МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«До захисту допущено»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)
18 грудня 2023 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня магістр

зі спеціальності 122 - Комп'ютерних наук,
освітньо-професійної програми «Інформатика»
на тему: «Інформаційні технології розпізнавання підозрілої активності в мережі»
здобувача групи ІН.м-22 Кириченка Володимира Олексійовича

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

Володимир
_____ КИРИЧЕНКО
(підпис)

Керівник,
к.т.н., доцент
доцент кафедри комп'ютерних наук В'ячеслав МОСКАЛЕНКО _____
(підпис)

Суми – 2023

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«Затверджую»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ
на здобуття освітнього ступеня магістр

зі спеціальності 122 - Комп'ютерних наук, освітньо-наукової програми «Інформатика»
здобувача групи ІН.м-22 Кириченка Володимира Олексійовича

1. Тема роботи: «Інформаційні технології розпізнавання підозрілої активності в мережі» затверджую наказом по СумДУ від «06» грудня 2023 р. № 1412-VI
2. Термін здачі здобувачем кваліфікаційної роботи до 18 грудня 2023 року
3. Вхідні дані до кваліфікаційної роботи _____
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити) *1) Аналіз проблеми предметної області, постановка й формування завдань дослідження. 2) Огляд технологій, що використовуються налаштування мовної моделі. 3) Розробка інтелектуальної системи для розпізнавання підозрілої активності в мережі» 4) Аналіз результатів.*
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)_____
6. Консультанти до проекту (роботи), із значенням розділів проекту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «____» _____ 20 ___ р.

Завдання прийняв до виконання _____ Керівник _____
(підпис) (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	*Аналіз проблеми предметної області, постановка й формування завдань дослідження*	08.12.2023	
2	*Огляд технологій, для створення мовних моделей*	09.12.2023	
3	*Розробка інтелектуальної системи виявлення підозрілої активності в мережі*	10.12.2023	
4	*Аналіз отриманих результатів*	15.12.2023	
5	*Оформлення пояснювальної записки до кваліфікаційної роботи*	16.12.2023	

Здобувач вищої освіти _____ Керівник _____
(підпис) (підпис)

АНОТАЦІЯ

Записка: 52 стор., 32 рис., 1 додаток, 25 джерел.

Обґрунтування актуальності теми роботи – Тема кваліфікаційної роботи присвячена застосуванні передових методів обробки природної мови (NLP) для виявлення підозрілих дій у текстових даних. Об'єктом дослідження є впровадження та ефективність мовної моделі, навченої виявляти потенційні загрози кібербезпеки в онлайн-комунікаціях.

Об’єкт дослідження — виявлення підозрілої активності.

Мета роботи — оцінці ефективності інформаційних технологій для розпізнавання підозрілої активності в мережі, зокрема, шляхом створення та тестування нової мовної моделі, яка оптимізована для ідентифікації потенційних загроз у текстових комунікаціях.

Методи дослідження — аналіз існуючих мовних моделей, експериментальне тестування розробленої моделі та оцінку її ефективності у виявленні підозрілої активності в мережі, використовуючи сучасні техніки обробки природної мови та алгоритми машинного навчання.

Результати — результати дослідження є створення власної мовної моделі котра виявляє підозрілу активність.

ІНФОРМАЦІЙНА СИСТЕМА, РОЗПІЗНАВАННЯ ПІДОЗРІЛОЇ АКТИВНОСТІ, NLP, LLM, TRANSFORMER.

ЗМІСТ

ВСТУП

Актуальність. У сучасному світі, де інформаційні технології розвиваються стрімко, забезпечення кібербезпеки стає все більш важливим.Зростання обсягів даних та їх значення для різних сфер вимагає розробки ефективних інструментів для їх захисту, особливо у контексті мережевих чатів.

Об'єкт дослідження. Кіберпростір і його захист, з особливим акцентом на мережеві чати.

Предмет дослідження. Методи та технології обробки природної мови (NLP) та їх застосування для виявлення підозрілої активності у текстових даних.

Гіпотеза. Сучасні мовні моделі можуть ефективно виявляти і аналізувати підозрілу мережеву активність, використовуючи алгоритми глибокого навчання та NLP для розуміння контексту та намірів у повідомленнях користувачів.

Наукова новизна. Практичний підхід до розробки мовної моделі, яка включає найсучасніші методи NLP та машинного навчання, враховуючи різні аспекти та виклики ідентифікації підозрілої активності.

Структура роботи. Дослідження включає аналіз проблеми, огляд існуючих технологій, розробку та експериментальне тестування інформаційної системи, а також аналіз отриманих результатів і висновки.

1 АНАЛІЗ ПРОБЛЕМИ І ПОСТАНОВКА ЗАДАЧІ

1.1 Сучасний стан та тенденції розвитку систем розпізнавання підозрілої мережевої активності

У сучасному цифровому світі, де інтернет став неодмінною частиною нашого повсякденного життя, безпека мережевих систем набуває вирішального значення. Зі зростанням залежності від онлайн-технологій збільшується і кількість потенційних кіберзагроз. Підозріла мережева активність — це широкий термін, який охоплює різноманітні дії або поведінку в мережі, які можуть вказувати на неавторизований доступ, шкідливі наміри або інші незаконні діяльності. Важливо розуміти та вміти ідентифікувати ці аспекти, адже вони є ключовими для забезпечення безпеки інформаційних систем та захисту даних.
Розпізнавання підозрілої мережевої активності — це складний процес, який вимагає глибокого розуміння як технічних, так і поведінкових патернів. Він включає аналіз мережевого трафіку, моніторинг поведінки користувачів, а також використання різних інструментів і технологій для виявлення аномалій. Далі будуть розглянуті ключові аспекти, які допомагають визначити підозрілу активність в мережі, щоб забезпечити ефективний захист від потенційних кіберзагроз.

Неавторизований доступ — це втручання в мережеві системи або бази даних без належних дозволів. Це одна з найпоширеніших форм кібератак, яка може призвести до витоку конфіденційної інформації, пошкодження даних, або навіть повного контролю над системою. Цей тип активності може бути здійснений як окремими особами, так і організованими групами.

Брутфорс-Атаки -це метод, при якому автоматизовані системи генерують велику кількість комбінацій імен користувачів та паролів, щоб вгадати правильні дані для входу. Такі атаки часто використовують списки найбільш поширених паролів.

Також є метод експлуатації вразливостей він включає виявлення та використання слабких місць в програмному забезпеченні або конфігурації системи. Зловмисники можуть використовувати спеціальні програми (експлоіти), які дозволяють обійти звичайні механізми безпеки.

Однією з найпопулярніших тактик це фішинг та інсайдери, тактика при якій користувачів обманом змушують розкрити свої персональні дані. Це може бути реалізовано через фальшиві електронні листи, веб-сайти або повідомлення, які імітують законні сервіси. У деяких випадках неавторизований доступ може бути здійснений внутрішніми співробітниками або особами, які мають деякий рівень допуску до системи.

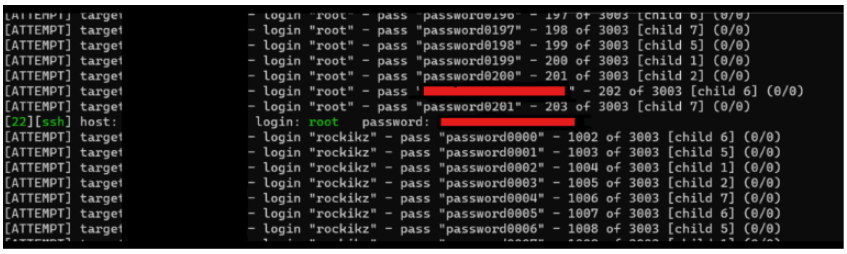


Рисунок 1.1 – Приклад роботи Брутфорс-Атаки

Заходи захисту:

1) Складні Паролі та Політики Безпеки. Використання складних паролів, які регулярно змінюються, може значно ускладнити брутфорс-атаки;

2) Багатофакторна та аутентифікація - це додатковий рівень безпеки, який вимагає від користувачів підтвердження своєї особи за допомогою декількох незалежних механізмів;

3) Оновлення та патчі безпеки являється головним залобом профілактики, регулярне оновлення програмного забезпечення та оперативне встановлення патчів безпеки можуть запобігти експлуатації відомих вразливостей.

Аномальний трафік у мережі відноситься до будь-яких незвичайних або несподіваних змін у шаблонах передачі даних. Ці зміни можуть включати неочікуване збільшення або зменшення обсягу трафіку, а також нерегулярні зразки передачі даних. Аномальний трафік часто є індикатором кібератак, таких як DDoS (розподілені атаки з відмовою в обслуговуванні) або інші мережеві загрози. До типових методів входить не тільки DDoS-Атаки, також є дуже популярним останнім часом це активність ботів, та шкідливе ПО(віруси та інші). DDoS-Атаки часто відрізняються від інших тім що при атакі зловмисники навмисно перевантажують мережу або сервери великою кількістю запитів, що

призводить до їхньої нездатності обслуговувати легітимний трафік. На відміну від DDoS ботнети можуть генерувати великі обсяги трафіку, організовуючи атаки або відправляючи спам, доволі часто ці атаки працюють по спільній схемі. В свою чергу різні види шкідливого програмного забезпечення можуть спричинити незвичайні шаблони трафіку, наприклад, відправляючи велику кількість даних з інфікованого комп'ютера.

В контексті кібербезпеки, аналіз та виявлення підозрілих дій в мережі є невід'ємною частиною захисту інформаційних систем. Інструменти моніторингу мережі відіграють ключову роль у виявленні аномалій у трафіку, що можуть вказувати на несанкціоновані або зловмисні дії. Застосування систем виявлення та попередження проникнень (IDS/IPS) дозволяє оцінити трафік на наявність зловмисних патернів або відомих підписів атак, забезпечуючи вчасне реагування на можливі загрози.

За допомогою алгоритмів машинного навчання можна аналізувати трафік на відхилення від звичайних шаблонів, що є особливо корисним для виявлення складних атак, таких як розподілені атаки з відмовою в обслуговуванні (DDoS). Обмеження швидкості передачі даних та використання розподілених мереж доставки змісту (CDN) можуть допомогти уникнути перевантаження мережі та забезпечити її більшу стійкість.

Експлуатація вразливостей – це процес, при якому атакуючі використовують слабкі сторони в програмному та апаратному забезпеченні для отримання несанкціонованого доступу або завдання шкоди системам. Вразливості в софті чи обладнанні можуть бути різноманітними: від помилок у коді до застарілих компонентів, які не були своєчасно оновлені. Такі дії можуть призвести до неправомірного доступу до системи, крадіжки даних чи їх знищення, що робить їх серйозною загрозою для кібербезпеки.

Загрози безпеці даних мають безліч форм. Наприклад, атаки SQL-ін'єкцій дозволяють зловмисникам втручатися в роботу баз даних через веб-форми. Cross-Site Scripting (XSS) – це введення шкідливих скриптів на веб-сторінки, що виконуються в браузерах користувачів та можуть викрасти їхні дані чи змінити поведінку веб-сайтів. Експлойти дня нуль використовуються для атак на

вразливості, про які ще не відомо розробникам, і відповідно, не існує патчів для їх усунення. Протидія цим атакам включає ряд заходів, таких як регулярні оновлення програмного забезпечення, аудит безпеки та тестування на проникнення, а також запровадження захисту від вищезгаданих атак через використання безпечних методів програмування та API.

Підозріла активність у соціальних мережах становить суттєвий ризик для індивідуальної та колективної безпеки та вимагає ретельного моніторингу та аналізу. Розпалювання конфліктів в соціальних мережах часто відбувається через публікації, що містять провокативні заяви або маніпулятивний контент, який може підбурювати до агресії або соціальної нестабільності. Постійні критичні висловлювання щодо влади, хоча й є частиною політичного дискурсу, іноді перетворюються на провокативні повідомлення, що можуть виходити за рамки конструктивної критики та порушувати норми законодавства або етики.

Образи в соціальних мережах, особливо коли вони містять ненормативну лексику або персональні нападки, можуть вести до цькування та психологічних травм. Ці дії можуть мати тривалі негативні наслідки для жертв та сприяти атмосфері інтернет-тролінгу та онлайн-напруги. Розповсюдження дезінформації та фейкових новин створює інформаційний шум, що ускладнює отримання правдивої інформації та підтримує сумнівні або неправдиві наративи. Ці аспекти разом формують серйозний виклик для підтримки здорової комунікативної екосистеми в соціальних мережах.

Розробка мовних моделей, які можуть автоматично ідентифікувати та класифікувати ці види підозрілої активності, є ключовим напрямком сучасних досліджень у галузі штучного інтелекту та обробки природної мови. Така модель повинна не тільки розрізняти окремі слова та фрази, але й аналізувати контекст та інтенції користувачів, що вимагає розуміння субтексту, іронії та неоднозначності мови.

1.2 Моделі і методи інтелектуального аналізу мовної інформації

Мовні моделі - це алгоритми, засновані на штучному інтелекті (ШІ) та обробці природної мови (NLP)[1] які вчяться розуміти, інтерпретувати та

генерувати людську мову. Далі будуть наведені і описані ключові аспекти того, як мовні моделі функціонують

Одним і майже найголовнішим є тренування на великих обсягах тексту. Мовні моделі тренуються на масивних наборах текстових даних. Це можуть бути книги, статті, веб-сторінки тощо. Під час тренування модель "вчиться" відповідним структурам мови, включаючи граматику, слівник та контекстуальне використан

Особливо важливою є архітектура трансформерів, яка використовує механізми уваги для обробки послідовностей слів. Це дозволяє моделі більш ефективно враховувати контекст та взаємозв'язки між словами. ня слів.

Також існують статистичні методи (наприклад, n-gram) та глибоке навчання. Ранні мовні моделі використовували статистичні методи, такі як n-gram, для передбачення наступного слова на основі попередніх слів. Ці моделі обмежувалися короткими відрізками тексту, які вони могли аналізувати.

Сучасні мовні моделі часто побудовані на основі нейронних мереж, зокрема архітектур глибокого навчання, таких як LSTM (Long Short-Term Memory)[2] та трансформери. Ці моделі ефективніше засвоюють мовні закономірності та контекст.

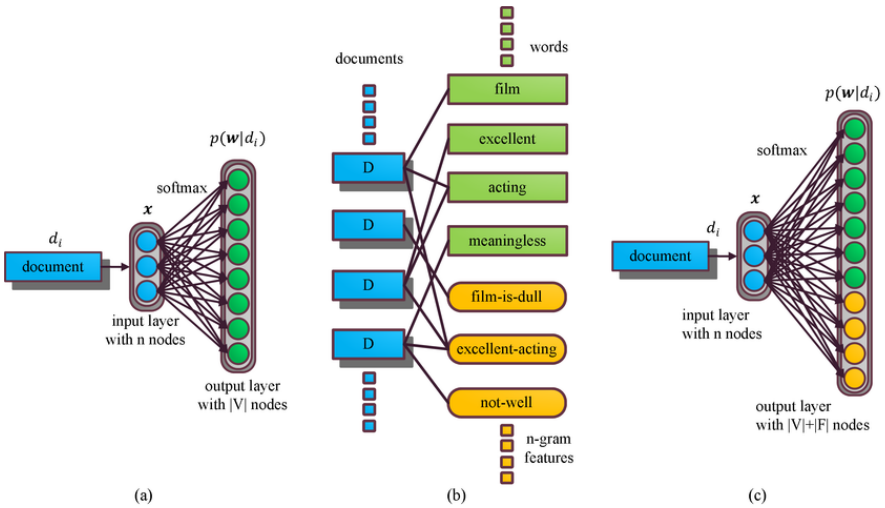


Рисунок 1.2 – DV-ngram модель[3]

Після тренування мовні моделі можуть виконувати такі завдання, як генерація тексту, автоматичний переклад, розпізнавання мовлення, відповіді на запитання та інші завдання, пов'язані з мовними даними. Мовні моделі, такі як

GPT (від OpenAI) або BERT (від Google), здатні до вражаючої обробки мови, використовуючи ці принципи для створення високоякісного, контекстуально відповідного тексту[4].

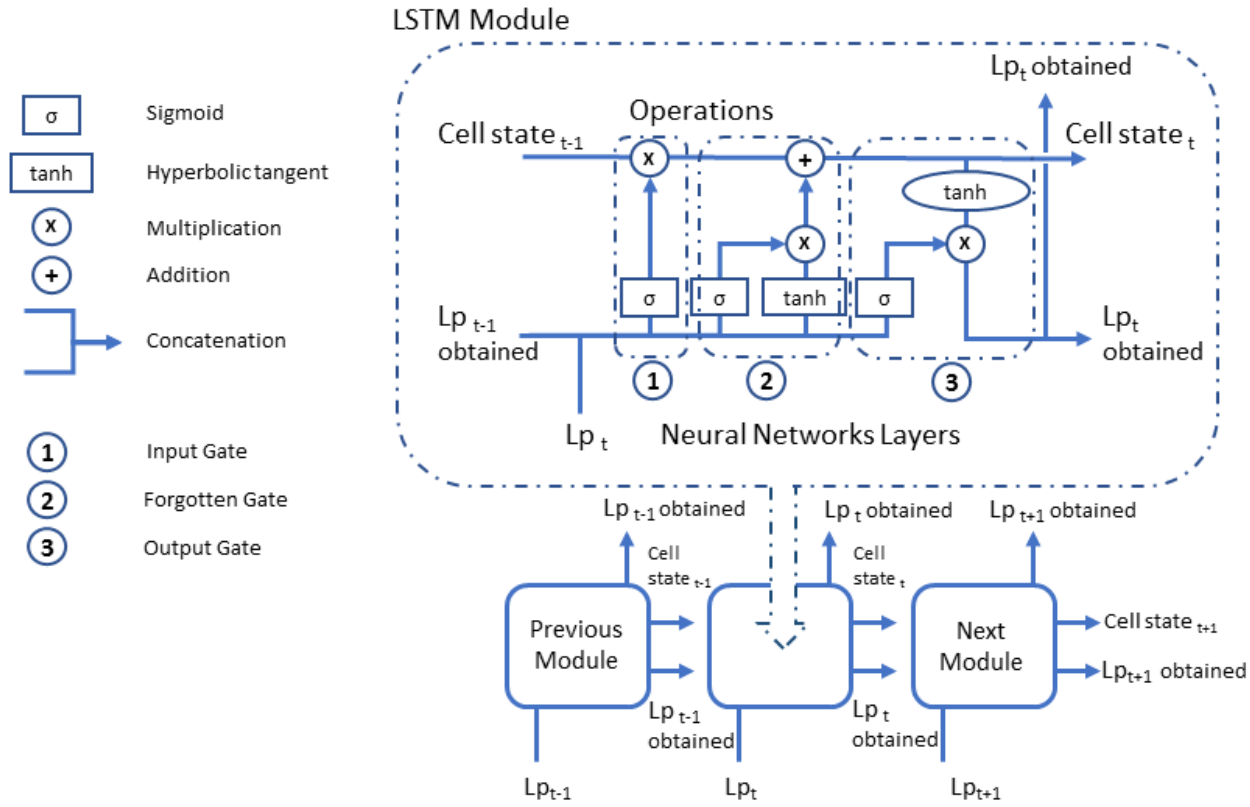


Рисунок 1.3 – Загальна схема довгострокової нейронної мережі (LSTM) [5]

Методи інтелектуального аналізу мовної інформації включають в себе ряд технік та підходів, які використовуються для розуміння, інтерпретації та обробки природної мови. Ці методи засновані на обробці природної мови (NLP) та штучному інтелекті (ШІ) та включають наступні: токенізація, морфологічний аналіз, синтаксичний аналіз, семантичний аналіз та інші.

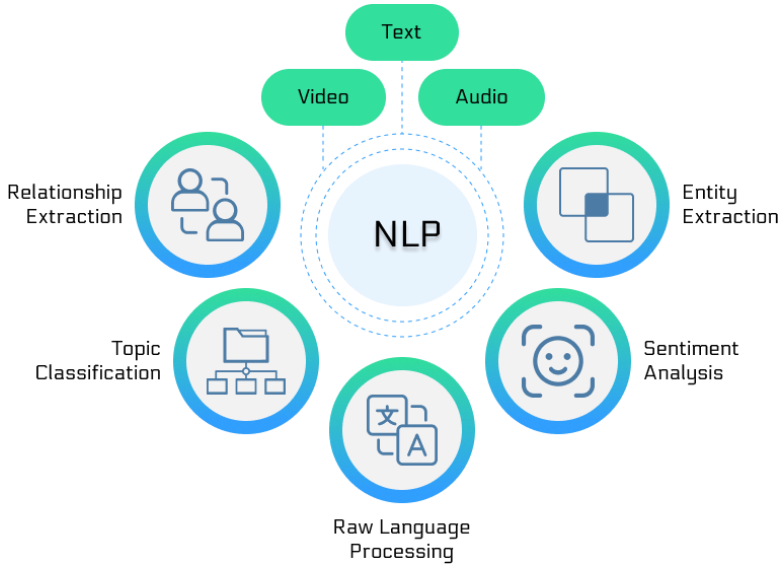


Рисунок 1.4 – Діаграма інтеграції обробки мови (NLP) з різними форматами вхідних даних та її застосування в різноманітних задачах [7].

Токенізація - це процес розділення тексту на окремі елементи, які називаються токенами. Ці токени можуть бути словами, фразами, реченнями або навіть окремими символами. Токенізація є важливим кроком у багатьох задачах NLP,[6] оскільки вона допомагає перетворити неперервний текст у структуровану послідовність, яку можна аналізувати. Різні методи токенізації включають використання простору та пунктуації як розділювачів, а також застосування складніших правил.

Морфологічний аналіз зосереджується на структурі слів, включаючи визначення коренів, суфіксів, префіксів та інших морфем. Цей аналіз дозволяє зрозуміти граматичні характеристики слів, такі як час, число, рід тощо. Він важливий для розуміння відмінностей між словами та їх правильного використання в реченнях.

Синтаксичний аналіз полягає в аналізі структури речень, визначенні взаємозв'язків між словами та фразами. Це включає ідентифікацію суб'єктів, присудків, об'єктів та інших частин речення. Синтаксичний аналіз використовує дерева розбору для представлення структури речення, допомагаючи зрозуміти граматичні відносини та залежності між словами.

Семантичний аналіз вивчає значення слів та речень, включаючи інтерпретацію концептів, об'єктів та відносин між ними. Цей аналіз дозволяє визначити, що саме мається на увазі в тексті, і виявляє змістовні зв'язки між різними елементами мови. Семантичний аналіз особливо важливий у контексті розуміння нюансів та багатозначності мови.

Прагматичний аналіз оцінює, як контекст та ситуація впливають на значення мови. Це включає розуміння інтенцій мовця, намірів, іронії та інших аспектів, які впливають на те, як мова сприймається слухачами. Прагматичний аналіз важливий для правильного інтерпретування мовних повідомлень в їх реальному контексті використання.

Розпізнавання емоцій та сентимент-аналіз виявляють емоційні стани або

настрої, які виражені у тексті. Це може включати визначення позитивних, негативних або нейтральних емоцій. Цей аналіз широко використовується для оцінки сприйняття продуктів, послуг та різних подій у соціальних медіа та відгуках споживачів[7].

Розпізнавання іменованих сутностей полягає в ідентифікації та класифікації важливих інформаційних елементів у тексті, таких як імена людей, організацій, географічних назв. Це дозволяє автоматично виділяти та категоризувати ключові інформаційні елементи, полегшуючи подальший аналіз або використання даних.

Машинний переклад забезпечує автоматичний переклад тексту з однієї мови на іншу. Цей процес включає розуміння семантики, граматики та контексту вихідного тексту, а також здатність генерувати адекватний переклад на цільовій мові. Розвиток машинного перекладу значно спростив комунікацію та доступ до інформації між різними мовними спільнотами.

Chatbots та діалогові системи - це програмні системи, розроблені для ведення розмови з людьми в натуральній мові. Вони використовують NLP та ШІ для розуміння запитів користувачів та генерації природніх та релевантних відповідей. Ці системи знаходять застосування в обслуговуванні клієнтів, автоматизації відповідей та як персональні помічники.

Ці методи широко використовуються в різноманітних застосуваннях, від систем автоматичної відповіді та персональних асистентів до аналізу соціальних медіа та автоматичного збору даних.

1.3 Великі мовні моделі LLM та збір даних

Великі мовні моделі (LLM) - це передові системи штучного інтелекту, які революціонізували спосіб взаємодії з машинами за допомогою природної мови. Вони навчаються на великих масивах текстових даних, що охоплюють широкий спектр людських знань і лінгвістичних структур, що дозволяє їм генерувати, перекладати, узагальнювати і відповідати на питання про текст з надзвичайною точністю.

Прикладом цієї технології є LLM GPT-3 (Generative Pretrained Transformer 3) від OpenAI - генеративний попередньо навчений трансформатор. Він був

навчений на наборах даних, які включають значну частину Інтернету, що дозволяє йому виконувати широкий спектр завдань, від написання творчих робіт до кодування. Його здатність розуміти і генерувати текст, подібний до людського, настільки розвинена, що він може писати есе, створювати вірші або навіть генерувати код у реальному часі, і все це з мінімальним вкладом з боку користувача.

Іншим прикладом є BERT (Bidirectional Encoder Representations from Transformers) від Google, який особливо вправно розуміє контекст слова в реченні, покращуючи таким чином результати пошуку, більш ефективно інтерпретуючи наміри, що стоять за пошуковими запитами.

Можливості LLM поширюються на кілька мов, що робить їх безцінними інструментами для перекладацьких служб. Вони можуть зберігати нюанси та ідіоматичні вирази вихідної мови, забезпечуючи не лише точний, але й контекстуально релевантний переклад.

Крім того, магістрів права використовують у розмовному штучному інтелекті, створюючи чат-ботів і віртуальних асистентів, які можуть вести більш природні та контекстно-орієнтовані діалоги. Вони можуть відповідати на запити клієнтів, надавати рекомендації і навіть керувати складними процесами обслуговування клієнтів.

Однак з великими можливостями приходить і велика відповідальність. Розгортання LLMs викликає етичні міркування щодо їхнього потенціалу увічнити упередження, присутні в їхніх навчальних даних, генерувати неправдиву інформацію або замінити людські робочі місця. Таким чином, розробка і впровадження цих моделей потребують ретельного розгляду і регулювання, щоб забезпечити їх етичне і корисне використання.

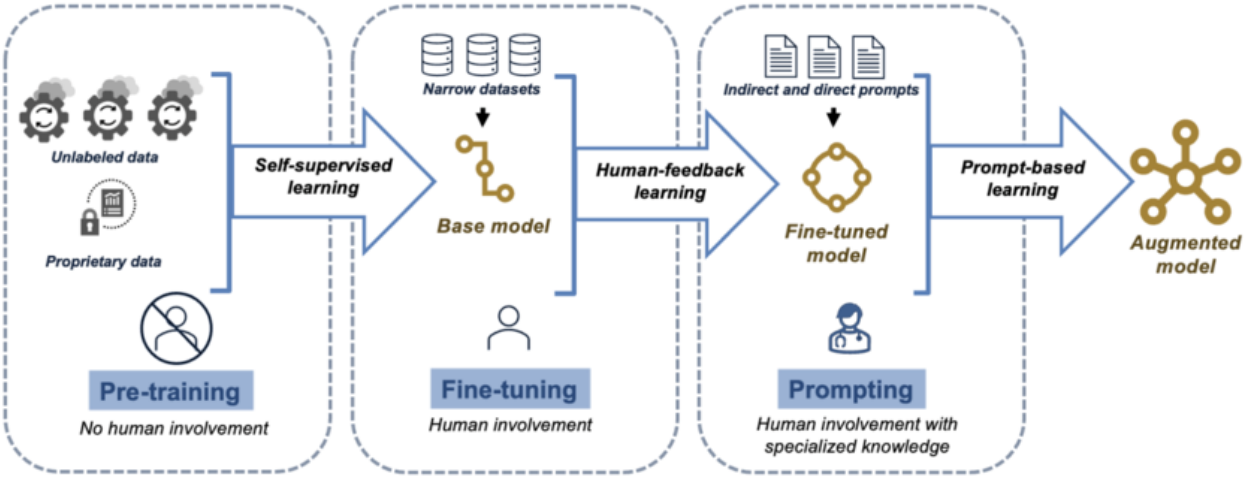


Рисунок 1.5 - Огляд процесу навчання LLM

Підсумовуючи, можна сказати, що LLM стоять на передовій технології NLP, надаючи інструменти, які можуть розуміти і генерувати людську мову з безпрецедентною витонченістю. Вони пропонують величезний потенціал для покращення комунікації, автоматизації завдань і надання інсайтів у різних галузях, що робить їх однією з найбільш трансформаційних технологій у сфері штучного інтелекту.

Для збору даних для навчання мовної моделі існує кілька методів, які зазвичай використовують дослідники та розробники:

1) Автоматизований веб-скрейпінг. Використання скриптів або інструментів для систематичного збору великих обсягів текстових даних з Інтернету. Наприклад, сканування новинних сайтів або форумів, щоб зібрати дані про використання різних мов;

2) Публічні набори даних. Використання доступних наборів даних зі сховищ, таких як GitHub, Kaggle або академічних баз даних, які пропонують безліч текстів різними мовами та форматами;

3) API та соціальні мережі. Використання API, що надаються такими платформами, як Twitter або Reddit, для доступу до великих потоків даних, які можуть включати широкий спектр мовних виразів і сленгу.

4) Краудсорсинг. Залучення таких платформ, як Amazon Mechanical Turk, для збору та маркування даних, де багато користувачів беруть участь у зборі та анотуванні даних;

5) Напівавтоматичні інструменти. Використання таких інструментів, як Google Forms, для збору даних від користувачів. Це може бути особливо корисно для збору певних типів тексту, наприклад, прикладів підозрілих дій;

6) Партнерства та співпраця. Співпраця з організаціями та установами, які можуть надати доступ до власних даних, що може бути особливо цінним для спеціалізованих мовних моделей.

У даному випадку використовували форму Google для збору прикладів та доступні набори даних з GitHub. Цей напівавтоматичний підхід дозволяє збирати цільові дані, які можуть мати безпосереднє відношення до передбачуваного застосування моделі, гарантуючи, що навчальний матеріал буде максимально наближений до реальних даних, з якими зіткнеться модель. Це може значно підвищити точність моделі у виявленні та класифікації різних типів підозрілої поведінки.

1.4 Формалізована постановка задачі

Розробка та навчання мовної моделі, спроможної автоматизовано виявляти підозрілу активність у мовних даних. Проблема полягає у виявленні та аналізі текстових повідомлень, які можуть містити потенційно шкідливий контент. Це включає, але не обмежується, спонуканням до суїциду, критикою, розпалюванням ненависті, а також ідентифікацією ботів і підозрілих запитань:

1) Збір та Обробка Даних: Зібрати великий і різноманітний набір текстових даних, що включає приклади потенційно шкідливого контенту. Провести попередню обробку даних, включаючи токенізацію та нормалізацію;

2) Розробка Мовної Моделі: Створити модель, яка може аналізувати текст та визначати потенційно шкідливі елементи. Модель повинна використовувати сучасні техніки NLP та машинного навчання, такі як нейронні мережі;

3) Тренування та Валідація Моделі: Навчити модель на зібраних даних, використовуючи методи глибокого навчання. Валідувати її ефективність на незалежних тестових наборах даних[8-9];

4) Аналіз та Інтерпретація Результатів: Оцінити якість роботи моделі, аналізуючи її здатність коректно виявляти різні типи підозрілого контенту;

5) Оптимізація та Удосконалення: Оптимізувати модель для підвищення точності, швидкості обробки та зменшення хибнопозитивних та хибнонегативних результатів.

Розробка ефективної мовної моделі, яка може точно ідентифікувати шкідливий контент у різних контекстах. Модель повинна сприяти підвищенню безпеки у цифровому просторі, зменшуючи ризики, пов'язані з негативним контентом. Цей пункт є ключовим у дослідженні, оскільки він концентрується на практичному застосуванні навчених моделей для вирішення конкретних проблем безпеки в мережі. Він вимагає глибокого розуміння як технічних аспектів машинного навчання та NLP, так і соціальних та психологічних факторів, які впливають на поведінку людей в мережі[10].

2 ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ РОЗПІЗНАВАННЯ ПІДОЗРІЛОЇ АКТИВНОСТІ В МЕРЕЖЕВИХ ЧАТАХ

2.1 Модель розпізнавання підозрілих повідомлень в мережевих чатах

Система виявлення підозрілого тексту (STD) - це програмний інструмент, призначений для автоматичного аналізу текстової інформації з метою виявлення ознак або паттернів, що можуть вказувати на небажаний змісту або нелегальні повідомлення.

Вона базується на комплексі технологій обробки природної мови (NLP) та аналізу даних [11]. Для цього використовуються алгоритми машинного навчання для навчання моделей розпізнавання паттернів, що вказують на підозрілу або небажану інформацію. До основних компонентів такої системи входять:

- **Токенізація тексту:** Розбиття текстового потоку на окремі токени або слова для подальшого аналізу.
- **Екстракція ознак:** Виділення характерних ознак із тексту, таких як ключові слова, фрази або синтаксичні конструкції, які можуть вказувати на підозрюваний контент.
- **Векторизація тексту:** Перетворення текстових даних в числовий вектор, що використовується для подальшого навчання моделей машинного навчання.
- **Моделі машинного навчання:** Використання класифікаторів або нейронних мереж для навчання на позначених даних та визначення та розпізнавання паттернів, що характеризують підозрілий текст.
- **Постановка висновків:** Визначення ймовірності та надання висновків щодо наявності підозрюваного контенту у вхідних текстових даних.

Ці технічні компоненти дозволяють ефективно автоматизувати виявлення підозрілого тексту, що допомагає у виявленні потенційних загроз або небажаного змісту у текстових джерелах [12].

Система виявлення підозрілого тексту (STD) класифікує текстовий $t_i \epsilon T$ із набору текстів T = {t1, t2, ..., tm} у клас $c_i \epsilon C$ із набору з двох класів $C = \{C_s, C_{ns}\}$. . Завдання STD — автоматично призначити ti до ci: < ti, ci >.

Вирішити, чи є текст підозрілим чи ні, не так просто навіть для мовних експертів через його складну морфологічну структуру. Багато варіацій у формуванні речень та відсутність визначення відповідної термінології, призводить до того, що важливо мати чітке визначення підозрілих текстів, щоб полегшити завдання STD. Для того, щоб представити цільне визначення, було проаналізовано кілька визначень: насильства, підбурювання, підозрілого та ненависті.

Більшість інформації, зібраної з різних джерел, узагальнено в таблиці 2.1:

Таблиця 2.1 – Визначення забороненого вмісту відповідно до різних веб-сайтів соціальних мереж, організацій та наукових досліджень

Джерело	Визначення
YouTube	«Вміст, який заохочує інші пропагувати або вчиняти насильство проти окремих осіб і груп на основі релігії, національного походження, етнічного походження, статі/гендеру, віку, раси, інвалідності, гендерної ідентичності/сексуальної орієнтації»
Facebook	«Вміст, який підбурює або пропагує серйозне насильство, реальну загрозу громадській або особистій безпеці, інструкції з виготовлення зброї, яка може поранити або вбити, а також погрози завдати фізичної шкоди приватним або публічним діячам».
Twitter	«Ви не повинні пропагувати тероризм або насильницький екстремізм, переслідувати або погрожувати іншим або розпалювати гнів проти певних груп чи груп людей».
Рада Європи	«Висловлювання, які підбурюють, поширюють, заохочують або виправдовують насильство проти конкретних осіб або груп з будь-якої причини»
Paula та інші.	«Мова, яка прославляє насильство та ненависть, підбурює людей до цільових груп на основі релігії, расового чи етнічного походження, зовнішності, гендерної ідентичності тощо»

Більшість з наведених визначень зосереджені на подібних атрибутах, таких

як підбурювання до насильства, пропаганда ненависті та тероризму, а також погрози особі чи групі людей. Ці визначення охоплюють ширший аспект підозрілого вмісту, включаючи відео, текст, зображення, мультфільми, ілюстрації та графіку. Тим не менш, ця робота зосереджується на виявленні підозрілого вмісту лише в текстовому форматі.

Проаналізувавши зміст і властивості цих визначень, було створене наступне визначення підозрілого тексту:

«Підозрілий текст – це текст, який має шахрайські мотиви, підбурює до насильства, заохочує до тероризму, пропагує насильницький екстремізм, підбурює до політичних партій або підбурює проти окремих осіб чи спільнот на основі певних ознак, таких як релігійні переконання, меншини, сексуальна орієнтація, раса та фізичні характеристики.»

Сучасні нейронні мережі являють собою захоплюючий новий підхід до створення тексту природною мовою. Значна частина початкових досліджень генераторів нейронного тексту була спрямована на розробку різних архітектур.

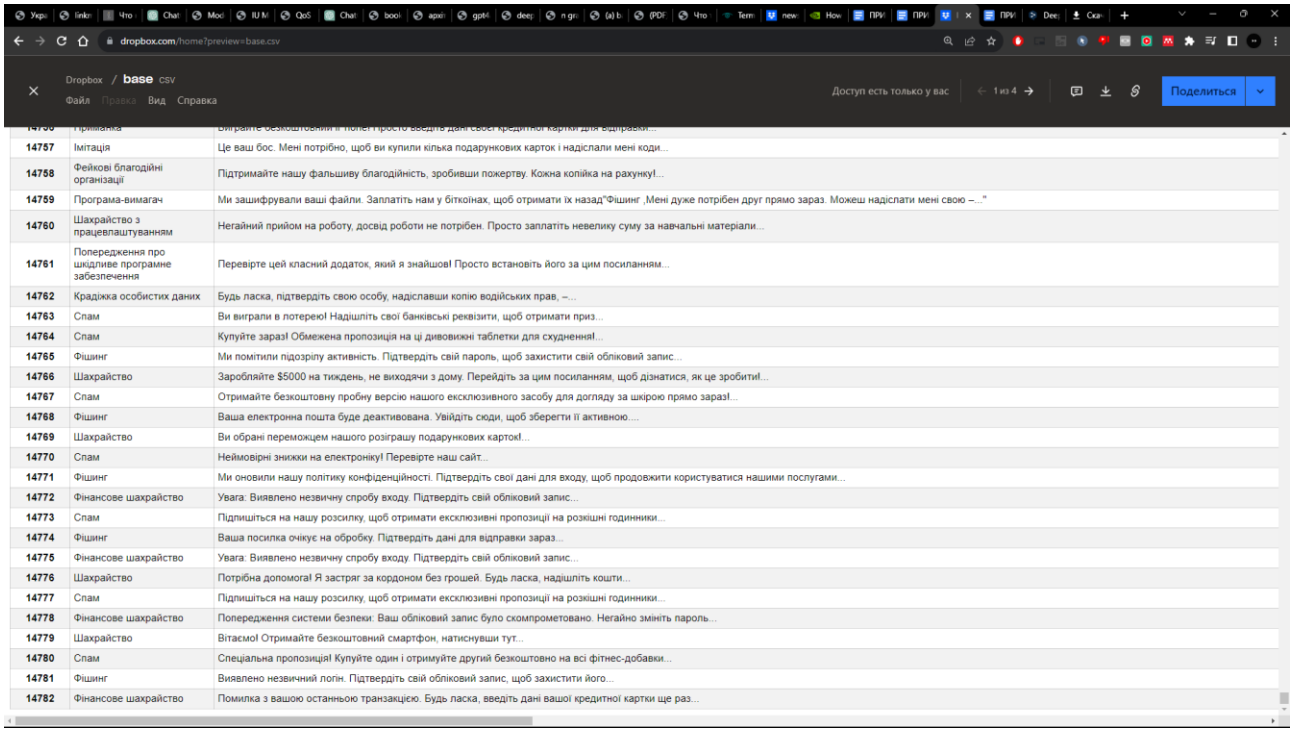
Однак нещодавня робота натякнула на те, яка стратегія декодування. тобто метод, який використовується для генерації рядків з моделі. може бути важливішим, ніж сама архітектура моделі. У світлі цього відкриття в літературі було представлено безліч стратегій декодування, кожна з яких претендує на створення більш бажаного тексту, ніж конкуруючі підходи.

2.2 Метод налаштування мовної моделі на виконання завдання

Настроювання мовної моделі на основі трансформерів для виявлення підозрілої активності в мережевих чатах починається з вибору відповідної попередньо навченої моделі, такої як BERT[13]або GPT-3. Ці моделі вже мають широкі знання про мову, отримані завдяки тренуванню на обширних текстових корпусах. Перший крок — це збір і підготовка тренувального датасету, який має відображати реальні умови використання і містити як підозрілі, так і нейтральні повідомлення. Такий підхід дозволяє моделі навчитися розрізняти нормальну поведінку від потенційно шкідливої або підозрілої. Попередня обробка даних

включає токенізацію, лематизацію, видалення шуму та нормалізацію тексту, що готує дані для ефективного машинного навчання.

Файн-тюнінг — це процес, де модель налаштовують за допомогою специфічного набору даних, щоб адаптувати її до конкретних вимог завдання. В цей процес входить налаштування гіперпараметрів, таких як швидкість навчання та кількість епох, а також валідація моделі на відокремленому наборі даних для гарантії, що модель правильно узагальнює, а не просто запам'ятовує тренувальні дані. Після успішного файн-тюнінгу модель проходить тестування на незалежних датасетах для оцінювання її здатності розпізнавати підозрілу активність в реальному світі. На цьому етапі використовуються метрики, такі як точність, виклик та F1-міра S[14], для кількісної оцінки продуктивності моделі. Останнім кроком є оптимізація моделі для її розгортання у виробництві, де вона може бути інтегрована в системи реального часу для моніторингу мережевих чатів. Цей процес також включає моніторинг роботи моделі після розгортання, щоб забезпечити її стабільність та надійність, реагуючи на нові виклики та адаптуючись до змін у поведінці користувачів та еволюції мовних патернів.



14757	Імітація	Це ваш бос. Мені потрібно, щоб ви купили кілька подарункових карток і надіслали мені коди...
14758	Фейкові благодійні організації	Підтримайте нашу фальшиву благодійність, зробивши пожертву. Кожна копійка на рахунку!...
14759	Програма-вимагач	Ми зашифрували ваші файли. Заплатіть нам у біткоїнах, щоб отримати їх назад"Фішинг ,Мені дуже потрібен друг прямо зараз. Можеш надіслати мені свою –..."
14760	Шахрайство з працевлаштуванням	Негайний прийом на роботу, досвід роботи не потрібен. Просто заплатіть невелику суму за навчальні матеріали...
14761	Попередження про шкідливе програмне забезпечення	Перевірте цей класний додаток, який я знайшов! Просто встановіть його за цим посиланням...
14762	Крадіжка особистих даних	Будь ласка, підтвердіть свою особу, надіславши копію водійських прав, –...
14763	Спам	Ви виграли в лотерею! Надішліть свої банківські реквізити, щоб отримати приз...
14764	Спам	Купуйте зараз! Обмежена пропозиція на ці дивовижні таблетки для схудненняІ...
14765	Фішинг	Ми помітили підозрілу активність. Підтвердіть свій пароль, щоб захистити свій обліковий запис...
14766	Шахрайство	Заробляйте $5000 на тиждень, не виходячи з дому. Перейдіть за цим посиланням, щоб дізнатися, як це зробити!...
14767	Спам	Отримайте безкоштовну пробну версію нашого ексклюзивного засобу для догляду за шкірою прямо зараз!...
14768	Фішинг	Ваша електронна пошта буде деактивована. Увійдіть сюди, щоб зберегти її активною....
14769	Шахрайство	Ви обрані переможцем нашого розіграшу подарункових карток!...
14770	Спам	Неймовірні знижки на електроніку! Перевірте наш сайт...
14771	Фішинг	Ми оновили нашу політику конфіденційності. Підтвердіть свої дані для входу, щоб продовжити користуватися нашими послугами...
14772	Фінансове шахрайство	Увага: Виявлено незвичну спробу входу. Підтвердіть свій обліковий запис...
14773	Спам	Підпишіться на нашу розсилку, щоб отримати ексклюзивні пропозиції на розкішні годинники...
14774	Фішинг	Ваша посилка очікує на обробку. Підтвердіть дані для відправки зараз...
14775	Фінансове шахрайство	Увага: Виявлено незвичну спробу входу. Підтвердіть свій обліковий запис...
14776	Шахрайство	Потрібна допомога! Я застряг за кордоном без грошей. Будь ласка, надішліть кошти...
14777	Спам	Підпишіться на нашу розсилку, щоб отримати ексклюзивні пропозиції на розкішні годинники...
14778	Фінансове шахрайство	Попередження системи безпеки: Ваш обліковий запис було скомпрометовано. Негайно змініть пароль...
14779	Шахрайство	Вітаємо! Отримайте безкоштовний смартфон, натиснувши тут...
14780	Спам	Спеціальна пропозиція! Купуйте один і отримуйте другий безкоштовно на всі фітнес-добавки...
14781	Фішинг	Виявлено незвичний логін. Підтвердіть свій обліковий запис, щоб захистити його...
14782	Фінансове шахрайство	Помилка з вашою останньою транзакцією. Будь ласка, введіть дані вашої кредитної картки ще раз...

Рисунок 2.1 – Файл з даними для навчання мовної моделі

Спираючись на процес тонкого налаштування мовної моделі з трансформаторною архітектурою для виявлення підозрілих дій у мережевих

чатах, розглядаємо набір даних, ретельно зібраний для представлення різноманітної та складної тканини спілкування Рис.2.1, що відбувається в Інтернеті. Цей набір даних, багатий мовний габілен, що охоплює 15 000 записів, втілює хитросплетіння і тонкощі лінгвістичних моделей, починаючи від безневинних розмов і закінчуючи зловісними відтінками афер, шахрайства і безлічі інших обманних практик. Кожен запис у цій колекції слугує для моделі навчальним матеріалом, що дозволяє їй розпізнати не лише явні ознаки шахрайства, але й завуальовані натяки, які можуть вислизнути від менш тонкого підходу.

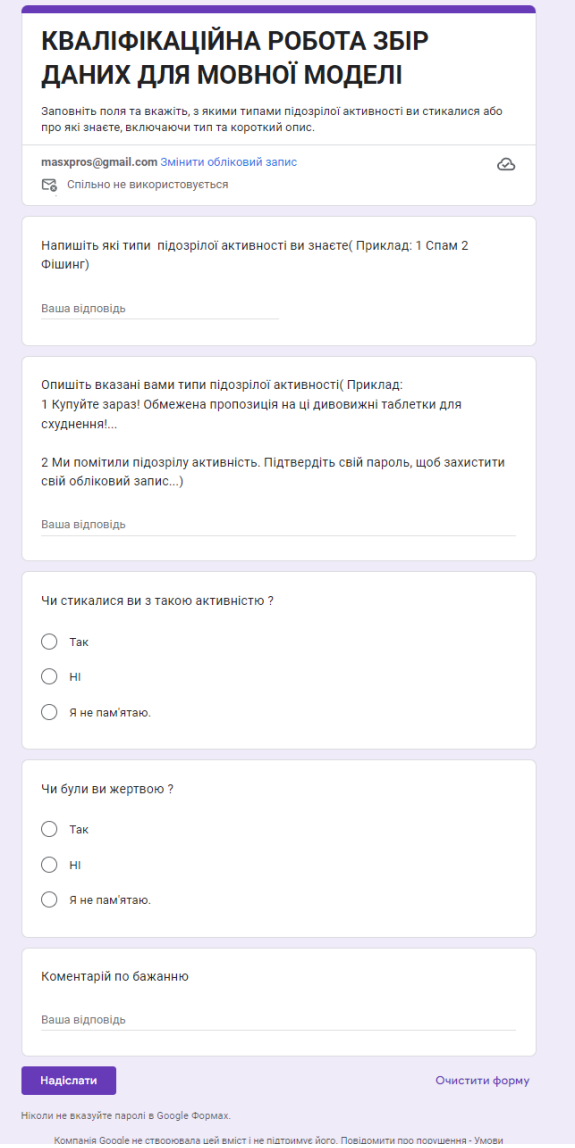


Рисунок 2.2 – Форма для збору даних

Дані брались з різних джерел та навіть напівавтоматично через гугл форму як зображено на Рис.2.2.

Навчання моделі – складна робота котра пов’язана з цим величезним лінгвістичним ландшафтом, керована передовими техніками NLP [15]. Вона починається з глибокого занурення в набір даних, де модель вчиться розпізнавати ознаки зловмисності, такі як примусова мова, терміновість дій та зловживання довірою. Проходячи через дані, модель точно налаштовує свої параметри - вагові коефіцієнти, швидкість навчання та шари - вдосконалюючи свою здатність розуміти контекст і видобувати сенс. Цей ітеративний процес нагадує роботу ремісника, який відточує своє ремесло, причому кожна епоха представляє собою шар складності та витонченості, що додається до зростаючого інтелекту моделі.

Щоб гарантувати, що модель не просто запам'ятала, а дійсно зрозуміла, валідація є ключовим фактором. Вона піддається суворому тестуванню з використанням невидимих даних, змушуючи її застосовувати набуті знання до нових сценаріїв. Саме тут виявляється справжня міра досконалості моделі, а такі показники, як точність, запам'ятовування та оцінка F1, дають кількісне свідчення її ефективності. Ці показники є маяками, які спрямовують процес тонкого налаштування, гарантуючи, що модель досягає тонкого балансу між чутливістю та специфічністю [16].

Після того, як модель довела свою спроможність на полігоні, фокус зміщується на готовність до розгортання. Вона проходить оптимізацію, щоб безперешкодно працювати в реальному часі в рамках мережевих чат-систем. Ця оптимізація є делікатним танцем обчислювальної ефективності та точності прогнозування, гарантуючи, що модель зможе безперебійно функціонувати у високошвидкісному світі онлайн-спілкування.

На завершальному етапі модель виходить з тренувального середовища в реальний світ, де вона починає своє чергування. Тут вона працює як цифровий вартовий, постійно скануючи потоки даних, що протікають через мережеві чати. Але робота не закінчується з розгортанням [17-18]. Цифрова сфера постійно розвивається, постійно з'являються нові форми комунікації та обману. Тому навчання моделі є безперервним, вимагаючи постійних оновлень і коригувань, щоб йти в ногу з мінливими пісками мови і людської поведінки. Цей

безперервний процес навчання є життєво важливим, оскільки дозволяє моделі залишатися надійним захисником від постійної хвилі онлайн-загроз.

2.3 Метод оцінювання ефективності розпізнавання підозрілих повідомлень

Оцінка ефективності мовної моделі, яка використовується для виявлення підозрілих повідомлень у мережевих чатах, вимагає використання різноманітних метрик і методів аналізу. Точність моделі демонструє її здатність правильно ідентифікувати підозрілі повідомлення, але вона може бути не достатньо інформативною у випадку нерівномірно розподілених класів. Виклик та точність подають більш глибоке розуміння про те, як модель ідентифікує класи інтересу, забезпечуючи інформацію про те, скільки реально підозрілих повідомлень було виявлено і яка частка ідентифікованих як підозрілі повідомлень насправді є такими.

Матриця плутанини — це спосіб оцінки продуктивності моделі класифікації. Це порівняння між основною істинністю (фактичними значеннями) і прогнозованими значеннями, випромінюваними моделлю для цільової змінної.

Матриця плутанини корисна в категорії навчання під наглядом машинного навчання з використанням позначеного набору даних [19].

Матриця плутанини представлена позитивним і негативним класами. Позитивний клас представляє ненормальний клас або поведінку, тому зазвичай представлений менше, ніж інший клас. Негативний клас, з іншого боку, представляє нормальність або нормальну поведінку.

Моделі створюються для максимально точного прогнозування того, що не є нормальним (виявлення аномалії). Наприклад, ми хотіли б передбачити потенційних спамерів як боти або потенційних злочинців як злочинців.

Ось чотири квадранти в матриці плутанини:

- Справжній позитивний результат (TP) – це результат, коли модель правильно передбачає позитивний клас;
- Справжній негативний результат (TN) – це результат, коли модель правильно передбачає негативний клас;

- Хибнопозитивний результат (FP) – це результат, коли модель неправильно передбачає позитивний клас;
- Помилково негативний результат (FN) – це результат, коли модель неправильно передбачає негативний клас.

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

Accuracy. Ця метрика відображає загальну точність класифікації, враховуючи як правильно класифіковані позитивні, так і негативні приклади.

$$\mathrm{P} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{2.2}$$

Precision. Ця метрика визначає, яка частина позитивно визначених прикладів дійсно є позитивними.

$$\mathrm{Sn} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{2.3}$$

Recall або Sensitivity. Ця метрика показує, яка частина дійсно позитивних прикладів була визначена як позитивні.

$$\mathrm{Sp} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}} \tag{2.4}$$

Specificity. Ця метрика показує, яка частина дійсно негативних прикладів була визначена як негативні.

$$\mathrm{F-score} = 2 \times \frac{\mathrm{P} \times \mathrm{Sn}}{\mathrm{P} + \mathrm{Sn}} \tag{2.5}$$

F-score. Ця метрика об'єднує Precision та Recall у гармонічний показник, що дозволяє оцінити точність моделі при класифікації [20-21].

Перехресна перевірка забезпечує високу надійність оцінки, дозволяючи перевіряти стабільність моделі на різних вибірках даних, тоді як матриця помилок дає детальний огляд правильних та неправильних класифікацій, допомагаючи точно ідентифікувати слабкі місця моделі. Додаткові діагностичні тести, такі як використання замаскованого контексту або введення спеціальних мовних ідіом, можуть виявити потенційні проблеми з моделлю, які не були зазначені під час стандартного процесу тестування. У вашому випадку, якщо у вас є два стовпці - кількість правильних відповідей (True Positives) і кількість неправильних відповідей (False Negatives), сума цих двох стовпців дорівнює

загальній кількості. Тому точність можна розрахувати як відношення кількості правильних відповідей до загальної кількості.

Обираючи між всіма метриками була обрана Accuracy для оцінки моделі через кілька причин. Ця метрика має свої важливі переваги, особливо в контексті оцінки загальної ефективності класифікатора.

Основні переваги використання Accuracy:

1. Простота та зрозумілість: Accuracy - це проста метрика, яка легко інтерпретується. Вона вимірює загальну точність моделі без необхідності враховувати складні деталі;

2. Загальна ефективність: Для багатьох задач, де важливо зрозуміти, наскільки часто модель робить правильні передбачення, Accuracy надає загальну картину про те, як добре модель виконує свою роботу;

3. Баланс між класами**:** Якщо класи у вашому наборі даних балансовані (один клас не переважає над іншим), то Accuracy буде надійною метрикою. Вона відображає точність класифікації для обох класів однаково.

Однак, слід зауважити, що у деяких випадках Accuracy може бути не найкращим вибором. Наприклад, коли класи у наборі даних незбалансовані, існує небагато екземплярів одного класу у порівнянні з іншим. У таких ситуаціях, модель може бути відмінною у класифікації більш представленого класу, але показувати погані результати у менш представленому класі. Для таких випадків краще розглядати інші метрики, такі як Precision, Recall або F1-score, які дозволяють краще оцінити ефективність моделі в таких умовах.

Під час валідації моделі важливо використовувати реалістичні сценарії, щоб забезпечити, що вона буде ефективною у реальних умовах використання. Це передбачає тестування на датасетах, які відображають різноманіття мовних виразів і соціальних поведінок, які зустрічаються у різних платформах соціальних мереж. Такий підхід гарантує, що модель не тільки навчена правильно ідентифікувати загальний тип контенту, а й здатна адаптуватися до постійно змінюваних форм спілкування та нових методів маніпуляції.

3 РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ РОЗПІЗНАВАННЯ ПІДОЗРІЛИХ ПОВІДОМЛЕНЬ В МЕРЕЖЕВИХ ЧАТАХ

3.1 Формування вхідних даних для настроювання і валідації

Формування вхідних даних для точного налаштування та перевірки мовної моделі передбачає структурований підхід, який гарантує, що дані добре підготовлені для ефективного навчання моделі. У контексті мовної моделі процес починається з визначення типу мови та сценаріїв, які повинна розуміти модель. З огляду на складність людської мови і тонкі нюанси, які можуть вказувати на підозру або зловмисність, вхідні дані повинні бути різноманітними і репрезентативними для реальної комунікації.

Перший крок передбачає збір великого масиву зразків тексту, які демонструють різні форми підозрілої поведінки, як вже зробили, використовуючи різні джерела. Ці зразки можуть варіюватися від фішингових електронних листів, шахрайських повідомлень, шахрайських заяв до більш складних форм обману, таких як тактика соціальної інженерії.

Після завершення етапу збору критично важливою стає попередня обробка даних. Вона включає в себе очищення тексту, щоб видалити нерелевантну інформацію або шум, який може відволікати модель під час навчання. Цей крок включає видалення непотрібного форматування, виправлення помилок або видалення неінформативних слів-заповнювачів. Нормалізація тексту, яка передбачає перетворення тексту в стандартний формат, також має вирішальне значення. Вона гарантує, що варіації у використанні мови, такі як використання сленгу або різні варіанти написання, не перешкоджатимуть процесу навчання моделі.

Рисунок 3.1 – Очищений та готовий текст в одному стилю

Далі відбувається токенізація - розбиття тексту на керовані частини або токени. Наприклад, речення розбиваються на слова або навіть підслова, що допомагає моделі розуміти будівельні блоки мови.

```
!pip install transformers==4.19.0
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting transformers==4.19.0
  Downloading transformers-4.19.0-py3-none-any.whl (4.2 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.2/4.2 MB 38.9 MB/s eta 0:00:00
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.8/dist-packages (from transformers==4.19.0) (2022.6.2)
Requirement already satisfied: requests in /usr/local/lib/python3.8/dist-packages (from transformers==4.19.0) (2.25.1)
Requirement already satisfied: filelock in /usr/local/lib/python3.8/dist-packages (from transformers==4.19.0) (3.9.0)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.8/dist-packages (from transformers==4.19.0) (6.0)
Collecting tokenizers!=0.11.3,<0.13,>=0.11.1
  Downloading tokenizers-0.12.1-cp38-cp38-manylinux_2_12_x86_64.manylinux2010_x86_64.whl (6.6 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6.6/6.6 MB 82.5 MB/s eta 0:00:00
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.8/dist-packages (from transformers==4.19.0) (21.3)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.8/dist-packages (from transformers==4.19.0) (1.21.6)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.8/dist-packages (from transformers==4.19.0) (4.64.1)
Collecting huggingface-hub<1.0,>=0.1.0
  Downloading huggingface_hub-0.12.0-py3-none-any.whl (190 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 190.3/190.3 KB 23.9 MB/s eta 0:00:00
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.8/dist-packages (from huggingface-hub<1.0,>=0.1.0->transformers==4.19.0) (4.4.0)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.8/dist-packages (from packaging>=20.0->transformers==4.19.0) (3.0.9)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.8/dist-packages (from requests->transformers==4.19.0) (2022.12.7)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.8/dist-packages (from requests->transformers==4.19.0) (2.10)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.8/dist-packages (from requests->transformers==4.19.0) (1.24.3)
Requirement already satisfied: chardet<5,>=3.0.2 in /usr/local/lib/python3.8/dist-packages (from requests->transformers==4.19.0) (4.0.0)
Installing collected packages: tokenizers, huggingface-hub, transformers
Successfully installed huggingface-hub-0.12.0 tokenizers-0.12.1 transformers-4.19.0
```

```
from transformers import GPT2LMHeadModel, GPT2Tokenizer
import torch
from torch.utils.data import Dataset # this is the pytorch class import
from transformers import DataCollatorForLanguageModeling
from transformers import TrainingArguments, Trainer
from transformers import StoppingCriteria, StoppingCriteriaList
```

Рисунок 3.2 – Встановлення transformers, та додавання до моделі GPT2LMHeadModel, GPT2Tokenizer

У деяких випадках, особливо в трансформаційних моделях, таких як BERT або GPT-3, токени можуть також призначатися цілим фразам або загальним виразам.

```
SPECIAL_TOKENS = {'bos_token':'<bos>','eos_token' :'<eos>', 'pad_token':'<pad>', 'sep_token': '<sep>'}
tokenizer.add_special_tokens(SPECIAL_TOKENS)
model.resize_token_embeddings(len(tokenizer))
```

```
Embedding(50260, 1024)
```

Рисунок 3.3 – Приклад коду з додаванням токенів

Згодом підготовлені дані розділяються на два набори: один для навчання, а інший для перевірки. Навчальний набір використовується для точного налаштування моделі. Він налаштовує внутрішні параметри та ваги моделі, щоб відобразити шаблони та структури підозрілої мови, представлені в навчальних даних. Перевірочний набір, однак, має вирішальне значення, оскільки він дає змогу оцінити роботу моделі на даних, яких вона не бачила під час навчання, гарантуючи, що модель добре узагальнює, а не просто запам'ятовує вхідні дані.

Після того, як модель точно налаштована на вхідних даних, розгорнеться в контрольованому середовищі для моніторингу її продуктивності. Цей етап моніторингу є життєво важливим для налаштування та коригування моделі на основі її точності, достовірності та здатності виявляти підозрілі дії в тексті.

Кінцевою метою є створення моделі, яка розпізнає не лише явні заяви про зловмисні наміри, а й більш тонкі підказки та шаблони, які можуть вказувати на ризик. Наприклад, модель може дізнатися, що певні фрази, які часто використовуються у фішингових листах, такі як "потрібні термінові дії" або "негайно перевірте свій обліковий запис", є сильними індикаторами підозрілого контенту.

```
!nvidia-smi
```

```
Fri Jan 29 23:48:31 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 516.94       Driver Version: 516.94       CUDA Version: 11.7     |
|-------------------------------+----------------------+----------------------+
| GPU  Name            TCC/WDDM | Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  NVIDIA GeForce ... WDDM  | 00000000:01:00.0 Off |                  N/A |
| 41%   58C    P0    67W / 200W |    483MiB /  8192MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A      2520    C+G   ...lPanel\SystemSettings.exe    N/A      |
|    0   N/A  N/A      3056    C+G   ...e\PhoneExperienceHost.exe    N/A      |
|    0   N/A  N/A      7284    C+G   C:\Windows\explorer.exe         N/A      |
|    0   N/A  N/A      8860    C+G   ...me\Application\chrome.exe    N/A      |
|    0   N/A  N/A      8876    C+G   ...wekyb3d8bbwe\Video.UI.exe    N/A      |
|    0   N/A  N/A      9024    C+G   ...5n1h2txyewy\SearchApp.exe    N/A      |
|    0   N/A  N/A      9844    C+G   ...perience\NVIDIA Share.exe    N/A      |
|    0   N/A  N/A     10504    C+G   ...cw5n1h2txyewy\LockApp.exe    N/A      |
|    0   N/A  N/A     11820    C+G   ...2txyewy\TextInputHost.exe    N/A      |
|    0   N/A  N/A     12652    C+G   ...sv5v3m8wq0b2\ExcelApp.exe    N/A      |
|    0   N/A  N/A     12912    C+G   ...m Files\iTunes\iTunes.exe    N/A      |
|    0   N/A  N/A     13020    C+G   ...y\ShellExperienceHost.exe    N/A      |
|    0   N/A  N/A     14760    C+G   ...8wekyb3d8bbwe\Cortana.exe    N/A      |
|    0   N/A  N/A     15212    C+G   ...ck\app-4.29.149\slack.exe    N/A      |
|    0   N/A  N/A     16220    C+G   ...zpdnekdrzrea0\Spotify.exe    N/A      |
|    0   N/A  N/A     16712    C+G   ...5n1h2txyewy\SearchApp.exe    N/A      |
|    0   N/A  N/A     17004    C+G   ...lls\wgc_renderer_host.exe    N/A      |
+-----------------------------------------------------------------------------+
```

Рисунок 3.4 – Відображення інформації про пристрої NVIDIA GPU

Загалом, ретельна підготовка вхідних даних та їх постійна перевірка відіграють ключову роль у процесі тонкого налаштування, що призводить до створення надійної мовної моделі, яка здатна винюхувати небезпеки в Інтернеті, приховані в тонкощах людського спілкування. Цей процес, проілюстрований у роботі, підкреслює поєднання всебічної підготовки даних зі складними методами моделювання ШІ для вирішення постійно зростаючих викликів кібербезпеки.

3.2 Короткий опис програмного забезпечення

Середовище програмування, структуроване на основі Python та Google Colab, пропонує динамічну платформу для розробки моделі NLP для виявлення підозрілих дій. Розгалужена екосистема Python використовується в повній мірі, включаючи такі бібліотеки, як Pandas та NumPy для маніпулювання та обробки даних, а також PyTorch для побудови та навчання моделей глибокого навчання.

Вибір Google Colab надає додаткову перевагу хмарного середовища з доступом до високопродуктивних обчислювальних ресурсів, таких як графічні процесори, що має вирішальне значення для обробки великих моделей, таких як GPT-2 і GPT-3.

```
from transformers import GPT2LMHeadModel, GPT2Tokenizer
import torch
from torch.utils.data import Dataset # this is the pytorch class import
from transformers import DataCollatorForLanguageModeling
from transformers import TrainingArguments, Trainer
from transformers import StoppingCriteria, StoppingCriteriaList
```

Рисунок 3.5 – Імпортування бібліотек

У цьому налаштуванні бібліотека transformers відіграє ключову роль. Вона надає модель GPT-2 і токенізатор, які є фундаментальними в обробці нюансів природної мови, дозволяючи моделі генерувати і розуміти складні текстові структури. Використовуючи попередньо навчену модель, та використовує величезний обсяг знань, якого вона вже досягла, а потім допрацьовуєте її з урахуванням конкретного набору даних, щоб адаптувати її розуміння до розпізнавання підозрілих шаблонів у тексті.

Отриманий і очищений набір даних перетворюється на формат, який модель може перетравити за допомогою токенізатора. Він інкапсулює текст у токени, які модель може обробити, і ці токени потім збираються разом за допомогою збирача даних. Збирач даних для лінгвістичного моделювання спеціально розроблений для завдань NLP, враховуючи тонкощі заповнення і масок уваги, забезпечуючи отримання добре структурованих і узгоджених пакетів даних для навчання моделі.

Класи TrainingArguments і Trainer з бібліотеки `transformers` організовують процес навчання. Вони дозволяють визначати параметри навчання та керувати циклом навчання відповідно. Об'єкт "Тренер" є високорівневим інтерфейсом, який абстрагується від низькорівневих деталей запуску тренувальних епох, збереження контрольних точок та ведення журналу, роблячи процес навчання більш зручним для користувача.

```
training_args = TrainingArguments(
    output_dir=f'{my_path}Checkouts', #The output directory
    overwrite_output_dir = True, #overwrite the content of the output directory
    num_train_epochs = 10, # number of training epochs
    per_device_train_batch_size = 3, # batch size for training
    per_device_eval_batch_size = 3,  # batch size for evaluation
    warmup_steps = 100,# number of warmup steps for learning rate scheduler
    gradient_accumulation_steps = 1, # to make "virtual" batch size larger
    save_steps = 3000
    )


trainer = Trainer(
    model=model,
    args=training_args,
    data_collator=data_collator,
    train_dataset=train_dataset,
    optimizers = (torch.optim.AdamW(model.parameters(),lr=1e-5),None) # Optimizer and lr scheduler
)
```

Рисунок 3.6 – Класи TrainingArguments і Trainer в коді

Навчання моделі, кероване цими інструментами, є не одноразовим процесом, а циклом вдосконалення. Клас `StoppingCriteria` гарантує, що модель не буде перенастроюватися або тренуватися за межами точки зменшення прибутковості, визначаючи конкретну умову, коли процес навчання повинен зупинитися.

```
class KeywordsStoppingCriteria(StoppingCriteria):
    def __init__(self, keywords_ids:list):
      self.keywords = keywords_ids

    def __call__(self, input_ids: torch.LongTensor, scores: torch.FloatTensor, **kwargs) -> bool:
      if input_ids[0][-1] in self.keywords:
        print(input_ids)
        return True
      return False
```

```
stop_criteria = KeywordsStoppingCriteria(tokenizer.encode(tokenizer.eos_token, return_tensors="pt").to(DEVICE))
```

Рисунок 3.7 – Класи StoppingCriteria в коді

Nvidia-smi ця команда надає потужний інструмент для відстеження завантаження GPU, що гарантує ефективне використання обчислювальних ресурсів під час навчального процесу.

Кожен компонент у програмному стеку робить свій внесок у комплексну систему, спрямовану на амбітне завдання автоматизації виявлення зловмисної поведінки в Інтернеті. Використовуючи Python в якості ядра, Google Colab в якості платформи та бібліотеку `transformers` в якості основного інструментарію, створили складне середовище, здатне впоратися зі складнощами мовного моделювання. Взаємодія між цими компонентами є свідченням передового стану технології NLP та її застосовності у вирішенні реальних проблем, таких як

кібербезпека.

У контексті навчання моделей машинного навчання, особливо в обробці природної мови (NLP), збирач даних - це компонент, який відповідає за підготовку пакетів даних для завантаження в модель для навчання або оцінювання. Коллайдер забезпечує правильне форматування та пакетування даних, тобто групує кілька навчальних прикладів в один пакет, одночасно дбаючи про заповнення, маскування та будь-які інші необхідні кроки попередньої обробки.

Data Collator має вирішальне значення при роботі з вхідними даними змінної довжини, що часто трапляється з текстовими даними. Різні тексти мають різну довжину, і для їхнього пакетного оброблення зазвичай потрібно вставляти коротші тексти відповідно до довжини найдовшого тексту в пакеті. Цей процес полегшує ефективну паралельну обробку на такому обладнанні, як графічні процесори.

Наприклад, DataCollatorForLanguageModeling часто використовується з трансформантними моделями, такими як BERT або GPT, де він не лише обробляє заповнення, але й динамічно маскує токени для завдання моделювання замаскованої мови, що є важливим для навчання таких моделей.

Collator є важливою частиною конвеєра навчання NLP-моделі, гарантуючи, що сирі текстові дані будуть правильно попередньо оброблені і перетворені у формат, який може обробити модель, що є критично важливим кроком на шляху до побудови точної і надійної мовної моделі.

3.3 Результати експериментальних досліджень

Завдання моделі полягає в тому, щоб ефективно розрізняти нормальний і підозрілий вміст. Результати цих тестів зазвичай вимірюються за допомогою різних метрик, включаючи точність, достовірність, пригадування і показник F1, щоб надати повну картину ефективності моделі.

Наприклад, п'ять повідомлень з потенційно шкідливим вмістом перевіряють здатність моделі правильно позначати і класифікувати повідомлення як підозрілі. Водночас п'ять нешкідливих повідомлень слугують

для перевірки того, що модель не дає хибних спрацьовувань.

Ось 10 прикладів повідомлень для тестування програми.

Потенційно шкідливе:

1) Привіт, я знайшов спосіб подвоїти ваші інвестиції за тиждень. Напишіть мені, щоб дізнатися подробиці;

2) Перейдіть за цим посиланням, щоб отримати безкоштовну ексклюзивну подарункову картку прямо зараз!;

3) Терміново: Ваш акаунт було скомпрометовано. Надайте свої дані для підтвердження особи;

4) Ви отримали спадок від далекого родича. Надішліть свої банківські реквізити, щоб його отримати;

5) Це податкова. Ви заборгували штраф. Необхідно негайно сплатити, щоб уникнути судового позову, посилання прикріплюємо.

Не зловмисні:

1) Давай зустрінемося за кавою наступного тижня. Як щодо понеділка?;

2) З днем народження, Марк! Бажаю тобі фантастичного дня!;

3) Нагадую: завтра о 10 ранку в головному конференц-залі відбудуться збори команди;

4) Мені сподобалася книга, яку ви порекомендували. Давай наступного разу обміняємося улюбленими книжками;

5) Дедлайн проекту продовжено до кінця місяця на прохання клієнта.

```
inp = input('Введіть текст або просто натисніть Enter:')
```

```
Введіть текст або просто натисніть Enter:Привіт, я знайшов спосіб по
```

Рисунок 3.8 – Від даних для тесту

Для початку водимо, потрібні дані в вікно та завантажуємо їх для перевірки. Після того як дані завантажились він буде продубльований як показано на Рис. 3.8.

```
[2]  inp = input('Введіть текст або просто натисніть Enter:')

     Введіть текст або просто натисніть Enter:Привіт, я знайшов спосіб подвоїти ваші інвестиції за тиждень. Напишіть мені, щоб дізнатися подробиці
```

Рисунок 3.9 – Ведені дані котрі готові до аналізу

Пісня всіх дій проводимо останню дію а саме перевірку, яка показала позитивний результат а саме виявило мотив фішингу іншими словами обману з подальшої крадіжки даних або коштів.

```
with torch.no_grad():
      out = model.generate(input_ids,
                           do_sample=True,
                           num_beams=3,
                           temperature=2.0,
                           top_p=0.9,
                           max_length = 200,
                           stopping_criteria=StoppingCriteriaList([stop_criteria]),
                           eos_token_id=tokenizer.eos_token_id,
                           bos_token_id=tokenizer.bos_token_id,
                           ).to(DEVICE)
print(tokenizer.batch_decode(out, skip_special_tokens=False)[0])
```

```
Setting `pad_token_id` to `eos_token_id`:50258 for open-end generation.
<bos> Фішинг <sep>Привіт, я знайшов спосіб подвоїти ваші інвестиції за тиждень. Напишіть мені, щоб дізнатися подробиці .  <eos>
```

Рисунок 3.10 – Приклад успішної роботи та виявлення

Ці повідомлення можна використовувати, щоб оцінити, чи програма ідентифікує та класифікує повідомлення з підозрілим вмістом, коректно ігноруючи при цьому безпечні повідомлення. Для цього потрібно провести декілька тестів, а саме з 10 повідомленнями.

```
inp = input('Введіть текст або просто натисніть Enter:')
```

```
Введіть текст або просто натисніть Enter:Нагадую: завтра о 10 ранку в головному конференц-залі відбудуться збори команди.
```

```
inp = inp if len(inp) > 0 else tokenizer.bos_token #токен
input_ids = tokenizer.encode(inp, return_tensors="pt").to(DEVICE)
```

```
with torch.no_grad():
      out = model.generate(input_ids,
                           do_sample=True,
                           num_beams=3,
                           temperature=2.0,
                           top_p=0.9,
                           max_length = 200,
                           stopping_criteria=StoppingCriteriaList([stop_criteria]),
                           eos_token_id=tokenizer.eos_token_id,
                           bos_token_id=tokenizer.bos_token_id,
                           ).to(DEVICE)
print(tokenizer.batch_decode(out, skip_special_tokens=False)[0])
```

```
Setting `pad_token_id` to `eos_token_id`:50258 for open-end generation.
<bos>Не знайдено<sep>Нагадую: завтра о 10 ранку в головному конференц-залі відбудуться збори команди. .  <eos>
```

Рисунок 3.11 – Приклад успішної роботи не виявлення

Далі проводимо тест на для всіх питань котрі ще не використовувались, та спостерігаємо як поведе себе програма з декількома рядками та як виявить

підозрілий вміст.

```
# inp = input('Введите начало заголовка или просто нажмите Enter:')
print("Введіть текст. Двічі натисніть Enter, щоб зупинитися.")
inp = user_input = []
while True:
    line = input()
    if line:
        user_input.append(line)
    else:
        break
full_input = '\n'.join(user_input)
print("You entered:")
print(full_input)
```

```
Введіть текст. Двічі натисніть Enter, щоб зупинитися.
Перейдіть за цим посиланням, щоб отримати безкоштовну ексклюзивну подарункову картку прямо зараз!
З днем народження, Марк! Бажаю тобі фантастичного дня!
Нагадую: завтра о 10 ранку в головному конференц-залі відбудуться збори команди.
Це податкова. Ви заборгували штраф. Необхідно негайно сплатити, щоб уникнути судового позову, посилання прикріплюємо.
Дедлайн проекту продовжено до кінця місяця на прохання клієнта.
Мені сподобалася книга, яку ви порекомендували. Давай наступного разу обміняємося улюбленими книжками
Ви отримали спадок від далекого родича. Надішліть свої банківські реквізити, щоб його отримати
Терміново: Ваш акаунт було скомпрометовано. Надайте свої дані для підтвердження особи

You entered:
Перейдіть за цим посиланням, щоб отримати безкоштовну ексклюзивну подарункову картку прямо зараз!
З днем народження, Марк! Бажаю тобі фантастичного дня!
Нагадую: завтра о 10 ранку в головному конференц-залі відбудуться збори команди.
Це податкова. Ви заборгували штраф. Необхідно негайно сплатити, щоб уникнути судового позову, посилання прикріплюємо.
Дедлайн проекту продовжено до кінця місяця на прохання клієнта.
Мені сподобалася книга, яку ви порекомендували. Давай наступного разу обміняємося улюбленими книжками
Ви отримали спадок від далекого родича. Надішліть свої банківські реквізити, щоб його отримати
Терміново: Ваш акаунт було скомпрометовано. Надайте свої дані для підтвердження особи
```

Рисунок 3.12 – Ведені дані

Спостерігаємо успішну роботу програми, а саме вдале виявлення підозрілої активності та специфікацію загроз. Також не було виявлено ніяких загроз в звичайному тексті який був доданий в довільному порядку з підозрілим, але стоїть жорстка класифікація контра не завжди являється вірною.

```
                        ).to(DEVICE)
print(tokenizer.batch_decode(out, skip_special_tokens=False)[0])
```

```
Setting `pad_token_id` to `eos_token_id`:50258 for open-end generation.
<bos> Фішинг <sep> Перейдіть за цим посиланням, щоб отримати безкоштовну ексклюзивну подарункову картку прямо зараз!.  <eos>
<bos> Не знайдено <sep> З днем народження, Марк! Бажаю тобі фантастичного дня!.  <eos>
<bos> Не знайдено <sep> Нагадую: завтра о 10 ранку в головному конференц-залі відбудуться збори команди..  <eos>
<bos> Імітація <sep> Це податкова. Ви заборгували штраф. Необхідно негайно сплатити, щоб уникнути судового позову, посилання прикріплюємо..  <eos>
<bos> Не знайдено <sep> Дедлайн проекту продовжено до кінця місяця на прохання клієнта..  <eos>
<bos> Не знайдено <sep> Мені сподобалася книга, яку ви порекомендували. Давай наступного разу обміняємося улюбленими книжками..  <eos>
<bos> Шахрайство з спадком <sep> Ви отримали спадок від далекого родича. Надішліть свої банківські реквізити, щоб його отримати..  <eos>
<bos> Крадіжка особистих даних <sep> Терміново: Ваш акаунт було скомпрометовано. Надайте свої дані для підтвердження особи.  <eos>
```

Рисунок 3.13 – Успішно пройдена перевірка

Для подальшої перевірки та порівняння ефективності була змінена мовна модель на GPT4All-J v1.3-groovy та LLama 7B. Котрі були протестовані на одному тестовому відрізку даних з 10 тестових питань.

При першій перевірці на основі GPT4All-J v1.3-groovy моделі були відсортовані дані і спочатку видаваються повідомлення котрі не містять підозрілу активність та є безпечними для користувача, після них ідуть повідомлення що уже містять підозрілу мережеву активність, отриманні такі дані зображенні на Рис 3.14.

```
з них не варто фокусуватися.
Давай зустрінемося за кавою наступного тиждень. Як щодо понеділка?; (Не має підозрілої активності)
З днем німеча, Марк! Бажаю тобі фантастичного дня!; (Не має підозрілої активності)
Нагадую: завтра о 10 ранку в головному конференц-залі відбудуться збори команди; (Не має підозрілої активності)
 Мені сподобалася книга, яку ви порекомендували. Давай наступного разу обміняємося улюбленими книжками; (Не має підозрілої активності)
Дедлайн проекту продовжено до кінця місяця на прохання клієнта. (Не має підозрілої активності)
З повідомлень, що містять підозрілу активність, є:
Привіт, я знайшов спосіб подвоїти ваші інвестиції за тиждень. Напишіть мені, щоб дізнатися подробиці; (Підозріла пропозиція про інвестиції)
Перейдіть за цим посиланням, щоб отримати безкоштовну ексклюзивну подарункову картку прямо зараз!; (Спроба підження, використою посилання на непряму адресу)
Терміново: Ваш акаунт було скомпрометовано. Надайте свої дані для підтвердження особи; (Підозріла повідомлення про скомпрометування акаунту)
Ви отримали спадок від далекого родича. Надішліть свої банківські реквізити, щоб його отримати; (Підозріла пропозиція про отримання спадоку)
Це податкова. Ви заборгували штраф. Необхідно негайно сплатити, щоб уникнути судового позову, посилання прикріплюємо. (Підозріла повідомлення про заборгування штрафу)
```

Рисунок 3.14 – Перевірка за допомогою GPT4All-J v1.3-groovy

Особливістю цієї перевірки став автоматичний опис кожної підозрілої активності та автоматичний поділ на дві секції.

```
/

1) Запрошення подвоїти інвестиції за тиждень, яке може бути шахрайством або фінансовою пірамідою. Надалі слід обережності та перевірки надійності джерела.
2) Посилання на ексклюзивну подарункову картку, яке може бути початком шахрайства з даними користувача. Не вдавайтеся до таких пропозицій без перевірки їх авторефералу або офіційного сайту банку.
3) Повідомлення про скомпрометоване облікове записи, яке може бути початком фази фішинг. Не надавайте свої дані в таких ситуаціях без перевірки джерела.
4) Звістка про спадок від родича, який може бути шахрайством з метою отримання банківських реквізитів користувачів. Завжди перевіряйте подібні листи та не надавайте свої дані без офіційної документації.
5) Повідомлення про податкову заборгованість, яке може бути шахрайством з метою отримання грошових коштів. Переконайтеся в реальності подібних ситуацій та не сплачуйте гроші без офіційної документації.
```

Рисунок 3.15 – Перевірка за допомогою LLama 7B

Остання перевірка буде здійснена за допомогою LLama 7B моделі, її особливості полягають в тому що вона автоматично видаляє повідомлення котрі не несуть загрози та коротко їх описує, але не класифікує дані Рис.3.15.

При малій кількісті даних правильний відсоток у кожної моделі 100%, але щоб сказати що це вірно, чи хибне твердження потрібно збільшити вхідні дані для тесту моделі, кількість даних збільшити в 10 разів, а саме до 100 повідомлень.

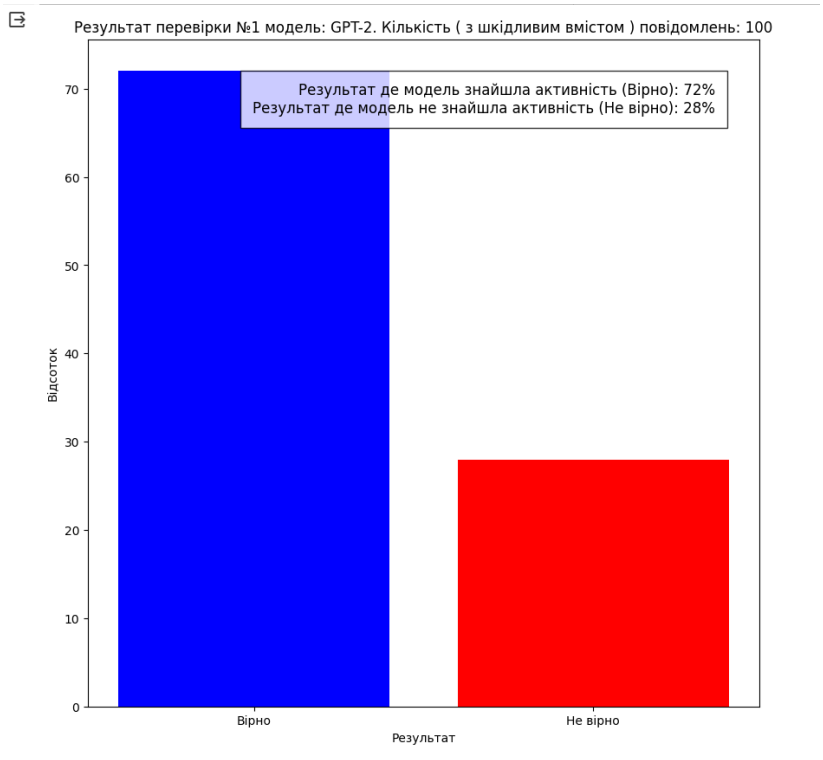


Рисунок 3.16 – Результати першої перевірки моделі GPT-2

Після збільшення даних в 10 разів, результати змінились, а саме результат показав що зі 100 повідомлень з шкідливим вмістом модель виявила тільки 72, а 28 загроз не виявила Рис.3.16. для більшої зручності та аналізу дані переведені в графічний вигляд.

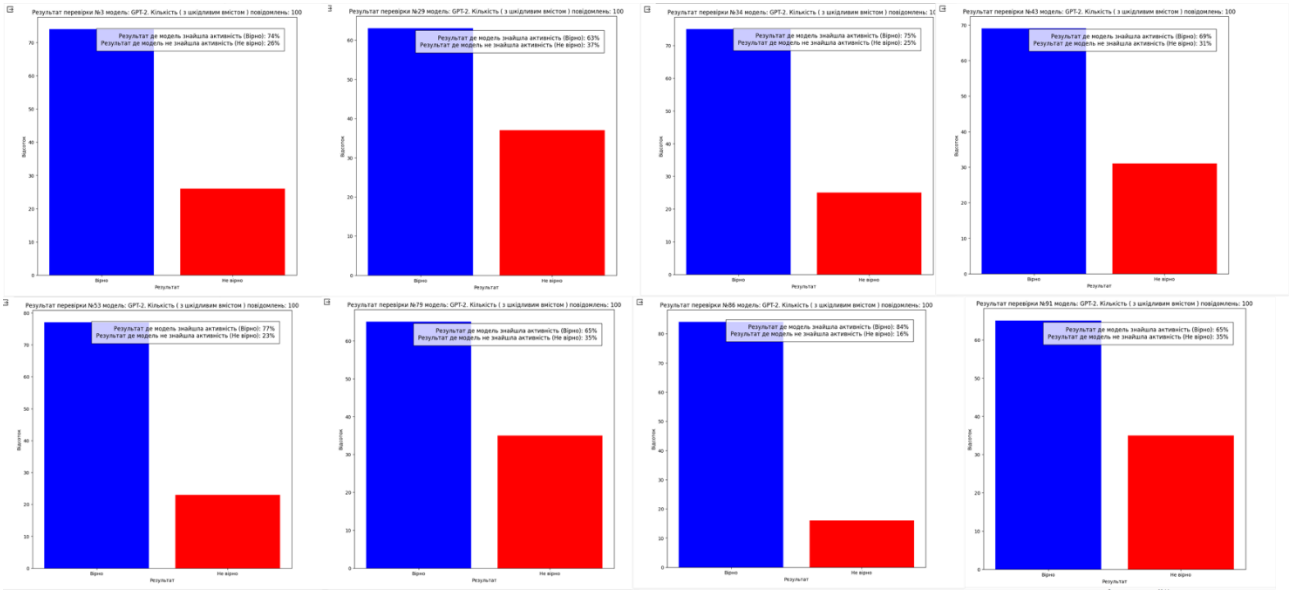


Рисунок 3.17 – Загальні результати перевірки за допомогою GPT-2

Щоб забезпечити точну оцінку ефективності, потрібно провести більше тестів, а саме 100 перевірок, це по 10000 повідомлень на кожну модель. Ця більш глибока перевірка має показати більш детальні дані та загальні інформацію.

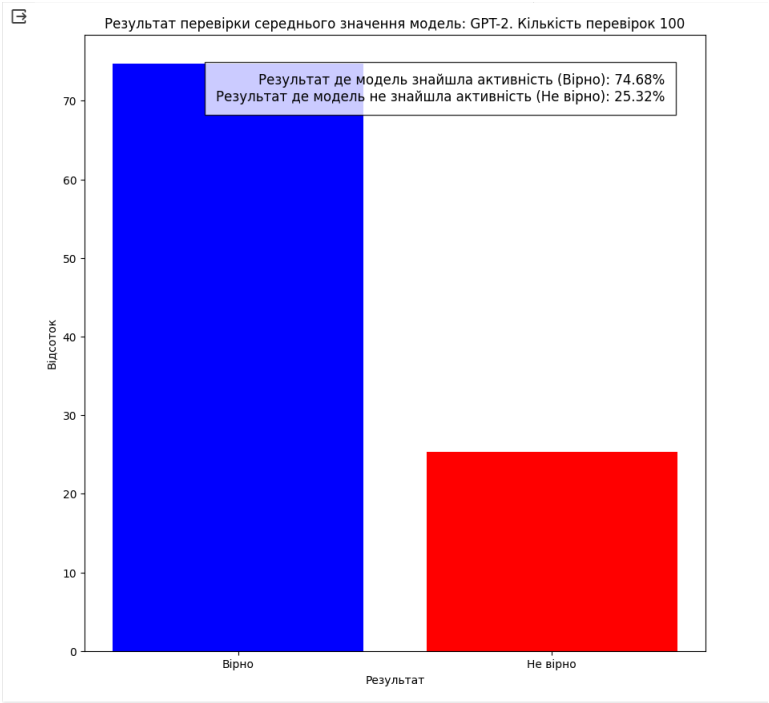


Рисунок 3.18 – Результати перевірки за допомогою GPT-2

Після всіх перевірок результати показали, що в середньому в 74,68% випадків модель GPT-2 показувала правильну відповідь, а в 25,32 повідомлення зі шкідливим вмістом були класифіковані як безпечні.

Після отриманих результатів змінюємо модель на LLama 7B та проводимо аналогічні тести, щоб зібрати дані для фінального порівняння моделей.

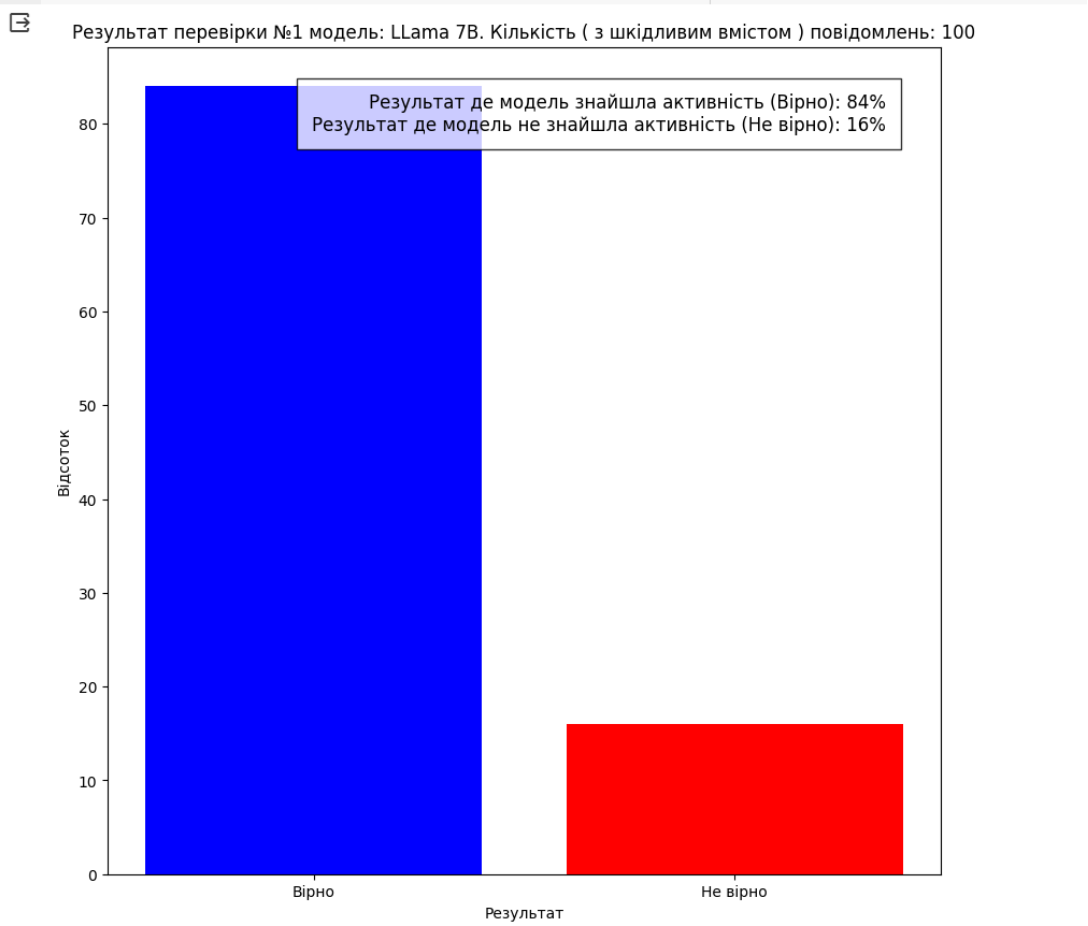


Рисунок 3.19 – Результати першої перевірки моделі LLama 7B

Після першої перевірки дані показують, що результати перевершують попередню модель GPT-2, а саме з пустуна 72% правильних відповідей підвищилися до 84%, а з 28% не вірних зменшилися до 16%.

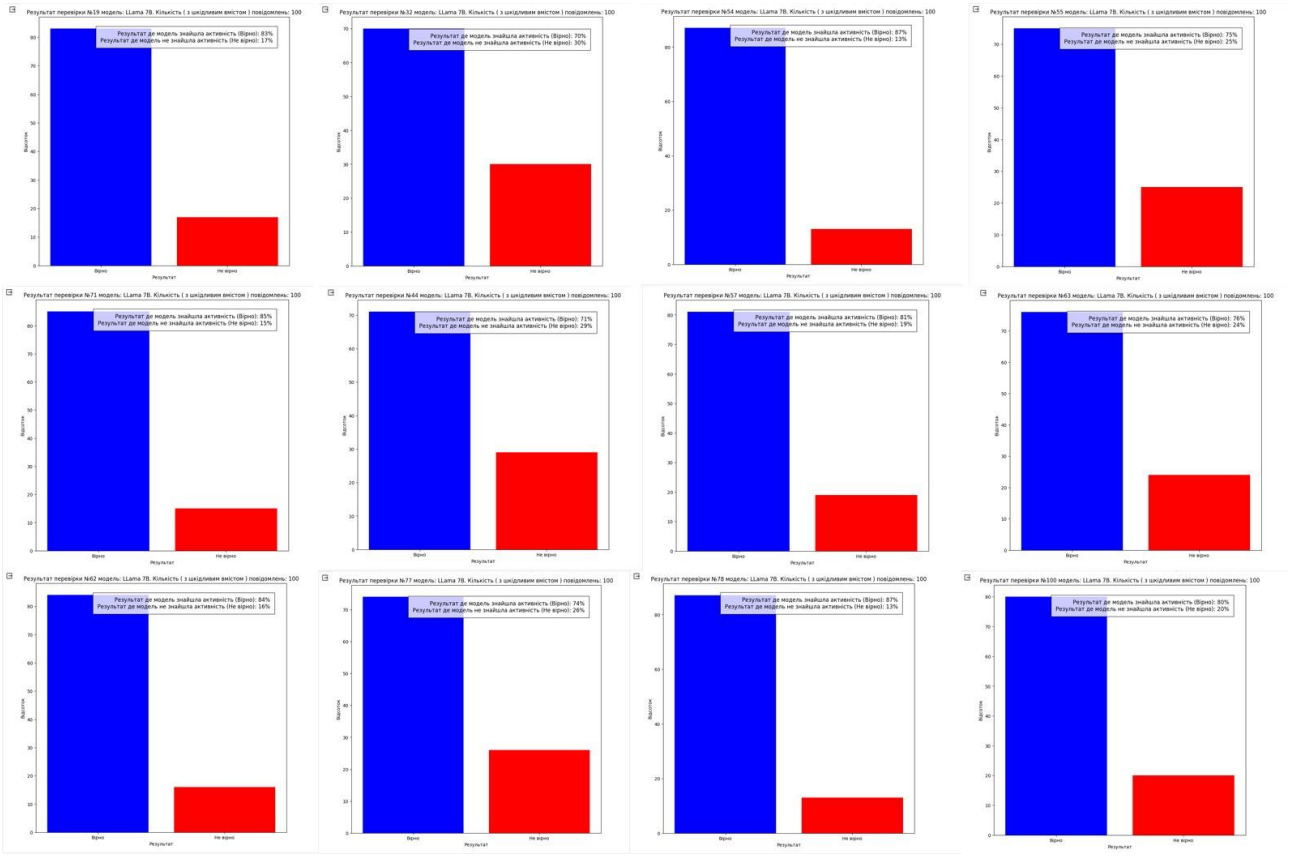


Рисунок 3.20 – Загальні результати перевірки за допомогою LLama 7B

При порівнянні з попередньою моделлю GPT-2 у графіків моделі LLama 7B мають в середньому більші значення, та більш плавні з меншим розривом у відсотках графіки, для більш детальної інформації потрібно порівнювати з повним збором даних.

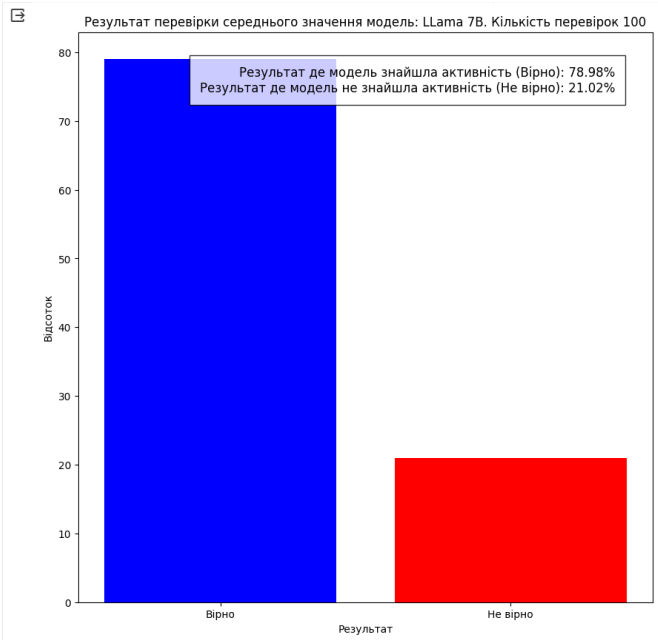


Рисунок 3.21 – Результати перевірки за допомогою LLama 7B

Після збільшення обсягу даних в 10 разів LLama 7B показала зміни у виявленні шкідливого вмісту. З 100 повідомлень зі шкідливим вмістом модель виявила 78.92% тоді як 21.02% загрози не були виявлені. Графічне представлення цих даних можна побачити на Рис.3.21.

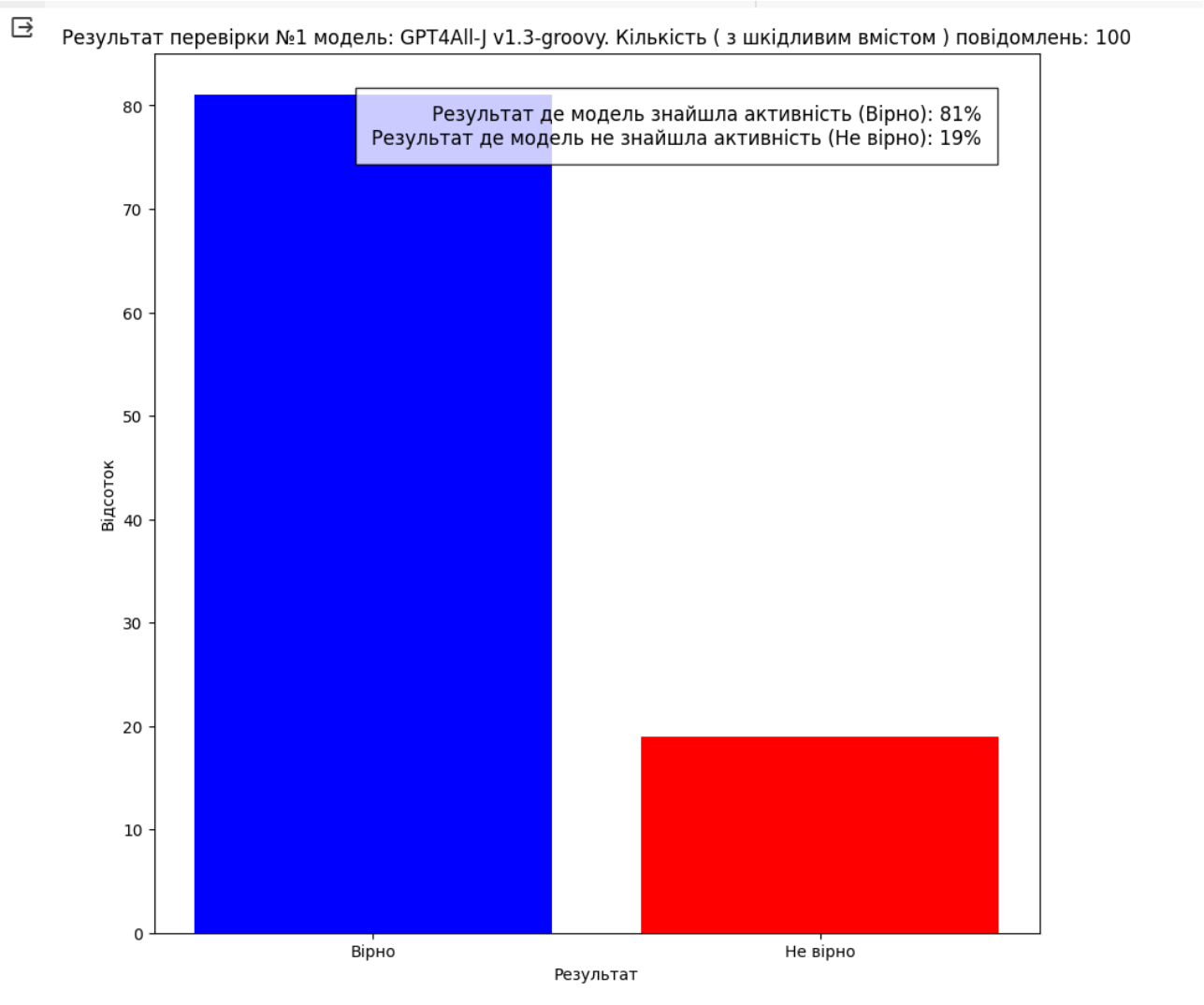


Рисунок 3.22 – Результати першої перевірки моделі GPT4All-J v1.3-groovy

Після першої перевірки модель GPT4All-J v1.3-groovy показала найкращі результати, а саме що ця модель демонструє точність на рівні 81.%. У 19% випадків повідомлення зі шкідливим вмістом були помилково класифіковані як безпечні.

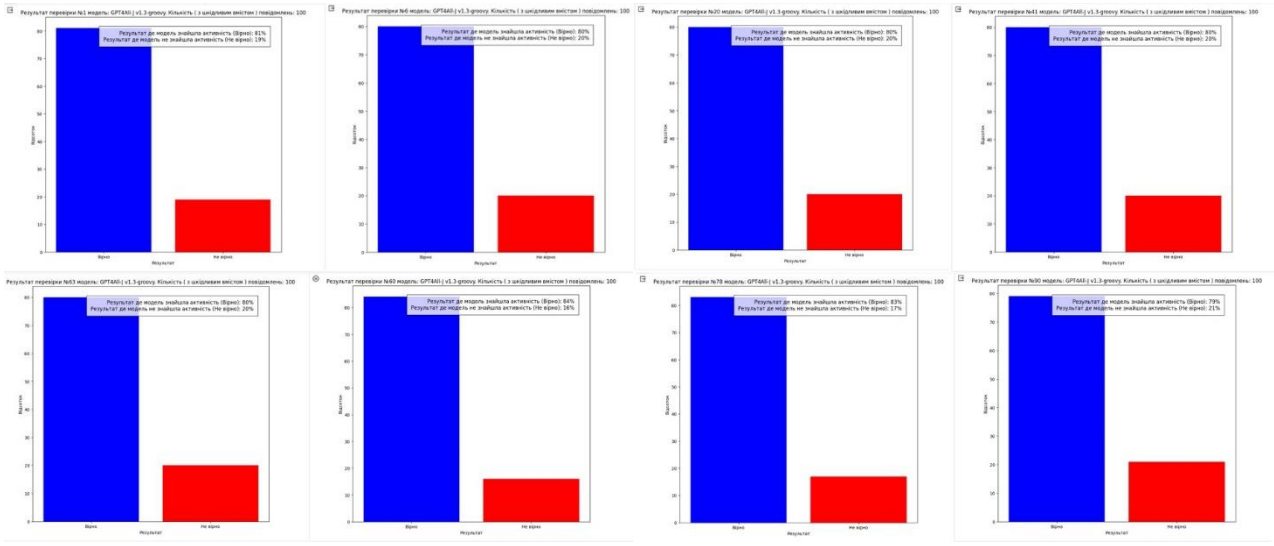


Рисунок 3.23 – Загальні результати перевірки за допомогою GPT4All-J v1.3-groovy

При загальному порівняні модель GPT4All-J v1.3-groovy показує мінімальний процент відхилення та набагато кращі результати ніж і інші моделі, але потрібно і враховувати що це і сама велика модель котра і потребує великих ресурсів на її роботу, в порівнянні з попередніми моделями.

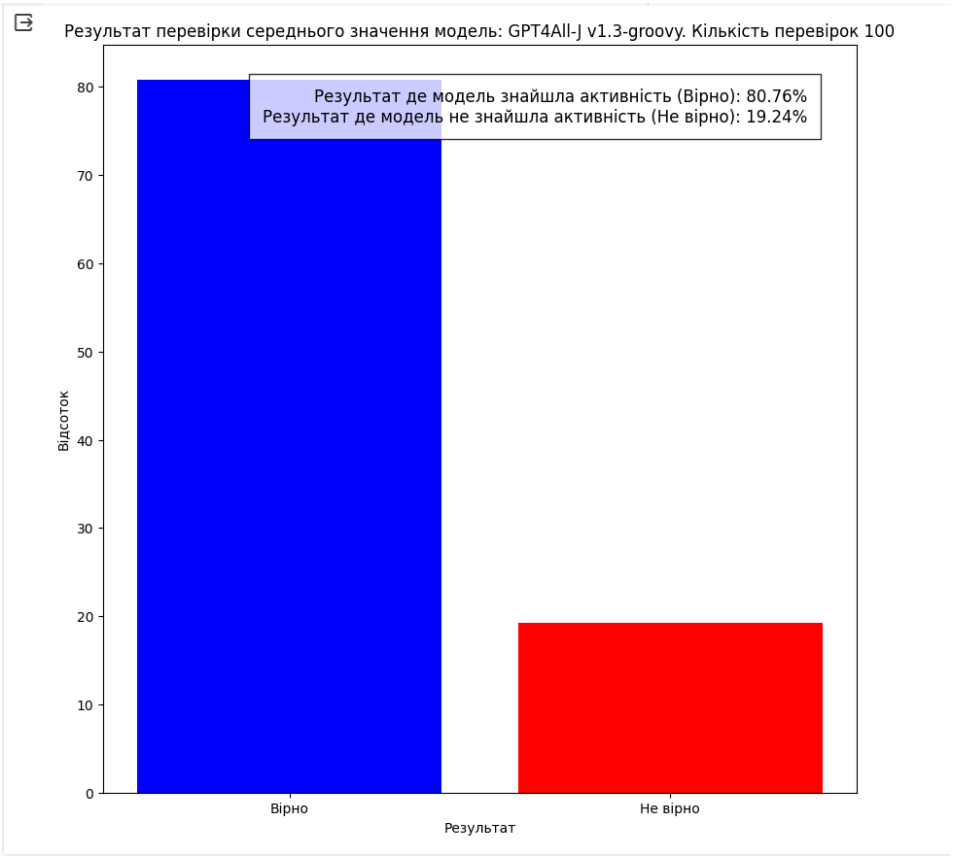


Рисунок 3.24 – Результати перевірки за допомогою GPT4All-J v1.3-groovy

Щодо моделі GPT4All-J v1.3-groovy, після проведення 100 перевірок з кожною моделлю на по 10 000 повідомлень, в середньому виявлено, що ця модель демонструє точність на рівні 80.76%. У 19.24% випадків повідомлення зі шкідливим вмістом були помилково класифіковані як безпечні.

Закон ансамблювання стверджує, що якщо кожен член ансамблю моделей має точність, яка перевищує 50%, то комбінування їх прогнозів може призвести до покращення результатів через диверсифікацію та зниження помилок.

Принцип роботи полягає у тому, що кожна модель вносить свій внесок у прогнозування, індивідуально виділяючи свої сильні сторони та компенсуючи слабкі. Наприклад, у випадку голосування, кінцеве рішення приймається на основі більшості голосів моделей.

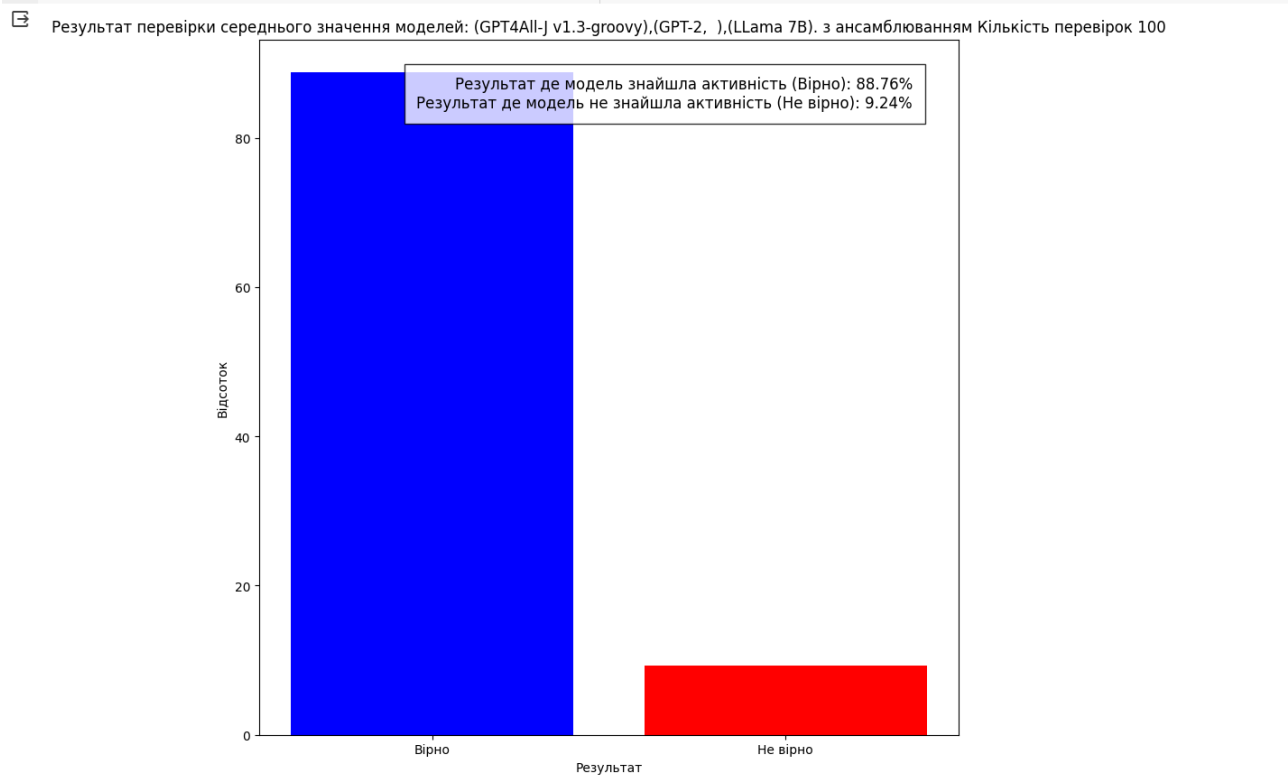


Рисунок 3.25 – Результати перевірки за допомогою ансамблювання

Підбір оптимальних моделей для ансамблювання та їх комбінування може здійснюватися шляхом тестування різних архітектур або методів комбінації результатів. Такий підхід може підвищити загальну точність системи через використання різних переваг кожної окремої моделі.

Розглянемо ключові відмінності між моделями GPT-2, LLama 7B і GPT4All-J v1.3-groovy:

Розмір та потужність моделі:

1. GPT-2: Має близько 1.5 мільярда параметрів. Це відносно менший розмір порівняно з іншими моделями, що може впливати на його здатність у розумінні складних контекстів;

2. LLama 7B: Ця модель має близько 7 мільярдів параметрів, що робить її значно потужнішою за GPT-2. Більший розмір дозволяє краще розуміння мови та глибше контекстуалізувати інформацію;

3. GPT4All-J v1.3-groovy: Ця модель ще більша за LLama 7B і, ймовірно, має декілька десятків мільярдів параметрів. Такий обсяг дозволяє їй враховувати значно більше даних та зв'язків між ними.

Якість генерації тексту:

1. GPT-2: Добре справляється з генерацією тексту, але через обмежений розмір може мати проблеми з докладністю та контекстом;

2. LLama 7B: Має високу якість генерації тексту та краще розуміння мови, що дозволяє створювати більш якісний контент;

3. GPT4All-J v1.3-groovy: Через величезну кількість параметрів може забезпечувати ще більш точний контекст для генерації тексту та відповідей.

Швидкість та доступність:

1. GPT-2: Зазвичай швидкий у роботі та доступний для використання через широку підтримку;

2. LLama 7B: Може бути повільнішим через більший розмір, але його якість генерації тексту виправдовує це;

3. GPT4All-J v1.3-groovy: Через величезну кількість параметрів може бути повільнішим у роботі та вимагати більше ресурсів;

Основні висновки з проведених тестів з моделями GPT-2, LLama 7B та GPT4All-J v1.3-groovy можна узагальнити наступним чином:

Ефективність відповідно до розміру даних:

- Початкові тести з невеликим обсягом даних показали 100% точність для кожної моделі, проте ці показники не відображають реальну ефективність, оскільки базуються на обмеженій кількості вхідних даних;
- Збільшення обсягу даних у 10 разів для кожної моделі спричинило

значні зміни у виявленні шкідливого вмісту, знизивши точність.

Результати тестів збільшеного обсягу даних:

- Після проведення додаткових 100 перевірок з 10 000 повідомлень на кожну модель, GPT-2 показала точність на рівні 74.68%. У цих тестах 25.32% повідомлень були помилково класифіковані як безпечні;
- LLama 7B виявила покращення у виявленні шкідливого вмісту, підвищивши точність до 84%, зменшивши помилкові класифікації до 16%;
- GPT4All-J v1.3-groovy показала найкращі результати серед усіх моделей з точністю на рівні 81%. У 19% випадків повідомлення зі шкідливим вмістом були помилково класифіковані.

Аналіз висновків:

- Модель GPT4All-J v1.3-groovy продемонструвала найвищий рівень точності, проте її великий обсяг та потреба у великих ресурсах слід враховувати при подальшому використанні;
- LLama 7B, хоча й показала значне покращення порівняно з GPT-2, все ще вимагає подальшої перевірки та оцінки на більшому обсязі даних для точної оцінки її ефективності;
- Результати вказують на необхідність більш глибокого аналізу та порівняння моделей на більшому обсязі даних для прийняття фінального рішення щодо найбільш ефективної моделі для використання в конкретному контексті.

Кожна модель має свої переваги та обмеження. GPT-2 - хороший вибір для загальних завдань. LLama 7B - забезпечує вищу точність та якість. GPT4All-J v1.3-groovy - надає ще більше деталізації та контексту, але може бути вимогливим у використанні. Отже, обираючи між ними, важливо враховувати потреби у конкретних задачах.

3.4 Подальший розвиток програмного продукту

Подальший розвиток програмного продукту спрямований на розширення його функціональності та залучення користувачів. Одним з таких потенційних напрямків розвитку є інтеграція моделі в Telegram-бот. Цей бот міг би

моніторити чат-групи або прямі повідомлення, аналізуючи розмови в режимі реального часу на предмет підозрілого контенту. Якщо він виявить потенційні загрози або сумнівний контент, він може зберігати повідомлення або посилання для подальшого вивчення.

Крім того, можна реалізувати функцію ручної перевірки, коли користувачі можуть надсилати текст боту для миттєвого аналізу. Якщо модель позначить контент як підозрілий, бот може попередити користувача або модератора про необхідність вжити додаткових заходів.

Ця інтерактивна система може слугувати потужним інструментом для менеджерів спільнот, команд безпеки та окремих користувачів, забезпечуючи рівень безпеки та модерації, який сьогодні стає все більш необхідним у цифрових комунікаційних просторах. Завдяки постійному оновленню та зворотному зв'язку з користувачами, модель може бути навчена, щоб стати більш точною та ефективною у виявленні загроз.

ВИСНОВКИ

У цій дослідницькій роботі були аналізовані існуючі мовні моделі для виявлення підозрілої активності в мережі, та розроблена власна модель, що продемонструвала високу ефективність у цій задачі. Використання передових інформаційних технологій та алгоритмів глибокого навчання відіграло ключову роль у реалізації цього проекту. Основними висновками роботи є підтвердження того, що сучасні мовні моделі можуть ефективно аналізувати та ідентифікувати підозрілу мережеву активність, забезпечуючи значний вклад у покращення засобів кібербезпеки.

Робота демонструє, що ретельно розроблена та налаштована мовна модель може адаптуватися до специфіки різних видів комунікацій, що робить її універсальним інструментом у боротьбі з кіберзагрозами. Також розглядається можливість застосування цієї моделі у різних контекстах та середовищах, що розширює її потенційну корисність.

Загалом, це дослідження відкриває нові перспективи для розробки більш вдосконалених систем кібербезпеки, що використовують мовні моделі для забезпечення безпеки в мережі. Таким чином, воно не лише вносить важливий вклад у наукове розуміння цієї проблематики, але й має практичне значення для підвищення рівня захисту в цифровому світі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Transformers [Електронний ресурс] – Режим доступу: https://insights.stackoverflow.com/survey/2021#technology
2. BERT [Електронний ресурс] – Режим доступу: https://huggingface.co/docs/transformers/model_doc/bert
3. rnn-model [Електронний ресурс] – Режим доступу: https://github.com/topics/rnn-model
4. recurrent-neural-network [Електронний ресурс] – Режим доступу: https://github.com/topics/rnn-model
5. recurrent-neural-network [Електронний ресурс] – Режим доступу: https://github.com/topics/recurrent-neural-network
6. Natural Language Processing (NLP) Collective [Електронний ресурс] – Режим доступу: https://stackoverflow.co/labs/nlp-collective/
7. What is NLP and how It is Implemented in Our Lives [Електронний ресурс] – Режим доступу: https://amazinum.com/insights/what-is-nlp-and-how-it-is-implemented-in-our-lives/
8. NLP Collectivehttps [Електронний ресурс] – Режим доступу:://stackoverflow.com/collectives/nlp
9. Large-language-model [Електронний ресурс] – Режим доступу: https://stackoverflow.com/questions/tagged/large-language-model?tab=Newest
10. What are Large Language Models [Електронний ресурс] – Режим доступу: https://machinelearningmastery.com/what-are-large-language-models/
11. What is a large language model (LLM) [Електронний ресурс] – Режим доступу: https://www.cloudflare.com/learning/ai/what-is-large-language-model/
12. TensorRT-LLM [Електронний ресурс] – Режим доступу: https://github.com/NVIDIA/TensorRT-LLM
13. Довідник з мови Python [Електронний ресурс] – Режим доступу: https://docs.python.org/uk/3/reference/index.html
14. Система імпорту [Електронний ресурс] – Режим доступу https://docs.python.org/uk/3/reference/import.html#importlib

15.3 Easy Methods For Improving Your Large Language Model [Електронний ресурс] – Режим доступу https://towardsdatascience.com/3-easy-methods-for-improving-your-large-language-model-68670fde9ffa

16. LLM [Електронний ресурс] – Режим доступу https://python.langchain.com/docs/modules/chains/foundational/llm_chain

17. LLM [Електронний ресурс] – Режим доступу https://llm.datasette.io/en/stable/

18. transformers 4.36.1 [Електронний ресурс] – Режим доступу https://pypi.org/project/transformers/

19. transformers [Електронний ресурс] – Режим доступу https://huggingface.co/docs/transformers/index

20. An Introduction to Using Transformers and Hugging Face [Електронний ресурс] – Режим доступу https://www.datacamp.com/tutorial/an-introduction-to-using-transformers-and-hugging-face

21. Transformer's from scratch in simple python. Part-I [Електронний ресурс] – Режим доступу https://medium.com/@hhpatil001/transformers-from-scratch-in-simple-python-part-i-b290760c1040

22. Transformers : Part 3- Why pytorch and not Tensorflow for Transformers [Електронний ресурс] – Режим доступу https://medium.com/@aneesha161994/transformers-part-3-why-pytorch-and-not-tensorflow-for-transformers-4bbc99a4267

ДОДАТОК

```
import pandas as pd
import numpy as np
!pip install transformers==4.19.0
from transformers import GPT2LMHeadModel, GPT2Tokenizer
import torch
from torch.utils.data import Dataset # this is the pytorch class import
from transformers import DataCollatorForLanguageModeling
from transformers import TrainingArguments, Trainer
from transformers import StoppingCriteria, StoppingCriteriaList
DEVICE = torch.device("cuda") if torch.cuda.is_available() else None
!pip install openpyxl
my_path = '/content/'
!wget https://www.dropbox.com/s/base/base.csv?dl=0 -O /content/news.csv
data = pd.read_csv(f'{my_path} base.csv')
data.drop_duplicates(subset = ['title'],inplace=True)
data.dropna(inplace=True)
def clear_data(text: str):
  if 'Чи є текст підозрілим' in text:
    return np.nan
  if len( text.split(' ')) < 20:
    return np.nan
  return text
data['text'] = data['text'].apply(clear_data)
data.dropna(subset=['text'], inplace=True)
!nvidia-smi
model_name_or_path = " haggingface/gpt3-medium"
tokenizer = GPT2Tokenizer.from_pretrained(model_name_or_path)
model = GPT2LMHeadModel.from_pretrained(model_name_or_path).to(DEVICE)
SPECIAL_TOKENS = {'bos_token':'<bos>','eos_token' :'<eos>',
'pad_token':'<pad>', 'sep_token': '<sep>'}
tokenizer.add_special_tokens(SPECIAL_TOKENS)
model.resize_token_embeddings(len(tokenizer))
torch.manual_seed(42)
class myDataset(Dataset):

  def __init__(self, data, tokenizer, gpt2_type="gpt2", max_length=150):
    self.tokenizer = tokenizer
    self.input_ids = []
    self.attn_masks = []

    for i in data.index.to_list():
      title = data['title'][i]
      description = data['text'][i] if type(data['text'][i]) == str else ''

      encodings_dict = tokenizer('<bos>'+ title + '<sep>' + description +
'<eos>',
                                 truncation=True,
                                 max_length=max_length,
                                 padding="max_length")
```

```
      self.input_ids.append(torch.tensor(encodings_dict['input_ids']))
      self.attn_masks.append(torch.tensor(encodings_dict['attention_mask']))

  def __len__(self):
    return len(self.input_ids)

  def __getitem__(self, idx):
    return {
        'input_ids': self.input_ids[idx],
        'attn_masks': self.attn_masks[idx]
    }
train_dataset = myDataset(data, tokenizer)
train_dataset[10]
data_collator = DataCollatorForLanguageModeling(tokenizer=tokenizer,
mlm=False)
training_args = TrainingArguments(
    output_dir=f'{my_path}Checkouts', #The output directory
    overwrite_output_dir = True, #overwrite the content of the output
directory
    num_train_epochs = 10, # number of training epochs
    per_device_train_batch_size = 3, # batch size for training
    per_device_eval_batch_size = 3,  # batch size for evaluation
    warmup_steps = 100,# number of warmup steps for learning rate scheduler
    gradient_accumulation_steps = 1, # to make "virtual" batch size larger
    save_steps = 3000
    )


trainer = Trainer(
    model=model,
    args=training_args,
    data_collator=data_collator,
    train_dataset=train_dataset,
    optimizers = (torch.optim.AdamW(model.parameters(),lr=1e-5),None) #
Optimizer and lr scheduler
)
trainer.train()
trainer.save_model(f'{my_path}model_with_summary')
tokenizer.save_vocabulary(f'{my_path}tokenizer')
tokenizer = GPT2Tokenizer.from_pretrained(f'{my_path}tokenizer')
model =
GPT2LMHeadModel.from_pretrained(f'{my_path}model_with_summary').to(DEVICE)
SPECIAL_TOKENS = {'bos_token':'<bos>','eos_token' :'<eos>',
'pad_token':'<pad>', 'sep_token': '<sep>'}
tokenizer.add_special_tokens(SPECIAL_TOKENS)
class KeywordsStoppingCriteria(StoppingCriteria):
    def __init__(self, keywords_ids:list):
      self.keywords = keywords_ids

    def __call__(self, input_ids: torch.LongTensor, scores:
torch.FloatTensor, **kwargs) -> bool:
      if input_ids[0][-1] in self.keywords:
        print(input_ids)
```

```
        return True
      return False
inp = input('Ведіть дані та натисніть Enter:')
inp = inp if len(inp) > 0 else tokenizer.bos_token #токен
input_ids = tokenizer.encode(inp, return_tensors="pt").to(DEVICE)
```