для текстів, статей чи документів

- Класифікація тексту: модель вчиться назначати деякі категорії текстовим документам, зокрема чи є документ позитивний чи негативний, тематику документа, тощо

- Генерація тексту: модель вчиться продовжувати тексти, генерувати історії, тощо.

- Вилучення інформації: модель вчиться вилучати структуровані дані з тексту, наприклад вилучати дати або імена.

Так само як і минула модель T5, Flan-T5 також має декілька версій з різною кількістю параметрів, що спрощує використання цієї мовної моделі на не дуже потужних системах.

Завдяки інструкційному донавчанню, трансформена модель Flan-T5 демонструє хороші результати з обробкою інформації що раніше не були доступні моделі, або були доступні проте в малій кількості. Це робить цю модель особливо корисною для задач де кількість прикладів, що були помічені обмежено або їх зовсім немає. Серед таких задач класифікацію тексту за заданим критерієм – виявлення фішингового листа.

Висновки до розділу 2

В другому розділі було проведено аналіз архітектури та принципів роботи трансформерних моделей, а також здійснено огляд сучасних компактних мовних моделей, що реалізують ці підходи. На основі вивченого матеріалу, можна зробити висновки, що архітектурні рішення трансформерних моделей мають низку переваг, що роблять їх придатними для задач класифікації текстових повідомлень, а саме виявлення фішингових листів.

Особливу увагу було приділено моделі RoBERTa, що демонструє високу ефективність порівняно з класичною моделлю BERT, через покращене донавчання. Також було розглянуто модель Qwen2-0.5B-Instruct, як компактний представник тільки-декодер архітектури. Модель Flan-T5 була проаналізована як приклад трансформерної моделі енкодер-декодер архітектури, здатної виконувати широкий спектр завдань завдяки інструкційному донавчанню.

На основі проведеного огляду можна стверджувати, що трансформерні моделі мають високу гнучкість у налаштуванні, здатні ефективно представляти складну текстову інформацію та здатні адаптуватись до змін у стилістиці та структурі фішингових повідомлень. Їх головна перевага над класичними методами полягає в здатності знаходити поверхневі та глибинні мовні зв'язки.

3 Експериментальне дослідження

3.1 Вибір датасету та критерії оцінки ефективності

У межах дослідження, присвяченого використанню трансформених моделей різних архітектур для виявлення фішингових електронних листів, було обрано датасет “Phishing Email Detection” розміщений на платформі “Kaggle” (платформа для публікування датасетів для відкритого користування) автором Субхадіпом Чакраборті.

Датасет містить в собі текстові частини електронних листів, які класифіковані за двома класами – фішингові(Phishing Email) та безпечні (Safe Email). Датасет містить ~18 600 електронних листів з яких ~11 350 безпечних листів та ~7250 фішингових або спам листів.

Також був обраний схожий за раніше обраним проте більший датасет “Phishing Email Dataset” використаний у роботі “ Novel Interpretable and Robust Web-based AI Platform for Phishing Email Detection”[25] що складається з ~82 500 електронних листів отриманих з різних джерел з яких ~43 000 це фішингові/спам електронні листи та ~40 000 звичайних електронних листів.

Для об’єктивної оцінки результатів роботи дотренованої трансформерної моделі з виявлення фішингових повідомлень буде використано низку метрик, зокрема F1-метрика, матриця помилок (з англ. “Confusion matrix”) та детальний звіт що надається бібліотекою “sklearn” мови “Python”.

F1-метрика була обрана як основний критерій ефективності моделі, оскільки вона дозволяє врахувати як точність моделі так і її повноту у контексті розподілу класів.

Оцінювання буде здійснюватись після завершення донавчання моделі на тестовій підмножині даних з датасетів обраних раніше. Результати прогнозування будуть порівняні з реальними мітками класів, після чого буде побудована матриця помилок для кращого розуміння результативності моделі

3.2 Експериментальне дослідження ефективності моделі RoBERTa

У межах дослідження було проведено донавчання попередньо натренованої трансформеної моделі RoBERTa, що була розглянута у розділі 2.2.1.

Для донавчання був використаний менший датасет “Phishing Email Detection”, що містив в собі текст класифікованих електронних листів та мітка класифікатор – 0 – безпечний лист, 1 – фішинговий/спам лист.

Тренування виконувалось за допомогою мови “Python” з використанням бібліотеки “transformers” від HuggingFace. Для завантаження та токенизації був використаний токенізатор “RobertaTokenizer”, а сама модель базувалась на “RobertaForSequenceClassification” (базова модель RoBERTa з доданою головою класифікатором) із двома вихідним класами. Модель була тренована двічі – один раз повністю, другий раз – була дотренована тільки голова класифікатор.

Початковий набір даних завантажено CSV файлу та поділено на пропорції 90/10 - навчальна та тестова частина відповідно. Токенізація проводилась з параметрами “padding=True” “truncation=True” що забезпечує вирівнювання довжин вхідних послідовностей.

Оптимізація здійснювалась за допомогою “AdamW” з параметром навчання “learning_rate=2e-5” (0.00002), а також з розкладом навчання “get_liner_schedule_with_warmup”. Для пришвидшення процесу донавчання була використана підтримка точності в 16-біт (fp16).

Модель тренувалась протягом двох епох з розміром пакету в 8. Кожні 500 кроків виконувалась перевірка точності та збереження чекпоинту донавчання на випадок необхідності термінової зупинки навчання без втрачання всього прогресу. Також під час тренування було відслідковано кількість використовуваної відеопам’яті.

Донавчання тривало приблизно 8 годин (Використана відеокарта – GTX-1650, 4 ГБ відеопам’яті). Після завершення донавчання було здійснено оцінку точності моделі на більшому датасеті що був зазначений раніше.

F1-метрика = 0,8808

Таблиця 3.1 – Таблиця метрик моделі RoBERTa при повному донавчанні

Клас	Точність	Повнота	F1-метрика	К-ть прикладів
0	0,98	0,78	0,87	6289
1	0,82	0,98	0,89	6293

Таблиця 3.2 – Таблиця помилок моделі RoBERTa при повному донавчанні

	Передбачений клас - 0	Передбачений клас - 1
Реальний клас - 0	4916	1373
Реальний клас - 1	112	6181

Отримані результати вказують на загальну високу якість класифікації моделлю RoBERTa. Особливо ефективною модель себе показала у виявленні листів з класом 1 (фішингові листи) з повнотою в 98%, що означає що майже всі фішингові приклади що були проаналізовані моделлю були виявлені. Це є особливо важливо для використання цієї моделі у реальному середовищі де є критично важливим відфільтровувати шкідливі повідомлення.

Проте модель дещо гірше справилась з детектуванням класу 0 (безпечні повідомлення), тобто деяка помітна кількість безпечних повідомлень була помилково класифікована як фішингова. Це може створити проблеми з надмірною фільтрацією листів, але це вважається допустимим в умовах де безпека є пріоритетною.

В другий раз відбувалось тренування тільки голови класифікатора, а сама модель залишалась незмінною.

F1-метрика = 0,8513

Таблиця 3.3 – Таблиця метрик моделі RoBERTa при донавчанні тільки голови класифікатора

Клас	Точність	Повнота	F1-метрика	К-ть прикладів
0	0,88	0,81	0,84	6162
1	0,83	0,89	0,86	6420

Таблиця 3.4 – Таблиця помилок моделі RoBERTa при донавчанні тільки голови класифікатора

	Передбачений клас - 0	Передбачений клас - 1
Реальний клас - 0	4977	1185
Реальний клас - 1	682	5738

Тренування цієї моделі зайняло 95 хвилин часу, та з постійним навантаженням на відеокарту в 1 ГБ відеопам'яті.

Середня точність моделі на тестовій вибірці склала 85%, що свідчить про загальну здатність моделі коректно класифікувати повідомлення. Середня F1-метрика вийшла 0,8513, що є достатньо високим показником для задач бінарної класифікації.

З матриці помилок видно, що модель схильна помилково класифікувати безпечні повідомлення як фішингові (1185 випадків), ніж фішингові повідомлення як безпечні (682 випадки).

Загалом, модель що тренувала тільки голову класифікатор показала себе трохи гірше за модель, що була дотренована повністю. Модель, що була тренувана повністю показує кращі характеристики, а саме краща F1-метрика, більша точність моделі, та значне збільшення повноти виявлення класу 1 (фішингове повідомлення).

На основі отриманих результатів можна зробити практичний висновок. Хоч повне донавчання трансферної моделі RoBERTa дозволяє досягти кращої точності, зокрема значного зменшення помилок другого роду, воно вимагає значно більшу кількість обчислювальних ресурсів. В проведеному експерименті знадобилось приблизно в 8 разів більше часу та в 3 рази більше відеопам'яті, в порівнянні з навчанням лише голови класифікатора.

Базуючись на цьому висновку, вибір методу дотренування має базуватись на пріоритетах і обмеженнях конкретної ситуації та застосування :

- Для великих організацій, де безпека інформаційної середовища є найбільшим пріоритетом, перевагу слід надавати точнішій моделі, навіть за умови необхідності більших потужностей комп'ютерних систем.

- У системах з обмеженими ресурсами, або де кібербезпека не є головним пріоритетом, буде доцільніше використовувати модель з меншою необхідністю в великих обчислювальних потужностях, хоч точність моделі буде нижчою. У такому випадку ця система більше підійде як модуль антивірусного програмного забезпечення для персональних комп'ютерів.

3.3 Експериментальне дослідження ефективності моделі Qwen2-0.5B Instruct

У межах дослідження було проведено донавчання попередньо натренованої трансформованої моделі Qwen2-0.5B Instruct , що була розглянута у розділі 2.2.2.

Для донавчання був використаний той самий датасет що і для донавчання моделі RoBERTa.

Тренування виконувалось за допомогою мови “Python” з використанням бібліотеки “transformers” від HuggingFace. Для завантаження та токенизації був використаний токенизатор “ AutoTokenizer”, а сама модель базувалась на “ AutoModelForSequenceClassification” (базова модель Qwen2 0.5B Instruct з

доданою головою класифікатором) із двома вихіднимим класами. Модель була тренована єдиний раз через недостатньо великі обчислювальні потужності, а саме була дотренована тільки голова класифікатор.

Початковий набір даних завантажено CSV файлу та поділено на пропорції 80/20 - навчальна та тестова частина відповідно. Токенізація проводилась з параметрами “padding=True” “truncation=True” що забезпечує вирівнювання довжин вхідних послідовностей.

Для пришвидшення процесу донавчання була використана підтримка точності в 16-біт (fp16).

Модель тренувалась протягом двох епох з розміром пакету в 1 для зменшення необхідних обчислювальних потужностей. Кожні 500 кроків збереження чекпоинту донавчання на випадок необхідності термінової зупинки навчання без втрачання всього прогресу. Також під час тренування було відслідковано кількість використовуваної відеопам’яті.

Донавчання тривало приблизно 28 годин та з постійним навантаженням в 50/50 Вт та 3,8/4 Гб відеопам'яті. Після завершення донавчання було здійснено оцінку точності моделі на тому самому датасеті що був використаний раніше для дотренування, для зменшення часу оцінки моделі (зменшення з 22.2 годин до 3.5 годин).

F1-метрика = 0,5457

Таблиця 3.5 – Таблиця метрик моделі Qwen2-0.5B Instruct при донавчанні тільки голови класифікатора

Клас	Точність	Повнота	F1-метрика	К-ть прикладів
0	0,8347	0,3277	0,4706	2234
1	0,4730	0,9029	0,6208	1493

Таблиця 3.6 – Таблиця помилок моделі Qwen2-0.5B Instruct при донавчанні тільки голови класифікатора

	Передбачений клас - 0	Передбачений клас - 1
Реальний клас - 0	732	1502
Реальний клас - 1	145	1348

Загальна точність класифікації склала 56%, що є невисоким показником. Середнє значення F1-метрики, було розраховане як середнє значення для кожного класу і становить 54,57%.

Отримані результати вказують на те що модель має низький рівень точності та явний дисбаланс до класу 1, що підтверджується даними отриманими з Confusion matrix, а саме видно що більшість прикладів було класифіковано як фішингові.

Загалом, за отриманими результатами, можна стверджувати, що ступінь придатності мовної моделі Qwen2-0.5B Instruct з доданою головою класифікатором для задачі бінарної класифікації є ледве задовільною та потребує подальших змін. Через недостатній рівень загальної точності та збалансованості класифікації, модель є непродуктивною для використання у практичних завданнях виявлення фішингових листів.

3.4 Експериментальне дослідження ефективності моделі Flan-T5

У межах дослідження було проведено донавчання попередньо натренованої трансформеної моделі Flan-T5 Small[23], що була розглянута у розділі 2.2.3.

Тренування виконувалось за допомогою мови “Python” з використанням раніше зазначених бібліотек.

Модель була донавчена на двох завданнях – виявлення фішингових листів (fine-tuned-flan-t5-small-phishing-right) на базі самих листів та виявлення фішингових посилань (fine-tuned-flan-t5-small-phishing-right-links).

Для донавчання моделі для виявлення фішингових листів був використаний той самий датасет що і для донавчання моделі RoBERTa. Для донавчання моделі для виявлення фішингових посилань було використано новий датасет створений з датасету “Phishing Email Detection” та “Phishing Email Dataset” шляхом вилучення посилань з відповідних фішингових листів.

Обидві моделі були донавчені шляхом надавання специфічного запиту, прикладу та правильної відповіді.

Для моделі що була донавчена на фішингових листах запит мав вигляд:

“Is this text phishing, yes or no? Text: [приклад]”

Для моделі що була донавчена на фішингових посиланнях:

“Is this link phishing, yes or no? Link: [приклад]”

Для більш стабільного навчання був використаний оптимізатор AdamW з параметром коефіцієнт навчання, що дорівнював 0.00005.

Початковий набір даних завантажено з CSV файлу та поділено на пропорції 80/20 - навчальна та тестова частина відповідно. Токенізація проводилась з параметрами “padding=True” “truncation=True” що забезпечує вирівнювання довжин вхідних послідовностей.

Обидві моделі тренувались протягом п’яти епох. Кожні 1000 кроків збереження чекпоинту донавчання на випадок необхідності термінової зупинки навчання без втрачання всього прогресу. Також під час тренування було відслідковано кількість використовуваної відеопам’яті.

Донавчання кожної моделі тривало приблизно 3 години та з постійним навантаженням в 50/50 Вт та 3,2/4 Гб відеопам'яті. Після завершення донавчання

було здійснено оцінку точності моделі на більшому датасеті “Phishing Email Dataset”.

F1-метрика = 0,8602

Таблиця 3.7 – Таблиця метрик моделі Flan-T5-Small при донавчанні на тексті фішингових листів

Клас	Точність	Повнота	F1-метрика	К-ть прикладів
0	0,9639	0,7478	0,8422	12376
1	0,7994	0,9729	0,8777	12788

Таблиця 3.8 – Таблиця помилок моделі Flan-T5-Small при донавчанні на тексті фішингових листів

	Передбачений клас - 0	Передбачений клас - 1
Реальний клас - 0	9255	3121
Реальний клас - 1	347	12441

Загальна точність класифікації склала 88%, що є високим показником.

Отримані результати вказують на успішне донавчання моделі для задачі класифікації фішингових листів, з невеликою перевагою до класу 1 (фішингове повідомлення).

F1-метрика = 0,9439

Таблиця 3.9 – Таблиця метрик моделі Flan-T5-Small при донавчанні на фішингових посиленнях

Клас	Точність	Повнота	F1-метрика	К-ть прикладів
0	0,9489	0,9757	0,9621	1809
1	0,9319	0,8637	0,8965	697

Таблиця 3.10 – Таблиця помилок моделі Flan-T5-Small при донавчанні на фішингових посиленнях

	Передбачений клас - 0	Передбачений клас - 1
Реальний клас - 0	1765	44
Реальний клас - 1	95	602

Загальна точність класифікації склала 94%, що є дуже високим показником.

Отримані результати вказують на успішне донавчання моделі для задачі класифікації фішингових листів з відсутністю переваги будь-якого класу.

Загалом, за отриманими результатами можна стверджувати, що ступінь придатності мовної моделі Flan-T5-Small для задачі бінарної класифікації є задовільною.

3.5 Експериментальне дослідження ефективності моделі GPT-4

Під час дослідження також було приділено увагу моделі GPT-4 через сервіс від розробників OpenAI - “ChatGPT”. Коротке дослідження відбулось у якості “zero-shot” класифікації, тобто без донавчання та надання будь-якої додаткової інформації.

Був використаний датасет “Phishing Email Detection”.

F1-метрика = 0.5898

Таблиця 3.11 – Таблиця метрик моделі GPT-4 при “zero-shot” класифікації

Клас	Точність	Повнота	F1-метрика	К-ть прикладів
0	0.68	0.59	0.63	11322
1	0.48	0.58	0.52	7328

Таблиця 3.12 – Таблиця помилок моделі GPT-4 при “zero-shot” класифікації

	Передбачений клас – 0	Передбачений клас - 1
Реальний клас - 0	6682	4640
Реальний клас - 1	3095	4233

Загальна точність класифікації склала 58%, що є дуже низьким показником, що граничить з випадковим вибором.

Отримані результати вказують на те, що навіть надзвичайно велика модель така як GPT-4 (~ 1 трлн. параметрів за приблизними оцінками) , що була донавчена на дуже великій вибірці можливих завдань не змогла показати достатніх результатів для практичного використання у якості класифікатора фішингових листів.

Висновки до розділу 3

В цьому розділі було проведено експериментальне дослідження ефективності різних трансформерних моделей для задачі виявлення фішингових повідомлень. Було здійснено донавчання трьох моделей: RoBERTa, Qwen2-0.5B-Instruct, Flan-T5-Small, а також було проаналізовано GPT-4 у режимі zero-shot. Для забезпечення об’єктивності оцінювання були застосовані такі метрики – F1-метрика, точність, повнота та матриця помилок.

Найвищі результати були досягнуті моделлю Flan-T5-Small, що пройшла донавчання на фішингових посиланнях, вилучених з датасету. Вона продемонструвала F1-метрику в 0.94, що свідчить про високу якість моделі. Також високу ефективність показала модель RoBERTa, дотренована повністю на текстах – F1-метрика - ~ 0.88 .

Порівнюючи з вже зазначеними моделями, Qwen2-0.5B-Instruct продемонструвала значно гірші результати, F1-метрика - ~ 0.54 , що свідчить про її обмежену ефективність у якості класифікатора фішингових листів. GPT-4, незважаючи на свій колосальний розмір та сучасну архітектуру, при zero-shot підході вона не змогла продемонструвати хороших результатів, F1-метрика - ~ 0.59 , що вказує на важливість донавчання незважаючи на розмір, комплексність та якість моделі.

Таким чином, результати проведеного експериментального дослідження підтвердили доцільність використання трансформерних моделей, для задачі виявлення фішингових листів.

Висновки

У межах виконаної дипломної роботи було проведено дослідження ефективності трансформерних моделей різних архітектур для виявлення фішингових електронних листів. Основною метою проведеного дослідження було визначення можливостей сучасних мовних трансформерних моделей для задач бінарної класифікації текстів електронної пошти.

Для досягнення цієї мети було визначено декілька трансформерних моделей, що є представниками різних підходів до обробки природної мови.

А саме RoBERTa - як представник тільки-енкодер архітектури, Qwen2-0.5B-Instruct – як представник тільки-декодер архітектури, Flan-T5-Small – як представник енкодер-декодер архітектури, та GPT-4 – як надзвичайно великий представник тільки-декодер архітектури та загалом дуже популярна модель для великої кількості задач. Було реалізовано декілька сценаріїв донавчання – як повного донавчання усього тіла моделі, так і лише голови класифікатора. Також було використано два датасети – менший датасет на 18 тисяч листів та більший датасет на 82 тисячі листів що дозволило оцінити здатність моделей до узагальнення.

Найкращі результати було отримано від повністю дотренованої моделі RoBERTa, що досягла F1-метрики в 0,8808 з повнотою 98% для 1-го класу (фішингові листи). Це означає що модель дуже ефективно виявляє фішингові повідомлення, що особливо важливо в умовах реального використання, де пропуск фішингового листа може мати критичні наслідки. Проте було виявлено, що точність для класу 0 (безпечні листи) нижча, що свідчить про схильність моделі до надмірного блокування легітимних повідомлень.

У випадку часткового донавчання, а саме лише голови класифікатора та сама модель RoBERTa досягла трохи гірших результатів, F1-метрика - 0,8513, з помітним зниженням повноти для обох класів. Однак, значною перевагою цього підходу є менше споживання обчислювальних ресурсів – приблизно у 8 разів

швидше навчання, та в 3 рази менше зайнятої відеопам'яті, що робить його доцільним для обмежених середовищ, наприклад на клієнтських пристроях або в легких антивірусних системах.

Модель Qwen2-0.5B-Instruct, була дотренована лише частково через брак обчислювальних потужностей, продемонструвала низький рівень точності в 56% та F1-метрику в 0,5457. Аналіз показав що модель має сильний дисбаланс в сторону фішингового класу, з дуже низькою повнотою для класу безпечних листів. Такі результати демонструють, що ця модель без глибокої адаптації не є придатною для використання в задачах класифікації тексту.

Проте мовна модель Flan-T5-Small, завдяки гнучкості архітектури типу енкодер-декодер та додатковому донавчання у вигляді інструкцій, показала високу ефективність. Модель для виявлення фішингових листів досягла результату F1-метрики 0,8602 , а модель для виявлення фішингових посилань показала F1-метрику в 0,9439 , що стало найвищим результатом серед усіх проведених досліджень.

Остання модель, що була навчена на виявленні фішингових посилань, виявилась найбільш збалансованою, з високим значенням як точності, так і повноти для обох класів. Це свідчить про доцільність використання спеціалізованих моделей для обробки певних частин листа, наприклад посилань що є найчастішими носіями загроз.

Модель GPT-4, попри свою потужність та популярність, показала незадовільні результати у форматі zero-shot класифікації (F1-метрика 0.5898). Це підтверджує, що навіть великі мовні без донавчання не можуть забезпечити достатній рівень точності у специфічних прикладних задачах, таких як фільтрація фішингових листів.

З дипломної роботи можна винести загальні висновки а саме:

-Повне донавчання моделі забезпечує найвищу якість класифікації, особливо для задач виявлення небажаного тексту, проте вимагає значних обчислювальних ресурсів.

-Донавчання лише голови класифікатора – це компромісний варіант, що дозволяє отримати хороші показники при мінімальних витратах ресурсів, але зі зниженням точності.

-Архітектура Flan-T5 виявилась дуже ефективною для задач класифікації, в умовах наявності інструкцій та невеликих вхідних даних.

-Тільки-декодер моделі як GPT або Qwen2 без глибокого донавчання не придатні до задач виявлення фішингу, незважаючи на загальну потужність.

Таким чином, результати цієї дипломної роботи підтверджують, що використання трансформерних моделей є ефективним і дуже гнучким підходом до виявлення фішингових електронних листів, що дозволяє адаптувати необхідне рішення як до надзвичайно складних корпоративних сценаріїв, так і до більш побутових сценаріїв.

Перелік джерел посилань

1. Method of analysis for analyzing phishing messages – Режим доступу:
https://www.researchgate.net/publication/362122194_Method_of_analysis_for_analyzing_phishing_messages
2. Classification and methods of detection of phishing attacks – Режим доступу:
https://www.researchgate.net/publication/382153168_CLASSIFICATION_AND_METHODS_OF_DETECTION_OF_PHISHING_ATTACKS
3. APWG Trends Report Q2 2024 – Режим доступу:
https://docs.apwg.org/reports/apwg_trends_report_q2_2024.pdf
4. APWG Trends Reports Archive – Режим доступу: <https://apwg.org/trendsreports/>
5. Senate Commerce Committee Report – Режим доступу:
<https://www.commerce.senate.gov/services/files/24d3c229-4f2f-405d-b8db-a3a67f183883>
6. DFS NY Twitter Report – Режим доступу: https://www.dfs.ny.gov/Twitter_Report
7. Cybersecurity Journal Article (стр. 7) – Режим доступу:
<https://csecurity.kubg.edu.ua/index.php/journal/article/view/591/464>
8. APWG Trends Report Q4 2023 – Режим доступу:
https://docs.apwg.org/reports/apwg_trends_report_q4_2023.pdf?_gl=1*_1flqlct*_ga*MTcxODQ4NjI2MS4xNzQ2NzA0MjY5*_ga_55RF0RHXSr*czE3NDkzNzcxNTYkbzQkZzEkdDE3NDkzNzcxNzYkajQwJGwwJGgw
9. IBM – Encoder-Decoder Model – Режим доступу:
<https://www.ibm.com/think/topics/encoder-decoder-model>
10. IBM – Attention Mechanism – Режим доступу:
<https://www.ibm.com/think/topics/attention-mechanism>

11. IBM – Self-Attention – Режим доступу: <https://www.ibm.com/think/topics/self-attention>
12. IBM – Recurrent Neural Networks – Режим доступу: <https://www.ibm.com/think/topics/recurrent-neural-networks>
13. Positional Encoding Explained – Режим доступу: <https://medium.com/thedeephub/positional-encoding-explained-a-deep-dive-into-transformer-pe-65cfe8cfe10>
14. IBM – Transformer Model – Режим доступу: <https://www.ibm.com/think/topics/transformer-model>
15. Vaswani et al. "Attention is All You Need" – Режим доступу: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf
16. Understanding Encoder and Decoder – Режим доступу: <https://magazine.sebastianraschka.com/p/understanding-encoder-and-decoder>
17. IBM – Tokens in WatsonX – Режим доступу: <https://www.ibm.com/docs/en/watsonx/saas?topic=solutions-tokens>
18. BERT (модель мови) – Режим доступу: [https://uk.wikipedia.org/wiki/BERT_\(%D0%BC%D0%BE%D0%B4%D0%B5%D0%B%D1%8C_%D0%BC%D0%BE%D0%B2%D0%B8\)](https://uk.wikipedia.org/wiki/BERT_(%D0%BC%D0%BE%D0%B4%D0%B5%D0%B%D1%8C_%D0%BC%D0%BE%D0%B2%D0%B8))
19. Arxiv: "RoBERTa: A Robustly Optimized BERT Pretraining Approach" – Режим доступу: <https://arxiv.org/pdf/1907.11692>
20. Arxiv: "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" – Режим доступу: <https://arxiv.org/pdf/1810.04805>

21. HuggingFace LLM Course – Режим доступу: <https://huggingface.co/learn/llm-course/en/chapter6/6>
22. Arxiv: "OPT: Open Pre-trained Transformer" – Режим доступу: <https://arxiv.org/pdf/2210.11416>
23. HuggingFace – FLAN-T5-Small – Режим доступу: <https://huggingface.co/google/flan-t5-small>
24. **Diogenes, Y., & Ozkaya, E.** *Cybersecurity – Attack and Defense Strategies: Infrastructure security with Red Team and Blue Team tactics.* - Режим доступу: <https://kupichitay.com.ua/product/cybersecurity-attack-and-defense-strategies-infrastructure-security-with-red-team-and-blue-team-tactics-yuri-diogenes-erdal-ozkaya/>
25. Novel Interpretable and Robust Web-based AI Platform for Phishing Email Detection. - Режим доступу: <https://arxiv.org/abs/2405.11619>
26. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. - Режим доступу: <https://arxiv.org/pdf/1910.10683>