https://doi.org/10.31649/1997-9266-2024-172-1-110-120

УДК 004.8: 004.91

В. Б. Мокін[1]
Б. Ю. Варер[1]
С. М. Левіцький[1]

ІНТЕЛЕКТУАЛЬНА ТЕХНОЛОГІЯ ВИЯВЛЕННЯ ТЕКСТОВИХ ДІПФЕЙКІВ З ВИКОРИСТАННЯМ ВЕЛИКИХ МОВНИХ МОДЕЛЕЙ

[1]Вінницький національний технічний університет

Стрімкий розвиток великих мовних моделей в останні роки породжує важливу проблему — зростання обсягу синтезованих текстів в інформаційному середовищі, що викликає загрозу поширення дезінформації. Відповідно удосконалення технологій виявлення таких текстів стає актуальним завданням.

В статті запропоновано інтелектуальну технологію автоматичної ідентифікації текстів, згенерованих за допомогою штучного інтелекту, зокрема великими мовними моделями. Дослідження базується на аналізі розв'язків конкурсу «LLM — Detect AI Generated Text» на платформі Kaggle. Для цього побудовано датасет, що містить приклади текстів двох класів: ті, що написані людиною, та ті, що згенеровані великими мовними моделями. Датасет зібрано з даних, що знаходяться в публічному доступі. Також проведено розвідувальний аналіз даних та продемонстровано основні особливості підготовленого датасету.

Проаналізовано популярні методи розв'язання задачі ідентифікації згенерованих великими мовними моделями текстів в межах конкурсу на платформі Kaggle. Формалізовано загальну структуру рішення та обґрунтовано основні фактори впливу на точність ідентифікації текстів, згенерованих штучним інтелектом. Розроблено алгоритм для підвищення точності рішення завдяки операціям перед- та післяоброблення, покращення тренувального датасету, оптимізації вибору моделей та методу їхнього ансамблювання тощо. Проведено експерименти, які продемонстрували ефективність запропонованої інтелектуальної технології.

Це дослідження робить внесок у розвиток технологій боротьби з дезінформацією та підкреслює важливість пошуку нових методів виявлення штучно створених текстів у сучасному інформаційному середовищі.

Ключові слова: текстові діпфейки, дезінформація, штучний інтелект, великі мовні моделі, Kaggle, ідентифікація синтезованих текстів, інтелектуальна технологія, чат-боти.

Постановка задачі та вихідні передумови

Практичне впровадження ChatGPT та його аналогів на основі великих мовних моделей в останні роки викликало справжню революцію у сфері генерування текстів для бізнес-рішень, написання комп'ютерних програм тощо. Висока якість текстів, які генеруються, за критерієм збігу з природномовним текстом, спричинила нову проблему: як відрізнити текст, написаний людиною, від тексту, написаного «машиною»? Досі існують задачі, де авторів ранжують за оцінками якості їхніх текстів (написання есе на різні теми в період навчання, мотиваційні листи під час вступу у заклади вищої освіти тощо). В таких задачах важливо вміти виявляти текстові діпфейки, тобто згенеровані штучним інтелектом «підробки» природномовного тексту, які потім люди видають як написані власноруч. Розвиток великих мовних моделей та сервісів на їхній основі спонукав до розвитку і цього доволі нового напряму — створення технологій і систем виявлення діпфейків. Успіхи в цьому напрямі сприяють розвитку великих мовних моделей та технологій їхнього застосування, а це, зі свого боку, дозволяє розвивати технології генерування діпфейків і так — в циклі. Отже, цей напрямок є лише піднапрямком напряму розвитку великих мовних моделей та технологій їхнього прикладного застосування. Цим напрямком займаються усі провідні лабораторії штучного інтелекту у більшості країн світу, оскільки природна мова у кожній країні своя.

Прикладами статей у сфері створення технологій і систем виявлення діпфейків є, до прикладу, роботи [1]—[4]. Але їхнім загальним недоліком є, по-перше, те, що недостатню увагу приділено питанню побудови тренувального датасету, адже подібні рішення доволі чутливі до змісту корпусу (набору) текстів. А по-друге, недостатньо системно та обґрунтовано вибираються рішення щодо передоброблення даних, вибору і тюнінгу моделей та варіантам післяоброблення даних.

Метою дослідження є підвищення точності виявлення текстових діпфейків шляхом створення методу синтезу оптимальної структури інтелектуальної технології для цього виявлення.

Ідея розв'язання задачі

Задачу виявлення текстових діпфейків найефективніше формалізувати і розв'язати як задачу машинного навчання з бінарним таргетом, тобто з цільовою ознакою (англ. "target"), яка набуває тільки два значення (чи є текст фейковим або ні). До того ж, точнішим буде рішення для варіанта «з вчителем», коли модель тренується на наперед розміченому датасеті, де точно відомо які тексти писала людина, а які — «машина». Для розв'язання таких задач, як правило, застосовується технологія з такими етапами [5]:

1. Побудова і розмітка датасету. Формування тренувального, валідаційного і тестового датасетів.
2. Передоброблення даних $\left(P_{before}\right)$.
3. Розвідувальний аналіз даних (може містити ще й етап інженерії ознак).
4. Побудова і ідентифікація (тюнінг — з англ. "tuning") моделей на тренувальному датасеті та вибір оптимального ансамбля моделей на основі передбачень на валідаційному датасеті.
5. Передбачення таргета (*Y*) для тестового датасету.
6. Післяоброблення результатів передбачення для тестового датасету $\left(P_{after}\right)$.
7. Аналіз ефективності передбачення.

Не всі етапи є обов'язковими і часто застосовуються в різній послідовності чи ітеративно з підциклами, якщо точність передбачення не є задовільною. В загальному вигляді, усі етапи є важливими. Якщо датасет є незбалансованим чи не є релевантним задачі, тоді всі інші етапи не будуть ефективними. Якщо дані не передобробити (P_{before}: усунути дублікати, пропущені та аномальні дані, нормалізувати ознаки тощо), тоді моделі можуть не ідентифікуватись якісно. Часто в задачах цього напрямку передбачають не просто бінарні варіанти таргета *Y*, а ймовірність того, що він набуває якогось значення — це дозволяє гнучкіше налаштувати правила післяоброблення P_{after} результатів передбачення і підвищити його точність.

Існує чимало прийомів реалізації кожного етапу та їхніх комбінацій. Наприклад, у статті [5] наведено приклад різних операцій передоброблення P_{before}. Для вибору їхніх оптимальних варіантів на кожному етапі, з чого й складається структура інтелектуальної технології в цілому, можна використати різні методи:

1) метод повного перебору варіантів — може бути задовгим у часі, якщо варіантів на кожному етапі буде забагато;
2) інтелектуальний метод з підкріпленням синтезу оптимального конвеєру операцій попереднього оброблення даних у задачах машинного навчання [5] — для усього розмаїття методів і даних можуть бути складності зі збіжністю алгоритму, що потребує окремого дослідження;
3) метод поетапного перебору варіантів, коли повний перебір здійснюватиметься на кожному етапі окремо — не так тривало, але й не так ефективно, оскільки для кожної моделі оптимальним є свій варіант операцій перед- та післяоброблення;
4) інші методи.

Пропонується використовувати другий метод. Хоча, у простих випадках чи у першому наближенні можна застосовувати й перший чи третій.

Для розв'язання поставленої задачі на нових тестових даних важливо ефективно розв'язати дві задачі:

Задача 1. Побудувати датасет, релевантний новим тестовим даним (етап 1).

Задача 2. Побудувати інтелектуальну технологію з передтренованою оптимальною моделлю на основі датасету з етапу 1 (етапи 2—7 алгоритму, наведеного вище).

Запропонуємо розв'язок цих задач з використанням огляду рішень конкурсу «LLM — Detect AI Generated Text» [6] (скорочено «DAIGT») на платформі датасайнтистів Kaggle, де учасники нама-

гались саме виявити чи наявні есе написані студентами, чи штучним інтелектом. В цьому конкурсі брали участь автори цієї статті як одна команда.

Побудова датасету для розв'язання задачі

Під час побудови тренувального датасету, за яким буде тренуватись модель для задачі «з вчителем» і бінарним таргетом, важливо враховувати такі критерії:

– релевантність потенційним тестовим даним;

– збалансованість щодо значень цільової ознаки (приблизно однакова кількість варіантів для кожного значення, умовно 0 або 1);

– достатня кількість прикладів для навчання моделі.

Цікаво врахувати досвід згаданого вище конкурсу [6], де була поставлена задача виявлення текстових діпфейків серед набору есе. Організаторами конкурсу представлено незбалансований датасет, в якому переважна більшість навчальних прикладів відносилась до категорії текстів, написаних людиною, у той час як прикладів, що згенеровані штучним інтелектом (ШІ) було вкрай мало (лише 3 есе, згенерованих ШІ проти 1375 есе, написаних людиною). В результаті, учасники змагання самостійно намагались побудувати датасети, згенеровані різними великими мовними моделями (англ. "Large Language Models" — LLM). У табл. 1 подано найпопулярніші в конкурсі датасети та використані для їхньої побудови LLM. Щодо кожного датасету наведено V — кількість голосів за цей датасет на платформі Kaggle та значення S (від англ. "Scoring") — максимальне значення метрики ROC AUC (з англ. "Receiver Operating Characteristic Area Under the Curve", що означає «площа під кривою, яка відображає ефективність роботи класифікатора за різних порогових значень»), отримане публічним ноутбуком чи зазначене у пості учасником змагання, який використав цей датасет напряму чи шляхом поєднання з іншими.

Таблиця 1

Порівняльна таблиця найбільш популярних датасетів в конкурсі [6]

№	Назва датасету	Використані LLM та інші датасети {D}	P_{before}	V	S
D_0	Datasets of the notebook "R100_Ensemble" [7]	D_1, D_6, D_{16} та низка приватних (неопублікованих) датасетів	—	—	0,987
D_1	"DAIGT V2 Train Dataset" [8]	D_2, D_5, D_6, D_8, D_{12}, D_{13}, D_{16}; T_{CC}	—	335	0,987
D_2	"LLM Generated Essays for the Detect AI Comp!" [9]	T_{GPT3}, T_{GPT4}	—	230	0,961
D_3	"DAIGT Proper Train Dataset" [10]	D_2, D_5, D_6, D_8, D_{13}; T_{M7b}	—	217	0,977
D_4	"LLM: 7 prompt training dataset" [11]	D_2, D_6, D_8, D_9, D_{12}, D_{13}; T_H	—	167	0,976
D_5	"DAIGT | External Dataset" [12]	T_H, T_{GPT3}	—	153	0,962
D_6	"persuade corpus 2.0" [13]	T_H	—	90	0,978
D_7	"daigt-v3-train-dataset" [14]	D_1, D_9, D_{14}, D_{15} T_{ada-1}, T_{bbg-1}, $T_{curie-1}$, $T_{davin-1}$, $T_{davin-2}$, $T_{davin-3}$	C_1, C_2, C_3, C_4	90	0,976
D_8	"daigt data – llama 70b and falcon180b" [15]	T_{L-70}, T_{F-180} T_{GPT4} (для генерації промптів)	C_4	79	0,976
D_9	"LLM: Mistral-7B Instruct texts" [16]	T_{M7b}	C_9	73	0,986
D_{10}	"DAIGT-V4-TRAIN-DATASET" [17]	D_7; T_{L*}	—	69	0,976
D_{11}	"DAIGT External Train Dataset" [18]	T_H, T_{GPT3}, $T_{davin-3}$, $T_{davin-2}$, $T_{davin-1}$, $T_{curie-1}$ [19], T_{BARD} [20]	C_5, C_6, C_7, C_8 [18]	66	0,977
D_{12}	"LLM-generated essay using PaLM from Google Gen-AI" [21]	T_{PaLM}	—	46	0,976
D_{13}	"Hello, Claude! 1000 essays from Anthropic…" [22]	T_{CI}	—	35	0,976
D_{14}	"[DAIGT] 3500 Essays from Intel Neural Chat 7b" [23]	T_{INC}	—	32	0,965
D_{15}	"GPT4 Rephrased LLM DAIGT Dataset" [24]	T_{GPT4}	—	17	—
D_{16}	"mock_test" [25]	T_H та згенеровані LLM тексти без зазначення їхнього походження	—	0	0,987

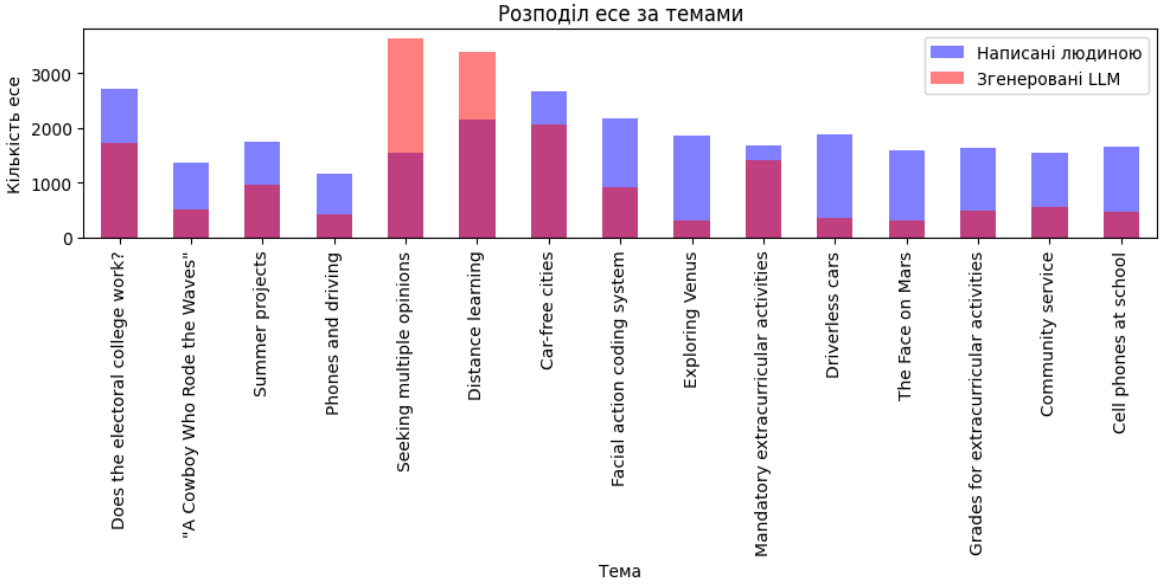
Примітки. 1. Позначення використаних LLM: T_{GPT3} — GPT 3.5 Turbo; T_{GPT4} — GPT 4; T_{CC} — Cohere Command; T_{M7b} — Mistral 7b Instruct; T_H — Тексти, написані людьми; T_{ada-1} — OpenAI text-ada-001; T_{bbg-1} — OpenAI text-babbage-001; $T_{curie-1}$ — OpenAI text-curie-001; $T_{davin-1}$ — OpenAI text-davinci-001; $T_{davin-2}$ — OpenAI text-davinci-002; $T_{davin-3}$ — OpenAI text-davinci-003; T_{BARD} — Google BARD (Gemini); T_{PaLM} — Google PaLM; T_{CI} — Claude Instant 1; T_{INC} — Intel Neural Chat 7b v3.1; T_{L-70} — LLaMA 2 70b; T_{L*} — LLaMA 2 finetuned on Persuade corpus; T_{F-180} — Falcon 180b.

2. Використані такі позначення операції передоброблення (очищення — англ. “Cleaning”) датасетів: C_1 — видалення пропущених значень NaN; C_2 — видалення дублікатів; C_3 — видалення коротких генерацій тексту; C_4 — процедура очищення текстів від «маркерів» генерації LLM (таких, як, наприклад, наявність назви LLM в результаті генерування) [24]; C_5 — видалення лише специфічного тексту “As an AI model …” без урахування інших можливих «маркерів» генерації LLM; C_6 — виправлення регістру літер; C_7 — виправлення пробілів та розривів рядка; C_8 — нормалізація апострофів (наприклад, виправлення dont’» на «don’t»), C_9 — додавання до 15 % помилок до тексту (для того, щоб модель вчилась краще обробляти тестовий текст, навіть з помилками).

Варто зазначити, що багато з наведених популярних датасетів містять в собі інші датасети та їхні комбінації (іноді з деяким післяобробленням, а іноді — просте поєднання). Водночас, в деяких рішеннях конкурсу також використовувалися комбінації різних датасетів, зокрема тих, що вже є об’єднанням. Деякі з датасетів також містять в собі дублікати, очевидні «маркери» генерації LLM, такі, як явне розділення есе на підрозділи, цитування промпта (або запиту — від англ. “prompt”), що використовувався для генерування есе тощо. Також існують генерації LLM, які, внаслідок невдалого промпту, за змістом не є есе взагалі. Ці фактори можуть негативно вплинути на ефективність моделі, якщо не будуть враховані під час навчання моделі.

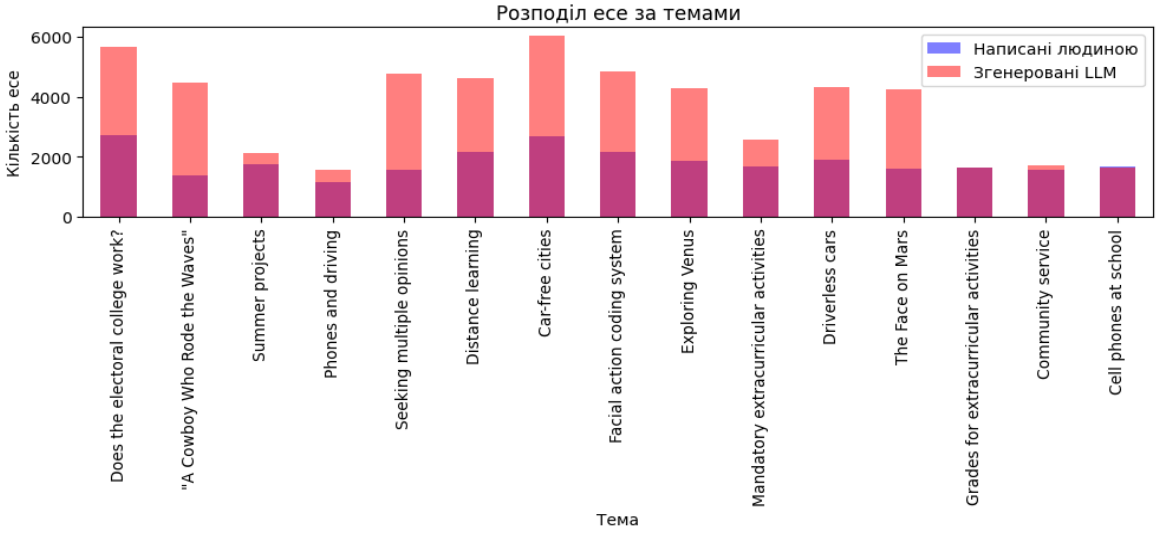
Для збільшення обсягу даних зі збереженням балансу, деякі з датасетів включають в себе есе, написані людьми, а не тільки згенеровані LLM, зокрема: корпус ArguGPT, що включає 4115 есе, написаних людьми [19], корпус PERSUADE 2.0, що включає 25000 есе, написаних учнями 6—12 класів у США [27] тощо. Також серед датасетів існують такі, що створені за допомогою кооперації LLM: наприклад, деякі з промптів для датасету D_8 згенеровано за допомогою GPT-4, після чого самі есе згенеровано LLaMA-2 70b та Falcon 180b.

Слід звернути увагу на те, що для побудови більшості із зазначених датасетів використано переважно специфічні для конкурсу промпти з обмеженим набором тем для есе, що може вплинути на здатність моделей, які навчаються на таких датасетах, до узагальнення та здатності ідентифікувати згенеровані LLM есе на довільшну тему. Аналіз показує, що 10 з наведених датасетів повторюють ті самі теми промптів або їхню підмножину: D_1, D_2, D_4, D_6, D_7, D_9, D_{10}, D_{12}, D_{14}, D_{15}. На рис. 1, для ілюстрації, показано побудовані графіки розподілу кількості есе за темами для датасетів D_1 та D_{10}.



а)

Рис. 1. Графіки розподілу кількості есе за темами для датасетів: *а* — D_1



б)

Продовження рис. 1. Графіки розподілу кількості есе за темами для датасетів: *б* — D_{10}

Проте, окрім датасетів, що містять тексти, згенеровані за схожими темами, серед наведених є датасети, що містять унікальні промпти: наприклад, D_5 містить 2421 згенерованих тексти есе, серед яких 2420 мають унікальний промпт.

Авторами статті у своєму рішенні для цього конкурсу використано датасет D_1, проте не враховано наявність нерелевантного шуму у даних через наявність текстів, що не є есе та текстів, згенерованих за унікальними поодинокими промптами, а також потенційно недостатньої кількості тренувальних прикладів за окремими темами. Врахувавши ці фактори, можна підвищити точність нашого рішення.

Пропонується розробити процедуру побудови датасету з урахуванням передоброблення та очищення даних, яке включатиме такі етапи: видалення «маркерів» LLM; видалення або редагування нерелевантних результатів генерування, які можуть бути визначені за допомогою кооперації LLM, де одна модель буде оцінювати релевантність результату іншої або — за допомогою методів NLP. Для забезпечення релеватності потенційним тестовим даним, промпти для генерації текстів можуть бути згенеровані з використанням LLM. Важливо, також, мінімізувати кількість нерелевантних промптів, які можуть внести в дані зайвий шум і ускладнити процес розроблення адекватної моделі.

Формально описану процедуру можна представити таким чином: нехай

$S = \bigcup_{n=1}^{\infty} B^n$ — множина послідовностей символів зі скінченного алфавіту B, де кожна послідовність має довільну довжину;

$L : S \times \Omega \to \rho(B)$ — велика мовна модель, як функція, що відображає множину послідовностей символів S та множину стохастичних параметрів Ω у розподіл ймовірності наступного символу послідовності $b \in B$;

$W : \rho(B) \times \Omega \to B$ — функція вибору, що повертає конкретний символ $b \in B$ з розподілу $\rho(B)$ з урахуванням стохастичних параметрів Ω. Спосіб вибору може ґрунтуватися на різних методах, таких як вибір символу з максимальною ймовірністю появи (argmax $\rho(B)$), вибір випадкового символу з розподілу $\rho(B)$ з урахуванням параметра температури $t \in \Omega$, який впливає на баланс між випадковістю та детермінованістю результату генерування, метод вибору top-k, де розглядаються k символів з найвищою ймовірністю, метод вибору top-p, де вибір обмежується мінімальною кількістю символів, сумарна ймовірність яких перевищує заданий поріг p тощо. Ці методи дозволяють регулювати ступінь випадковості та різноманітності у виборі символу, забезпечуючи оптимальний баланс між точністю та креативністю в процесі генерування тексту.

Тоді можна представити процес генерування тексту за допомогою LLM, як ітеративний процес G конкатенації вибраного функцією W продовження до поточної послідовності, що починається з

початкового промпту s_0:

$$G(L,W,\Omega,s_0)=\oplus_{n=1...N}W\left(L\left(s_{n-1},\Omega\right),\Omega\right). \quad (1)$$

Для побудови датасету згідно з вищеописаною процедурою, введемо такі позначення:

– множини промптів: P_0 — множина заданих (наявних) промптів для генерування есе; P_p — множина промптів для генерування нових промптів для генерування есе; P_C — множина промптів для фільтрування та виправлення згенерованих текстів есе;

– функції генерування чи оброблення тексту чи есе: $G_P(P)=G(L_P,W_P,\Omega_P,p\in P)$ — функція генерування нових промптів для подальшого генерування текстів есе; $G_{E1}(P)=G\left(L_E,W_E,\Omega_E,G_P(P)\right)$ — функція генерування тексту есе за згенерованим промптом; $G_{E2}=G\left(L_E,W_E,\Omega_E,p\in P_0\right)$ — функція генерування тексту есе за наявним промптом; $G_C(E)=G\left(L_C,W_C,\Omega_C,\left(p\in P_C\right)\oplus E\right)$ — функція фільтрування та виправлення тексту есе, де E — текст есе.

Тоді функція генерування корпусу з n есе має такий вигляд:

$$D\left(G_P,G_{E1},G_{E2},G_C,P_0,P_C,n\right)=\left\{G_C\left(G_{E1}(P_P)\cup G_{E2}\right)\mid i=1...n\right\}. \quad (2)$$

Отже, проаналізовано популярні датасети, проведено розвідувальний аналіз даних та описано формальну процедуру побудови датасету з використанням кооперації великих мовних моделей та передоброблення даних.

Аналіз популярних та ефективних технологічних рішень поставленої задачі у конкурсах Kaggle

Після побудови гарного датасету ще треба розв’язати задачу вибору структури інтелектуальної технології у вигляді оптимальних варіантів передоброблення даних (їх може бути декілька у певній послідовності), вибору оптимальних моделей чи їхнього ансамблю, вибору операцій післяоброблення. Пропонується така система нотацій цих варіантів:

1. Для вибору операцій передоброблення пропонується використати систему позначень зі статті [5], але дії позначати, як наведено вище — C_{ij}.

2. Для вибору моделей пропонується ієрархічна система з позначенням M_{ij} (з англ. “Model” — модель), яка доповнюється такими видами позначень E_q для різних способів ансамблювання моделей машинного навчання [28].

3. Для вибору правил післяоброблення пропонується ієрархічна система з позначенням R_{ij} (з англ. “Rule” — правило).

А тоді кожне рішення можна буде записати як у табл. 2, де подано приклади такого роду операцій з використаннням таких позначень:

– передоброблення даних: видалення спеціальних символів (C_{10}); нормалізація Unicode (NFC) — процес, який забезпечує стандартизацію різних двійкових представлень символів до одного двійкового значення (C_{11}); Byte-Pair Encoding токенізація — ітераційне об’єднання найчастішої пари послідовних байтів або символів у текстовому корпусі, доки не буде досягнуто попередньо визначеного розміру словника (C_{12}); деобфускація — виправлення орфографічних помилок та обфускації в даних (C_{13}); без явного передоброблення (C_{14});

– великі мовні моделі (M_0): Mistral 7B (M_{00}); DeBERTa (M_{01}); Llama 2 7B (M_{02}); Tiny Llama 1.1B (M_{03}); Mamba-790M (M_{04}); Phi-2 (M_{05}); Ghostbuster (M_1) [39];

– трансформери: RoBERTa (M_{11});

– інші моделі машинного навчання: MultinomialNB (M_2); SGD-класифікатор (M_3); LightGBM-класифікатор (M_4); Gradient Boosting регресор (M_5); Random Forest регресор (M_6); лінійна регресія (M_7); метод опорних векторів (OneClassSVM) (M_8); XGBoost-класифікатор (M_9);

RandomForest класифікатор (M_{10}); інші класифікатори (M_{12});

– ансамблі: зважене голосування (англ. “weighted voting”) (E_0); голосування (VotingClassifier) (E_1); середнє арифметичне значення результатів (E_2); прогнози моделі M_{01}, що потрапили в 40—60 перцентилі, замінені прогнозами моделі F_0 і M_3 (E_3);

– післяоброблення: R_0 — ранжування; R_1 — застосування сигмоїдальної функції у нейромережах.

Таблиця 2

Порівняльна таблиця рішень конкурсу [6]

№	Джерело	P_{before}	Вибрані моделі	Ансамблювання моделей	P_{after}	V	S
1	[27]	C_6, C_7, C_{10}, C_{12}, C_{14}	M_{00}, M_{01}, M_{02}, M_{03}, M_1	E_0	R_0	50	0,987
2	[28]	C_{12}, C_{13}	M_{01}, M_2, M_3, M_4	E_0		64	0,974
3	[29]	C_{12}	M_{01}, M_2, M_3, M_4, M_5, M_6, M_7	E_0	R_1	63	0,973
4	[30]	—	M_{01}, M_{04}	E_2	—	81	0,972
5	[31]	—	M_{05}, M_8	—	—	15	0,969
6	[32]	C_6, C_{11}, C_{12}	M_{01}, M_3	E_3	—	44	0,965
7	[33]	—	M_4, M_9, M_{10}	E_1	—	65	0,956
8	[34]	—	M_{01}, M_{11}, M_3,	E_1	—	53	0,947
9	[35]	—	M_{01}, M_{12}	E_0	—	15	0,942
10	[36]	C_2	M_{01}, M_{11}	E_0	—	25	0,938

Аналіз табл. 2 показує, що найпопулярнішими є такі рішення:

– передоброблення P_{before}: виправлення регістру літер C_6, Byte-Pair Encoding токенізація C_{12};

– моделі: DeBERTa (M_{01}), MultinomialNB (M_2), SGD-класифікатор (M_3), LightGBM-класифікатор (M_4);

– ансамбль: зважене голосування E_0;

– післяоброблення P_{after}: ранжування R_0.

Перше місце на конкурсі здобув розв’язок:

$$Y = R_0\left(E_0\left(M_{00}(X_1), M_{01}(X_1), M_1\left(M_{02}(X_1)\right), M_1\left(M_{03}(X_1)\right)\right)\right); \quad (3)$$

$$X_1 = C_{12}\left(C_6\left(C_{10}\left(C_1(X)\right)\right)\right). \quad (4)$$

Отже, здійснено нотацію операцій та формалізацію задачі, знайдено найпопулярніші рішення на конкурсі.

Алгоритм розв’язання поставленої задачі та огляд авторського рішення

Авторами статті розроблено рішення, яке має такий формалізований вигляд:

$$Y = E_0\left(M_4(X_1), M_3(X_1), M_9(X_1)\right); \quad (5)$$

$$X_1 = C_{12}\left(C_2\left(C_1(X)\right)\right). \quad (6)$$

Пропонується алгоритм, який дозволить покращити точність поточного рішення:

1. Побудова набору даних з використанням функцій (1) і (2), формування тренувального, валідаційного і тестового датасетів.
2. Тонке налаштування великої мовної моделі з використанням тренувального датасету з п. 1.
3. Передоброблення даних з використанням операцій {C}.
4. Машинне навчання моделей з множини $\{M\}$.
5. Побудова ансамблю E_0 моделей з п. 4, тобто — середньозважена сума їхніх передбачень.
6. Передбачення таргета *Y* для тестового датасету.
7. Післяоброблення (ранжування R_0) результатів передбачення для тестового датасету.

Блок-схема цього алгоритму показана на рис. 2.

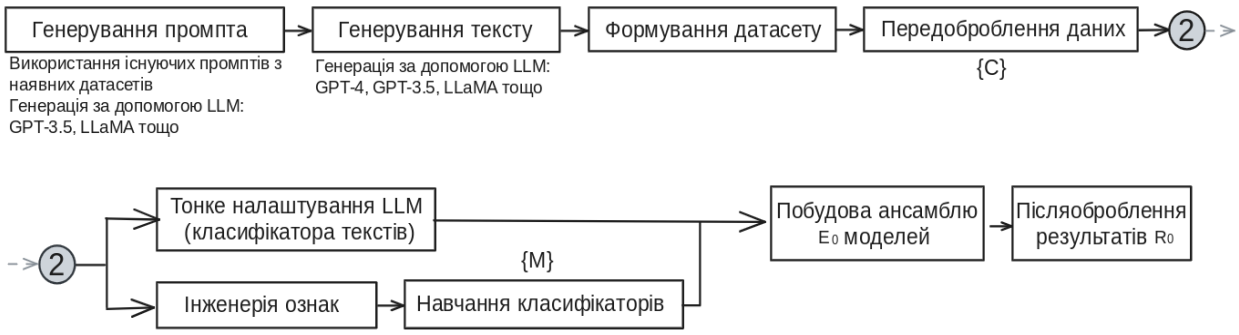


Рис. 2. Блок-схема алгоритму розв’язання задачі

Застосування цього алгоритму дозволить досягти підвищення точності.

Приклад розв’язання поставленої задачі та дослідження ефективності

Візьмемо датасет з переможного рішення, але, для пришвидшення розрахунків, візьмемо тільки варіант D16. Побудуємо декілька варіантів рішень з використанням запропонованого алгоритму (рис. 2). Результат передбачення подано у табл. 3.

Таблиця 3

Варіанти розв’язання задачі за оптимальним алгоритмом для датасету D_{16}

№ п/п	P_{before}	Вибрані моделі	Ансамблювання моделей	P_{after}	S
1	C_6, C_7, C_{10}, C_{12}, C_{14}	M_{00}, M_{01}, M_{02}, M_{03}, M_1	E_0	R_0	0,992
2	C_2, C_6, C_7, C_{10}, C_{11}	M_{01}, M_3, M_7	E_3	—	0,814
3	C_2, C_6, C_7, C_{10}, C_{11}	M_3, M_6, M_7	E_0	—	0,759
4	C_2, C_6, C_7, C_{10}, C_{11}	M_3, M_7	E_0	—	0,695
5	C_{14}	M_7	—	—	0,660

Як видно з табл. 3, врахування більшої кількості рекомендацій суттєво збільшує точність з 0,66 до 0,992 за метрикою ROC AUC. А отже, можна припустити (і автор переможного рішення конкурсу Kaggle це довів), що врахування усього комплексу рекомендацій дозволить досягти точності майже 1. Варто зазначити, що переможне рішення теж можна удосконалити, з урахуванням рекомендацій алгоритму на рис. 2, використовуючи ідеї інших рішень. А отже, на запитання чи може сучасний стан науки відрізнити текст, написаний людиною, від тексту, згенерованого великими мовними моделями, відповідь — так, це можливо.

Висновки

Розглянуто задачу бінарної класифікації для автоматичної ідентифікації штучно згенерованих текстів. Авторами дослідження формалізовано та проаналізовано розв’язки з конкурсу «LLM – Detect AI Generated Text» на платформі Kaggle, визначено популярні та ефективні технологічні рішення поставленої задачі.

Обгрунтовано, що саме поєднання різних операцій формування тренувального і валідаційного датасетів, передоброблення даних, гарний вибір моделей та способу їхнього ансамблювання, варіантів післяоброблення та припасування їх в єдине ціле, може дати суттєве зростання точності, що й продемонстрували переможці конкурсу. Виконано розвідувальний аналіз формалізованої інформації про основні переможні рішення конкурсу.

Розроблено алгоритм для підвищення точності виявлення текстових діпфейків, з урахуванням найкращих ідей переможців конкурсу, за рахунок покращення тренувального датасету, застосування операцій перед- і післяоброблення, ансамблювання моделей та ін. Зроблено формалізований

опис запропонованого алгоритму. Проведено експерименти на спрощених датасетах, які показали доцільність запропонованого підходу. В ході експериментів вдалося підвищити точність рішення ідентифікації текстів, згенерованих штучним інтелектом. Доведено, що сучасний стан науки, принаймні для розглянутої у статті задачі, поки дозволяє практично точно визначити чи текст написаний людиною, чи машиною.

Запропонований алгоритм інтелектуальної технології може бути використаний і на інших промптах, тобто для текстів іншого профілю і змісту, але тоді треба буде провести інший системний аналіз за аналогічною методологією.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

[1] R. R. Soto et al., “Few-Shot Detection of Machine-Generated Text using Style Representations,” arXiv preprint, arXiv:2401.06712, 2024.

[2] B. P. Kumar, M. S. Ahmed, and M. Sadanandam, “DistilBERT: A Novel Approach to Detect Text Generated by Large Language Models (LLM),” Feb. 2024, https://doi.org/10.21203/rs.3.rs-3909387/v1 .

[3] Z. Wu, and H. Xiang, “MFD: Multi-Feature Detection of LLM-Generated Text”, Aug. 2023, https://doi.org/10.21203/rs.3.rs-3226684/v1 .

[4] OpenAI, “New AI classifier for indicating AI-written text,” 2023. [Online]. Available: https://openai.com/blog/new-ai-classifier-for-indicating-ai-written-text. Accessed: Feb 15, 2024.

[5] В. Б. Мокін, і М. В. Дратований, «Інтелектуальний метод з підкріпленням синтезу оптимального конвеєру операцій попереднього оброблення даних у задачах машинного навчання,» Наукові праці ВНТУ, вип. 4, Груд 2022. https://doi.org/10.31649/2307-5376-2022-4-15-25 .

[6] J. King, P. Baffour, S. Crossley, R. Holbrook, and M. Demkin, “LLM – Detect AI Generated Text,” 2023 [Online]. Available: https://www.kaggle.com/competitions/llm-detect-ai-generated-text. Accessed: Feb 15, 2024.

[7] N. Broad “R100_Ensemble,” 2023 [Online]. Available: https://www.kaggle.com/code/nbroad/r100-ensemble/input. Accessed: Feb 15, 2024.

[8] D. Kłeczek, “DAIGT V2 Train Dataset,” 2023 [Online]. Available: https://www.kaggle.com/datasets/thedrcat/daigt-v2-train-dataset. Accessed: Feb 15, 2024

[9] R. Osmulski, “LLM Generated Essays for the Detect AI Comp!” 2023 [Online]. Available: https://www.kaggle.com/datasets/radek1/llm-generated-essays. Accessed: Feb 15, 2024.

[10] D. Kłeczek, “DAIGT Proper Train Dataset,” 2023 [Online]. Available: https://www.kaggle.com/datasets/thedrcat/daigt-proper-train-dataset. Accessed: Feb 2, 2024.

[11] C. McBride Ellis, “LLM: 7 prompt training dataset,” 2023 [Online]. Available: https://www.kaggle.com/datasets /carlmcbrideellis/llm- 7-prompt-training-dataset . Accessed: Feb 15, 2024.

[12] A. Paullier, “DAIGT | External Dataset,” 2023 [Online]. Available: https://www.kaggle.com/datasets/alejopaullier/daigt-external-dataset. Accessed: Feb 15, 2024.

[13] N. Broad, “Persuade corpus 2.0,” 2023 [Online]. Available: https://www.kaggle.com/datasets/nbroad/persaude-corpus-2/. Accessed: Feb 2, 2024.

[14] D. Kłeczek, “Daigt-v3-train-dataset,” 2023 [Online]. Available: https://www.kaggle.com/datasets/thedrcat/daigt-v3-train-dataset. Accessed: Feb 15, 2024.

[15] N. Broad, “Daigt data – llama 70b and falcon180b,” 2023 [Online]. Available: https://www.kaggle.com/datasets /nbroad/daigt-data-llama-70b-and-falcon180b . Accessed: Feb 15, 2024

[16] C. McBride Ellis, “LLM: Mistral-7B Instruct texts,” 2023 [Online]. Available: https://www.kaggle.com/datasets /carlmcbrideellis /llm-mistral-7b-instruct-texts . Accessed: Feb 15, 2024.

[17] D. Kłeczek, “DAIGT-V4-TRAIN-DATASET,” 2023 [Online]. Available: https://www.kaggle.com/datasets/thedrcat /daigt-v4-train-dataset. Accessed: Feb 15, 2024.

[18] D. Kłeczek, “DAIGT External Train Dataset,” 2023 [Online]. Available: https://www.kaggle.com/datasets /thedrcat/daigt-external-train-dataset. Accessed: Feb 15, 2024.

[19] Y. Liu et al., “ArguGPT: evaluating, understanding and identifying argumentative essays generated by GPT models,” arXiv preprint, arXiv:2304.07666, 2023.

[20] K. Hayawi, S. Shahriar, and S. Mathew, “The Imitation Game: Detecting Human and AI-Generated Texts in the Era of Large Language Models,” arXiv preprint, arXiv:2307.12166, 2023.

[21] M. Rizqi, “LLM-generated essay using PaLM from Google Gen-AI,” 2023 [Online]. Available: https://www.kaggle.com/datasets/kingki19/llm-generated-essay-using-palm-from-google-gen-ai. Accessed: Feb 15, 2024.

[22] D. Hanley, “Hello, Claude! 1000 essays from Anthropic…,” 2023 [Online]. Available: https://www.kaggle.com/datasets/darraghdog/hello-claude-1000-essays-from-anthropic. Accessed: Feb 15, 2024.

[23] P. Srikanth, “[DAIGT] 3500 Essays from Intel Neural Chat 7b,” 2023 [Online]. Available: https://www.kaggle.com/datasets/phanisrikanth/daigt-essays-from-intel-neural-chat-7b. Accessed: Feb 15, 2024.

[24] N. Matatov, “GPT4 Rephrased LLM DAIGT Dataset,” 2023 [Online]. Available: https://www.kaggle.com/datasets /snassimr/gpt4-rephrased-llm-daigt-dataset. Accessed: Feb 15, 2024.

[25] R. Biswas, et al., “mock_test,” 2023 [Online]. Available: https://www.kaggle.com/datasets/conjuring92/mock-test. Accessed: Feb 15, 2024.

[26] N. Broad, “Clean llama 70b data,” 2023 [Online]. Available: https://www.kaggle.com/code/nbroad/clean-llama-70b-data/notebook . Accessed: Feb 15, 2024

[27] S. Crossley, et al., “A large-scale corpus for assessing written argumentation: PERSUADE 2.0,” Zenodo, Aug. 2023, https://doi.org/10.1016/j.asw.2023.100667 .

[28] “Scikit-learn: Machine Learning in Python,” Sklearn.ensemble Module [Online]. Available: https://scikit-learn.org/stable/modules/classes.html#module-sklearn.ensemble . Accessed: Feb 15, 2024.

[29] N. Broad, “Comprehensive 1st Place Write-Up,” 2023 [Online]. Available: https://www.kaggle.com/competitions/llm-detect-ai-generated-text/discussion/473295 . Accessed: Feb 15, 2024.

[30] Y. Maslov, “3rd place solution,” 2023 [Online]. Available: https://www.kaggle.com/competitions/llm-detect-ai-generated-text/discussion/470333. Accessed: Feb 15, 2024

[31] E. Demir, “[4th Place Solution] A Summary of Combined Arms Approach,” 2023 [Online]. Available: https://www.kaggle.com/competitions/llm-detect-ai-generated-text/discussion/470179 . Accessed: Feb 15, 2024.

[32] J. Day, “5th place solution: 1.7 million training examples + domain adaptation,” 2023 [Online]. Available: https://www.kaggle.com/competitions/llm-detect-ai-generated-text/discussion/470093 . Accessed: Feb 15, 2024.

[33] D. Cozzolino, “6nd place solution with code,” 2023 [Online]. Available: https://www.kaggle.com/competitions/llm-detect-ai-generated-text/discussion/471831 . Accessed: Feb 15, 2024.

[34] H. Mei, “[7th Place Solution] Generate Data with Non-Instruction-Tuned Models,” 2023 [Online]. Available: https://www.kaggle.com/competitions/llm-detect-ai-generated-text/discussion/470643 . Accessed: Feb 15, 2024.

[35] A. Meda, “[8th LB Solution] Linguistic Features: PPL & GLTR,” 2023 [Online]. Available: https://www.kaggle.com/competitions/llm-detect-ai-generated-text/discussion/470224 . Accessed: Feb 15, 2024.

[36] D. Hanley, “[1st Public/9th Private] LLMLab - Solution Summary,” 2023 [Online]. Available: https://www.kaggle.com/competitions/llm-detect-ai-generated-text/discussion/470255. Accessed: Feb 15, 2024.

[37] U. Erii, “12th place solution: DeBERTa + TF-IDF,” 2023 [Online]. Available: https://www.kaggle.com/competitions/llm-detect-ai-generated-text/discussion/470396 . Accessed: Feb 15, 2024/

[38] R. Banthia, “13th place solution - Transformers only,” 2023 [Online]. Available: https://www.kaggle.com/competitions/llm-detect-ai-generated-text/discussion/470593 . Accessed: Feb 15, 2024.

[39] Verma, Vivek, et al. “Ghostbuster: Detecting Text Ghostwritten by Large Language Models,” arXiv preprint, arXiv:2305.15047, 2023.

Рекомендована кафедрою системного аналізу та інформаційних технологій ВНТУ

Стаття надійшла до редакції 20.02.2024

Мокін Віталій Борисович — д-р техн. наук, професор, завідувач кафедри системного аналізу та інформаційних технологій; e-mail: vbmokin@vntu.edu.ua ;

Варер Борис Юхимович — аспірант кафедри системного аналізу та інформаційних технологій; e-mail: androbor17@gmail.com ;

Левіцький Сергій Мойсейович — аспірант кафедри системного аналізу та інформаційних технологій; e-mail: levitsky.serhii@gmail.com

V. B. Mokin[1]
B. Yu. Varer[1]
S. M. Levitskyi[1]

Intelligent Technology for Detecting Text-Based Deepfakes Using Large Language Models

[1]Vinnytsia National Technical University

The rapid development of large language models in recent years has generated a significant problem — the increase in the volume of synthesized texts in the information environment, which poses a threat of the spread of misinformation. Accordingly, improving technologies for detecting such texts becomes a relevant ask.

This article proposes an intelligent technology for the automatic identification of texts generated by artificial intelligence, especially large language models. The research is based on the analysis of solutions from the "LLM — Detect AI Generated Text" competition on the Kaggle platform. For this purpose, a dataset was constructed that contains examples of texts from

two classes: those written by humans and those generated by large language models. The dataset was compiled from data that is publicly available. An exploratory data analysis was also conducted, demonstrating the main features of the prepared dataset.

The article analyzes popular solutions for the problem of identifying texts generated by large language models within the Kaggle competition. It formalizes the general structure of the solution and justifies the main factors affecting the accuracy of identifying texts generated by artificial intelligence. An algorithm was developed to increase the accuracy of the solution through pre-processing and post-processing operations, improving the training dataset, optimizing the selection of models, and their ensemble method, among others. Experiments were conducted, demonstrating the effectiveness of the proposed intelligent technology.

This research contributes to the development of technologies to combat misinformation and highlights the importance of finding new methods to detect artificially created texts in modern information environment.

Keywords: text deepfakes, misinformation, artificial intelligence, large language models, Kaggle, identification of synthesized texts, intelligent technology, chat-bots.

Mokin Vitalii B. — Dr. Sc. (Eng.), Professor, Head of the Chair of System Analysis and Information Technology, e-mail: vbmokin@vntu.edu.ua ;

Varer Borys Yu. — Post-Graduate Student of the Chair of System Analysis and Information Technology, e-mail: androbor17@gmail.com ;

Levitskyi Serhii M. — Post-Graduate Student of the Chair of System Analysis and Information Technology, e-mail: levitsky.serhii@gmail.com