

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ ТЕХНОЛОГІЧНИЙ УНІВЕРСИТЕТ

Кваліфікаційна наукова праця
на правах рукопису

Никоненко Андрій Олександрович

УДК 004.85:004.912:004.056

ДИСЕРТАЦІЯ

**ГІБРИДНИЙ МЕТОД ІДЕНТИФІКАЦІЇ АВТОМАТИЧНО
ЗГЕНЕРОВАНИХ ПРИРОДНОМОВНИХ ТЕКСТІВ**

122 – Комп’ютерні науки

12 – Інформаційні технології

Подається на здобуття наукового ступеня доктора філософії

Дисертація містить результати власних досліджень.

Використання ідей, результатів і текстів інших авторів

мають посилання на відповідне джерело

_____ А. О. Никоненко

Науковий керівник Фауре Еміль Віталійович, доктор технічних наук, професор

Черкаси – 2026

АНОТАЦІЯ

Никоненко А. О. Гібридний метод ідентифікації автоматично згенерованих природномовних текстів – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії за спеціальністю 122 Комп'ютерні науки (12 Інформаційні технології). Черкаський державний технологічний університет, Міністерство освіти і науки України, Черкаси, 2026.

У дисертаційному дослідженні розв'язано актуальну науково-прикладну задачу, що полягає в розробці гібридного методу ідентифікації автоматично згенерованих природномовних текстів, стійкого до трансформаційних атак маскування та епістемічної невизначеності.

Широке впровадження великих мовних моделей (LLM), крім позитивних наслідків, призвело до появи цілої низки викликів, серед яких можна виділити створення та розповсюдження неправдивої інформації, використання ШІ в соціальній інженерії, маніпулювання суспільством та підрив принципів академічної доброчесності. У сценаріях неетичного використання LLM їх найбільша перевага - здатність до генерації текстів, які лексично та структурно неможливо відрізнити від текстів, написаних людиною, - стає найбільшою загрозою, оскільки виявлення штучно згенерованих текстів є неможливим без використання спеціальних інструментів. Боротьба з новими викликами в інформаційній безпеці вимагає створення надійних інструментів ідентифікації штучно згенерованого контенту - детекторів ШІ. Проведене дослідження показало, що наявні на ринку системи маскування згенерованих даних проводять глибокі структурні та семантичні трансформації текстів, що веде до приховування сліду ШІ та суттєво знижує ефективність наявних детекторів щодо їх можливостей розпізнавання згенерованих текстів. З іншого боку, більшість сучасних рішень стикається з проблемою епістемічної невизначеності під час аналізу даних поза навчальним розподілом, наприклад, текстів, написаних неносіями мови, що веде до підвищеного рівня хибнопозитивних спрацювань і підриву довіри до засобів детекції загалом.

З огляду на вищезазначене, розробка математичної моделі сліду ІІІ та гібридного методу ідентифікації автоматично згенерованих природномовних текстів, стійкого до трансформаційних атак маскуваннн та епістемічної невизначеності, є актуальною науково-прикладною задачею, що має важливе значення для розвитку систем контент-модерації, інформаційної безпеки та платформ академічної доброчесності.

Для визначення завдань дослідження проведено систематизацію та аналіз існуючих підходів до ідентифікації автоматично згенерованих текстів. Встановлено, що попри наявність великої кількості підходів до виявлення ІІІ, кожен з них містить як серйозні теоретичні обмеження щодо застосовності, так і вразливості до сценаріїв маскуваннн, що робить процес визначення згенерованих даних досить ненадійним. Доведено, що детекція ІІІ, заснована лише на одному рівні ознак (лексичному, синтаксичному чи семантичному), має недоліки як з боку вразливості до трансформаційних атак, так і з боку високого рівня хибнопозитивних спрацювань. Неідеальна точність наявних класифікаторів, теоретичні та практичні обмеження щодо їх застосування, велика кількість атак та типів маскуваннн і відсутність стійких та швидких засобів інтерпретації результатів вказують на потребу в розробці як надійного гібридного методу ідентифікації автоматично згенерованих текстів, так і інтерпретаційної моделі для візуалізації доказів детекції.

Вразливість сучасних методів детекції напнрнму пов'язана зі сприйняттям сліду ІІІ як статичного об'єкта, що дає змогу легко маніпулювати точністю детекції шляхом внесення збурень через перефразування, заміну символів або вихід нових версій моделей чи архітектур. Для підвищення стійкості АІ детекції проти трансформаційних атак пропонується замість масштабування детектора (збільшення розміру моделі та обсягу тренувальних даних) змінити концепцію сліду ІІІ так, щоб отримати стійкий багатомірний об'єкт, що включає ознаки трьох рівнів: семантичного, стилістичного і структурного. Зокрема, розроблено математичну модель сліду ІІІ в багатовимірному просторі ознак $\mathbf{F} = \mathbf{F}_{lex} \times \mathbf{F}_{syn} \times \mathbf{F}_{sem}$ (лексичних, синтаксичних та семантичних), яка формалізує його у вигляді вектора $T_{AI}(X) = \Phi(f_{lex}(X), f_{syn}(X), f_{sem}(X))$. Також введено основні рівні аналізу:

лексичний ($f_{lex}(X)$), що фіксує імовірнісне згладжування тексту генеративними моделями через перплексію та вибуховість (burstiness); синтаксичний ($f_{syn}(X)$), що вимірює структурну однотонність згенерованого тексту через дивергенцію Кульбака-Лейблера для N-грам POS-тегів відносно людського еталона, а також через дисперсію глибини синтаксичних дерев; семантичний ($f_{sem}(X)$), що виявляє аномально високу локальну узгодженість ембедінгів сусідніх речень та глобальний дрифт тематики.

Крім того, удосконалено концептуальну модель оцінки робастності систем детекції сліду ШІ, яка шляхом введення формалізованого поняття Точки зламу та інтеграції математичної моделі багаторівневого простору ознак дає змогу перейти від наявних емпіричних підходів оцінки робастності на фіксованих датасетах до встановлення прямого математичного взаємозв'язку між інтенсивністю структурних трансформацій тексту (коефіцієнт заміни RR), збереженням його семантичної цілісності (I_{sem}) та ймовірністю виявлення (P_{det}). Модель виділяє три області, що демонструють ефективність атаки, та пропонує метод оцінки робастності детектора в умовах невизначеності.

Вперше розроблено гібридний метод ідентифікації автоматично згенерованих природномовних текстів, який, на відміну від класичних детекторів, що сприймають слід ШІ як статичний і легко змінюваний об'єкт, шляхом використання каскадного ансамблю незалежних класифікаторів із додатковими механізмами валідації дає змогу проводити аналіз міжрівневої узгодженості ознак через адаптивне зважування та розрахунок сигналу розбіжності $S_{mismatch}$. На відміну від традиційного простого об'єднання ознак в один вектор, запропонований метод розподіляє аналіз між трьома незалежними модулями: лексичним - M_{lex} , синтаксичним - M_{syn} та семантичним - M_{sem} . Кожен із модулів генерує власну незалежну оцінку, яка передається до метакласифікатора для прийняття остаточного рішення. Оскільки під час трансформаційної атаки методам маскуванню вкрай важко змінити стилістику, синтаксис та семантику одночасно і пропорційно, будь-яка атака створює розрив між

рівнями. Цей розрив аналізується через механізм адаптивного зважування, що дає змогу використовувати дворівневу логіку ваг: $w_j(X) = w_j^{stat} \cdot \alpha_j(X)$, тобто вивчена статистична вага множиться на динамічний коефіцієнт довіри. Залежно від типу виявленої атаки, алгоритм змінює довіру до різних модулів, завдяки чому спроба обходу детектора на одному рівні автоматично стає тригером для підвищення ймовірності детекції на інших рівнях. Такий підхід фіксує когнітивний дисонанс між лексичним, синтаксичним та семантичним рівнями тексту під час застосування трансформаційних атак, роблячи спроби маскування на одному рівні тригером для посилення детекції на іншому.

Разом з цим набув подальшого розвитку метод забезпечення робастності систем обробки природної мови в умовах епістемічної невизначеності та даних поза навчальним розподілом. Класичні методи детекції генерують точкові ймовірності з надмірною впевненістю і не містять інформації про те, чи відповідає текст, що аналізується, текстам із тренувальної вибірки. Це робить їх критично вразливими до даних поза навчальним розподілом, наприклад, текстів неносіїв мови, академічних текстів або нестандартних стилів письма, що призводить до зростання кількості хибнопозитивних спрацювань. Для розв'язання цієї проблеми розроблено надбудову, яка дає змогу перетворювати евристичні точкові прогнози на математично обґрунтовані множини прогнозів $\Gamma^\epsilon(x)$. Обчислюючи міри неконформності на калібрувальній вибірці, алгоритм ICP гарантує, що істинний клас тексту належатиме до згенерованої множини із заданим рівнем статистичної достовірності $1 - \epsilon$. Це перетворює точкові прогнози метакласифікатора на множини прогнозів з гарантованою статистичною достовірністю, що суттєво знижує рівень хибнопозитивних спрацювань, уможливорює глобальну ідентифікацію аномалій та генерацію порожніх множин замість хибних класифікацій.

Крім того, уперше розроблено інтерпретаційну модель обґрунтування доказів детекції пояснюваного ШІ, яка, на відміну від класичних ітеративних методів LIME/SHAP, не потребує значних обчислювальних ресурсів і демонструє стабільність на текстових даних. Її створення розв'язує проблему чорної скриньки,

що існує у випадку класичних детекторів, результатам яких часто бракує прозорості для ухвалення рішень експертами. Шляхом використання внутрішніх ваг метакласифікатора w_j^{stat} та адаптивних коефіцієнтів довіри $\alpha_j(X)$ ця модель дає змогу генерувати три рівні атрибуції: глобальний, локальний та Карту доказів. Глобальний профіль атрибуції дає змогу визначити, які ознаки відіграють найбільшу роль під час аналізу текстів гібридним методом загалом. Локальний профіль атрибуції показує вплив кожного з параметрів метакласифікатора на фінальне рішення щодо конкретного тексту, а інтерактивна Карта доказів візуалізує аномалії лексичної перплексії та структурних загроз.

У роботі проведено експериментальну валідацію розроблених математичних моделей та гібридного методу ідентифікації згенерованих текстів на спеціально сформованих тестових даних. Крім текстів, написаних людьми та згенерованих Instant моделями, тестові дані також включають тексти MoE та Reasoning-моделей, тексти після трансформаційних атак парафразерів T5, DIPPER та комерційних систем маскування. Експериментально доведено, що для SOTA-детектора Fast-DetectGPT Точка зламу настає вже за мінімальних змін (коефіцієнт заміни $RR = 0.03$), тоді як результати гібридного методу підтверджують суттєве підвищення стійкості детекції з уникненням досягнення Точки зламу навіть під час зміщення праворуч до $RR > 0.9$. Експериментально досліджено та адаптовано методи моделювання та виділення сліда ШІ для сучасних архітектур MoE та Reasoning. На відміну від підходів, що фокусуються на мікроструктурних артефактах генерації через оцінку щільності ядра KDE чи спектральний аналіз ритму FFT, запропоновані методи дають змогу надійно ідентифікувати когнітивний слід моделі.

На основі розробленої математичної моделі та гібридного методу створено програмний комплекс ідентифікації автоматично згенерованих текстів. Систему розгорнуто з використанням хмарної інфраструктури AWS, що забезпечує її високу відмовостійкість, масштабованість та можливість паралельної обробки запитів під час аналізу великих масивів даних у режимі реального часу. Загальна асимптотична складність системи, що складається з гібридного методу та підсистеми

інтерпретації, дорівнює $O(N^2)$, а середній час обробки одного документа у хмарному середовищі AWS становить 0,33 секунди. На відміну від наявних рішень, що демонструють критичне падіння повноти детекції в Області 2 (зона успішних атак) під час трансформаційних атак, гібридний метод, завдяки архітектурі каскадного ансамблю та аналізу міжрівневої узгодженості, дає змогу повністю нівелювати зону успішного маскування згенерованого контенту. Запропонована архітектура метакласифікатора, використання алгоритмів канонізації та механізму адаптивного зважування ознак дають змогу досягти показника точності $F1 = 0.92$ на стаціонарних даних та зберегти повноту виявлення (Recall) на рівні 88.77% за жорсткого обмеження рівня хибнопозитивних спрацювань (FPR) до 1%. В умовах протидії сучасним комерційним системам маскування (StealthGPT, AIHumanize, Phrasly) система перевершує світові аналоги, забезпечуючи середній Recall на рівні 77.16% та найнижчий серед аналогів рівень успішності атак ($ASR = 22.84\%$).

Впровадження методики застосування фреймворку ICP дало змогу суттєво знизити рівень хибнопозитивних спрацювань (FPR) під час аналізу текстів неносіїв мови (на прикладі датасету IELTS) з 4.81% до 0.98%. Інтеграція фреймворку ICP дозволила системі розпізнати епістемічну невизначеність цих текстів та згенерувати порожні множини в складних випадках. Підсумковий коефіцієнт порятунку Rescue Rate становив 79.71%, що дає змогу говорити про суттєву мінімізацію ризиків несправедливих звинувачень авторів-людей.

Практичне значення одержаних результатів полягає у створенні готового до промислового використання інструментарію для систем контент-модерації, інформаційної безпеки та платформ перевірки академічної доброчесності.

Ключові слова: Гібридний метод, Бінарна класифікація, Нейронні мережі, Алгоритми машинного навчання, Текстовий документ, Ансамблі класифікаторів, Рівень хибнопозитивних результатів, Інформаційна безпека, Методи прийняття рішень, Виявлення аномалій.

ABSTRACT

Nykonenko, A. O. A Hybrid Method for the Identification of Automatically Generated Natural Language Texts – Qualifying scientific work on the rights of a manuscript.

Dissertation for the degree of Doctor of Philosophy in Computer Science (specialty 122, 12 Information Technology). Cherkasy State Technological University, Ministry of Education and Science of Ukraine, Cherkasy, 2026.

This dissertation addresses a pressing scientific and practical problem: the development of a hybrid method for identifying automatically generated natural-language texts that is resistant to transformational masking attacks and epistemic uncertainty.

The widespread adoption of large language models (LLMs), in addition to its positive effects, has given rise to a number of challenges, including the creation and dissemination of misinformation, the use of AI in social engineering, the manipulation of the public, and the undermining of academic integrity. In scenarios of unethical LLM use, their greatest advantage - the ability to generate texts that are lexically and structurally indistinguishable from those written by humans - becomes the greatest threat, as detecting artificially generated texts is impossible without the use of specialized tools. Addressing new challenges in information security requires the development of reliable tools for identifying artificially generated content - AI detectors. The study indicated that commercially available systems for masking generated data perform deep structural and semantic transformations of texts, which hide the AI trace and significantly reduce the effectiveness of existing detectors in their ability to recognize generated texts. On the other hand, most modern solutions face the problem of epistemic uncertainty when analyzing data outside the training distribution, such as texts written by non-native speakers, which leads to an increased rate of false positives and undermines trust in detection methods in general.

In consideration of the above, the development of a mathematical model of an AI trace and a hybrid method for identifying automatically generated natural-language texts

that is resistant to transformational masking attacks and epistemic uncertainty is a pressing scientific and applied problem of significant importance for the evolution of content moderation systems, information security, and academic integrity platforms.

To define the research objectives, we systematized and analyzed existing approaches to the identification of automatically generated texts. It was found that despite the existence of a wide range of approaches to AI detection, each of them has both serious theoretical limitations regarding applicability and vulnerabilities to masking scenarios, which makes the process of identifying generated data fairly unreliable. It has been proven that AI detection based solely on a single feature level (lexical, syntactic, or semantic) has shortcomings both in terms of vulnerability to transformational attacks and in terms of a high rate of false positives. The imperfect accuracy of existing classifiers, theoretical and practical limitations regarding their application, the large number of attacks and types of obfuscation, and the lack of robust and fast tools for interpreting results point to the need to develop both a reliable hybrid method for identifying automatically generated texts and an interpretation model for visualizing detection evidence.

The vulnerability of modern detection methods is directly linked to the interpretation of AI trace as a static object, which can be easily manipulated to affect detection accuracy by introducing perturbations through paraphrasing, character substitution, or the release of new versions of models or architectures. To improve the robustness of AI detection against transformational attacks, we propose, instead of scaling the detector (increasing the model size and the volume of training data), to modify the concept of the AI trace to obtain a robust multidimensional object that includes features at three levels: semantic, stylistic, and structural. Specifically, a mathematical model of the AI trace in a multidimensional feature space $\mathbf{F} = \mathbf{F}_{lex} \times \mathbf{F}_{syn} \times \mathbf{F}_{sem}$ (lexical, syntactic, and semantic) has been developed, which formalizes it as a vector $T_{AI}(X) = \Phi(f_{lex}(X), f_{syn}(X), f_{sem}(X))$. The main levels of analysis were also introduced: lexical ($f_{lex}(X)$), which captures the probabilistic smoothing of text by generative models through perplexity and burstiness; syntactic ($f_{syn}(X)$), which measures the structural monotony of the generated text via the Kullback-Leibler divergence for

N-grams of POS tags relative to a human benchmark, as well as via the variance in the depth of syntactic trees; semantic ($f_{sem}(X)$), which detects abnormally high local consistency of embeddings of neighboring sentences and global thematic drift.

Additionally, we have refined the conceptual model for assessing the robustness of AI trace detection systems. By introducing the formalized concept of a Breakdown point and integrating a mathematical model of a multi-level feature space, this model enables a transition from existing empirical approaches to robustness assessment on fixed datasets to the establishment of a direct mathematical relationship between the intensity of structural text transformations (replacement rate RR), the preservation of its semantic integrity (I_{sem}), and the probability of detection (P_{det}). The model identifies three areas that demonstrate the effectiveness of an attack and proposes a method for assessing the robustness of the detector under conditions of uncertainty.

For the first time, a hybrid method has been developed for identifying automatically generated natural-language texts, which, unlike classical detectors that treat AI traces as static and easily modifiable objects, by using a cascaded ensemble of independent classifiers with additional validation mechanisms, allows for the analysis of cross-level consistency of features through adaptive weighting and the calculation of the $S_{mismatch}$ discrepancy signal. Unlike the traditional simple combination of features into a single vector, the proposed method distributes the analysis among three independent modules: lexical (M_{lex}), syntactic (M_{syn}), and semantic (M_{sem}). Each module generates its own independent assessment, which is passed to the meta-classifier for the final decision. Since it is extremely difficult for obfuscation methods to simultaneously and proportionally alter style, syntax, and semantics during a transformational attack, any attack creates a gap between the levels. This gap is analyzed through an adaptive weighting mechanism, which allows the use of two-level weight logic: $w_j(X) = w_j^{stat} \cdot \alpha_j(X)$, where the learned statistical weight is multiplied by a dynamic reliability coefficient. Depending on the type of detected attack, the algorithm adjusts the degree of trust to different modules, so that an attempt to bypass the detector at one level automatically triggers an increase in the

probability of detection at other levels. This approach captures the cognitive dissonance between the lexical, syntactic, and semantic levels of the text when transformational attacks are used, making attempts at obfuscation at one level a trigger for enhanced detection at another.

Along with this, a method for ensuring the robustness of natural language processing systems under conditions of epistemic uncertainty and out-of-distribution data has been further developed. Classical detection methods generate point probabilities with excessive confidence and do not provide information on whether the analyzed text matches texts from the training set. This makes them critically vulnerable to out-of-distribution data, such as texts written by non-native speakers, academic texts, or non-standard writing styles, leading to an increase in the number of false positives. To address this issue, an extension has been developed that allows converting heuristic point predictions into mathematically grounded sets of predictions $\Gamma^\epsilon(x)$. By computing non-conformity measures on a calibration sample, the ICP algorithm guarantees that the true class of the text will belong to the generated set with a specified statistical confidence level of $1 - \epsilon$. This transforms the point predictions of the meta-classifier into sets of predictions with guaranteed statistical confidence, which significantly reduces the false positive rate, enables global anomaly detection, and generates empty sets instead of misclassifications.

Furthermore, this study presents for the first time an interpretation model for justifying the detection evidence of explainable AI, which, unlike the classical iterative LIME/SHAP methods, does not require significant computational resources and demonstrates stability on text data. Its creation solves the black-box problem inherent in classical detectors, whose results often lack the transparency required for expert decision-making. By utilizing the internal weights of the meta-classifier (w_j^{stat}) and adaptive confidence coefficients $\alpha_j(X)$, this model enables the generation of three levels of attribution: global, local, and an Evidence Map. The global attribution profile allows us to determine which features play the greatest role in the overall hybrid text analysis. The local attribution profile shows the influence of each of the meta-classifier's parameters on

the final decision regarding a specific text, while the interactive Evidence Map visualizes anomalies in lexical perplexity and structural threats.

An experimental validation of the developed mathematical models and the hybrid method for identifying generated texts was conducted on specially constructed test data. In addition to texts written by humans and generated by Instant models, the test data also includes texts from MoE and Reasoning models, as well as texts following transformational attacks by the T5 and DIPPER paraphraser and commercial masking systems. It has been experimentally proven that for the SOTA detector Fast-DetectGPT, the Breakdown point occurs even with minimal changes (replacement rate $RR = 0.03$), whereas the results of the hybrid method confirm a significant increase in detection robustness, without reaching the Breakdown point even when shifted to the right to $RR > 0.9$. AI trace modeling and extraction methods for modern MoE and Reasoning architectures have been experimentally investigated and adapted. Unlike approaches that focus on microstructural generation artifacts through KDE kernel density estimation or FFT spectral analysis of rhythm, the proposed methods provide a reliable way to identify the model's cognitive trace.

Based on the developed mathematical model and hybrid method, a software package for identifying automatically generated texts has been created. The system has been deployed using the AWS cloud infrastructure, which ensures its high fault tolerance, scalability, and the ability to process requests in parallel when analyzing large datasets in real time. The overall asymptotic complexity of the system, consisting of the hybrid method and the interpretation subsystem, is $O(N^2)$, and the average processing time for a single document in the AWS cloud environment is 0.33 seconds. Unlike existing solutions, which demonstrate a critical drop in detection completeness in Region 2 (the zone of successful attacks) during transformational attacks, the hybrid method, due to its cascading ensemble architecture and inter-level consistency analysis, allows for the complete elimination of the zone of successful masking of generated content. The proposed meta-classifier architecture, the use of canonicalization algorithms, and the adaptive feature weighting mechanism allow us to achieve an F1 score of 0.92 on stationary data and maintain a Recall of 88.77% while strictly limiting the false positive rate (FPR) to 1%.

When countering modern commercial obfuscation systems (StealthGPT, AIHumanize, Phrasly), the system outperforms global counterparts, achieving an average recall of 77.16% and the lowest attack success rate (ASR = 22.84%) among comparable systems.

The implementation of the ICP framework methodology made it possible to significantly reduce the false positive rate (FPR) during the analysis of texts written by non-native speakers (using the IELTS dataset as an example) from 4.81% to 0.98%. The integration of the ICP framework enabled the system to recognize the epistemic uncertainty of these texts and generate empty sets in complex cases. The resulting Rescue Rate was 79.71%, indicating a significant reduction in the risk of unfair accusations against human authors.

The practical significance of these results lies in the creation of a ready-to-use toolkit for content moderation systems, information security, and academic integrity verification platforms.

Keywords: Hybrid method, Binary classification, Neural networks, Machine learning algorithms, Text document, Classifier ensembles, False positive rate, Information security, Decision-making methods, Anomaly detection.

Список опублікованих праць за темою дисертації:

– статті в іноземних виданнях, в яких опубліковані основні наукові результати дисертації:

1. Faure, Emil, and Andrii Nykonenko. "Analyzing the nature of AI footprint: noise-to-text method." *Proceedings of the Computational Intelligence Application Workshop (CIAW 2024)*, Lviv, Ukraine, October 10-12, 2024. CEUR Workshop Proceedings, Vol-3861, paper 14. Online [CEUR-WS.org/Vol-3861/paper14.pdf](https://ceur-ws.org/Vol-3861/paper14.pdf). ISSN 1613-0073. DOI: <https://doi.org/10.13140/RG.2.2.15592.02561>. Видання індексується в наукометричній базі Scopus та DBLP.

Особистий внесок автора полягає у розробленні нового методу ітераційної трансформації даних “noise-to-text” для дослідження природи цифрового відбитку штучного інтелекту, проведенні експериментальних досліджень із використанням

моделі gpt-3.5-turbo та аналізі чутливості сучасних систем детекції ШІ (Turnitin, GPTZero, Originality.ai та інших) і становить 0,5 друк. арк.

– статті у наукових фахових виданнях України, в яких опубліковані основні наукові результати дисертації:

2. Nykonenko, A. "The impact of artificial intelligence on modern education: prospects and challenges." *Artificial Intelligence* 2 (2023): 10-15. ISSN 2710-1673. ONLINE ISSN 2710-1681. DOI: <https://doi.org/10.15407/jai2023.02.010>. Фаховий категорії Б.

3. Nykonenko, A. "How text transformations affect AI detection." *Artificial Intelligence* 4 (2024): 233-241. ISSN 2710-1673. ONLINE ISSN 2710-1681. DOI: <https://doi.org/10.15407/jai2024.04.233>. Фаховий категорії Б.

4. Никоненко, А. О. "Систематичний огляд досягнень в галузі LLM від GPT-3 до міркуючих агентів." *Stuc. intelekt* 30.3 (2025): 32-43. ISSN 2710-1673. ONLINE ISSN 2710-1681. DOI: <https://doi.org/10.15407/jai2025.03.032>. Фаховий категорії Б.

– наукові праці, що засвідчують апробацію матеріалів дисертації:

5. Никоненко А.О. Штучний інтелект як засіб трансформації освіти // Тези доповідей III Міжнародної науково-практичної конференції «Інформаційні технології в освіті та науці» (ІТОН-2023), м. Запоріжжя-Мелітополь, 25-26 травня 2024 р., с.328-330.

6. Никоненко А.О. Підходи до визначення текстів згенерованих штучним інтелектом // Тези доповідей VII Міжнародної науково-практичної конференції «Інформаційні технології в освіті, науці і техніці» (ІТОНТ-2024), ЧДТУ, м. Черкаси, 23-24 травня 2024 р., с.142-144.

7. E. Faure, A. Nykonenko. Noise-to-text method analysis for evaluating AI-generated texts. Proceedings of the 1st International Scientific and Practical Conference «COMPUTATIONAL INTELLIGENCE AND SMART SYSTEMS» (CISS-2024). 10–12 October 2024, Lviv, Ukraine.

Особистий внесок автора полягає в обґрунтуванні концепції та розробленні алгоритму оцінювання систем аналізу ШІ-генерованих текстів на основі методу “noise-to-text”, проведенні серії експериментів із трансформації даних за допомогою LLM та систематизації результатів тестування сучасних детекторів і становить 0,15 друк. арк.

8. Никоненко А.О. Вплив трансформацій тексту на виявлення штучного інтелекту // Тези доповідей XXIV міжнародної науково-технічної конференції «Штучний інтелект та інтелектуальні системи - АІІС'2024», м. Київ, Україна, 18-19 жовтня 2024 р.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ.....	18
ВСТУП.....	25
РОЗДІЛ 1 АНАЛІЗ ПОТОЧНОГО СТАНУ ПРЕДМЕТНОЇ ОБЛАСТІ. ПОСТАНОВКА НАУКОВОГО ЗАВДАННЯ.....	34
1.1. Основи генерації та ідентифікації природномовних текстів.....	34
1.1.1. Класифікація моделей генерації тексту.....	35
1.1.2. Визначення сліду ШІ та його прояви.....	39
1.2. Аналіз існуючих підходів до детекції згенерованих текстів.....	44
1.2.1. Статистичні та лінгвістичні методи ідентифікації.....	45
1.2.2. Методи ідентифікації, засновані на нейромережах.....	48
1.2.3. Ключові обмеження існуючих методів детекції.....	53
1.3. Постановка завдання дослідження.....	56
1.3.1. Аналіз вразливостей детекторів до трансформаційних атак.....	56
1.3.2. Вимоги до нового покоління методів детекції.....	59
1.3.3. Задача дослідження.....	61
1.4. Висновки до розділу 1.....	63
Список використаних джерел до розділу 1.....	65
РОЗДІЛ 2 МОДЕЛІ ТА ГІБРИДНИЙ МЕТОД ІДЕНТИФІКАЦІЇ СЛІДУ ШІ.....	71
2.1. Формалізація сліду ШІ в багатовимірному просторі ознак.....	72
2.1.1. Математична модель сліду ШІ.....	72
2.1.2. Аналіз негаусового розподілу ознак генеративних моделей.....	82
2.2. Концептуальна модель стійкості детекції.....	88
2.2.1. Метрики семантичної цілісності та функції втрат.....	88
2.2.2. Визначення критичних меж маскуванню.....	90
2.2.3. Метод оцінки робастності детектора в умовах невизначеності.....	93
2.3. Розробка гібридного методу ідентифікації.....	96
2.3.1. Архітектура гібридного методу.....	97
2.3.2. Протидія структурним атакам.....	102
2.3.3. Алгоритм прийняття фінального рішення.....	105
2.4. Інтерпретаційна модель обґрунтування результатів детекції.....	109
2.4.1. Розробка моделі профілю атрибуції. Локальні атрибути.....	110
2.4.2. Глобальні атрибути.....	113
2.4.3. Карта доказів.....	115
2.5. Висновки до розділу 2.....	117
Список використаних джерел до розділу 2.....	121
РОЗДІЛ 3 ЕКСПЕРИМЕНТАЛЬНА ВАЛІДАЦІЯ МОДЕЛЕЙ.....	124

3.1. Формування та характеристика наборів даних.....	125
3.1.1. Дані, створені людьми.....	125
3.1.2. Дані, створені ШІ.....	129
3.2. Моделювання та синтез трансформаційних атак.....	136
3.3. Метрики оцінювання ефективності та робастності.....	141
3.4. Перевірка математичних моделей сліду ШІ.....	146
3.4.1. Валідація характеру сліду ШІ для МоЕ моделей.....	146
3.4.2. Валідація характеру сліду ШІ для Reasoning моделей.....	152
3.4.3. Експериментальна перевірка моделі Точки зламу.....	157
3.5. Висновки до розділу 3.....	167
Список використаних джерел до розділу 3.....	169
РОЗДІЛ 4 ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА ВАЛІДАЦІЯ ГІБРИДНОГО МЕТОДУ ІДЕНТИФІКАЦІЇ СЛІДУ ШІ У СКЛАДНИХ УМОВАХ.....	172
4.1. Аналіз ефективності та робастності гібридного методу.....	173
4.1.1. Базова точність та порівняння із SOTA на стаціонарних даних.....	173
4.1.2. Стійкість до трансформаційних атак та зміщення Точки зламу.....	178
4.1.3. Ефективність проти комерційних систем маскування.....	183
4.1.4. Внесок модулів у робастність.....	185
4.2. Робота в умовах невизначеності та швидкодія.....	188
4.2.1. Валідація методу ICP для ідентифікації OOD-даних.....	188
4.2.2. Аналіз обчислювальної складності та часових витрат.....	194
4.3. Архітектура та програмна реалізація системи ідентифікації.....	198
4.3.1. Модульна структура та логіка каскадного ансамблю.....	199
4.3.2. Вибір технологічного стека та реалізація ICP.....	202
4.4. Реалізація інтерпретаційної моделі обґрунтування.....	205
4.4.1. Програмна генерація профілю атрибутів та Карти доказів.....	205
4.4.2. Аналіз прикладних кейсів.....	208
4.5. Висновки до розділу.....	215
Список використаних джерел до розділу 4.....	218
ВИСНОВКИ.....	220
ДОДАТКИ.....	227
ДОДАТОК А Список опублікованих праць за темою дисертації.....	228
ДОДАТОК Б Розподіл моделей за наборами даних.....	230
ДОДАТОК В Розподіл системних prompts за наборами даних.....	233
ДОДАТОК Г Характеристики датасету атак типу А.....	243
ДОДАТОК Д Розподіл Fast-DetectGPT AI Score для датасету атак типу В.....	244

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

LLM	Large Language Model - Велика Мовна Модель
GPT	Generative Pre-trained Transformer - Генеративний Попередньо навчений Трансформер
SOTA	State-Of-The-Art - Найсучасніший рівень розвитку технологій
NN	Neural Networks - Нейронні мережі
AGI	Artificial General Intelligence - Загальний або Повний штучний інтелект, що має когнітивні здібності, близькі до людських
AI / ШІ	Artificial Intelligence / Штучний Інтелект
Benchmark	Набір стандартизованих тестів, що використовується в індустрії ШІ для оцінки якості моделей.
Inference / Інференс	Виведення/висновок - процес видачі прогнозів натренованою моделлю
Prediction / Прогноз	Для NN це результат Inference
MoE	Mixture of Experts - Мікс Експертів, архітектура LLM, що дає змогу використовувати лише частину нейронів під час інференсу
Prompt / Промпт / Prompting style	Підказка - Спосіб постановки задачі LLM природною мовою
Chain-of-thought (CoT)	Ланцюжок думок - Спосіб генерації відповіді LLM шляхом послідовних проміжних кроків, що приводять до остаточної відповіді

Multi-step Reasoning / Reasoning Agent	Багатокрокове міркування - Prompting style типу Chain-of-thought, що вкладений в модель на системному рівні
Open Source / Open Weight	Публічно доступний (для програмного забезпечення / моделей)
Reasoning Trace	Слід Міркування - ланцюжок роздумів згенерований моделлю при використанні Chain-of-thought
Multimodal Models	Багатомодальні Моделі - моделі, що підтримують кілька видів вхідних даних (наприклад, текст і аудіо; або код, зображення і відео)
Grounded Language	Заземлена Мова - спеціальне представлення слів в Multimodal Models, що зв'язує слово з його відповідником у реальному світі
ML	Machine Learning - Машинне Навчання
Drift / Дрифт	Зсув - у ML означає зниження якості моделі з плином часу через наростання різниці між тренувальними та реальними даними
AI Detector / AI Детектор	Детектор ШІ
AI Trace / AI Footprint / Слід ШІ	Набір характеристик тексту, що притаманний даним, згенерованим LLM, на відміну від тексту написаного людиною
Confidence	Впевненість - наскільки модель впевнена в своєму прогнозі
Perplexity / перплексія	Розгубленість/Невизначеність - у випадку LLM характеризує наскільки модель невпевнена в

наступному токени, можна трактувати як зворотне значення до Confidence

Burstiness	Вибуховість - міра, що характеризує варіативність тексту
Adversarial attack	Змагальні атаки - тип атаки на ML модель, що характеризується незначними змінами вхідних даних для маніпулювання виходом (prediction)
Тексти певного домену	Це інформація, що стосується конкретної теми чи галузі
Benchmark / Бенчмарк	Стандартний набір тестових даних і метрик, що використовується для порівняння якості різних моделей
Bypassing/ Humanization/ Masking / Маскування	Використання різних adversarial attacks з метою зміни створеного LLM тексту таким чином, щоб детектор визнав його написаним людиною
Features / Ознаки	Ознаки або властивості - індивідуальні вимірювані характеристики в наборі даних, які використовуються при навчанні моделі
N-grams / N-грами	Послідовності з N елементів, як правило, символів, токенів чи слів
POS Tag / (Part-Of-Speech) Tag	Морфологічна розмітка
Feature engineering / Інженерія ознак	Процес виділення числових ознак для навчання моделей з неструктурованих даних
Authorship Attribution	Визначення авторства тексту

RLHF	Reinforcement Learning from Human Feedback - Навчання з підкріпленням на основі зворотного зв'язку від людини - метод узгодження поведінки LLM з людськими очікуваннями
Humanized / Bypassed	Гуманізація текстів - зміна характеристик створених AI текстів таким чином, щоб вони виглядали більш “людиноподібними”
Fine-Tuned / Fine-Tuning	Доналаштовані / Тонко налаштовані - попередньо навчені моделі, що адаптовані до виконання більш вузької задачі
Out-of-Distribution / OOD	Дані, що належать до іншого статистичного розподілу, ніж дані, що було використано для тренування моделі
Робастність	Це здатність системи зберігати свою функціональність, попри збурення, похибки, невизначеності або зміни в її роботі
Watermarking / Водяний знак	Метод внесення в текст спеціальних маркерів на етапі генерації
XAI / Explainable Artificial Intelligence	Пояснюваний ІІІ - надання людині можливості зрозуміти, яким чином модель дійшла певного висновку, що дає змогу перевірити висновки та підвищує довіру. Як правило, протиставляється методу чорної скриньки.
LIME	Local Interpretable Model-agnostic Explanations - Локальні Інтерпретовані Незалежні від Моделі Пояснення
SHAP	SHapley Additive exPlanations - Адитивні Пояснення

Шеплі

Black-Box	Чорна Скринька - в комп'ютерних науках означає взаємодію з системою, про принципи роботи якої попередньо нічого не відомо
AI Score	Оцінка ШІ або Оцінка виявлення АІ - в залежності від методу детекції вказує або на відсоток тексту згенерованого АІ, або на ймовірність того, що весь текст було згенеровано АІ
False Negative / FN	Хибно-Негативні спрацювання класифікатора - Помилки другого роду
False Positive / FP	Хибно-Позитивні спрацювання класифікатора - Помилки першого роду
TPR / Recall	True Positive Rate / Частка Істинно Позитивних Результатів - вимірює частку позитивних випадків, які були правильно ідентифіковані моделлю
FPR	False Positive Rate / Рівень Хибнопозитивних Результатів - частка негативних випадків, які помилково класифікуються як позитивні
POS/Part-of-Speech	Частина мови (іменник, прикметник і т. ін.)
Divergence / дивергенція	Розходження - міра, що вказує наскільки один розподіл імовірностей відрізняється від іншого, еталонного розподілу імовірностей
Dependency Parse Tree	Дерево розбору залежностей - в NLP представлення граматичної структури речення через зв'язки між словами

VSM	Vector Space Models / Векторні Моделі - спосіб представлення документів через вектори спільного векторного простору
Embedding / Ембедінг	Вбудовування/Вкладання слів - в NLP форма векторного представлення тексту
Kernel Density Estimation (KDE)	Оцінка щільності ядра - це статистичний метод для оцінки неперервної щільності розподілу випадкової величини
Ексцес (kurtosis)	У статистиці – це характеристика форми розподілу даних, яка показує “крутість” або “гостроту” вершини розподілу порівняно з нормальним розподілом (дзвоноподібною кривою)
Fast Fourier Transform / FFT	Швидке Перетворення Фур'є - швидкий алгоритм обчислення дискретного перетворення Фур'є, дає змогу переводити сигнал із часової області у частотну область
Uncertainty Quantification / UQ	Квантифікація Невизначеності - це процес кількісного оцінювання та мінімізації невизначеностей в обчислювальних та реальних системах
Conformal Forecasting / CF	Конформне Прогнозування - фреймворк, який дає змогу отримувати статистично обґрунтовані оцінки впевненості (інтервал з рівнем достовірності) для прогнозів моделі машинного навчання
Validity	Гарантована достовірність - в Conformal Forecasting статистична гарантія того, що інтервал містить істинне значення з заздалегідь заданою імовірністю

Inductive Conformal Prediction / ICP	Індуктивне Конформне Передбачення - альтернативний до CF метод отримання статистичних гарантій точності більш ефективним способом
Attribution Profile	Профіль атрибуції - список ключових факторів, що впливають на рішення моделі-детектора разом з їх силою впливу

ВСТУП

1. Актуальність теми

Зародження статистичних методів та поступовий перехід від них до машинного навчання відбулися в середині 20 століття. 1980-ті роки характеризувалися великою кількістю теоретичних розробок, а в 1990-ті наявні обчислювальні потужності дали змогу проводити практичні експерименти та створити перші комерційні застосунки з використанням як лінійних моделей, так і рекурентних нейронних мереж. У 2000-х роках обсяг наявних даних і обчислювальних потужностей дав змогу галузі набути широкого комерційного застосування. Нова дисципліна отримала назву Data Mining і фокусувалася переважно на роботі з табличними даними, обробці природномовних текстів і зображень; основну практичну популярність отримали методи навчання без вчителя (на кшталт кластеризації) через брак специфічних розмічених даних та методи класичного машинного навчання (через обмеженість як даних, так і обчислювальних ресурсів). На початку 2010-х років стався прорив у галузі обробки зображень через застосування глибокого навчання, а наприкінці 2010-х - прорив в обробці мови через винахід архітектури Transformers. Також у 2018 році з'явилася перша генеративна версія трансформерів - модель GPT-1, яка на той момент була досить повільною моделлю з обмеженою якістю генерації та спектром використання.

Справжня революція відбулась у 2020-х роках, коли технології ШІ стали доступними широкому загалу. Весь розвиток ШІ йшов шляхом зниження порогу входу в галузь: ранні класифікатори дослідникам потрібно було створювати самостійно, потім з'явилися пакети програмного забезпечення з реалізацією популярних архітектур, ще пізніше - готові натреновані моделі та фреймворки, що суттєво спрощували процеси тренування, тюнінгу і використання моделей. Але лише у 2020-х роках з'явилася можливість працювати з моделями напряму, комунікуючи природною мовою замість коду, а пізніше і шляхом передачі зображень та аудіо/відео. Це повністю прибрало фактор наявності спеціальних знань як необхідну умову для роботи з ШІ-моделями. Відповідно, LLM отримали широке

розповсюдження і застосування у всіх галузях людського життя (від домашнього лікаря до юриста і психолога), навіть у тих сценаріях, до використання в яких вони не були призначені.

Широкий доступ до технології та простота її використання, крім позитивних наслідків, призвели до появи цілої низки негативних застосувань та зловживань: підроблення інформації, маніпулювання суспільною думкою, використання в соціальній інженерії та академічній недоброчесності. Ситуація ускладнюється тим, що останні покоління LLM здатні генерувати тексти, які лексично та структурно неможливо відрізнити від текстів, написаних людиною. Боротьба з новими викликами в інформаційній безпеці вимагає створення надійних інструментів ідентифікації штучно згенерованого контенту - детекторів ШІ.

Наразі існує велика кількість детекторів ШІ, що демонструють високу ефективність на даних, близьких до їх тренувального розподілу. Проте постійний вихід нових версій LLM та поява таких архітектур, як MoE та Reasoning, атаки перефразуванням та створення комерційних систем маскування (зокрема, StealthGPT, AINhumanize, Phrasly) створюють значні виклики для традиційних методів ідентифікації ШІ. Наведені фактори призводять до глибоких структурних та семантичних трансформацій, що веде до приховування сліду ШІ та суттєво знижує ефективність наявних детекторів щодо їх можливостей розпізнавання згенерованих текстів. З іншого боку, більшість наявних рішень стикається з проблемою епістемічної невизначеності під час аналізу даних поза навчальним розподілом (OOD), наприклад, текстів, написаних неносіями мови. Детектор не здатний коректно обробляти дані, які відрізняються від його тренувальної вибірки, що призводить до високого рівня хибнопозитивних спрацювань і ризику несправедливих звинувачень авторів-людей.

Суттєвий внесок у розвиток методів штучного інтелекту, обробки природної мови та побудови базових архітектур виявлення згенерованого контенту зробили такі провідні закордонні науковці, як Крістофер Меннінг, Єджин Чой, Ерік Мітчелл, Тіанксінг Хе та інші розробники сучасних систем детекції ШІ. Серед вітчизняних вчених, чії фундаментальні та прикладні праці заклали базу розвитку

математичного моделювання, системного аналізу, інформаційної безпеки та інтелектуального аналізу даних, варто відзначити дослідження Ю. В. Триуса, М. В. Підгорного та інших провідних фахівців у галузі комп'ютерних наук та системного аналізу.

Попри значні досягнення згаданих вчених, проблема забезпечення комплексної робастності систем ідентифікації згенерованих текстів в умовах активного маскування залишається недостатньо розв'язаною. Зокрема, бракує формалізованого опису математичної моделі, що відображала б взаємозв'язок між втратою семантичної цілісності тексту та падінням ймовірності детекції під час маскувальних атак. Потребують удосконалення підходи до використання методів квантифікації невизначеності для мінімізації хибних спрацювань на даних поза навчальним розподілом. Крім того, існує гостра потреба у розробці обчислювально ефективних методів пояснювального ШІ для підтримки прозорості прийнятих рішень. З огляду на вищезазначене, розробка математичної моделі сліду ШІ та гібридного методу ідентифікації автоматично згенерованих природномовних текстів, стійкого до трансформаційних атак маскування та епістемічної невизначеності, є актуальною науково-прикладною задачею, що має важливе значення для розвитку систем контент-модерації, інформаційної безпеки та платформ академічної доброчесності.

2. Мета і завдання дослідження

Мета: Підвищення точності та робастності ідентифікації автоматично згенерованих природномовних текстів в умовах маскування та епістемічної невизначеності шляхом розробки гібридного методу, що інтегрує лексичні, синтаксичні та семантичні ознаки сліду ШІ.

Завдання дослідження:

1. Провести систематизацію та аналіз існуючих підходів до ідентифікації автоматично згенерованих текстів, виявити їхні вразливості до методів маскування та проблеми генералізації.

2. Розробити математичну модель сліду ШІ в багатовимірному просторі ознак та побудувати на її основі концептуальну модель Точки зламу, яка відображає математичний взаємозв'язок між втратою семантичної цілісності тексту та падінням ймовірності детекції, для прогнозування чутливості системи до трансформаційних атак.
3. Розробити гібридний метод ідентифікації згенерованих текстів на основі каскадного ансамблю класифікаторів з механізмами адаптивного зважування та аналізу міжрівневого дисонансу ознак.
4. Удосконалити метод забезпечення робастності систем ідентифікації згенерованих текстів шляхом інтеграції фреймворку індуктивного конформного передбачення (ICP) для квантифікації епістемічної невизначеності, розпізнавання даних поза навчальним розподілом та мінімізації хибнопозитивних спрацювань.
5. Розробити інтерпретаційну модель пояснюваного ШІ на основі вагових коефіцієнтів моделі-детектора для генерації профілів атрибуції та карт доказів, що відображають вплив ознак різних рівнів (лексичних, синтаксичних, семантичних) на кінцеве рішення системи.
6. Розробити програмний комплекс та провести експериментальні дослідження для оцінювання точності ідентифікації, обчислювальної ефективності та робастності до трансформаційних атак порівняно із сучасними аналогами.

3. Об'єкт і предмет дослідження

Об'єкт дослідження: Процеси ідентифікації автоматично згенерованих природномовних текстів у складних інформаційних умовах (зокрема, в умовах використання комерційних систем маскування, трансформаційних атак та невизначеності джерела генерації).

Предмет дослідження: Моделі та методи ідентифікації автоматично згенерованих природномовних текстів у складних інформаційних умовах.

4. Методи дослідження

Методологічною основою дослідження є системний підхід, теорія прийняття рішень та математичне моделювання випадкових процесів:

1. Системний аналіз та системний підхід - для формалізації предметної області та представлення процесу детекції як складної взаємодії об'єкта (текст), середовища (LLM, системи маскування) та детектора.
2. Методи обробки природної мови та інтелектуального аналізу даних - для виділення багаторівневих лінгвістичних ознак (екстракція семантичних ембедингів, парсинг дерев залежностей, аналіз лексичної ентропії) та формування механізму семантичної валідації.
3. Математична статистика та теорія ймовірностей - для формалізації математичної моделі сліду ШІ, проведення спектрального аналізу (FFT), а також застосування індуктивного конформного передбачення (ICP) для розрахунку довірчих інтервалів.
4. Чисельне моделювання та комп'ютерний експеримент - для реалізації гібридного методу, моделювання трансформаційних атак (перифразування, структурні модифікації) та валідації ефективності методу на наборах даних.

5. Наукова новизна одержаних результатів

Наукова новизна дисертаційної роботи полягає у розробці та теоретико-методичному обґрунтуванні моделей і гібридного методу ідентифікації автоматично згенерованих природномовних текстів, що підвищує стійкість детекції до трансформаційних атак та забезпечує інтерпретованість прийнятих рішень в умовах невизначеності.

Вперше:

- розроблено гібридний метод ідентифікації автоматично згенерованих текстів, який, завдяки застосуванню каскадного ансамблю з механізмами адаптивного зважування та аналізу міжрівневої узгодженості ознак, дає змогу виявляти дисонанс між лексичним, синтаксичним та семантичним рівнями тексту, забезпечуючи стійкість до комерційних систем

маскування і зміщуючи Точку зламу детектора до критичних меж втрати семантичної цілісності;

- розроблено інтерпретаційну модель обґрунтування доказів детекції, яка, на відміну від обчислювально ресурсомістких ітеративних ХАІ-методів, використовує внутрішні ваги метакласифікатора для прямої генерації глобального та локальних профілів атрибуції та карт доказів, що дає змогу знизити часову складність побудови пояснень до $O(1)$ та забезпечити прозорість рішень детектора ІІІ.

Одержали подальший розвиток:

- метод забезпечення робастності моделей обробки природної мови в умовах агресивного маскування згенерованого тексту та епістемічної невизначеності шляхом інтеграції фреймворку індуктивного конформного передбачення (ICP), що дає змогу ефективно виявляти аномалії, генерувати порожні множини замість хибних висновків та кардинально знижувати рівень хибнопозитивних спрацювань у сценаріях аналізу даних поза навчальним розподілом.

Удосконалено:

- концептуальну модель оцінки робастності систем детекції шляхом введення формалізованого поняття Точки зламу, яке встановлює математичний взаємозв'язок між інтенсивністю структурних змін тексту, збереженням його семантичної цілісності та ймовірністю виявлення, що дає змогу використовувати модель як прогностичний інструмент оцінки чутливості детектора в умовах невизначеності та активного маскування згенерованого контенту.

6. Практичне значення одержаних результатів

1. На основі запропонованих моделей розроблено програмний комплекс ідентифікації автоматично згенерованих текстів, який забезпечує підвищену точність та стійкість до маскування порівняно з передовими світовими

аналогами, що робить його ефективним інструментом для промислового використання у системах контент-модерації та перевірки доброчесності.

2. Програмна реалізація гібридного методу дала змогу експериментально довести його перевагу над передовими світовими аналогами (зокрема, Fast-DetectGPT, Binoculars, Radar) із досягненням показника $F1 = 0.92$ на даних із відомим розподілом та збереженням повноти виявлення (Recall) на рівні 88.77% в умовах жорсткого обмеження рівня хибних спрацювань ($FPR = 1\%$).
3. Розроблено та впроваджено методику застосування фреймворку ICP, яка дала змогу суттєво знизити рівень хибнопозитивних спрацювань під час аналізу текстів з високим рівнем епістемічної невизначеності; зокрема, на складних текстах неносіїв мови (IELTS), показник FPR було знижено з 4.81% до 0.98%, що підтверджує здатність системи глобально мінімізувати ризики несправедливих звинувачень авторів-людей незалежно від специфіки контенту.
4. Експериментально підтверджена ефективність програмної реалізації гібридного методу у протидії комерційним засобам маскування тексту (зокрема, StealthGPT, AIHumanize, Phrasly) із доведеним зниженням рівня успішності атак (ASR) до 22.84%, що є найнижчим показником серед досліджуваних аналогів, дає змогу рекомендувати його для впровадження у платформи перевірки академічної доброчесності, системи контент-модерації та інструменти кібербезпеки. Водночас інтегровані засоби пояснювального ШІ, зокрема карти доказів, надають експертам вичерпну аналітичну базу для ухвалення фінальних рішень.

7. Особистий внесок здобувача.

Дисертаційна робота є самостійним завершеним науковим дослідженням здобувача. Усі результати, що виносяться на захист, зокрема розробка гібридного методу, математичної моделі Точки зламу, впровадження фреймворку ICP та інтерпретаційної моделі ХАІ, отримані автором особисто. У наукових працях,

опублікованих у співавторстві (перелік наведено у Додатку Б), особистий внесок здобувача полягає в такому:

- у статті [1] автором особисто розроблено метод ітераційної трансформації даних “noise-to-text” для дослідження природи цифрового відбитка ШІ; проведено серію експериментальних досліджень із використанням моделі GPT-3.5-Turbo; виконано порівняльний аналіз чутливості сучасних систем детекції до структурних змін тексту;
- у матеріалах конференції [7] здобувачем обґрунтовано концепцію та розроблено алгоритм оцінювання систем аналізу ШІ-генерованих текстів на основі методу “noise-to-text”; проведено експерименти з трансформації даних за допомогою LLM та систематизовано результати тестування детекторів.

Усі інші наукові праці, наведені у списку опублікованих робіт за темою дисертації (Додаток А), а саме статті у фахових виданнях України [2, 3, 4] та тези доповідей на конференціях [5, 6, 8], опубліковані здобувачем одноосібно.

8. Апробація результатів дисертації.

Результати досліджень дисертаційної роботи доповідалися та обговорювалися на таких національних та міжнародних конференціях: III Міжнародна науково-практична конференція «Інформаційні технології в освіті та науці» (ІТОН-2023) (м. Запоріжжя-Мелітополь, 25-26 травня 2024 р.); VII Міжнародна науково-практична конференція «Інформаційні технології в освіті, науці і техніці» (ІТОНТ-2024) (ЧДТУ, м. Черкаси, 23-24 травня 2024 р.); 1st International Scientific and Practical Conference «COMPUTATIONAL INTELLIGENCE AND SMART SYSTEMS» (CISS-2024) (Lviv, 10–12 October 2024); XXIV міжнародна науково-технічна конференція «Штучний інтелект та інтелектуальні системи» (AIIS-2024) (м. Київ, 18-19 жовтня 2024 р.); International Conference on AI, 5G and QUANTUM SECURITY (Erasmus+ SECURE project) (Online, 19 March 2026).

9. Публікації.

За результатами дослідження опубліковано 10 наукових праць: 1 наукова стаття в іноземних виданнях, що індексуються у Scopus; 3 наукові статті у фахових виданнях України; 4 тези доповідей на наукових конференціях (Додаток А).

10. Структура та обсяг дисертації.

Дисертація включає вступ, 4 розділи, висновки та 5 додатків. Обсяг дисертації – 248 сторінок, з них основного тексту – 188 сторінок. Дисертація містить 45 рисунків, 40 таблиць в основному тексті та посилання на 134 використаних джерел.

РОЗДІЛ 1

АНАЛІЗ ПОТОЧНОГО СТАНУ ПРЕДМЕТНОЇ ОБЛАСТІ. ПОСТАНОВКА НАУКОВОГО ЗАВДАННЯ

Поява перших Великих Мовних Моделей (LLM) на базі архітектури (GPT) у 2018 році у науковому товаристві не була сприйнята як великий прорив чи революційна технологія. Радше, це був ще один цікавий варіант застосування новітньої на той момент архітектури Трансформерів (Transformer), запропонованої роком раніше в роботі [1]. Реліз GPT-2 [2] викликав більшу зацікавленість та слугував SOTA базою для тогочасних AI-стартапів. Проте справжній прорив у використанні LLM стався лише з релізом моделі GPT3-Instruct [3] та GPT-3.5 під кодовою назвою ChatGPT в листопаді 2022 року. Якість роботи моделі та рівень розуміння природної людської мови були достатньо високими та кардинально відрізнялися від усіх інших застосунків на базі нейронних мереж (NN). Менш ніж за 2 місяці після релізу модель перетнула позначку в 100 мільйонів користувачів і почала чинити серйозний вплив на різні аспекти людського життя, і освіти зокрема [4]. Перша реакція на ChatGPT була настільки позитивною, що породила дві гіпотези, що продовжують існувати й сьогодні:

- LLM мають свідомість
- Поява AGI - справа найближчих кількох років

Розглянемо детальніше, що таке LLM, на яких принципах вони побудовані, що їх відрізняє від людського інтелекту та AGI.

1.1. Основи генерації та ідентифікації природномовних текстів

Галузь створення LLM суттєво виросла з моменту виходу GPT-1, особливо активний прогрес відбувся у 2023-2025 роках, ініційований успіхом ChatGPT. Попри велику кількість компаній, що створюють LLM-based продукти, розробкою фундаментальних (foundational) моделей займається лише невелика група організацій. Побудова фундаментальних моделей вимагає великої кількості ресурсів (як обчислювальних, так і даних), наявності в команді провідних спеціалістів зі

штучного інтелекту (ШІ) та майже необмеженого фінансування. Наразі розробка нових версій LLM може коштувати сотні мільйонів доларів навіть для компаній, що вже успішно функціонують у цій сфері та мають комерційно вдалі моделі. Причому конкуренція досить висока і створення нової LLM, навіть більш успішної за суперників за тестами (benchmarks), зовсім не гарантує комерційного успіху та повернення витрачених інвестицій.

У таких жорстких умовах, коли поріг входу і ризик втрати суттєвих фінансів дуже високий, а винагорода не гарантована, тільки досить невелика кількість компаній готові вступити в перегони ШІ. І лише за умови чіткого бачення, чому їм потрібна своя LLM і як вона буде використовуватися в майбутньому. Для прикладу можна навести компанію Microsoft, що входить до десятки найбільших компаній світу, проте не має своєї власної мовної моделі, віддаючи перевагу партнерству з OpenAI.

1.1.1. Класифікація моделей генерації тексту

Можна виділити десять компаній, що відповідають за створення близько 90% потужних фундаментальних моделей, розповсюджених на ринку, що активно використовуються в реальних продуктах і оцінюються за стандартними benchmarks. Ось ці 10 компаній: OpenAI, Anthropic, Google, Meta, Mistral AI, xAI, DeepSeek, Qwen, AI21 Labs та Cohere. Крім вказаних десяти компаній, існують також інші гравці, що залишилися поза нашим оглядом, проте також впливають на загальний розвиток ринку LLM, хоча і не так суттєво. Серед таких гравців варто відзначити:

- Amazon з моделями Titan, Nova та Project Olympus
- Nvidia з моделлю Nemotron-4-340B
- Databricks з моделлю DBRX
- Technology Innovation Institute (TII) з серією моделей Falcon
- EleutherAI з серіями моделей GPT-Neo, Pythia, Polyglot-Ko
- Salesforce з серією моделей XGen

За стратегічною спрямованістю, основних гравців можна розділити на 3 групи:

- 1) Великі компанії, що намагаються домінувати в усіх сферах одночасно: OpenAI, Anthropic, Google, Meta
- 2) Компанії сфокусовані на ефективності та оптимальності: Mistral AI, DeepSeek, AI21 Labs
- 3) Компанії сфокусовані на конкретних комерційних сценаріях: Qwen, Cohere, xAI

Початок перегонів у галузі штучного інтелекту поклала “гіпотеза масштабування” (scaling hypothesis), запропонована OpenAI [5] у 2020 році. Ця робота фактично стверджувала, що ключ до якості ШІ полягає в розмірі моделі, тому всі компанії групи 1 почали будувати все більші моделі, використовуючи все більше ресурсів. Перегони продовжувалися кілька років і фактично закрили галузь для менших гравців, оскільки для того, щоб наздогнати поточну SOTA потрібні були початкові вкладення в інфраструктуру в сотні мільйонів доларів. Ці компанії продовжують випускати моделі, що претендують на звання найкращої універсальної моделі, та змагаються за найвищі щаблі лідерборду за широким набором benchmarks.

Компанії другої групи не могли за обсягом інвестицій змагатися з компаніями першої групи, а значить, не могли претендувати на вищі щаблі лідерборду. Тож вони винайшли іншу стратегію - оптимізувати побудову LLM. Річ у тому, що моделі “великої четвірки”, хоча і були якісними, проте використання їх залишалося дуже дорогим, що унеможливлювало багато потенційних сценаріїв використання. Таким чином, на ринку існувала ніша для менш потужних, проте все ще достатньо якісних моделей за суттєво нижчу ціну. Цю нішу і зайняли компанії другої групи, змінивши фокус індустрії з парадигми масштабування на парадигму ефективності шляхом оптимізації, побудови нових архітектур та перегляду ключових принципів.

Третя група компаній має фокус на комерційному застосуванні LLM, тобто вони створюють спеціалізовані моделі під конкретні потреби та запити бізнесу. Ці моделі характеризуються точністю роботи та надійністю, розширеними можливостями використання зовнішніх інструментів, наявними розмірами під різні

потреби й т.ін. У більшості випадків вони не представлені в лідерах різних тестів, проте, завдяки вузькій спеціалізації, здатні приносити найбільшу користь бізнесу.

Щодо класифікації моделей, то у статті [6] нами запропоновано наступні категорії (рис. 1.1), що описують рівні розвитку LLM. Варто відзначити, що лише перший рівень є обов'язковим для будь-якої компанії, досягнення ж рівнів 2 і 3 може відбуватися незалежно.

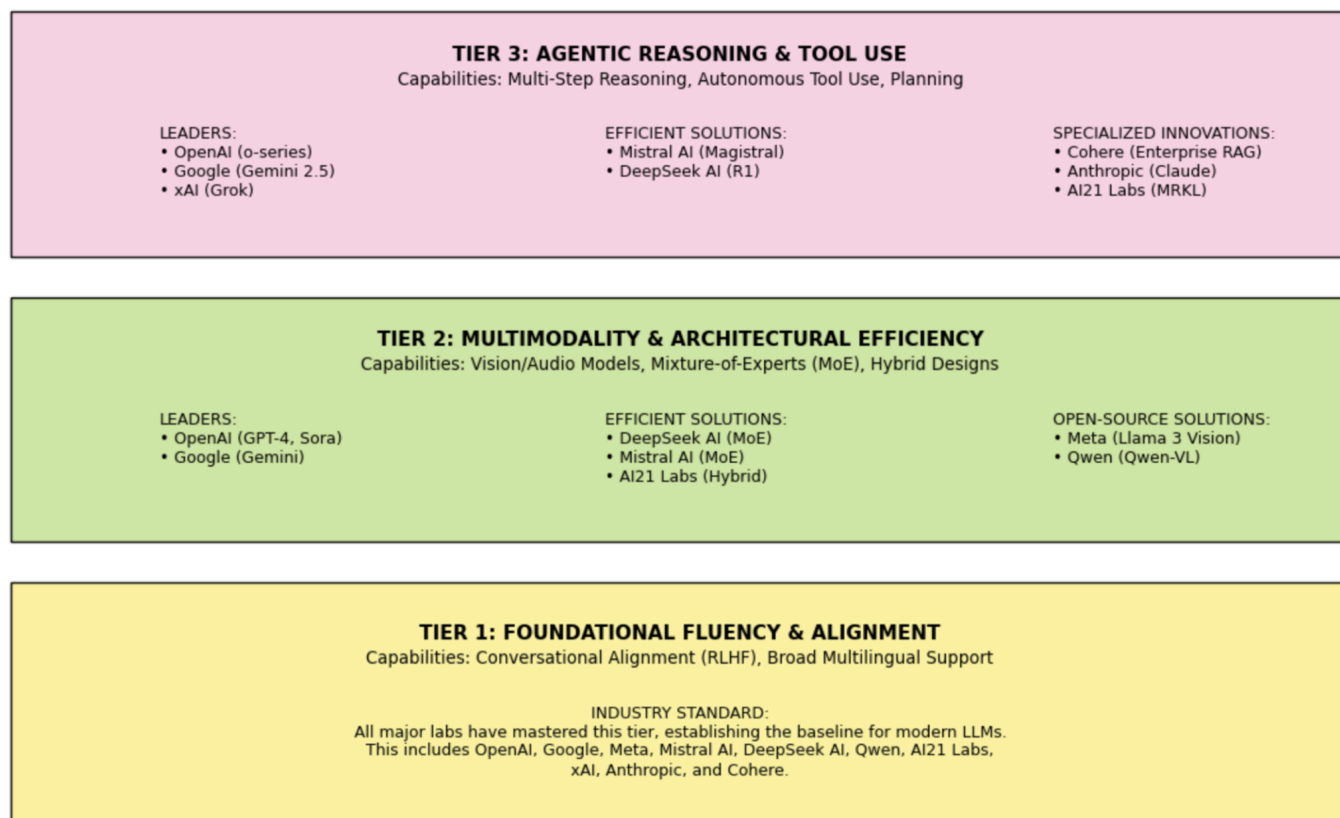


Рисунок 1.1 - Рівні розвитку великих мовних моделей

З погляду архітектури суттєво виділяються моделі, побудовані на базі МоЕ (Мікс Експертів). Дана архітектура була запропонована ще в роботі 1991 року [7], але справжню популярність отримала після її застосування у LLM компанією DeepSeek у роботах [8] і [9]. Після релізу моделі DeepSeek-R1, архітектура стала новою парадигмою. Її ключовою перевагою є те, що для кожного токена активізуються не всі нейрони, а лише спеціальний “розріджений маршрут” у мережі [10], що динамічно обирається механізмом маршрутизації. Використання цієї архітектури створює специфічні артефакти в тексті, що помітні на лінгвістичному рівні. Робота [11] стверджує, що різноманітність експертів зростає зі збільшенням

кількості шарів NN, а робота [12], присвячена дослідженню роботи МоЕ для випадку багатомовних моделей, вказує на те, що початкові та кінцеві шари мережі є мовно-специфічними, а середні шари є мово-незалежними, універсальними. Таким чином, у звичайній щільній LLM у генерації токенів беруть участь всі нейрони рівномірно, у випадку ж МоЕ вибір нейронів є не рівномірно-монотонним, а підпорядкованим деякому процесу вищого рівня, має специфічні патерни, та навіть спеціалізацію нейронів різних шарів. Тобто тексти, створені LLM на базі МоЕ, мають свій унікальний спосіб генерації, що слідує певним патернам і залишає специфічні відбитки (чи слід). Робота [13] підтверджує, що цей слід піддається детекції.

З погляду функціональності, окремим класом виділяються моделі, здатні до Multi-step Reasoning (Багатокрокового міркування); ця техніка є логічним продовженням Chain-of-thought prompting style (підказки в стилі ланцюжок думок), який реалізовано на системному рівні. Вперше була запропонована OpenAI з релізом моделей o-серії [14] та розповсюджена в open source (публічний доступ) з релізом DeepSeek-R1 [9]. Ключовою особливістю підходу є те, що модель не генерує відповідь на поставлене питання відразу, замість цього вона починає його аналізувати крок за кроком, перевіряти свої попередні висновки й, таким шляхом, у цілому суттєво підвищувати якість відповідей. З лінгвістичного погляду, такий метод функціонування моделі створює специфічний артефакт - reasoning trace (слід міркування). Цей слід, як правило, складається з окремих логічних кроків, націлених на розв'язання задачі, та суттєво відрізняється як від тексту, написаного людиною, так і від типових текстів, згенерованих LLM [15].

З погляду вхідних даних, варто виокремити Multimodal Models (Мультимодальні Моделі) - моделі, здатні працювати одночасно з різними класами вхідних даних (наприклад, текст, зображення та аудіо). Першими представниками цього класу стали GPT-4 [16], Gemini [17] та LLAMA-3.2-Vision [18]. Для такої роботи моделі потрібно створити власний "grounded language" (заземлену мову) [19-20], що впливає на розподіл токенів при генерації та суттєво відрізняється від

типового тексту згенерованого звичайною LLM, оскільки вибір фінального токена обумовлюється впливом нетекстового входу.

Ще однією причиною, що суттєво впливає на лінгвістичний слід моделі і, як результат, на можливості її детекції є drift (зсув) - явище, коли з часом продуктивність моделі падає. У випадку LLM варто виокремити два різних типи цього явища: model drift та identity drift. Model drift характеризується зміною поведінки моделі під час виходу нових версій чи взаємодії моделі з новими даними [21]. Identity drift (зсув індивідуальності) характеризується неможливістю моделі утримувати стиль спілкування чи персону протягом довгих діалогів [22]. Саме через ці причини AI Detector (Детектор ШІ), що успішно визначав GPT-3.5 у 2023 році, навряд чи буде вдало визначати GPT-5 у 2025.

1.1.2. Визначення сліду ШІ та його прояви

Одним з основних фокусів дослідження цієї дисертаційної роботи є вивчення концепції AI Trace (слід ШІ) - такого набору характеристик тексту, що притаманний LLM і непритаманний текстам, написаним людиною. Ці характеристики є досить тонкими і їх важко виявити без використання спеціального інструментарію або підготовки. Складність виявлення AI Trace є природною, оскільки, з одного боку, LLM було навчено на текстах, написаних людьми, тож вони імітують “усереднений” людський стиль; з іншого ж боку, якість і природність згенерованого мовною моделлю тексту є однією з ключових характеристик, за якими LLM оцінюються. Однією з ключових причин слабого розповсюдження та обмеженості сценаріїв використання перших LLM на кшталт GPT-1 та GPT-2 була саме штучність/синтетичність та неприродність згенерованого тексту. З лінгвістичного погляду такий текст міг бути орфографічно і граматично коректним, проте при читанні було дуже важко зрозуміти його зміст. Наразі ж, мовні моделі здатні генерувати набагато якісніший текст, тому питання виявлення AI Trace стає значно складнішим.

Можливість виявлення сліду ШІ є вирішальним фактором для вирішення двох завдань: ідентифікації автоматично згенерованих текстів та надання доказів (бажано, візуальних) синтетичної природи тексту. У роботі [23] професійним редакторам запропонували покращити якість текстів згенерованих ШІ, а потім запропонували іншим експертам оцінити, який з текстів якісніший - оригінал; оригінал, відредагований іншим ШІ; чи варіант, відредагований людиною. Варіанти після редагування людиною оцінювалися як якісніші у 65% випадків. Іншим цікавим результатом дослідження став перелік проблем у ШІ-згенерованих текстах, які підлягали виправленню. У рамках дисертаційного дослідження ця інформація становить інтерес, оскільки вона проливає світло на питання, на яких рівнях можна розрізняти синтетичні та природні тексти. Основні категорії виправлень, відповідно до [23]: Cliche (табл. 1.1), Unnecessary/Redundant Exposition, Purple Prose, Poor Sentence Structure, Lack of Specificity and Detail, Awkward Word Choice and Phrasing, Tense Inconsistency. Крім вказаних категорій виправлень на лексично-граматичному рівні, також виділяються виправлення суто синтаксичного (табл. 1.2) та лексичного рівнів (рис. 1.2).

Таблиця 1.1 Приклад кліше

Categorization	Paragraph with rewrites
Cliche	As Sarah stepped off the bus, the scent of pine and damp earth enveloped her. [.....] In the kitchen, she found herself reaching for the cabinet where her mother always kept the coffee, only to stop short. The realization that she was alone here, truly alone, settled over her like a heavy blanket. This time, though, she was alone. Her mother would never come back. She sank into a chair at the old oak table[....]

Таблиця 1.2 Синтаксичні патерни, що найчастіше потребували виправлення

Syntactic Pattern	% of Times Edited
DT NN IN NN CC	54
NN IN NN CC NN	35
DT JJ NN IN PRP\$	40
DT NN IN JJ NN	27
IN DT NN IN NN CC	45

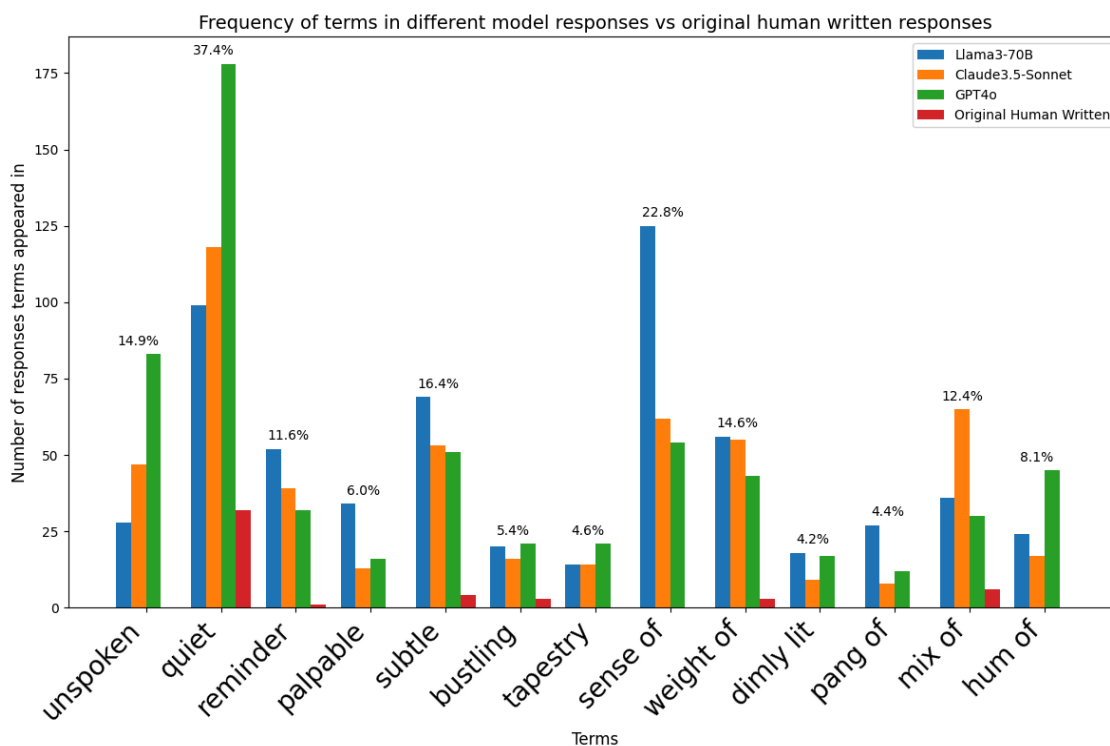


Рисунок 1.2 - Частота використання нетипових слів різними LLM

Робота [24] робить наступний крок і пропонує підхід до ідентифікації текстів згенерованих ШІ через незалежне виявлення AI Trase на трьох різних рівнях: семантичному, стилістичному і структурному. Приклад опису керівних принципів для виявлення таких характеристик наведено на рис. 1.3.

У рамках цього дисертаційного дослідження нами буде визначено три основні рівні, на яких можна виділити слід ШІ: лексичний, синтаксичний та семантичний. Безумовно, запропоновані рівні пов'язані між собою, і деякі ознаки сліду ШІ можна віднести відразу до кількох рівнів, тому далі буде наведена максимально точна характеристика кожного з них.

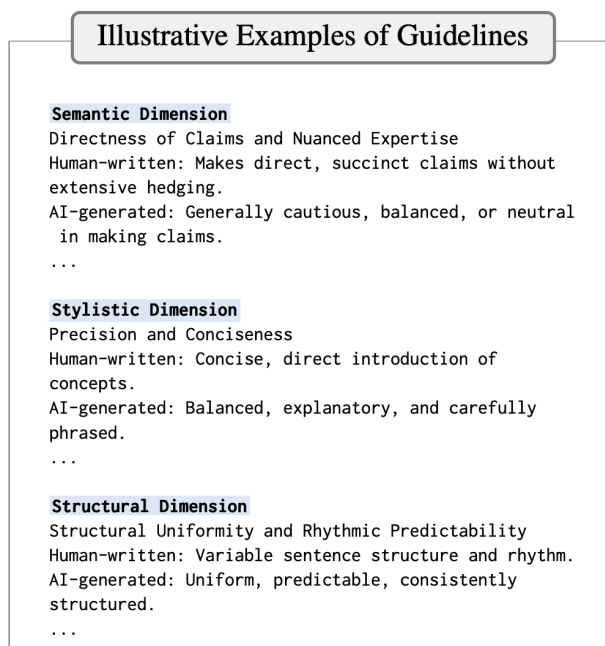


Рисунок 1.3 - Приклад характеристик сліду ШІ за рівнями

Лексичний рівень характеризується специфічним вибором слів, які або не характерні для тексту, написаного людиною взагалі, або рідко використовуються в певному контексті, або використовуються ШІ суттєво частіше, ніж зазвичай використовує людина. Як правило, дослідження на кшталт [25] виділяють дві типові характеристики AI текстів: low perplexity (низька невизначеність, тобто висока прогнозованість тексту) та обмежена burstiness (вибуховість - мала варіативність perplexity). Іншими словами, LLM мають тенденцію до генерації рівномірного тексту з вибором токенів з високою ймовірністю (найбільш імовірної послідовності слів), без різких переходів (від пасажів з низькою perplexity до неочікуваних формулювань), з тенденцією до використання формалізмів та кліше, як описано в [24]. Цей рівень є найбільш поверхневим і легко визначається методами статистичного аналізу. Гарним прикладом інструменту візуалізації AI trace на основі perplexity може виступати GLTR [26]. Інструмент є застарілим, оскільки вимірює perplexity на основі розподілу притаманного GPT-2, проте все ще гарно демонструє сам принцип (рис. 1.4).

In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English. The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science. Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved. Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow. Pérez and the others then ventured further into the valley. "By the time we reached the top of one peak, the water looked blue, with some crystals on top," said Pérez.

Рисунок 1.4 - Розмітка ймовірності токенів у LLM тексті від GLTR

Синтаксичний рівень - тут мова іде про патерни організації тексту, його структуру, зв'язність та ритм. Відповідно до теорій риторичної та текстової когерентності, текст, створений людиною, має змінну структуру речень та ритм. Людина несвідомо залишає відбиток своєї свідомості у тексті через вибір слів і конструкцій, довжину речень та складну аргументацію, через що створений людиною текст не буде однорідним [24]. Наше власне дослідження [27] показує, що порушення синтаксичної структури тексту серйозно впливає на можливості сучасних AI detectors до ідентифікації таких текстів, що, своєю чергою, доводить, що синтаксична структура згенерованого LLM тексту містить елементи AI trace, що є важливими для детекторів. Дослідження [28], присвячене Adversarial attacks (змагальним атакам - атакам з метою змінити вихід моделі) на AI detectors розглядає три рівні атак, один з яких, word level (атаки на рівні зміни слів) безпосередньо спрямований на зміну синтаксичної структури, що ще раз підкреслює наявний консенсус щодо важливості сліду ШІ, представленого на цьому рівні.

Семантичний рівень відповідає за найглибший AI trace. Тексти, згенеровані SOTA LLM, характеризуються високою локальною узгодженістю - плавними логічними переходами від речення до речення, проте можуть страждати через відсутність узгодженості в глобальному контексті - наприклад, викладене на початку довгого документа може повністю суперечити викладеному в його кінці. У дослідженні [24], яке цитувалося вище, один із запропонованих рівнів - семантичний, про який вони говорять, що AI trace цього рівня характеризується обережним, збалансованим чи нейтральним тоном під час висування тверджень. У нашому дослідженні [29] показано, що існує суттєва різниця в детекції даних

згенерованих на стадії 5 (кожне речення має сенс, та немає зв'язку між реченнями) та стадії 6 (кожен параграф має сенс, та немає зв'язку між параграфами). Деякі дослідження, наприклад, [30] пропонують спеціальні типи атак, що зачіпають більш поверхневі рівні AI тексту (наприклад, заміна деяких слів на синоніми), проте свідомо зберігають семантичний рівень без змін. У той самий час, дослідження [28] стверджує, що більшість сучасних детекторів є стійкими до атак низького рівня (наприклад, описки та заміна символів), проте залишаються вразливими до атак семантичного рівня.

Таким чином, слід III, що наявний у текстах, згенерованих сучасними LLM, це багаторівневий артефакт, що на концептуальному рівні пов'язаний з особливостями функціонування великих мовних моделей. Цей артефакт може видозмінюватися від покоління до покоління моделі, на нього можуть впливати як нові тренувальні дані, так і вибір специфічної архітектури моделі, наявність інших модальностей чи reasoning можливостей. Через певні архітектурні обмеження та специфіку тренування моделей неможливо побудувати LLM, інференс якої не буде містити характерного AI trace. Питання детекції AI спирається на можливості детектора оперувати на одному чи кількох рівнях AI trace з метою виявлення типових відбитків або патернів; а заходи з маскування AI-generated контенту пов'язані з adversarial attacks на різні рівні сліду III, як описано вище. Тому слід III відіграє ключову роль у питанні можливості ідентифікації текстів створених AI.

1.2. Аналіз існуючих підходів до детекції згенерованих текстів

Маючи уявлення про те, як побудований процес генерації, та який слід великі мовні моделі залишають у даних, можна розглянути особливості процесу детекції текстів згенерованих AI. На сьогодні існує багато різних підходів до вирішення вказаного завдання, проте всі їх можна згрупувати у два великі класи, яким і присвячена ця частина дослідження. Для більшості AI детекторів існує проблема, що вони можуть добре працювати на окремому домені чи специфічному бенчмарку (наборі тестів), проте погано пристосовані для викликів у реальному використанні:

крос-доменних сценаріїв, аналізу нових моделей та застосування методів маскування (результат adversarial attacks). Така нестійка поведінка детекторів призвела у 2023 році до появи великої кількості робіт, які критикували AI Detection та заперечували можливість вирішення цього завдання технічними методами взагалі [31-33].

1.2.1. Статистичні та лінгвістичні методи ідентифікації

Стилометрія - кількісний метод аналізу текстів, що використовує статистичний аналіз для аналізу стилю тексту. Як правило, стилometрія використовує такі features, як N-grams (послідовності з N елементів) слів, символів, POS (Part-Of-Speech) Tags, токенів, вимірює типові довжини слів і речень, використання службових та стоп-слів, пунктуації та деякі інші. По суті, цей тип аналізу базується на методах feature engineering (інженерія ознак), що були напрацьовані до буму NN моделей на архітектурі трансформерів, і використовує підходи, які було створено для розв'язання задачі Authorship Attribution (визначення авторства) чи інших близьких задач. Ці методи добре зарекомендували себе у розв'язанні задач на кшталт визначення тематики тексту, NER (Named Entity Recognition - визначення іменованих сутностей), класифікації текстів, Anaphora Resolution (вирішення анафор - процес визначення до якого слова відноситься поточна згадка в тексті, наприклад, займенник), Coreference Resolution (встановлення кореферентності - визначення всіх згадок одного об'єкту в тексті), визначення авторства та інших. Використання цього сімейства методів до завдання AI detection спирається на базову гіпотезу, що кожна LLM має свій власний авторський стиль, як і людина. Відповідно, тоді можна застосувати методи Authorship Attribution, розглядаючи LLM як автора, з метою побудови системи, здатної відрізнити цього автора від усіх інших.

Дослідження [34] стверджує, що, на відміну від людини, ранні версії LLM, під час генерації, зберігали послідовний стиль незалежно від того, чи правдиву інформацію вони створювали, чи вигадану. Це може вказувати на наявність стійкого

“авторського” стилю. Робота [35] порівнює ефективність стилометрії, порівняно з аналізом людиною, і вказує на те, що стилометрія працює ефективніше.

З іншого боку, наші власні дослідження з Authorship Attribution [36] та дослідження інших авторів показують, що LLM не завжди зберігає цілісний авторський стиль, він може змінюватися залежно від, наприклад, домену або інших параметрів, встановлених під час генерації. Що є досить природним, адже стиль письма ШІ з певної тематики залежить від тренувальних даних з цієї тематики та низки інших факторів. Робота [37] звертає увагу на нестабільність особистості моделі, що здатна сильно змінюватися залежно від контексту, чи prompting style. Робота [38] зазначає суттєвий вплив технік RLHF (Reinforcement Learning from Human Feedback - навчання з підкріпленням на основі зворотного зв'язку від людини) та стиль генерації моделей, уніфікуючи його та затираючи індивідуальні відбитки. Робота [39] показує, що перефразування тексту здатне повністю знищити/приховати авторський стиль, а робота [40] говорить про суттєву вразливість стилометрії до adversarial attacks.

Perplexity. Як було сказано вище, perplexity може виступати іншою кількісною характеристикою, що дає змогу визначати згенеровані AI тексти. Цей спосіб детекції базується на припущенні, що AI текст створюється з центру розподілу ймовірностей моделі, отже всі вибрані токени є найбільш імовірними продовженнями поточної послідовності слів, водночас текст, створений людиною, є більш неочікуваним. Тобто, вимірюючи perplexity система визначає, наскільки текст, що аналізується, є передбачуваним з погляду моделі [41]. Текст, що створено моделлю, для неї самої (тобто для вивченого моделлю розподілу ймовірностей токенів) має виглядати дуже очікуваним (низька perplexity), водночас текст, створений людиною, повинен мати неочікувані комбінації токенів, отже мати високу perplexity з т.з. моделі. На даному принципі базується робота багатьох інструментів на кшталт GLTR [26].

Методи засновані на perplexity є досить популярними - цілий набір AI detectors, таких як GPTZero, ZeroGPT, QuillBot AI Detector, Crossplag та Sapling засновані на них. Проте підхід має ряд теоретичних та практичних обмежень. Одним

з обмежень виступає питання `proxy-model`, тобто моделі, що використовується для видачі ймовірності токенів. З погляду ефективності та швидкодії бажано використовувати невелику модель; з іншого боку, немає гарантії, що вивчений розподіл ймовірностей у моделі GPT-2 буде відповідати такому в більших моделях, на кшталт GPT-4 чи GPT-5. Також, як було розглянуто раніше, наразі існує як мінімум 10 сімейств фундаментальних моделей LLM і не можна очікувати, що розподіл ймовірностей у GPT, LLAMA, Claude, Gemini, Mistral та інших буде однаковим чи близьким, не кажучи вже про архітектурні особливості та різницю в розмірах моделей навіть в одному сімействі.

Прогрес у навчанні моделей теж накладає певні обмеження на детекцію на основі `perplexity` - наприклад, через використання RLHF моделі вивчають варіювання стилю, притаманне людині, і генерують тексти з більшою `burstiness` [31]. Разом з тим, деякі домени, що вимагають формального написання текстів, наприклад, наукові публікації, часто містять низькоентропійний підбір токенів, що типово для текстів написаних машиною [42]. Іншою проблемою, характерною для цього підходу є те, що тексти, написані неносіями мови, як правило, тяжіють до вибору типових послідовностей слів і, відповідно, мають низьку `perplexity` і низьку `burstiness`, що характерно для текстів згенерованих LLM [43]. Оскільки ця група методів, по суті, намагається розглядати питання AI detection як задачу бінарної класифікації, то їй характерний чіткий розподіл на два класи, через що виникає велика сіра зона, куди потрапляють тексти, написані людиною і потім відшліфовані ШІ, чи навпаки, створені ШІ та потім `humanized` (гуманізовані - оброблені спеціальним чином, щоб виглядати створеними людьми). Детектор на основі `perplexity` не в змозі впоратися з текстами в сірій зоні й часто видає неправильні результати класифікації [44].

Отже, клас методів детекції заснованих на `perplexity` є досить розповсюдженим на ринку, він має свої слабкі та сильні сторони. В цілому, методи на основі `perplexity` можуть непогано працювати в контрольованому середовищі, особливо коли одна й та сама модель виступає і в ролі автора текстів і в ролі джерела даних про очікуваний розподіл ймовірностей у рамках одного стилю і домену. Якщо ж аналіз

проводиться у складніших умовах, то моделі стає набагато важче відрізнити текст написаний людиною і згенерований AI [41]. Деякі детектори, на кшталт DetectGPT, Binoculars, RAIDAR намагаються обійти обмеження притаманні perplexity шляхом поєднання її з використанням інших методів.

1.2.2. Методи ідентифікації, засновані на нейромережах

Fine-Tuned Classifiers. Тонко налаштовані класифікатори - це попередньо навчені NN моделі, які потім були спеціально дотреновані для розв'язання задачі AI detection. Найбільш часто використовуються класифікатори на базі трансформерної моделі BERT/RoBERTa (110M - 155M параметрів), як певний компроміс між якістю та точністю детекції, проте, іноді застосовуються і класифікатори, побудовані на значно більших моделях, наприклад Radar [45] побудовано з використанням таких моделей як: Pythia, Dolly 2.0, Palmyra, Camel, GPT-J, Dolly 1.0, LLaMA, Vicuna (3B-7B параметрів). Такі класифікатори можуть досягати дуже високих показників точності (до 99%) [46], особливо якщо використовуються проти моделей, які були в тренувальній вибірці. Головною перевагою цього класу моделей є те, що використання великих NN дає змогу їм виконати всю feature engineering роботу замість людини - вони здатні автоматично виділити ключові характеристики та складні патерни, що відрізняють LLM-стиль від людського. Здатність моделей автоматично виділяти багатомірні features дає змогу їм досягати значно вищих оцінок у тестах, порівняно зі статистичними моделями, описаними в 1.2.1 [47].

Дві основні проблеми, які найчастіше згадуються у контексті цих методів, це, по-перше, слабка адаптованість до даних, що відрізняються від тренувальних (Out-of-Distribution). Ця проблема може проявлятися як нездатність коректно ідентифікувати тексти, згенеровані моделлю, якої не було в тренувальних даних, або складності з класифікацією даних з інших доменів [46]. Наприклад, класифікатор натренований на новинах буде мати складності з опрацюванням есе, а класифікатор натренований на моделях сімейства GPT буде мати складності з детекцією Claude. Робота [48] звертає увагу, що традиційний підхід використання fine-tuned

класифікаторів не дає змоги вирішити разом і проблему узагальнення (Out-of-Distribution) і робастності. Також цей клас детекторів вразливий до модифікацій тексту шляхом перефразування та інших класів adversarial attacks.

Zero-shot methods. Якщо класифікатори попереднього класу працюють завдяки тому, що вивчають як виглядає згенерований LLM текст на базі сотень тисяч прикладів, то моделі цього класу концентруються на тому, як AI “думає”. Для ідентифікації згенерованих текстів вони використовують ймовірнісний характер LLM і не потребують розмічених навчальних даних, по суті, вони становлять собою наступний крок еволюції perplexity моделей, і працюють на базі тієї самої гіпотези про те, що LLM використовують для генерації вужчий, концентрований набір умовних ймовірностей порівняно з людиною.

Серед переваг цього класу моделей варто зазначити високі можливості узагальнення, оскільки такі моделі як DetectGPT, Binoculars, та DNA-GPT використовують статистичні артефакти притаманні самому процесу генерації, які є більш стійкими до появи нових моделей, та проблеми Out-of-Distribution (OOD) з текстами з нових доменів [49]. Іншою суттєвою перевагою є відсутність потреби в тренувальних даних, що дає змогу уникнути упередженості та викривлень, які інакше могли бути вивчені з тренувальних даних [50].

Серед основних недоліків цього класу методів варто згадати наступні:

- 1) Висока ціна використання - для того, щоб зробити prediction, моделі потрібно мати доступ до ймовірностей токенів тексту, які можна отримати тільки з LLM, тому моделі необхідно відправити весь текст до LLM. Також, не всі комерційні LLM можуть повертати ймовірності токенів [50].
- 2) Базово, ці моделі вважалися стійкими до атак, проте останні дослідження [51] показали, що сучасні методи перефразування з використанням LLM, спеціально проінструкованої уникати детекції, дозволяють знизити рівень розпізнавання AI текстів майже на 99%.
- 3) Нові архітектури, такі як MoE та Reasoning Agents здатні створювати значний

стилістичний дрифт, який робить метод ненадійним [22].

Метод продовжує еволюціонувати, наприклад, нещодавно було запропоновано метод “виправлення” текстів, які були піддані adversarial модифікаціям, що дає змогу підвищити точність їх детекції [52].

Watermarking. Основна ідея використання “водяних знаків” [53] на етапі генерації контенту базується на обов’язковому використанні токенів зі спеціального словника. Під час детекції модель-детектор, маючи словник спеціальних токенів, мала просто порахувати їх відсоток у тексті, у разі перевищення певного порогового значення текст отримував мітку AI-згенерованого. Таким чином, можна було отримати абсолютно невидимий для користувача “водяний знак”, який можна було легко розпізнати спеціальним детектором. Ідея була досить розповсюдженою у 2023-2024 роках і обіцяла скасувати потребу в складніших детекторах ціною невеликого зниження якості згенерованого контенту. Метод мав бути стійким до редагування контенту, адже видалення невеликої кількості ідентифікаторів шляхом перефразування не мало впливати на точність детекції.

Більш пізні дослідження [54-56] виявили критичні недоліки запропонованого методу. Найбільшою вразливістю виявилось те, що зловмисник може відтворити список токенів зі спеціального словника моделі, використовуючи спеціальні prompting techniques і витративши на це всього близько 50\$. Маючи приблизний словник токенів “ключа”, зловмисник може повністю дискредитувати метод, оскільки це дає йому можливість як видалення watermark зі згенерованих текстів, так і вставлення його в тексти, написані людиною. Через що watermark повністю втрачає свою силу цифрового підпису і юридичну силу.

Іншим неочікуваним недоліком watermarking виявилася низька робастність до атаки перефразуванням, як було показано в роботі [31]. Хоча watermarking методи було розроблено з розрахунку на підвищену стійкість до adversarial attacks загалом і перефразування зокрема, атака рекурсивним перефразуванням, запропонована в [31] показала свою високу ефективність. Робота [57] вказує на те, що хоча watermarking

методи є стійкими до певних типів змін даних, навіть просте редагування на кшталт заміни символів чи перефразування дає змогу уникнути детекції.

Загалом методи на базі watermarking мають зараз не найкращі часи. Коли їх було запропоновано кілька років тому, вважалося, що вони здатні розв'язати проблему AI detection майже гарантовано та очікувалося, що всі провідні розробники фундаментальних моделей вбудують їх у свої LLM. У такому випадку, єдиною практичною проблемою залишалося б те, що певні open source LLM можуть продовжити виходити без водяних знаків. Через кілька років, після проведених досліджень, стало очевидним, що перед watermarking стоїть два серйозні виклики, які потрібно вирішити для збереження перспектив практичного застосування:

- Легкість виявлення цифрового відбитка водяного знаку. Ускладнення відбитку можливе лише шляхом більш агресивного примушення LLM до використання певних токенів, що призведе до погіршення її якості роботи.
- Вразливість до перефразувань та глибоких змін тексту.

XAI-Driven Detection. Цей клас методів побудовано на базі засобів пояснюваного ШІ, тобто таких технік, де модель тим чи іншим способом може пояснити людині причини свого рішення, наприклад, підсвітивши ключові слова, речення чи features, які було використано під час отримання певного висновку. Серед XAI (Explainable AI - Пояснювальний ШІ) методів два найбільш помітних це LIME [58] та SHAP [59].

LIME (Local Interpretable Model-agnostic Explanations - Локальні Інтерпретовані Незалежні від Моделі Пояснення) модель працює на базі припущення, що хоча рішення великих NN (наприклад, AI детекторів) глобально нелінійні, проте їх можна апроксимувати простою лінійною моделлю (на кшталт лінійної регресії, чи дерева рішень) на невеликому локальному контексті. LIME не виносить рішень щодо AI детекції, вона працює поверх іншого детектора і намагається пояснити його рішення. Для цього на певному тексті LIME генерує велику кількість змін і кожен з таких змінених варіантів відправляє до AI detector. Отримавши результат для кожного варіанта (AI/Human), модель використовує

накопичені дані для тренування простої лінійної моделі, з якої потім отримують ваги (позитивні та негативні) для кожного токена. Ці ваги демонструють потенційний вплив конкретного токена на рішення оригінального великого AI детектора.

SHAP (SHapley Additive exPlanations - Адитивні Пояснення Шеплі) модель базується на теорії кооперативних ігор, вона розглядає процес прогнозу як гру, де слова, токени та інші ознаки виступають у ролі гравців, що співпрацюють разом для отримання результату - прогнозу AI детектору. Для кожного слова обчислюється значення Шеплі - середній внесок цього слова в prediction серед усіх комбінацій слів у реченні. Це значення демонструє, наскільки присутність слова у реченні впливає на prediction порівняно з його відсутністю. Результатом роботи моделі є стійкі оцінки важливості для кожного токена.

Слабким моментом LIME є нестійкість його оцінок, оскільки кожен текст оцінюється окремо, у відриві від інших текстів, тож вага одного токена може суттєво відрізнятись в різних текстах. На додаток, повторний аналіз одного і того ж тексту також може видавати різні ваги для одного токена, оскільки вони залежать від процесу рандомного семплювання під час створення тренувальних даних [60]. Хоча SHAP не має вказаної слабкості, проте він залишається дуже обчислювально складним порівняно з іншими методами [61] і має певні теоретичні обмеження, оскільки базується на припущенні, що кожне слово є незалежним; таким чином, він не може враховувати вплив контексту і словосполучень [62].

Також дослідження [63] вказує на недостатню доказову базу точності роботи ХAI методів, адже не побудовано достатньо великих benchmark наборів даних для оцінки точності цих методів і їх результативність заявляється на базі окремих прикладів. Іншим недоліком цього класу методів є потенційна загроза [64] їх використання зловмисниками для уникнення детекції - якщо після використання ХAI всі ключові токени, що впливають на рішення детектора відомі, то це полегшує як процес зміни тексту для уникнення детекції, так і процес “зламу” детектора шляхом автоматичного маскування сліду III ще на етапі prompt engineering.

1.2.3. Ключові обмеження існуючих методів детекції

На базі проведеного дослідження можна стверджувати, що наразі не існує методу детекції ШІ, який одночасно є робастним та інтерпретованим. Щодо робастності, то всі наявні методи: статистичні, неймережеві і навіть watermarking тим чи іншим чином вразливі до Out-of-Distribution даних, сценаріїв перефразування, humanization, маскування та навіть простої заміни символів і слів. Додаткові складності становить питання появи нових архітектур LLM та проблема хибнопозитивних спрацювань детекторів на текстах, написаних у формальному стилі чи неносіями мови. Щодо інтерпретованості, то всі методи детекції спираються на black-box підхід (підхід Чорної Скриньки), а ХАІ методи намагаються лише додати новий шар зверху, у спробі пояснити можливі принципи роботи детектора. Оскільки ХАІ радше припускають, як би міг працювати black box, а не описують його справжні керівні принципи, при цьому самі залишаючись не дуже надійними та обмеженими, то людина залишається без надійних засобів розуміння причин винесення того чи іншого рішення детектором. Нерозуміння базових принципів роботи детекторів, їх вразливість до певних сценаріїв та відсутність надійних методів інтерпретації породжують недовіру до AI detection взагалі [31-33], сприяють відмові від використання детекторів ШІ і рекомендують покладатися на аналіз тексту людиною. Попри ряд серйозних недоліків автоматичні методи детекції суттєво перевищують за точністю людський аналіз [65-67] і виступають у ролі основного стримуючого фактору, що стоїть на заваді хвилі ШІ-згенерованої дезінформації, неконтрольованому використанню ШІ в освіті та загрозам кібербезпеки.

Методи на основі стилометрії та perplexity є достатньо ефективними з погляду обчислювальних ресурсів, проте дуже потерпають від нестабільності ідентичності моделей. Авторський стиль моделі дуже залежить не лише від внутрішніх параметрів, таких як архітектура чи тренувальні дані, але і від зовнішніх параметрів, наприклад, температури, prompting style, наданого контексту та інших. Вплив на ці параметри здатен серйозно змінити характеристики стилю, на які націлені

статистичні методи. Крім того, методи на основі perplexity схильні до хибнопозитивних спрацювань у певних сценаріях.

Fine-Tuned Classifiers достатньо добре себе показують під час роботи з відомими їм моделями, оскільки вони здатні вивчити специфічні артефакти притаманні відбитку конкретної моделі. Тож ключовим обмеженням цього класу методів залишається вимога до якомога найширшого покриття відомих моделей і представлення створених ними даних у тренувальному наборі. Методи дуже вразливі щодо питання уніфікації/генералізації, як стосовно нових моделей, так і стосовно нових доменів. Модифікація текстів також може суттєво впливати на точність їх роботи.

Zero-Shot Methods та Watermarking намагаються виправити слабкості попередніх методів шляхом використання внутрішніх характеристик моделей та ін'єкції криптографічної інформації в текст на етапі генерації. Zero-Shot Methods потерпають від обчислювальної складності, адже, по суті, для аналізу тексту їм потрібна схожа кількість ресурсів, до тієї, що було витрачено на його генерацію; також їм потрібен доступ до внутрішніх ймовірностей моделі, що згенерувала текст, що не завжди доступно у випадку комерційних моделей. Іншою проблемою залишається питання вибору LLM, ймовірності якої потрібно використовувати в кожному конкретному випадку, та обмеження, що накладаються використанням проксі-моделей. Watermarking пропонував ультимативне розв'язання проблеми AI detection, проте виявився вразливим до Spoofing Attacks (спуфінг-атаки - атаки підміни). Метод має і інші слабкі сторони, наприклад, вразливість до атак перефразування чи заміни слів/символів, проте саме вразливість до Spoofing Attacks зробила метод фактично повністю непридатним для практичного використання до моменту винайдення способу протидії.

XAI-Driven методи не є самостійними методами детекції, а працюють як додатковий рівень над класичним AI detector з метою пояснити рішення детектора користувачеві й таким чином підвищити довіру. XAI методи мають свої власні недоліки на кшталт високого використання обчислювальних ресурсів, нестійкості результатів та ігнорування ширшого контексту.

Зведений аналіз розглянутих методів детекції наведено у табл. 1.3.

Таблиця 1.3 Основні методи детекції ШІ

Detection Technique	Primary Mechanism	Critical Limitation	Adversarial Vulnerability
Стилометрія	Аналіз лінгвістичних ознак для пошуку “авторського стилю”	Нестабільність - відсутність у LLMs стабільного стилю	Prompt Engineering - “пиши в стилі X”
Perplexity	Вимірює статистичні характеристики на кшталт передбачуваності / ентропії тексту	Упередженість - висока кількість хибнопозитивних спрацювань на неносіях мови та формальних текстах	"Burstiness" Injection - “змінюй довжину речень, використовуй неочікувані слова”
Fine-Tuned Classifiers	Навчання NN на розміченому наборі даних з Human/AI мітками	Проблема генералізації - нові архітектури і out-of-distribution дані	Paraphrasing - використання автоматичних парафразерів
Zero-Shot Methods	Визначає відхилення ймовірностей або перехресну perplexity	Вартість та доступ - висока вартість і потреба у внутрішній інформації LLM	Adversarial Paraphrasing / Humanization
Watermarking	Вбудовує статистичний сигнал під час генерації	Spoofing - відбиток може бути викрито і підмінено	Spoofing / Recursive Paraphrasing
XAI-Driven Detection	Атрибуція ознак (SHAP/LIME) щоб пояснити чому текст було класифіковано	Обчислювальна складність та нестабільність	Explanation Manipulation - зловмисник може створити AI текст, що буде пояснюватися як Human

1.3. Постановка завдання дослідження

Проведений аналіз свідчить про наявність серйозних недоліків у наявних методах детекції ШІ як з боку їх надійності (робастності) так і інтерпретованості. Кожен з розглянутих методів має свою специфіку і базується на певних припущеннях, що не завжди виконуються в реальному світі. Невідповідність базових гіпотез реальним умовам породжує певні концептуальні недоліки в роботі методу, що обмежують його практичну застосовність, проте дозволяють використовувати як надійний інструмент у контрольованих умовах. Таким чином, розуміння теоретичних принципів роботи AI detection методів дає змогу зробити перехресну перевірку, наприклад, наукової публікації методами інших класів, якщо методи на основі perplexity повернули високий AI score (оцінку ШІ). Проте всі методи страждають від adversarial attacks, і перехресна перевірка не в змозі допомогти в цьому випадку. Розглянемо детальніше основні механізми таких атак.

1.3.1. Аналіз вразливостей детекторів до трансформаційних атак

Загалом, на основі огляду актуальних досліджень, можна виділити чотири основні групи трансформаційних атак, які відрізняються націленістю на різні рівні AI Trace та способами модифікації тексту.

1. Реконструкція синтаксису і семантики.

Ця атака базується на припущенні, що виявлення ШІ працює на базі визначення певних n-grams та типових синтаксичних структур, вивчених з тренувальних даних, тобто низької burstiness LLM тексту. Відповідно, якщо нападник зможе відокремити синтаксис від семантики і знову зібрати текст з тим самим змістом, але на базі іншої синтаксичної структури, то атака буде успішною.

- Рекурсивне та дискурсне перефразування. На відміну від попереднього покоління парафразерів, які в основному займалися простою заміною синонімів, сучасні парафразери на кшталт DIPPER здатні проводити

глибоку переробку текстів на рівні абзаців. Вони мають налаштування для контролю ступеня зміни контенту та лексичного різноманіття (lexical diversity) і здатні цілком замінити статистичний слід LLM, підмінивши його нейтральним чи хаотичним [67]. Це руйнує типові n-grams, що впливає на методи на кшталт стилістичних, perplexity та watermarking.

- Багаторазовий переклад. Схожого ефекту - збереження близької семантики при значній зміні синтаксису та особливо словника можна досягти шляхом кількох циклів перекладу, наприклад, з англійської на українську і назад. Оскільки алгоритм машинного перекладу обирає токени іншим чином ніж LLM, то типовий слід ШІ буде з високою ймовірністю зруйновано [68].

2. Штучне вирівнювання ймовірнісного розподілу.

Атака спрямована в основному на zero-shot методи, а саме на їх дві основні ознаки: викривлення розподілу ймовірностей та локальну оптимальність вибору токенів. Атака намагається штучно підвищити ентропію тексту для імітації стилю людини.

- Adversarial перефразування - техніка використовує комбінацію детектора і парафразера і є одним з популярних засобів побудови сучасних humanizers. Суть методу лежить в ітеративному тренуванні парафразера, коли результати його роботи разом з AI score, отриманим від детектора, подаються йому назад у вигляді тренувальних даних. У результаті модель-парафразер вивчає патерн модифікації даних таким чином, щоб з максимальною ймовірністю отримати мітку Human. Дослідження [51] показує, що такі методи гарно переносяться і патерн, вивчений для обходу одного детектора добре спрацьовує і для обходу інших.
- Імітація високої температури генерації - кожна LLM має параметр температури (не завжди доступний у користувацькому інтерфейсі, проте часто доступний через API), який відповідає за зв'язність і логічність згенерованого тексту. Якщо задача генерації вимагає створення фіксованої структури, наприклад, JSON, то цей параметр встановлюють

у 0 (мінімальне значення); наприклад, для генерації програмного коду часто використовують відмінні від 0, проте малі значення; генерація типового тексту відбувається також при досить невисоких значеннях. Високі значення температури дозволяють моделі більш часто використовувати токени з низькою ймовірністю, що підвищує креативність, проте порушує зміст тексту. Перефразування тексту моделлю з високою температурою порушує патерн низької perplexity шляхом додавання шуму, що робить його прозорим для zero-shot methods [69].

3. Підміна символів.

Цей тип атаки широко відомий ще з часів перших систем визначення плагіату, але продовжує зберігати свою ефективність і проти AI detectors. Метод базується на експлуатації різниці між сприйняттям тексту людиною і машиною.

- Використання гомогліфів і Unicode символів - атака базується на застосуванні символів, які для людини виглядають однаково, або дуже схоже, а для машини є різними. Наприклад, заміна всіх літер “a” у латинському тексті кириличною літерою “а”. Така заміна плутає токенізатор і він починає розбивати слова на токени зовсім іншим способом, що створює унікальний набір токенів, якого модель-детектор раніше не зустрічала [70].
- Вставка невидимих символів - цей метод може приймати два види: (1) заміна пробілів на випадковий символ того ж кольору, що і фон тексту, така заміна є невидимою для людського ока, проте для машини перетворює текст на одне суцільне слово, що руйнує всі ознаки детектора; (2) вставка невидимих символів, наприклад, пробілів нульової ширини, або інших символів, присутність яких візуально не змінює текст - такий підхід руйнує очікувані послідовності токенів і плутає детектор [70].

4. Оптимізована чистка.

Метод націлений виключно на watermarking способи детекції, його основна мета полягає в очищенні тексту від слідів відбитка водяного знаку таким чином, щоб мінімізувати різницю між оригінальним текстом та його очищеною версією.

- Black-Box ScruBBing Attack. Гарним прикладом цього методу є модель b4, яка використовує проксі-модель для визначення нормального розподілу токенів, після чого замінює токени у вихідному тексті таким чином, щоб зменшити Kullback-Leibler divergence (розходження Кульбака-Лейблера) від нормального розподілу. Таким чином, досягається математично гарантоване очищення watermark при збереженні семантики [71].

1.3.2. Вимоги до нового покоління методів детекції

Загалом, підсумовуючи описане в підрозділі 1.2 і пункті 1.3.1, можна зобразити зв'язок між методами детекції та способами атаки таким чином (табл. 1.4). Проведений аналіз доводить, що немає жодного методу, який був би достатньо робастним в умовах цілеспрямованих атак. Обґрунтуванням НН1 (розробка гібридного методу ідентифікації автоматично згенерованих природномовних текстів) виступає той факт, що всі наявні методи детекції, як правило, спираються лише на один рівень ознак: статистично-стильові властивості згенерованих даних, синтаксичні характеристики, чи ймовірісно-семантичні патерни. Відповідно, кожен з таких методів є вразливим до атаки, спрямованої саме на підміну базових ознак його рівня, що є ключовими в детекції. Тому єдиним надійним способом побудови робастного детектора виглядає створення гібридного методу, що буде інкорпорувати ознаки з усіх трьох рівнів: стилістичного, структурного та семантичного. Такий детектор, з теоретичного погляду, має бути стійким до всіх типових атак і вразливим лише до трирівневої adversarial атаки. Дисертаційне дослідження не ставить мети зробити unbreakable detector (детектор, що неможливо зламати), через практичні

обмеження, проте, варто зазначити, що процес adversarial attacks має свої межі. З теоретичного погляду немає обмежень на кількість ітерацій у циклі навчання adversarial paraphraser у зв'язці з цільовим детектором для досягнення суттєвого зниження ефективності детектора; на практиці ж adversarial paraphraser також має два суттєвих обмеження: (1) текст на виході має бути зв'язним з помірною кількістю помилок; (2) текст має бути близьким за змістом до оригіналу. Якщо запропонований гібридний детектор буде демонструвати достатню робастність на даних, модифікованих з урахуванням цих двох обмежень, то завдання дослідження виконано успішно.

Таблиця 1.4 Зв'язок між трансформаційними атаками та методами детекції

Attack Category	Specific Attack Method	Primary Targeted Detector Type
Semantic Reconstruction	DIPPER / Recursive Paraphrasing	Stylometry & Perplexity
Semantic Reconstruction	Cross-Lingual Translation	Watermarking (KGW, EXP)
Adversarial Distribution	Adversarial Paraphrasing	Fine-Tuned Classifiers (RoBERTa, RADAR)
Adversarial Distribution	TempParaphraser	Zero-Shot (DetectGPT, Binoculars)
Tokenization Exploits	Homoglyph Substitution	Neural Classifiers & Embeddings
Optimization Scrubbing	b4 Scrubbing Attack	Watermarking (Unigram, KGW)
XAI Manipulation	Explanation Fooling	XAI-Driven Detection

Удосконалення концептуальної моделі оцінки робастності систем детекції сліду ІІІ (НН4) є необхідним як для побудови гібридного методу ідентифікації (НН1), так і для створення надійного механізму визначення критичних меж маскування текстів. Проведений у цьому розділі аналіз, систематизація та виявлення

обмежень існуючих методів детекції вказують не лише на потребу розробки гібридного методу (НН1), але і на необхідність подальшого розвитку методів забезпечення робастності систем обробки природної мови в умовах агресивного маскування згенерованого контенту та епістемічної невизначеності (НН3). Водночас нестача надійних і швидких методів інтерпретації результатів детекції доводить необхідність розробки нової інтерпретаційної моделі ХАІ (НН2). Вона має забезпечити візуалізацію обґрунтованих доказів детекції шляхом використання внутрішніх ваг метакласифікатора для генерації глобального та локального профілів атрибуції та карт доказів, знизивши часову складність побудови пояснень до $O(1)$.

1.3.3. Задача дослідження

Дослідження літератури, проведене в розділі 1, вказує на велику зацікавленість темою генерації текстів ШІ загалом та AI detection конкретно серед наукової спільноти та комерційних організацій. Актуальність теми пов'язана з широким проникненням ШІ в наше повсякденне життя і необхідністю тримати це проникнення під контролем у чутливих сферах, таких як освіта, медицина, юриспруденція та багато інших. Попри велику актуальність та практичну значущість теми та її цінність для широкого загалу, пропозиція засобів визначення ШІ залишається обмеженою і базується на п'яти ключових підходах. У зв'язку з високою конкуренцією в галузі, жоден з підходів не лишився не протестованим на надійність і, як показали подальші дослідження, всі існуючі методи детекції мають проблеми з надійністю, особливо в сценаріях маскування. Іншою суттєвою проблемою залишається відсутність стабільних та обчислювально прийнятних методів пояснення рішень детектора (ХАІ).

Таким чином, виникає важлива *науково-прикладна задача* з підвищення точності та надійності ідентифікації автоматично згенерованих природномовних текстів. Розв'язанню цієї задачі й присвячено дисертаційне дослідження. Отже, *метою дисертаційного дослідження* є підвищення точності та надійності ідентифікації автоматично згенерованих природномовних текстів в умовах

маскування та епістемічної невизначеності шляхом розробки гібридного методу, що інтегрує лексичні, синтаксичні та семантичні ознаки сліду ШІ.

Об'єктом дослідження є процеси ідентифікації автоматично згенерованих природномовних текстів у складних інформаційних умовах (зокрема, в умовах використання комерційних систем маскування, трансформаційних атак та невизначеності джерела генерації).

Предметом дослідження є гібридний метод ідентифікації автоматично згенерованих природномовних текстів у складних інформаційних умовах.

Для досягнення вказаної мети потрібно вирішити наступні завдання:

- Провести систематизацію та аналіз існуючих підходів до ідентифікації автоматично згенерованих текстів, виявити їхні вразливості до методів маскування та проблеми генералізації.
- Розробити математичну модель сліду ШІ в багатовимірному просторі ознак та побудувати на її основі концептуальну модель Точки зламу, яка відображає математичний взаємозв'язок між втратою семантичної цілісності тексту та падінням ймовірності детекції, для прогнозування чутливості системи до трансформаційних атак.
- Розробити гібридний метод ідентифікації згенерованих текстів на основі каскадного ансамблю класифікаторів з механізмами адаптивного зважування та аналізу міжрівневого дисонансу ознак.
- Застосувати метод індуктивного конформного передбачення (ICP) для квантифікації невизначеності, розпізнавання даних поза навчальним розподілом та мінімізації хибнопозитивних спрацювань.
- Розробити інтерпретаційну модель пояснюваного ШІ на основі вагових коефіцієнтів моделі-детектора для генерації профілів атрибуції та карт доказів, що відображають вплив ознак різних рівнів (лексичних, синтаксичних, семантичних) на кінцеве рішення системи.
- Розробити програмний комплекс та провести експериментальні дослідження для оцінювання точності ідентифікації, обчислювальної ефективності та робастності до трансформаційних атак порівняно із сучасними аналогами.

1.4. Висновки до розділу 1

У першому розділі розглянуто ключові компанії-розробники великих мовних моделей та їх стратегії створення фундаментальних моделей, основні архітектури LLM та принципи генерації контенту. Проаналізовано особливості генерації текстових даних, описано концепцію сліду ШІ, причини його виникнення, особливості його трансформації з виходом нових версій моделей та змінами архітектури, його рівні та принципи функціонування. Наведено основні підходи до визначення автоматично згенерованих текстів та які елементи AI Trace кожен з підходів використовує; особливу увагу зосереджено на базових гіпотезах, що лежать в основі кожного методу, і створюють певні теоретичні обмеження щодо їх застосовності. Перелічено найбільш актуальні способи модифікації тексту, що націлені на маскуванню AI Trace і, таким чином, уникнення детекції. Розглянуто чому кожен з методів модифікації тексту може успішно працювати проти певних видів детекції.

Завдяки використанню методів системного аналізу вдалося комплексно оцінити вразливості наявних підходів та показати, що, попри високу актуальність теми LLM та серйозний вплив ШІ-застосунків на щоденне життя та освіту [4], наявні інструменти детекції залишаються обмежено ефективними. Попри велику кількість підходів до виявлення ШІ, кожен з них містить як серйозні теоретичні обмеження щодо застосовності, так і вразливості до сценаріїв маскуванню, що робить процес визначення згенерованих даних досить ненадійним (низька точність розпізнавання згенерованого контенту) через високу кількість False Negative (помилкова класифікація AI тексту як Human).

З іншого боку існують проблеми з помилковою класифікацією написаних людиною текстів (низька точність розпізнавання людського контенту) через високу кількість False Positive (помилкова класифікація Human тексту як AI). Проблема False Positive, як правило, стоїть набагато більш гостро ніж проблема False Negative. Якщо умовна система-детектор ШІ має TPR (True Positive Rate - Частка Істинно Позитивних Результатів) на рівні 90% (тобто зі 100 AI текстів система правильно

визначить 90, а неправильно 10), то система може активно використовуватися в реальних сценаріях. Якщо умовна система-детектор має FPR (False Positive Rate - Рівень Хибнопозитивних Результатів) на рівні 10% (тобто зі 100 Human текстів система правильно визначить 90, а неправильно 10), то система не може використовуватися в реальних сценаріях. Тобто вимоги щодо точності визначення контенту, написаного людиною, - FPR мають набагато вищу вагу, ніж вимоги щодо точності визначення контенту, написаного LLM (TPR). Як показало дослідження, деякі типи детекції мають упередженість проти певних типів людських текстів, наприклад, perplexity методи проти текстів написаних у науковому стилі або неносіями мови. Проти інших методів, наприклад, watermarking, є спеціальні атаки, що примушують ці методи неправильно класифікувати модифіковані людські тексти. Відсутність стабільних та відносно швидких методів пояснюваного III лише погіршує ситуацію, оскільки кожного разу, коли експерт отримує позитивне спрацювання класифікатора, він постає перед питанням, наскільки можна покладатися на результат, виданий чорною скринькою.

Неідеальна точність наявних класифікаторів, теоретичні та практичні обмеження щодо їх застосування, велика кількість атак та типів маскування і відсутність стійких та швидких засобів інтерпретації результатів вказують на потребу в розробці як надійного гібридного методу ідентифікації автоматично згенерованих текстів, так і інтерпретаційної моделі для візуалізації доказів детекції. Отримані результати відповідають виконанню Завдання 1 дисертаційного дослідження.

Список використаних джерел до розділу 1

1. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30
2. Radford, Alec, et al. "Language models are unsupervised multitask learners." *OpenAI blog* 1.8 (2019): 9.
3. Ouyang, Long, et al. "Training language models to follow instructions with human feedback." *Advances in neural information processing systems* 35 (2022): 27730-27744.
4. Nykonenko, A. (2023). The impact of artificial intelligence on modern education: prospects and challenges. *Artificial Intelligence*, 2. Pp. 10-15.
5. Brown, Tom, et al. "Language models are few-shot learners." *Advances in neural information processing systems* 33 (2020): 1877-1901.
6. Nykonenko, A. (2025). A Systematic Review of LLM Advancements from GPT-3 to Reasoning Agents. *Artificial Intelligence*, 3. Pp. 32-43.
7. Jacobs, Robert A., et al. "Adaptive mixtures of local experts." *Neural computation* 3.1 (1991): 79-87.
8. Liu, Aixin, et al. "Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model." *arXiv preprint arXiv:2405.04434* (2024).
9. Guo, Daya, et al. "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning." *arXiv preprint arXiv:2501.12948* (2025).
10. Shazeer, Noam, et al. "Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer." *International Conference on Learning Representations (ICLR)*, 2017.
11. Lo, Ka Man, et al. "A Closer Look into Mixture-of-Experts in Large Language Models." *Findings of the Association for Computational Linguistics: NAACL 2025*, 2025. *arXiv*, arXiv:2406.18219.
12. Bandarkar, Lucas, et al. "Multilingual Routing in Mixture-of-Experts." *arXiv*, 2025, arXiv:2510.04694.

13. Hua, Yongxiang, et al. "Input Domain Aware MoE: Decoupling Routing Decisions from Task Optimization in Mixture of Experts." *arXiv*, 2025, arXiv:2510.16448.
14. OpenAI. "Learning to Reason with LLMs." OpenAI, 2024, openai.com/index/learning-to-reason-with-llms/.
15. Hao, Yuanzhen, and Desheng Wu. "Fact Verification on Knowledge Graph via Programmatic Graph Reasoning." *Findings of the Association for Computational Linguistics: EMNLP 2025*. 2025.
16. Achiam, Josh, et al. "GPT-4 Technical Report." *arXiv preprint arXiv:2303.08774*, 2023, arxiv.org/abs/2303.08774.
17. Team, G., et al. "Gemini: A Family of Highly Capable Multimodal Models." *arXiv preprint arXiv:2312.11805*, 2023, arxiv.org/abs/2312.11805.
18. Lee, J., et al. "Efficient LLaMA-3.2-Vision by Trimming Cross-attended Visual Features." *arXiv preprint arXiv:2504.00557*, 2025, arxiv.org/abs/2504.00557.
19. Alayrac, Jean-Baptiste, et al. "Flamingo: a visual language model for few-shot learning." *Advances in neural information processing systems* 35 (2022): 23716-23736.
20. Wang, Junchi, and Lei Ke. "Llm-seg: Bridging image segmentation and large language model reasoning." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024.
21. Drift Detection in Large Language Models: A Practical Guide, accessed November 7, 2025, <https://medium.com/@tsiciliani/drift-detection-in-large-language-models-a-practical-guide-3f54d783792c>
22. Choi, Junhyuk, et al. "Examining Identity Drift in Conversations of LLM Agents." *arXiv preprint arXiv:2412.00804* (2024).
23. Chakrabarty, Tuhin, Philippe Laban, and Chien-Sheng Wu. "Can ai writing be salvaged? mitigating idiosyncrasies and improving human-ai alignment in the writing process through edits." *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. 2025.

24. Li, Jiatao, et al. "AGENT-X: Adaptive Guideline-based Expert Network for Threshold-free AI-generated text detection." *arXiv preprint arXiv:2505.15261* (2025).
25. Can Turnitin Detect Perplexity.ai? - Walter Writes AI, accessed November 7, 2025, <https://walterwrites.ai/can-turnitin-detect-perplexity-ai/>
26. Gehrmann, Sebastian, Hendrik Strobelt, and Alexander M. Rush. "Gltr: Statistical detection and visualization of generated text." *arXiv preprint arXiv:1906.04043* (2019).
27. Nykonenko, A. (2024). How text transformations affect AI detection. *Artificial Intelligence*, 4. Pp. 233-241.
28. Teja, Lekkala Sai, et al. "Modeling the Attack: Detecting AI-Generated Text by Quantifying Adversarial Perturbations." *arXiv preprint arXiv:2510.02319* (2025).
29. E. Faure, A. Nykonenko: Analyzing the nature of AI footprint: noise-to-text method. Proceedings of the Computational Intelligence Application Workshop (CIAW 2024), Lviv, Ukraine, October 10-12, 2024, [CEUR-WS.org](https://ceur-ws.org/Vol-3861/paper14.pdf), online [CEUR-WS.org/Vol-3861/paper14.pdf](https://ceur-ws.org/Vol-3861/paper14.pdf).
30. Kadhim, Ahmed K., et al. "Adversarial attacks on AI-generated text detection models: A token probability-based approach using embeddings." *arXiv preprint arXiv:2501.18998* (2025).
31. Sadasivan, V. S., Kumar, A., Balasubramanian, S., Wang, W., & Feizi, S. (2023). Can AI-generated text be reliably detected? *arXiv preprint arXiv:2303.11156*.
32. D. Weber-Wulff, A. Anohina-Naumeca, S. Bjelobaba, T. Foltýnek, J. G. Guerrero-Dib, O. Popoola, P. Sigut, L. Waddington, Testing of Detection Tools for AI-generated Text, *International Journal for Educational Integrity* 19(1) (2023) 26.
33. S. S. Ghosal, S. Chakraborty, J. Geiping, F. Huang, D. Manocha, A. S. Bedi, Towards Possibilities & Impossibilities of AI-generated Text Detection: A Survey, *arXiv preprint arXiv:2310.15264* (2023).
34. Schuster, Tal, et al. "The limitations of stylometry for detecting machine-generated fake news." *Computational Linguistics* 46.2 (2020): 499-510.
35. Zaitsu, Wataru, et al. "Stylometry can reveal artificial intelligence authorship, but

- humans struggle: A comparison of human and seven large language models in Japanese." *PLoS One* 20.10 (2025): e0335369.
36. Marchenko, O., Anisimov, A., Nykonenko, A., Rossada, T., & Melnikov, E. (2017, June). Authorship attribution system. In *International conference on applications of natural language to information systems* (pp. 227-231). Cham: Springer International Publishing.
 37. Tosato, Tommaso, et al. "Persistent Instability in LLM's Personality Measurements: Effects of Scale, Reasoning, and Conversation History." *arXiv preprint arXiv:2508.04826* (2025).
 38. Kirk, Robert, et al. "Understanding the effects of rlhf on llm generalisation and diversity." *arXiv preprint arXiv:2310.06452* (2023).
 39. A Ship of Theseus: Curious Cases of Paraphrasing in LLM-Generated Texts
 40. Kumarage, Tharindu, et al. "Stylometric detection of ai-generated text in twitter timelines." *arXiv preprint arXiv:2303.03697* (2023).
 41. Lindberg, Karl, and Isak Nobel. "AI or Human? Detecting Machine-Generated Text Through Perplexity." (2025).
 42. Popkov, Andrey A., and Tyson S. Barrett. "AI vs academia: Experimental study on AI text detectors' accuracy in behavioral health academic writing." *Accountability in Research* 32.7 (2025): 1072-1088.
 43. Liang, Weixin, et al. "GPT detectors are biased against non-native English writers." *Patterns* 4.7 (2023).
 44. Saha, Shoumik, and Soheil Feizi. "Almost ai, almost human: The challenge of detecting ai-polished writing." *arXiv preprint arXiv:2502.15666* (2025).
 45. Hu, Xiaomeng, Pin-Yu Chen, and Tsung-Yi Ho. "Radar: Robust ai-text detection via adversarial learning." *Advances in neural information processing systems* 36 (2023): 15077-15095.
 46. Basani, Advik Raj, and Pin-Yu Chen. "Diversity boosts AI-generated text detection." *arXiv preprint arXiv:2509.18880* (2025).
 47. Liu, Xin, Yang Li, and Kan Li. "Enhancing the Robustness of AI-Generated Text Detectors: A Survey." *Mathematics* 13.13 (2025): 2145.

48. Zhou, Yinghan, et al. "Kill two birds with one stone: generalized and robust AI-generated text detection via dynamic perturbations." Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers). 2025.
49. Hans, Abhimanyu, et al. "Spotting llms with binoculars: Zero-shot detection of machine-generated text." arXiv preprint arXiv:2401.12070 (2024).
50. Mitchell, Eric, et al. "Detectgpt: Zero-shot machine-generated text detection using probability curvature." International conference on machine learning. PMLR, 2023.
51. Cheng, Yize, et al. "Adversarial Paraphrasing: A Universal Attack for Humanizing AI-Generated Text." arXiv preprint arXiv:2506.07001 (2025).
52. Zhu, Xiaowei, et al. "DNA-DetectLLM: Unveiling AI-Generated Text via a DNA-Inspired Mutation-Repair Paradigm." arXiv preprint arXiv:2509.15550 (2025).
53. Kirchenbauer, John, et al. "A watermark for large language models." International Conference on Machine Learning. PMLR, 2023.
54. Jovanović, Nikola, Robin Staab, and Martin Vechev. "Watermark stealing in large language models." arXiv preprint arXiv:2402.19361 (2024).
55. Liu, Aiwei, et al. "A survey of text watermarking in the era of large language models." ACM Computing Surveys 57.2 (2024): 1-36.
56. Zhang, Zhaoxi, et al. "Large language model watermark stealing with mixed integer programming." arXiv preprint arXiv:2405.19677 (2024).
57. Nemecek, Alexander, Yuzhou Jiang, and Erman Ayday. "Watermarking Without Standards Is Not AI Governance." arXiv preprint arXiv:2505.23814 (2025).
58. Ribeiro, Marco Tulio, Sameer Singh, and Carlos Guestrin. "" Why should i trust you?" Explaining the predictions of any classifier." Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. 2016.
59. Lundberg, Scott M., and Su-In Lee. "A unified approach to interpreting model predictions." Advances in neural information processing systems 30 (2017).
60. Nguyen, Hung Truong Thanh, et al. "Evaluation of explainable artificial

- intelligence: Shap, lime, and cam." Proceedings of the FPT AI Conference. 2021.
61. Rahulamathavan, Yogachandran, Misbah Farooq, and Varuna De Silva. "PLEX: Perturbation-free local explanations for llm-based text classification." arXiv preprint arXiv:2507.10596 (2025).
 62. Shukla, Manish. "Interpreting BERT Using LIME and SHAP." (2025).
<https://engrxiv.org/preprint/view/5078>
 63. Saarela, Mirka, and Vili Podgorelec. "Recent applications of Explainable AI (XAI): A systematic literature review." *Applied Sciences* 14.19 (2024): 8884.
 64. Mohammadi, Hadi, et al. "Explainability-Based Token Replacement on LLM-Generated Text." arXiv preprint arXiv:2506.04050 (2025).
 65. Liu, Jae QJ, et al. "The great detectives: humans versus AI detectors in catching large language model-generated medical writing." *International Journal for Educational Integrity* 20.1 (2024): 8.
 66. Yeadon, Will, et al. "Evaluating AI and human authorship quality in academic writing through physics essays." *European Journal of Physics* 45.5 (2024): 055703.
 67. Krishna, Kalpesh, et al. "Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense." *Advances in Neural Information Processing Systems* 36 (2023): 27469-27500.
 68. Ghanim, Mansour Al, et al. "Uncovering the Hidden Threat of Text Watermarking from Users with Cross-Lingual Knowledge." arXiv preprint arXiv:2502.16699 (2025).
 69. Huang, Junjie, et al. "TempParaphraser: "Heating Up" Text to Evade AI-Text Detection through Paraphrasing." *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. 2025.
 70. Creo, Aldan, and Shushanta Pudasaini. "Evading AI-generated content detectors using homoglyphs." arXiv e-prints (2024): arXiv-2406.
 71. Huang, Baizhou, Xiao Pu, and Xiaojun Wan. "b4: A black-box scrubbing attack on llm watermarks." *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. 2025.

РОЗДІЛ 2

МОДЕЛІ ТА ГІБРИДНИЙ МЕТОД ІДЕНТИФІКАЦІЇ СЛІДУ ШІ

У першому розділі досліджено різні рівні сліду ШІ та його особливості, включаючи вплив різних архітектур LLM та типів моделей (Multimodal, Multi-step Reasoning, MoE), параметрів генерації, Prompting Style, контексту, заданої персони та інших. Розглянуто, яким чином використовується AI Tracer для розв'язання задачі детекції та чим відрізняються підходи детекції, що базуються на використанні різних рівнів AI Tracer, їх теоретичні та практичні обмеження та умови, за яких вони працюють найбільш ефективно. Окремо описано основні вразливості кожного методу та поширені типи атак, що дають змогу супротивнику обійти механізми детекції. Проведене дослідження показало суттєві недоліки в SOTA методах AI Detection, особливо щодо їх робастності та інтерпретованості, які вимагають розробки нового методу детекції, що давав би адекватну відповідь на поставлені виклики. Вразливість сучасних методів детекції напряму пов'язана зі сприйняттям сліду ШІ як статичного об'єкта, що дає змогу легко маніпулювати точністю детекції шляхом внесення збурень через перефразування, заміну символів, вихід нових версій моделей чи архітектур. Для підвищення стійкості AI детекції проти трансформаційних атак пропонується замість масштабування (збільшення розміру детектора та обсягу тренувальних даних) змінити концепцію сліду ШІ так, щоб отримати стійкий багатомірний об'єкт, що включає ознаки трьох рівнів: семантичного, стилістичного і структурного.

Основним фокусом даного розділу є вирішення завдань 2-5 дисертаційного дослідження. Розділ пропонує підґрунтя для створення гібридного методу детекції, стійкого до використання в умовах невизначеності та трансформаційних атак. Окрім вирішення зазначених завдань, матеріал цього розділу формує математичну основу для всіх пунктів наукової новизни дисертаційної роботи, забезпечуючи формалізований перехід від моделювання сліду ШІ до його практичної ідентифікації та інтерпретації. Розділ сформовано таким чином, щоб спочатку дати математичне визначення об'єкта дослідження (підрозділ 2.1), потім визначити теоретичні межі поняття робастності детекції (підрозділ 2.2), дати алгоритмічний опис гібридного

методу (підрозділ 2.3), і представити опис пояснюваної моделі для інтерпретації результатів детекції (підрозділ 2.4).

2.1. Формалізація сліду ШІ в багатовимірному просторі ознак

У цьому підрозділі перейдемо від загального формулювання задачі до формального опису основних об'єктів, що використовуються у дисертаційному дослідженні. У першому розділі нами введено поняття AI Trace (слід ШІ) - це статистичне явище, яке пов'язане зі специфікою створення тексту великими мовними моделями. На відміну від людини, яка може підібрати слова, перебуваючи під дією великої кількості факторів, як внутрішніх, так і зовнішніх, LLM завжди формує текст шляхом підбору наступного найбільш імовірного токена. З математичного погляду цей процес описується як мінімізація від'ємної логарифмічної правдоподібності (log-likelihood) ланцюжка токенів. Відповідно, під час генерації тексту, модель розв'язує оптимізаційну задачу, що призводить до згладжування розподілу ймовірностей токенів і створює статистичний патерн, що відображається під час аналізу тексту на багатьох рівнях [1].

Раніше сигнал у LLM завжди передавався статичними маршрутами, проте підвищення складності архітектур призвело до появи динамічного формування маршрутів (наприклад, у MoE). Перехід від dense models до sparse models суттєво змінює AI Trace і вимагає його формалізації з урахуванням негаусової природи features, що створюються просунутими архітектурами на кшталт MoE та Reasoning Agents. Тому в цьому підрозділі буде розглянуто AI Trace не як скалярне значення, а як вектор \mathbf{T}_{AI} у просторі features \mathbf{F} (ознаки стилістичного, структурного, семантичного рівнів).

2.1.1. Математична модель сліду ШІ

Відповідно до класифікації рівнів сліду ШІ, запропонованої нами в [2] та уточненої у першому розділі (пункт 1.1.2) дисертаційного дослідження, простір ознак \mathbf{F} складається з наступних трьох вимірів:

- 1) Лексичний рівень (або стилістичний) (F_{lex}) - характеризується використанням специфічної лексики (слів або токенів) і описує стилістичні особливості згенерованого тексту. Включає різні частотні характеристики токенів, N-grams, ентропію, perplexity, burstiness і т. ін.
- 2) Синтаксичний (або структурний) рівень (F_{syn}) - описує особливості структури згенерованого тексту. Включає частоти POS Tags, синтаксичну структуру речень та особливості їх конструкції, ритм (POS Tags N-grams), глибину та довжину речень і т. ін.
- 3) Семантичний рівень (F_{sem}) - описує особливості семантики згенерованого тексту. Включає аналіз цілісності, зв'язності та логічної послідовності тексту.

Для формування багатовимірного простору ознак застосовуються методи обробки природної мови, що дають змогу виділити специфічні метрики на кожному рівні. Сам простір ознак F задається так:

$$F = F_{lex} \times F_{syn} \times F_{sem} \quad (2.1)$$

Варто зазначити, що кожен із просторів F_{lex} , F_{syn} та F_{sem} є багатовимірним ($F_{lex} \subset \mathbb{R}^k$, $F_{syn} \subset \mathbb{R}^m$, $F_{sem} \subset \mathbb{R}^n$, де k, m, n — кількість ознак відповідного рівня). Відповідно, слід ШІ T_{AI} для тексту X формується як вектор у загальному просторі ознак $F \subset \mathbb{R}^{k+m+n}$, а компоненти $f_{lex}(X), f_{syn}(X), f_{sem}(X)$ є векторами відповідних ознак:

$$T_{AI}(X) = \Phi(f_{lex}(X), f_{syn}(X), f_{sem}(X)) \quad (2.2)$$

де Φ є функцією об'єднання, що відповідає за нелінійний характер залежності між рівнями.

Далі визначено функцію f для кожного з рівнів:

- 1) **Лексичний рівень f_{lex}** . Можна стверджувати, що на лексичному рівні слід ШІ в основному представлено через розподіл ймовірностей. Процес навчання LLM неявним чином стимулює їх максимізувати ймовірності розподілу,

вивченого з тренувальних даних (тобто брати певний “середній” розподіл для будь-якої послідовності tokenів, вивчений з усіх контекстів, наявних в тренувальних даних). Таким чином, LLM тяжіють до вибору tokenів з найбільш ймовірного регіону ймовірнісного розподілу - щоразу намагаючись вибирати один з найбільш ймовірних tokenів у ролі продовження кожної послідовності. Подальше використання додаткових технік вирівнювання виходу моделей для формування безпечних та корисних відповідей, на кшталт RLHF, у ролі побічного результату, призводить до подальшого звуження розподілу [3].

- а) **Perplexity (PPL)**. Це стандартна метрика [4] для того, щоб визначити, наскільки добре модель передбачає приклад. Для тексту T , заданого послідовністю tokenів (t_1, t_2, \dots, t_N) perplexity відносно моделі M визначається так:

$$PPL(T) = \exp\left(-\frac{1}{N} \sum_{i=1}^N \ln P_M(t_i | t_{1:i-1})\right) \quad (2.3)$$

де $P_M(t_i | t_{1:i-1})$ - умовна ймовірність токена t_i у моделі M за умови контексту $t_{1:i-1}$.

Перплексія показує середнє значення кількості найбільш ймовірних варіантів tokenів, що модель має на кожному кроці, звідси її зв'язок з ентропією Шеннона [5-6]:

$$PPL(T) = 2^{H(T)} \quad (2.4)$$

де $H(T)$ - ентропія Шеннона.

Очікується, що $PPL(T_{\text{HUMAN}}) > PPL(T_{\text{AI}})$, оскільки в середньому люди обирають слова з меншою умовною ймовірністю для передачі певних нюансів або використовуючи свій власний стиль мовлення чи викладення думок. У такому випадку, можна сказати, що текст, написаний людиною, характеризується вищою ентропією. Проте, як вже

обговорювалося у Розділі 1, покладатися лише на perplexity у детекції AI є недостатнім через слабкості методу до певних технік маскуванню, на кшталт складних промптів чи підвищення температури генерації.

- b) **Burstiness**. Вибуховість демонструє зміни perplexity у часі, або іншими словами, вимірює різноманітність тексту. З погляду лінгвістики, людське письмо характеризується високою різноманітністю - короткі, прості речення (висока прогнозованість, мала ентропія, низька perplexity) можуть чергуватися з довгими, складними конструкціями (низька прогнозованість, висока ентропія, висока perplexity) [7]. З погляду теорії інформації, такі чергування створюють різкі стрибки ("вибухи") нової інформації. З формального погляду, burstiness вимірює дисперсію ентропії, у той час як perplexity вимірює середнє значення ентропії.

Дамо формальне визначення burstiness на базі перплексії речень: нехай $S = \{s_1, s_2, \dots, s_K\}$ - це набір речень у тексті T , а $ppl(s_j)$ - перплексія речення j . Тоді визначимо p як набір значень перплексії $p = \{ppl(s_1), ppl(s_2), \dots, ppl(s_K)\}$, а для визначення burstiness використаємо Фактор Фано (індекс дисперсії) [8]:

$$\beta = \frac{\sigma_p^2}{\mu_p} = \frac{\frac{1}{K-1} \sum_{j=1}^K (ppl(s_j) - \mu_p)^2}{\frac{1}{K} \sum_{j=1}^K ppl(s_j)} \quad (2.5)$$

де σ_p^2 - дисперсія (мінливість) перплексії між реченнями,

μ_p - середнє значення (мат. сподівання) перплексії всіх речень T ,

K - кількість речень у тексті T ,

$ppl(s_j)$ - перплексія j -го речення (s_j).

Використання burstiness дає змогу кількісно вимірювати відхилення стилю. Низька β (значення близькі до 1) вказує на монотонний стиль і однорідний потік ентропії, що характерні для AI

стилю. Висока β (значення суттєво більші за 1) вказує на наявність переходів від низької до високої ентропії (вибухів), що характерні для людського стилю. У задачах стилometriї, як правило, використовується нормалізована *burstiness* в інтервалі $[-1, 1]$, що дає змогу виконувати порівняння між документами різної довжини [9]:

$$\beta_{norm} = \frac{\sigma_p - \mu_p}{\sigma_p + \mu_p} \quad (2.6)$$

де σ_p - стандартне відхилення перплексії,

μ_p - середнє значення (мат. сподівання) перплексії всіх речень T .

β_{norm} показує рівномірність ритму. Для AI текстів характерне його значення, близьке до -1, тоді як для написаних людиною текстів очікується $\beta_{norm} > 0$.

2) **Синтаксичний рівень f_{syn}** . Якщо на лексичному рівні основну увагу було зосереджено на тому, що говориться в тексті, то на синтаксичному рівні у фокусі уваги те, як саме це сказано. У більшості випадків LLM має досить високий рівень володіння мовою (справедливо для мов з великою кількістю доступних онлайн-ресурсів), проте спосіб вираження думок часто може схилитися до обмеженого набору синтаксичних шаблонів і конструкцій. Як приклад тут можуть виступати синтаксичні патерни, наведені в Розділі 1 у табл. 1.2 пункту 1.1.2.

а) **Divergence**. Для вимірювання f_{syn} буде використовуватися *divergence* (дивергенція) POS Tags N-grams. Нехай $T_{POS} = (pos_1, pos_2, \dots, pos_N)$ - набір POS-тегів, отриманих шляхом аналізу тексту однією з NLP-бібліотек, що відповідає набору токенів (t_1, t_2, \dots, t_N) тексту T . Тоді для тексту, що аналізується, можна обчислити розподіл імовірностей триграм POS-тегів $P_{obs}(pos_i, pos_{i+1}, pos_{i+2})$ і порівняти його з еталонним розподілом P_{ref} .

обчисленим на великому корпусі текстів, написаних людьми. Для порівняння P_{obs} з P_{ref} буде використовуватися дивергенція Кульбака-Лейблера [10], яку також часто називають відносною ентропією і позначають $D_{KL}(P||Q)$. Дивергенція Кульбака-Лейблера - це певний тип статистичної відстані, що вказує на те, наскільки апроксимуючий розподіл імовірностей Q відрізняється від істинного розподілу імовірностей P .

Оскільки формула Кульбака-Лейблера не є симетричною, то важливим питанням буде, який з варіантів її застосування $D_{KL}(P_{obs}||P_{ref})$ чи $D_{KL}(P_{ref}||P_{obs})$ є більш коректним для даного дослідження? Може здатися, що другий запропонований варіант ($D_{KL}(P_{ref}||P_{obs})$) є більш коректним, оскільки використовує P_{ref} як істинний розподіл імовірностей, а P_{obs} у ролі його апроксимації. Таке застосування зміщує фокус на триграми, які часто трапляються в еталонному тексті (P_{ref}), але рідко в тексті, що аналізується (P_{obs}). Тобто формула штрафує за невикористання важливих патернів, що мають бути в людському тексті. Обернене ж формулювання $D_{KL}(P_{obs}||P_{ref})$ фокусується на триграмах, що часто трапляються в аналізованому тексті (P_{obs}), але рідко в еталонному тексті (P_{ref}). Тобто таке застосування формули штрафує за використання нетипових для людини патернів.

Відомо, що при генерації тексту LLM схильється до використання деякого “середнього” синтаксису, вивченого на навчальних даних; цей середній синтаксис має бути близьким до P_{ref} (за умови, що для отримання P_{ref} використовувався достатньо великий, різномірний корпус). Разом з тим, як було показано в пункті 1.1.2, AI має схильність до використання деяких “оптимальних” ланцюжків POS Tags, що

здаються моделі найбільш безпечними для різних задач. Очікується, що саме ці оптимальні ланцюжки й будуть найчастіше проявлятися в згенерованому тексті. Отже, потрібне таке використання D_{KL} , яке дасть змогу виявляти невідповідність між частими патернами спостережуваного тексту й еталоном. Тому буде використовуватися обернене формулювання $D_{KL}(P_{obs}||P_{ref})$ для обчислення f_{syn} .

У цьому випадку D_{KL} на множині всіх можливих POS Tags триграм G розраховується наступним чином:

$$D_{KL}(P_{obs}||P_{ref}) = \sum_{g \in G} P_{obs}(g) \log \left(\frac{P_{obs}(g)}{P_{ref}(g)} \right) \quad (2.7)$$

Формула (2.7) дає змогу обчислювати, наскільки розподіл синтаксичних патернів у аналізованому тексті відрізняється від типового людського письма. Низьке значення D_{KL} буде вказувати на середній, згладжений синтаксис з малою кількістю структурних відхилень, що є типовою характеристикою штучно згенерованого тексту. Високе значення D_{KL} вказує на нетиповий, складний або специфічний (“авторський стиль”) синтаксис. З іншого боку, якщо LLM буде надмірно використовувати обмежений список безпечних шаблонів POS-тегів, то це також може призвести до високого значення D_{KL} . Для розрізнення цих двох ситуацій вводиться додаткова метрика дисперсії синтаксичної глибини.

- b) **Дисперсія синтаксичної глибини.** Введемо метрику для вимірювання дисперсії глибини Dependency Parse Tree (Дерева розбору залежностей) речення і позначимо її як σ_{depth}^2 . Глибиною Dependency Parse Tree будемо називати довжину найдовшого зі шляхів у дереві розбору від його кореня до листків. Обчислення дисперсії буде проходити в кілька етапів:

- i) Розбір (синтаксичний парсинг) кожного речення s_j тексту T для побудови дерева залежностей.
- ii) Визначаємо глибину дерева $depth_j$ для кожного речення s_j як найдовший зі шляхів від кореня дерева (слово, що не залежить від будь-якого іншого) до всіх його листків.
- iii) Обчислюємо дисперсію σ_{depth}^2 для глибин ($depth_1, depth_2, \dots, depth_K$):

$$\sigma_{depth}^2 = \frac{1}{K-1} \sum_{j=1}^K (depth_j - \mu_d)^2 \quad (2.8)$$

де K - кількість речень у тексті T ,

$depth_j$ - максимальна глибина дерева залежностей j -го речення,

μ_d - середня синтаксична глибина всіх речень T .

Дерева залежностей є гарним інструментом для вимірювання синтаксичної складності тексту, оскільки відображають внутрішню організацію речення через залежності між словами. Очікується, що LLM схильні до мінімізації глибини синтаксичних дерев для уникнення помилок і спрощення сприйняття тексту. Відповідно, низька σ_{depth}^2 (близька до 0) буде вказувати на малу варіабельність глибини речень, тобто однотонність, характерну LLM. Висока σ_{depth}^2 - вказує на високу варіативність складності речень, що є типовим для написаного людиною тексту.

- 3) **Семантичний рівень f_{sem}** . На цьому рівні йдеться про семантичну зв'язність та цілісність контенту щодо думок, ідей та підходів. Оскільки моделі-трансформери, як правило, генерують текст на базі наявного контексту (тобто до розгляду беруться попередні M tokenів, де M - довжина контекстного вікна), то кожен наступний token є оптимальним продовженням

попереднього тексту (з погляду вивченого моделлю розподілу). Тобто зв'язність згенерованого тексту в певному невеликому околі від конкретного токена буде дуже високою, іноді аж надто високою, порівняно з нормальним, написаним людиною текстом. Водночас через особливості механізмів генерації, ширший контекст може втрачати зв'язність і цілісність [11].

Для аналізу семантичної зв'язності буде використовуватися VSM (Vector Space Models - векторні моделі) для побудови embeddings (ембедінги) [12]. Представлення тексту у векторному вигляді пройшло значний шлях від простої позиції-індексу в словнику, через TF/IDF і word-vectors (вектори слів) до контекстно-семантичних ембедінгів, що обчислюються сучасними моделями на кшталт BERT і RoBERTa. Нехай $E(\bullet)$ - це embedding функція, що забезпечує проєкцію речення s у вектор $v \in R^d$. Такі embeddings становлять собою семантику, або зміст речення і можуть бути використаними для знаходження різких тематичних переходів або інших проблем у зв'язності тексту. Як спосіб визначення семантичної зв'язності та узгодженості буде вимірюватися cosine similarity (косинусна подібність) між парами векторів суміжних речень. Косинусна подібність визначається наступним чином:

$$Sim(s_j, s_{j+1}) = \cos(\theta_j) = \frac{v_j \cdot v_{j+1}}{|v_j| |v_{j+1}|} \quad (2.9)$$

де v_j та v_{j+1} - embedding вектори речень s_j та s_{j+1} ,

θ_j - кут між цими векторами.

Введемо вектор локальної узгодженості (coherence) $C = \{Sim(s_1, s_2), Sim(s_2, s_3), \dots, Sim(s_{K-1}, s_K)\}$. На основі цього вектора можна порахувати дві важливі характеристики семантичної узгодженості:

- Середню узгодженість (μ_d). Очікується висока середня узгодженість у текстах згенерованих ШІ, оскільки LLM натреновані для генерації семантично близьких речень і не схильні до вводу неочікуваних

концептів чи змін теми, за умови відсутності спеціального prompt engineering чи маніпуляцій з температурою.

- Дисперсію узгодженості (σ_c^2). Очікується нижча дисперсія узгодженості в текстах згенерованих ШІ, оскільки люди більш схильні до різких змін теми, відступів, цитувань і переходів між ідеями.

Для відстежування можливих галюцинацій і дрифту тематики в довгих текстах вводиться метрика Глобальної Дистанції до Центроїда (Global Centroid Distance) - D_{global} . Спочатку обчислюється основна тема тексту (Глобальний Центроїд) v_{global} , як середнє значення всіх ембедінгів речень:

$$v_{global} = \frac{1}{K} \sum_{j=1}^K v_j \quad (2.10)$$

Далі вимірюємо середню відстань від Global Centroid до кожного речення:

$$D_{global} = \frac{1}{K} \sum_{j=1}^K (1 - \text{sim}(v_j, v_{global})) \quad (2.11)$$

Висока D_{global} може бути ознакою дрифту теми тексту - коли на початку тексту йдеться про одну тему, а в кінці зовсім про іншу. Хоча жодна з запропонованих вище семантичних метрик не може сама чітко розділити написаний людиною і ШІ тексти, проте їх комбінації дають набагато більш чітко інтерпретований результат. Наприклад, висока D_{global} у поєднанні з високою μ_d майже напевно свідчать про дрифт теми та можуть бути потужним маркером згенерованого тексту.

2.1.2. Аналіз негаусового розподілу ознак генеративних моделей

Перелічені вище features, як правило, покладаються на нормальний або гаусівський розподіл даних, що аналізуються. Як описано в пункті 1.1.1, деякі просунуті архітектури на кшталт МоЕ або Multi-step Reasoning можуть генерувати негаусівський розподіл даних або створювати велику кількість аномалій. У цьому пункті буде описано, як поведінка таких моделей має бути врахована у вирішенні завдання AI Detection.

- 1) **Особливості МоЕ моделей.** Сучасні LLM, побудовані з використанням архітектури Mixture of Experts, мають ряд суттєвих переваг з погляду економії ресурсів як під час навчання моделі, так і на етапі інференсу. Особливістю цих моделей є можливість використовувати не всі нейрони кожного шару (як це роблять “щільні” моделі), а обирати лише певні блоки (експертів) залежно від задачі, що вирішується. Як правило, модель має певну кількість блоків-експертів, які активуються спеціальним механізмом (Router); звідси й походить назва архітектури. Прикладами таких моделей можуть виступати DeepSeek-V3, Grok-1, Mixtral 8x7B та багато інших; через особливості роботи механізму активації їх часто називають “розрідженими” моделями на противагу “щільним” моделям.

Формально вихід y для вхідного сигналу x з МоЕ-шару мережі описується так:

$$y = \sum_{i=1}^n G(x)_i E_i(x) \quad (2.12)$$

де $E_i(x)$ - вихід i -ї мережі-експерта,

$G(x)_i$ - функція шлюзування/gating function; зазвичай Top-k Softmax,

індекс i означає i -й елемент вектора

По суті, робота МоЕ-моделей нагадує функціонування мультимодальних моделей, проте роль різних модальностей тут виконують мережі-експерти

(наприклад, експерти з літератури, фізики, математики та написання коду). Якщо взяти до уваги той факт, що $G(x)$, як правило, повертає не одного експерта, а кілька (залежить від того, який k було обрано для Softmax), причому $G(x)$ використовується для кожного шару мережі, отже, результатом є дуже складна і заплутана ситуація з генерацією. Виходить, що перший токен може бути згенеровано як результат виходу (e_1, e_2) на першому шарі, (e_1, e_5) на другому та (e_6, e_7) на третьому; а наступний токен може бути згенеровано комбінацією (e_3, e_4) , (e_1, e_3) , (e_5, e_7) . Такий текст буде виглядати дуже неоднорідним, оскільки він становить собою результат роботи комбінації нейронних мереж, де кожен токен було згенеровано випадковою підмножиною з них. Для таких текстів будуть типовими висока ентропія і висока perplexity, імовірно, і burstiness такого тексту буде також високою, адже з погляду “щільної” моделі переходи імовірності його токенів виглядатимуть дуже неочікувано.

Зрозуміло, що для текстів з такими характеристиками використання середнього і дисперсії не є достатніми інструментами. Спираючись на методи математичної статистики, дослідимо негаусівський розподіл ознак генеративних моделей за допомогою оцінки щільності ядра (KDE). Метод дає змогу проводити неперервну оцінку розподілу випадкової величини й визначити форму функції її розподілу. Основна різниця між KDE та традиційними методами вимірювання середніх оцінок у вибірці полягає в тому, що в другому випадку занадто згладжуються спостережувані значення і розмиваються сліди індивідуальних експертів. Використання KDE має дати змогу побачити піки, що відповідають моделям-експертам, у функції розподілу без надмірного усереднення [13].

Формально KDE описується в наступному вигляді:

$$\widehat{f}_h(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x - x_i}{h}\right) \quad (2.13)$$

де K - функція ядра, симетрична, з інтегралом рівним 1,

h - параметр згладжування,

x_i - спостережувані значення ознаки в тексті,

n - кількість спостережень

Використання $\hat{f}_h(x)$ для визначення природи тексту має такі наслідки.

Для МоЕ моделей очікувано побачити на графіку кілька піків - це явна ознака мультимодальності системи генерації. Якщо порівнювати вигляд функції $\hat{f}_h(x)$ для “щільної” архітектури та “розрідженої”, то в першому випадку очікується гладкий одномодальний розподіл спостережуваної ознаки, а у другому випадку спостерігатимуться статистичні викривлення - наприклад, множинні максимуми або інші аномалії.

Також важливою характеристикою є ексцес E (kurtosis), який показує крутість піка розподілу та важкість його хвостів. Для нормального розподілу $E=0$, значення $E>0$ вказує на високий пік і багато екстремальних значень (важкі хвости), а значення $E<0$ вказує на низький пік, легші хвости. Визначається ексцес наступним чином:

$$E = \frac{\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^4}{\left(\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2\right)^2} \quad (2.14)$$

де x_i - спостережувані значення ознаки в тексті,

\bar{x} - середнє арифметичне значення ознаки в тексті,

Чисельник цієї формули μ_4 - центральний момент четвертого порядку, а знаменник - квадрат вибіркової дисперсії σ^4 . Для порівняння з нормальним розподілом у статистиці, як правило, використовують надлишковий ексцес, що обчислюється як $E-3$. Ця інформація важлива з того боку, що значення $E>3$

вказує на суттєву відмінність розподілу від нормального - тобто модель покладається на вузькі розподіли, що генеруються експертами.

Обидві формули (2.13) і (2.14) можна застосовувати до всього вектора ознак $(f_{lex}, f_{syn}, f_{sem})$ для визначення архітектурної сигнатури моделі.

2) Особливості Reasoning моделей. Моделі, здатні до багатокрокового міркування (на кшталт моделей o-серії від OpenAI та DeepSeek-R1), використовують спеціальну техніку Chain-of-Thought для генерації більш точних відповідей. У просунутих випадках, покрокове міркування дає змогу моделі будувати ціле дерево міркувань, відсікаючи невдалі гілки та генеруючи фінальний результат на базі найбільш вдалих шляхів міркування. З погляду детекції згенерованих даних Reasoning моделі цікаві тим, що вони не генерують відповідь відразу, як звичайні LLM, замість цього вони будують Reasoning Trace, який не завжди видимий користувачу, проте який завжди суттєвим чином впливає на фінальний результат моделі. У випадку звичайної моделі на розподіл токенів у її відповіді найбільше впливають три фактори: текст запиту користувача, обрані налаштування моделі (наприклад, температура) та вивчені ваги. У випадку Reasoning Agent з'являється четвертий параметр - ланцюжок думок, де кожен наступний крок найбільше залежить від попередніх міркувань; наявність Reasoning Trace дає змогу моделі генерувати відповіді, що досить далекі і за стилем, і за змістом від оригінального запиту користувача.

У фінальній відповіді моделі ланцюжок думок, як правило, відсутній, проте сама відповідь містить певний його відбиток [14]. Моделюється це через Періодичний Семантичний Сигнал; в основі цього моделювання лежить структурний патерн, якому підкоряється будь-який процес міркування моделі: Гіпотеза->Аналіз->Рішення. Отже, фінальний результат моделі буде складатися з поєднання елементів багатьох таких міркувань, що має створювати певні коливання як ентропії, так і семантичної зв'язності

(відповідь моделі має складатися з набору окремих сильно семантично зв'язаних всередині блоків), тобто мати певний “ритм”. Разом з тим, текст написаний людиною має бути аперіодичним, без різких коливань семантичної зв'язності та явних повторюваних патернів.

Для виявлення такої покрокової генерації буде використовуватися швидке перетворення Фур'є (FFT) як “детектор ритму”. Як вимірювану величину використовуватимуть ентропію речень. Аналіз базується на припущенні, що під час процесу мислення ентропія має змінюватися наступним чином:

- Початок циклу: формулювання гіпотези (висока ентропія - формулювання нової ідеї)
- Середина циклу: аналіз (ентропія зменшується)
- Кінець циклу: рішення/висновок (ентропія мінімальна)

Тобто висувається гіпотеза, що процес мислення моделі формує певні “хвилі” ентропії, і оскільки фінальний результат моделі базується на цих хвилях, то логічно припустити, що він також буде містити їх певний відбиток. З іншого боку, текст написаний людиною буде містити певні хаотичні зміни ентропії та не підпорядковуватися ритму CoT. FFT дає змогу проаналізувати ці зміни ентропії в часі.

Нехай $H = \{H(s_0), H(s_1), \dots, H(s_{K-1})\}$ - ентропія речень тексту T . Тоді для проведення спектрального аналізу необхідно обчислити $F(\omega)$ (спектральний підпис моделі) наступним чином:

$$F(\omega) = \sum_{j=0}^{K-1} H(s_j) e^{-i2\pi j\omega/K} \quad (2.15)$$

де ω - частота повторюваності певного патерну, $\omega \in \{0, 1, \dots, K-1\}$,

$F(\omega)$ - показує амплітуду (силу) частоти ω в тексті,

$H(s_j)$ - ентропія j -го речення,

$e^{-i\pi jw/K}$ - комплексний експоненціальний множник (ядро перетворення), у такій ситуації працює як еталонний ритм.

- i - уявна одиниця,
- 2π - повний цикл у радіанах,
- j - номер речення (виконує роль індексу поточного часового кроку),
- ω - індекс частоти для перевірки,
- K - загальна кількість речень у тексті (для нормалізації)

Оскільки невідомі ні розмір патерну, ні кількість його повторень в одержаному тексті, то використовують $\omega \in \{0, 1, \dots, K-1\}$, щоб перевірити всі можливі варіанти кількості повних циклів повторення. Наприклад, $\omega = 1$ означає, що весь текст містить лише один цикл, а $\omega = 5$ означає, що певний патерн повторився в тексті 5 разів; $\omega = 0$ ігнорується. Найбільший інтерес становить період $T = K/\omega$, який показує, через скільки речень модель повторює свій крок. Для визначення того, яке значення ω має використовуватися для обрахунку T , потрібно знайти частоту ω_{max} з максимальною амплітудою (або максимум спектра); значення $|F(\omega)|$ дає змогу це зробити.

Чисто математично, формула $T = K/\omega_{max}$ завжди поверне значення T для будь-якої послідовності слів, тому потрібен додатковий параметр, що дає змогу валідувати наявність ритму, а не лише визначати його період. Для цього буде використовуватися наступна формула:

$$Score_{freq} = \frac{\max_{w \neq 0} |F(\omega)|}{\text{average}_{w \neq 0} |F(\omega)|} \quad (2.16)$$

де $\max_{w \neq 0} |F(\omega)|$ - найбільше значення спектра даного тексту,

$\text{average}_{w \neq 0} |F(\omega)|$ - середня сила сигналу в тексті (фон)

Параметр $Score_{freq}$ показує відношення сигналу до шуму. Якщо значення $Score_{freq}$ вище за певний поріг, це означає, що в тексті наявний ритм, і можна обчислити значення T . Комбінація значень T і $|F(\omega)|$ дає змогу проводити поглиблений статистичний аналіз структури CoT тексту. Якщо значення $Score_{freq}$ нижче порогового, це означає, що ритму немає і текст не було згенеровано за допомогою CoT.

2.2. Концептуальна модель стійкості детекції

У даному підрозділі розглянуто поняття робастності та питання моделювання поведінки детектора при обробці модифікованого тексту. У першому розділі дисертаційного дослідження проаналізовано типові методи атаки на AI Detectors. Методи можуть суттєво відрізнятися залежно від таргетованого детектора та базових принципів його роботи, проте загалом можна сказати, що фокусом типової атаки є зміна AI Trace при максимальному збереженні семантики тексту. Робастність визначає можливості системи працювати в умовах збурень, невизначеності та зовнішнього впливу; у рамках реалізації Наукової новизни 4 описано концептуальну модель для прогнозування робастності детектора без прямого оцінювання стійкості його роботи у всіх можливих сценаріях модифікації тексту.

2.2.1. Метрики семантичної цілісності та функції втрат

Спочатку введемо математичний апарат для оцінювання атак модифікації тексту. Кожну атаку можна виміряти за допомогою двох параметрів: ступеня спотворення AI Trace (що і є метою атаки) та відхилення від оригінальної семантики (що є своєрідною платою за маскування). Зрозуміло, що метою будь-якої практичної Bypassing системи є уникнення детекції (максимізація першого параметру) при збереженні оригінальної семантики (мінімізація другого параметру). Крайніми випадками будуть ситуації створення повністю нечитабельного тексту, що не

визначається AI Detector (дуже високе спотворення AI Trace, але низьке збереження оригінальної семантики) та мінімальне перетворення тексту без уникнення детекції (низьке спотворення AI Trace, але високе збереження оригінальної семантики).

Ступінь втрати AI Trace позначатиметься L_{trace} (AI Trace Loss), а ступінь зміни семантики позначатиметься як I_{sem} (Semantic Integrity). Зрозуміло, що обидві метрики має сенс вимірювати тільки для створених AI текстів. Існує кілька сучасних способів виміряти зміни семантики тексту [15] і [16] а також класичні метрики на кшталт BLEU [17]. Класичні метрики базуються на підрахунку кількості збігів N-грам tokenів та POS-тегів, що явно не є достатнім для оцінки семантичної відповідності. З іншого боку, метрики, описані в [15] і [16], є часто занадто потужними (а також повільними та дорогими, як у випадку [15]) чи занадто сфокусованими на певному домені (як у випадку [16]). Тому основним інструментом вимірювання слугуватиме метрика SBERT [18], яка наразі є де-факто стандартом для такого класу задач. Альтернативним варіантом базової метрики може бути BERTScore [19], що дає змогу проводити глибший аналіз.

Нехай $E(X)$ це векторне представлення тексту X , а $E(X')$ це векторне представлення тексту після модифікації. Тоді семантична цілісність I_{sem} визначається через косинусну подібність:

$$I_{sem}(X, X') = \frac{E(X) \cdot E(X')}{\|E(X)\| \|E(X')\|} \quad (2.17)$$

Значення I_{sem} демонструє ступінь успішності атаки з погляду збереження семантики, якщо воно вище певного порогового значення (як правило, 0.85-0.90), то можна стверджувати, що оригінальну семантику було збережено.

Для обчислення L_{trace} потрібно ввести функцію детекції AI Trace - $D(X) \in [0, 1]$, яка вказує на ймовірність того, що текст було згенеровано LLM. Тоді втрата AI Trace визначається так:

$$L_{trace}(T, X) = D(X) - D(T(X)) \quad (2.18)$$

де $T(X)$ - оператор трансформування тексту

Високі значення L_{trace} свідчать про успішність атаки з погляду приховання AI Trace, низькі значення говорять про недостатність трансформації T для уникнення детекції. Тут виникає цікавий момент: основною метою системи-парафразера, що здійснює атаку, є досягнення високого значення L_{trace} , але, згідно з формулою (2.18), L_{trace} не можна виміряти без використання детектора D . Тобто, як демонструє формула (2.18), L_{trace} залежить як від D , так і від T ; отже, для певної атаки T ефективність парафразера може суттєво відрізнятись залежно від детектора D , що використовується.

Робастність детектора D визначається його здатністю протистояти втраті сигналу AI Trace при використанні Bypassing технік, тобто детектор, що здатен зберігати низьке значення L_{trace} для різних типів атак, буде вважатися робастним. При цьому, з практичних міркувань, накладено обмеження на сценарії, що розглядаються, звужуючи їх лише до тих, де I_{sem} зберігається високою. Сценарії з низьким I_{sem} описано в нашій роботі [11]. Візуально подати робастність детектора можна за допомогою Trade-off Curve (криву компромісу) як графік залежності L_{trace} від I_{sem} .

2.2.2. Визначення критичних меж маскування

У роботі [2] нами показано, що підсумковий показник AI Score згенерованого й модифікованого тексту залежить від сили застосованих модифікацій: радикальніші трансформації (на кшталт видалення слів) призводять до суттєвішого зниження оцінки. У цьому пункті описано математичну модель, що зв'яже ступінь трансформації з імовірністю детекції тексту. Для цього необхідно ввести кілька нових понять:

- 1) Replacement rate (коефіцієнт заміни) - метрика, що вимірює ступінь зміни тексту і визначається як кількість змінених токенів (N_{mod}) до загальної кількості токенів (N_{total}):

$$RR = \frac{N_{mod}}{N_{total}} \quad (2.19)$$

Для більш складних випадків можна використовувати зважений Replacement rate (RR_w), що враховує інформаційну цінність зроблених замін (наприклад, заміна іменника має більшу вагу ніж прийменника):

$$RR_w = \frac{\sum_{i \in mod} w_i}{N_{total}} \quad (2.20)$$

де w_i - вага токена, що залежить від його частини мови, $w_i \in [0, 1]$

- 2) Breakdown Point (BP, Точка зламу) - таке значення метрики RR , при якому текст переходить зі стану “визначається детектором” ($D(T(X)) \geq Threshold$) до стану “не визначається” ($D(T(X)) < Threshold$). $Threshold$ - значення з проміжку $[0,1]$, яке визначається емпірично для кожного детектора для досягнення бажаного балансу між FPR і Recall. Для класичних задач бінарної класифікації, коли важливість False Positive і False Negative приблизно однакова, значення $Threshold$ приймають за 0.5.
- 3) Специфіка функціонування детектора. На базі введених раніше понять можна виділити три основні регіони роботи детектора:
- а) Область 1 - зона стійкої детекції, де значення RR після атаки залишається низьким (наприклад, $RR < 0.10$). Це трансформації на кшталт додавання незначних помилок, видалення чи додавання пунктуації з нечастою заміною слів і структури речень. У цій області $I_{sem} \approx 1$ і робастний детектор має зберігати високий Recall (імовірність детекції $P_{det} \approx 1$).

- b) Область 2 - зона успішного маскування, це діапазон значень RR , в якому I_{sem} залишається вище певного порогового значення I_{min} і текст продовжує зберігати прийнятну семантику, проте типовий детектор стає схильним до неправильного визначення природи тексту ($P_{det} < 0.50$). Хоча I_{sem} формально і не залежить від значення RR , проте в задачі маскування природа процесу трансформації гарантує, що вищі значення RR будуть знижувати I_{sem} , тобто ці два значення будуть обернено пропорційні. Область 2 описує зону успішних bypassing attacks.
- c) Область 3 - ця зона характеризується високими значеннями RR ($RR > 0.5$), при цьому ймовірність $I_{sem} < I_{min}$ стає надзвичайно високою. Попри те, що в цій зоні $P_{det} \ll 0.50$, практичної користі тексти, модифіковані таким чином, не мають. Прикладом такої трансформації може слугувати заміна всіх слів у тексті випадковими.

Концептуальною метою розробки гібридного методу детекції є зміщення точки зламу (BP) праворуч по осі RR , збільшуючи Область 1 і зменшуючи Область 2. На рис. 2.1 зображено, як могла б виглядати залежність P_{det} і I_{sem} від RR для певного байпасера B і детектору D . Оскільки точка зламу задає початок Області 2 для певного детектора D , тоді якщо для гібридного методу точка BP буде знаходитися правіше (зменшуючи Область 2), то гібридний метод буде більш робастним щодо байпасера B . У випадку ідеального детектора BP має знаходитися на лівій межі Області 3, або ще правіше.

Візуалізація моделі "Точки зламу" (Breakdown Point)

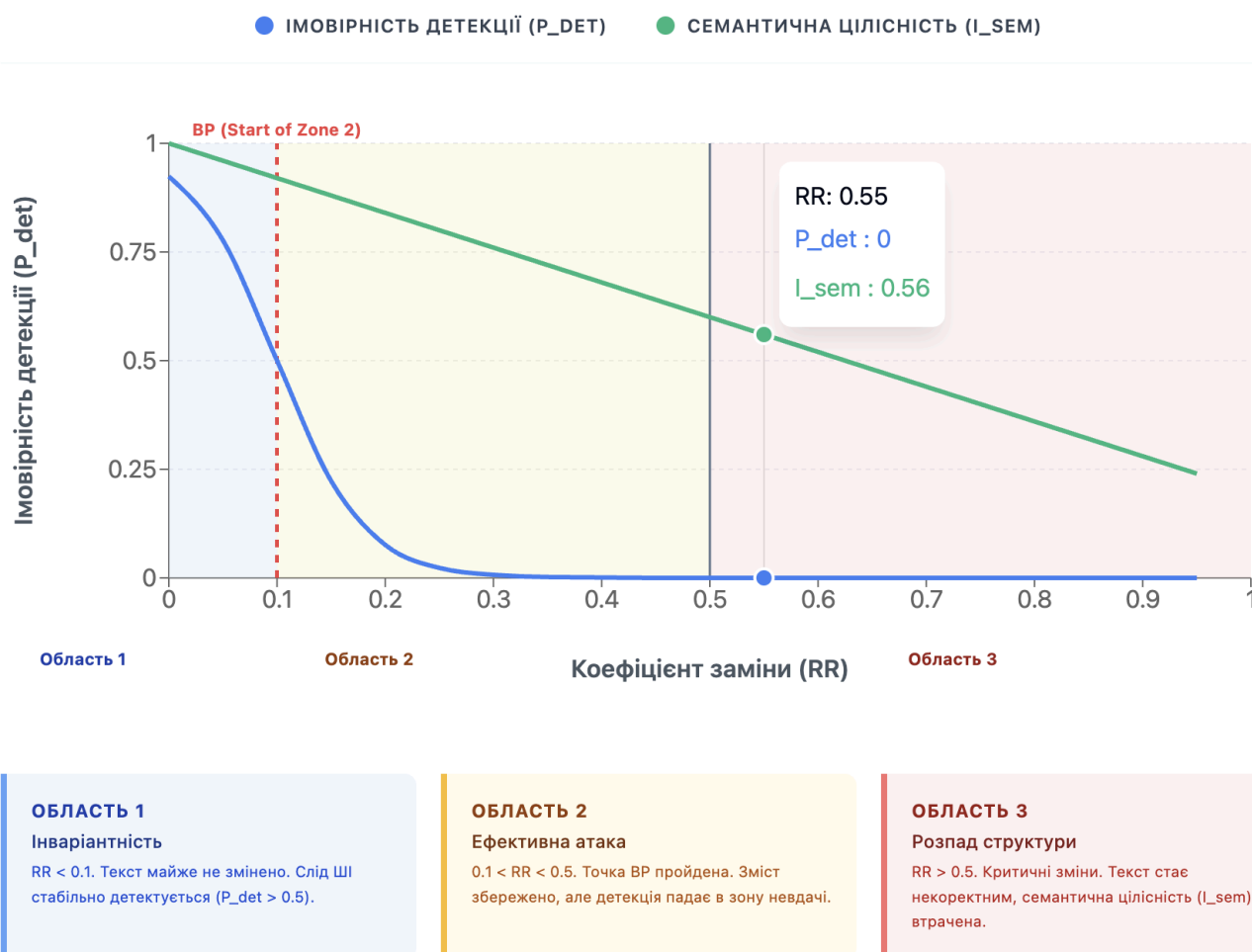


Рисунок 2.1 - Умовний графік, що демонструє залежність P_{det} і I_{sem} від RR

2.2.3. Метод оцінки робастності детектора в умовах невизначеності

Ще однією проблемою, яку потрібно вирішити при побудові AI Detector, є точність роботи на Out-of-Distribution (OOD) даних. Загалом поняття Out-of-Distribution є досить широким і в контексті машинного навчання вказує на дані, що не було представлено в навчальній вибірці. На практиці це може означати багато різних речей, наприклад, формат вхідних даних - коли детектор було натреновано на текстах, отриманих з PDF, а планується використовувати для аналізу як PDF, так і DOCX. Тоді, оскільки програмні засоби для отримання тексту з різних

форматів файлів можуть мати свою специфіку і генерувати різні викривлення тексту (наприклад, форматування таблиць чи списків) при читанні його з DOCX, порівняно з читанням з PDF, дані з DOCX будуть вважатися OOD. Більш типовими прикладами OOD даних для детектора будуть дані з інших доменів або дані, згенеровані новими LLM, що вийшли на ринок уже після тренування детектора. Зрозуміло, що єдиним надійним розв'язанням цієї проблеми є регулярне перетренування детектора з використанням даних від всіх SOTA LLMs.

З іншого боку, стабільність детектора на OOD даних є однією з важливих складових робастності, тож потрібно розробити певні підходи для підвищення стійкості. Основним методом слугуватиме квантифікація невизначеності (Uncertainty Quantification / UQ) [20] за допомогою конформного прогнозування (Conformal Forecasting / CF) [21]. Далі буде розглянуто ці два поняття детальніше.

Uncertainty Quantification дослівно перекладається як “вимірювання невизначеності”, у даному випадку невизначеність - це міра невпевненості в результатах моделі-детектора. Якщо в якомусь гіпотетичному сценарії детектор D видає результат, що текст T створено LLM з імовірністю 85%, то UQ дає змогу зрозуміти причини виникнення інших 15% та наскільки достовірними є 85%. Загалом квантифікація невизначеності дає змогу зрозуміти як різні типи вхідних даних впливають на результати моделі. У випадку низької впевненості детектора у результаті UQ дає змогу визначити її причини: шум у тексті (алеаторна невизначеність) чи OOD дані (епістемічна невизначеність). Алеаторна невизначеність - це невизначеність, пов'язана з самим предметом аналізу, наприклад, якщо текст, що аналізується, містить змішану природу (половина речень написана AI, а половина людиною). Цей тип невизначеності неможливо усунути шляхом покращення моделі, адже вона породжена природою даних що досліджуються. Епістемічна невизначеність породжується самою моделлю через відсутність необхідних знань, наприклад, дані, створені новою LLM, або текст на абсолютно нову для моделі тематику чи іншою мовою, ніж тренувальні дані. Цей тип

невизначеності пов'язаний з неідеальністю моделі і його можна усунути шляхом додавання нових даних до тренувального набору [22].

Для розв'язання проблеми епістемічної невизначеності залучено апарат теорії ймовірностей та математичної статистики, зокрема метод конформного передбачення. Conformal Forecasting (Конформне Прогнозування) - це фреймворк, що дає змогу як визначити тип невпевненості моделі, так і надати гарантовану достовірність (Validity) - статистичну гарантію того, що інтервал містить істинне значення з заздалегідь заданою імовірністю. Типовою проблемою нейронних мереж є overconfidence - коли останній шар NN (або вихід), на кшталт Softmax, видає занадто високі значення confidence, тобто ситуація, коли NN робить хибний прогноз з впевненістю в 99% або вище. У такій ситуації високий confidence score моделі зовсім не гарантує статистичну точність прогнозу, і застосування conformal forecasting стає виправданим для отримання validity. Тобто використання CF дає змогу перетворити прогноз моделі детектора зі стану “це AI-generated текст” у стан “це AI-generated текст, з гарантією 95%”. З іншого боку, можливість CF визначати тип невпевненості дає змогу моделі уникати невірних прогнозів на OOD даних, наприклад, повертаючи порожню множину результатів при аналізі даних нових LLMs.

Введемо формальні означення для всіх описаних вище понять. Передбачувану множину з CF, що містить істинне значення з імовірністю $1 - \epsilon$, будемо позначати $\Gamma^\epsilon(x)$ та обчислювати наступним чином:

$$\Gamma^\epsilon(x) = \{y: \alpha(x, y) \leq q\} \quad (2.21)$$

де $\Gamma^\epsilon(x)$ - передбачувана множина - набір міток з множини {Human, AI}, які система вважає правдоподібними при рівні помилки ϵ ,

y - кандидат на мітку (система перевіряє всі мітки на відповідність умові включення),

$\alpha(x, y)$ - бал невідповідності (nonconformity score) - функція, що

вимірює наскільки мітка y не відповідає тексту x на базі калібрувальних даних,

q - вказує на порогове значення функції α , яке показує максимально допустимий рівень невідповідності, при якому мітка y ще може вважатися допустимою

Використання $\alpha(x, y)$ дає змогу уникнути ситуації, коли детектор робить прогноз в умовах, коли $T_{AI}(x)$ (вектор ознак AI Trace для тексту x) лежить далеко від центроїдів обох класів (Human, AI). У цьому випадку множина $\Gamma^{\epsilon}(x)$ буде порожньою. У випадку алеаторної невизначеності $\Gamma^{\epsilon}(x) = \{Human, AI\}$ - тобто система не може визначити тип вхідних даних через шум у них (або модифікації). У випадку епістемічної невизначеності $\Gamma^{\epsilon}(x) = \emptyset$ - тобто система не може зрозуміти, які мітки можна застосувати до тексту x , оскільки ніколи раніше не зустрічала подібних даних (випадок OOD).

Загалом, використання CF як додаткової статистичної функції контролю якості роботи детектора дає змогу підвищити його надійність та прозорість у сценаріях аналізу OOD даних.

2.3. Розробка гібридного методу ідентифікації

У цьому підрозділі основний фокус спрямовано на розробку гібридного методу ідентифікації автоматично згенерованих текстів. Робота, виконана в попередніх частинах Розділу 2 з формалізації ознак, що мають увійти до складу AI Trace, та розробки концептуальної моделі стійкості детекції дає змогу перейти до безпосередньої побудови методу. Архітектура гібридного методу має враховувати слабкі місця і недоліки наявних методів детекції, які було описано в Розділі 1, та бути максимально захищеною від типових трансформаційних атак.

В основі запропонованого підходу лежить архітектура каскадного ансамблю, де різні типи ознак (лексичні, структурні та семантичні) розподілені за різними рівнями і виконують функцію кросвалідації рішення детектора. Тобто замість об'єднання всіх ознак, запропонованих у підрозділі 2.1, в один вектор, вони залишаються в трьох різних незалежних векторах, кожен з яких опрацьовується окремою моделлю-класифікатором. Ключовою новизною запропонованого методу є саме механізм взаємної валідації рішень, прийнятих на різних рівнях. Якщо, наприклад, лексичний рівень визначає текст як Human-written, але структурний рівень вказує на наявність чіткого ритму у тексті, то це буде явною ознакою застосування техніки маскування на лексичному рівні (наприклад, Burstiness Injection). Однією з основних переваг запропонованого методу є те, що проведення bypassing атаки на одному рівні підвищує силу прояву AI Trace на іншому.

2.3.1. Архітектура гібридного методу

Як відповідь на вимоги до нового покоління детекторів, описані в пункті 1.3.2, на основі системного підходу розроблено ієрархічну структуру незалежних класифікаторів, що працюють під керуванням метакласифікатора, який відповідає за прийняття остаточного рішення. На відміну від звичайних ансамблів класифікаторів, в яких фінальне рішення приймається шляхом голосування (наприклад, majority voting), у запропонованому методі фінальне рішення приймає окремий метакласифікатор. Основна роль цього метакласифікатора це виявлення та аналіз розбіжностей між різними рівнями та прийняття рішення про їх причини. Застосування метакласифікатора дає змогу змістити боротьбу з атаками з того рівня, де вони відбуваються (і де їх надзвичайно важко ідентифікувати), на той рівень, де доступна повна інформація про текст і де слід трансформаційних атак стає більш явним.

Основні елементи системи:

- Лексичний (статистично-стилістичний) модуль (M_{lex}). Він реалізує ознаки лексичного рівня F_{lex} , описані у пункті 2.1.1, такі як perplexity, burstiness, ентропію та інші. Також до складу модуля входить аналіз ознак МоЕ архітектур, використовуючи оцінку щільності ядра та ексцес, описані формулами (2.13) та (2.14). Модуль націлено на визначення базового відбитка сліду LLM, і він є надзвичайно чутливим до нього. Зворотною стороною такої чутливості є його вразливість до модифікацій тексту на кшталт додавання граматичних помилок, підвищення температури генерації, а також інших обмежень, характерних для цього класу методів і описаних у пункті 1.2.3.
- Синтаксичний (структурний) модуль (M_{syn}). Він реалізує ознаки синтаксичного рівня F_{syn} , описані у пункті 2.1.1, такі як дивергенція POS Tags N-grams, глибина та довжина речень і інші особливості їх конструкції. Також до складу модуля входить аналіз ознак Reasoning моделей, використовуючи спектральний аналіз ритму, описаний формулою (2.15). Цей рівень ігнорує зміст тексту і фокусується на його структурних характеристиках. Цей рівень виступає захистом від деяких типів перефразування та незначних модифікацій тексту (наприклад, заміна синонімів).
- Семантичний модуль - реалізує ознаки семантичного рівня F_{sem} , фокусується на особливостях змісту згенерованого тексту, його цілісності, логічності та зв'язності. Включає аналіз цілісності, зв'язності та логічної послідовності тексту. Одними з основних вимірюваних параметрів тут будуть середня узгодженість та дисперсія узгодженості.
- Механізм семантичної валідації (M_{sem}). Це ключовий елемент моделі, що відповідає за аналіз логічної узгодженості між рівнями, він базується на гіпотезі, що трансформаційні атаки можуть створювати дисонанс між різними рівнями тексту. Для визначення сили такого дисонансу буде обчислюватися сигнал розбіжності $S_{mismatch}$. Індивідуальні модулі {

$M_{lex}, M_{syn}, M_{sem}$ } становлять собою незалежні класифікатори, кожен з яких повертає імовірність ($M_i \in [0, 1]$) того, що текст було згенеровано

III. Для обчислення сигналу розбіжності $S_{mismatch}$ потрібно використати наступну формулу:

$$S_{mismatch} = w_{lexsem} |M_{lex} - M_{sem}| + w_{lexsyn} |M_{lex} - M_{syn}| \quad (2.22)$$

До формули (2.22) варто зробити кілька пояснень. Сигнал розбіжності обчислюється як сума абсолютних різниць передбачень різних модулів помножена на коефіцієнти важливості (w_{lexsem} та w_{lexsyn}). Може виникнути питання відсутності у формулі третього доданка: $|M_{syn} - M_{sem}|$. Річ у тому, що M_{lex} використовується як певний базовий сенсор, і за замовчуванням припускається, що він міг бути введений в оману. У деякому сенсі лексичний модуль відповідає за розпізнавання “зовнішнього вигляду тексту”. Отож очікувано, що більшість атак буде націлено саме на цей рівень і значення M_{lex} буде змінюватися досить суттєво.

З іншого боку, з дослідження проведеного в пункті 2.2.1 відомо, що збереження високого значення I_{sem} (формула 2.17) є необхідною умовою будь-якої успішної трансформаційної атаки (для уникнення потрапляння модифікованого тексту в Область 3). Тому очікувано, що успішні трансформаційні атаки будуть намагатися максимально зберігати семантику тексту і, як результат, зберігати значення M_{sem} . Тобто під час атак M_{sem} виступає в ролі майже нерухомої точки, а M_{lex} - сильно рухомої, що виправдовує наявність доданка $|M_{lex} - M_{sem}|$.

Цінність метрики M_{syn} полягає в тому, що вона дає змогу проаналізувати текст за іншими параметрами порівняно з M_{sem} . Стабільність цієї характеристики під час атак визначається кількома факторами:

- 1) Складність зміни синтаксичних шаблонів, порівняно з лексичними патернами. З одного боку, у людей використання синтаксичних шаблонів

та вибір функціональних слів є значною мірою автоматичним процесом, що погано формалізується, а з іншого боку, деякі архітектури LLM несуть свій специфічний структурний відбиток, що важко прибрати.

- 2) Складність значної зміни синтаксису тексту без значної зміни семантики.
- 3) Крихкість синтаксису. Глибоке структурне перетворення тексту є досить дорогим процесом з погляду ризику руйнування дискурсу і граматики (внаслідок чого модифікований текст може потрапити в Область 3).

Тому очікувано, що M_{syn} теж буде демонструвати досить стабільну поведінку під час атак, хоча можливі більші відхилення, порівняно з M_{sem} . Цим пояснюється наявність доданка $|M_{lex} - M_{syn}|$.

Доданок $|M_{syn} - M_{sem}|$ відсутній через очікування, що визначення трансформаційних атак буде покладено саме на M_{lex} , а різниця $M_{syn} - M_{sem}$ буде залишатися відносно стабільною через природу цих параметрів.

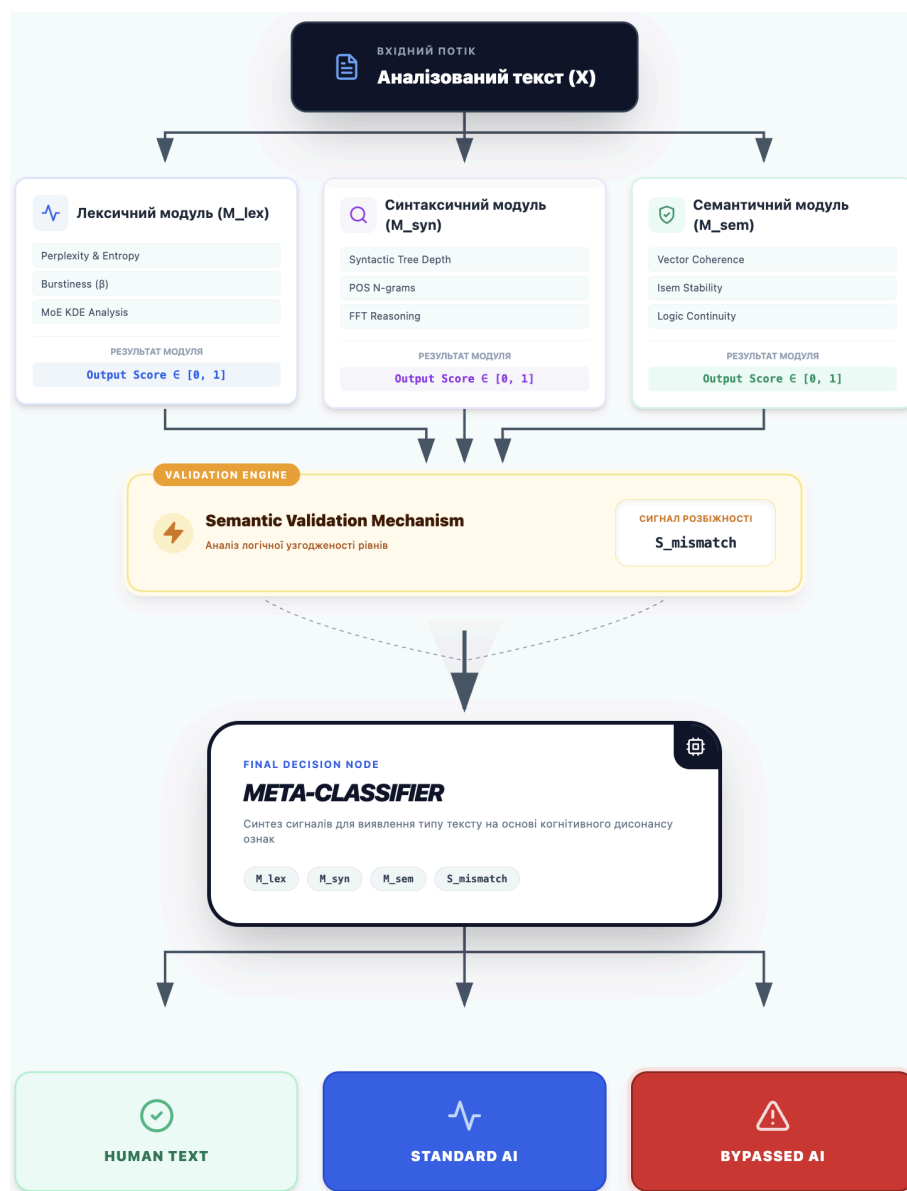


Рисунок 2.2 Архітектура гібридного методу

Можливі сценарії:

- 1) Текст написано людиною. $S_{mismatch} \rightarrow 0$, $M_i \rightarrow 0$ - всі модулі згодні, що мітка має бути Human, а $S_{mismatch}$ має низьке значення, вказуючи на відсутність атаки.
- 2) Текст написано LLM. $S_{mismatch} \rightarrow 0$, $M_i \rightarrow 1$ - всі модулі згодні, що мітка має бути AI, а $S_{mismatch}$ має низьке значення, вказуючи на відсутність атаки.

- 3) Текст написано LLM та оброблено Bypasser. $S_{mismatch} \gg 0$, $M_{lex} \rightarrow 0$, $M_{syn} \rightarrow 1$, $M_{sem} \rightarrow 1$ - лексичний модуль вважає текст написаним людиною, а два інші модулі вважають його AI, відповідно, $S_{mismatch}$ має високе значення, вказуючи на наявність трансформаційної атаки.

Метакласифікатор (для кращої інтерпретованості ваг будемо використовувати логістичну регресію) отримує на вхід результати роботи всіх трьох модулів та значення $S_{mismatch}$; високі значення $S_{mismatch}$ дають змогу класифікатору позначати текст як “bypassed AI”. Архітектуру гібридного методу зображено на рис. 2.2.

2.3.2. Протидія структурним атакам

Структурні атаки - це атаки на базову структуру тексту та її складові елементи - окремі символи, пунктуацію, пропуски, елементи форматування і таке інше. Роботи [23] та [24] звертають увагу на все більшу загрозу атак на основі гомогліфів, а стаття [25] наголошує на необхідності протидії атакам з використанням спеціальних символів. Самі ці атаки не є чимось новим і використовувалися ще 10 років тому для уникнення виявлення плагіату. Їх основна сила в простоті, але при цьому вони залишаються надзвичайно потужними. Тобто роботи [23-25] говорять про успішність використання старого інструменту проти нової технології - виявлення сліду AI.

Основна ідея структурної атаки проста - необхідно внести в текст на символьному рівні таку кількість змін, щоб з погляду машини він став зовсім іншим. При цьому необхідно, щоб з погляду людини текст залишався все тим же самим. Досягається це кількома шляхами: заміною одних символів на інші, що виглядають дуже схоже (наприклад, латинської ‘a’ на кириличну ‘а’), вставкою додаткових символів, що не мають візуального відображення (наприклад, zero-width space), або заміною пунктуації на схожі символи (наприклад, математичні). Такі заміни

кардинально порушують процес токенизації, тому інформація, що подається на вхід моделі-детектору, суттєво відрізняється від того, що і як написано в тексті.

З погляду математичної моделі робастності, описаної у пункті 2.2.1, структурна атака є одним з типів трансформації T , кінцевою метою якої є L_{trace} (2.18). Для зменшення впливу цього типу атак на L_{trace} потрібно попередньо обробити текст і привести до деякої канонічної форми. Для цього буде використано алгоритм канонізації.

Алгоритм канонізації:

Вхід - необроблений текст T_{raw}

Вихід - нормалізований текст $T_{canonical}$ та мітка атаки F_{attack}

- 1) **Видалення невидимих символів.** Проходить пошук і видалення символів, що не мають візуального відображення, на кшталт символів категорії Unicode Format (161 символ). Якщо кількість виявлених спеціальних символів перевищує пороговий відсоток від довжини тексту, то F_{attack} встановлюється в значення True.
- 2) **Нормалізація гомогліфів.** Для отримання канонічної форми кожного символу буде застосовуватися форма нормалізації NFKC (Normalization Form Compatibility Composition) [26]. Для захисту від атак, описаних у [24], коли одне слово може містити як кириличні, так і латинські символи, визначається основний алфавіт речення і всі символи в мішаних словах приводяться до домінантного алфавіту речення. При перевищенні кількістю мішаних слів порогового відсотка F_{attack} встановлюється в значення True.
- 3) **Нормалізація пробілів.** Всі типи табуляції та пробілів (категорія Space Separator) замінюються на один символ пробілу (U+0020). Також проводиться виявлення і заміна послідовностей пробілів на один символ U+0020.

Додатково виділимо два типи інваріантних (незмінюваних) ознак, що демонструють високу стійкість до перетворення перефразування і допомагають зрозуміти оригінальну природу тексту:

- 1) **Розподіл службових слів.** Використання службових слів (у NLP їх, як правило, називають стоп-словами) є значною мірою підсвідомим процесом. Людині важко контролювати частоту та шаблони використання артиклів, прийменників і сполучників. З іншого боку, LLM може намагатися оптимізувати цей процес і схилитися до сталих патернів, тоді як людина буде мати більш хаотичний шаблон використання. χ^2 відстань буде вимірюватися на основі статистики критерію згоди Пірсона [27] між розподілом стоп-слів документа та еталонним людським профілем:

$$D_{\chi^2}(X, Reference) = \sum_{i=1}^k \frac{(O_i - E_i)^2}{E_i} \quad (2.23)$$

де E_i - очікувана частота i -го слова в еталонному корпусі текстів, написаних людиною,

O_i - реальна частота i -го слова в тексті X ,

k - кількість слів в обраному списку службових слів

У випадку атаки (типу підміни символів або вставки додаткових пробілів) у формулі (2.23) починає рости чисельник формули, через появу нових невідомих раніше токенів і відсутність очікуваних токенів.

Значення D_{χ^2} може сигналізувати про кілька речей: низьке значення свідчить про дотримання середніх характеристик з людського корпусу - сигналізує про AI, середні значення - можуть говорити про текст, написаний людиною, аномально високі значення говорять про наявність структурної атаки (F_{attack} встановлюється в значення True).

2) **Глибина синтаксичної залежності.** Це значення розраховується як середня відстань між головним словом (коренем дерева) та залежними словами. Припускається, що базові атаки (включаючи деякі види перефразування) не зможуть суттєво змінити складність синтаксичних дерев. По суті, це значення μ_d з формули (2.8). Нехай текст X складається з K речень, а речення k складається з N_k tokenів $\{w_1, w_2, \dots, w_{N_k}\}$, тоді середня глибина синтаксичних дерев тексту μ_d обчислюється так:

$$\mu_d = \frac{1}{K} \sum_{k=1}^K \left(\frac{1}{N_k} \sum_{i=1}^{N_k} d(w_i) \right) \quad (2.24)$$

А глибина d для токена w_i обчислюється наступним чином:

$$d(w_i) = \begin{cases} 0, \text{ якщо } w_i - \text{корінь} \\ d(head(w_i)) + 1, \text{ інакше} \end{cases} \quad (2.25)$$

де $d(head(w_i))$ - батьківський вузол для w_i

2.3.3. Алгоритм прийняття фінального рішення

На останньому етапі система має прийняти остаточне рішення про природу тексту. На вхід системи подається вектор ознак $V(X) = \{M_{lex}, M_{syn}, M_{sem}, S_{mismatch}, Score_{freq}, F_{attack}\}$, на базі якого модель

метакласифікатор має прийняти фінальне рішення (S_{final}). Вектор ознак складається з двох груп ознак: прогностичні ($M_{lex}, M_{syn}, M_{sem}$) і керівні ($S_{mismatch}, Score_{freq}, F_{attack}$). Важливо зазначити, що елементи вектора V мають різні діапазони:

- $M_{lex}, M_{syn}, M_{sem}$ - це виходи моделей-класифікаторів, тобто імовірність того, що текст створено AI, відповідно, ці значення належать діапазону $[0,1]$;
- $S_{mismatch} \in [0, 2]$;
- $F_{attack} \in \{0, 1\}$;
- для $Score_{freq}$ - вихід класифікатора Reasoning моделей, а саме імовірність, тому значення належить діапазону $[0,1]$.

Тому перед використанням вектора V для обчислення S_{final} потрібно всі його компоненти привести до фіксованого інтервалу $[0,1]$. Зробити це можна за допомогою техніки Min-Max Scaling:

$$\hat{V}_j(X) = \frac{V_j(X) - \min(V_j)}{\max(V_j) - \min(V_j)} \quad (2.26)$$

де $\max(V_j)$ і $\min(V_j)$ - параметри розподілу, визначені на тренувальній вибірці

$$S_{final} = \sigma\left(\sum_{j=1}^N w_j V_j(X) + b\right) \quad (2.27)$$

де S_{final} - імовірність того, що текст було згенеровано AI; $S_{final} \in [0, 1]$, де 1 - максимальна впевненість у штучній природі тексту,

σ - сигмоїд (логістична функція активації),

j - індекс елемента у векторі V ,

$V_j(X)$ - значення нормалізованої j -ї ознаки,

w_j - адаптивний ваговий коефіцієнт j -ї ознаки,

b - вільний член

У прийнятті фінального рішення до уваги беруться не лише коефіцієнти, вивчені метакласифікатором, але і набір правил, які описано в роботі вище. Додаткове керування системою за допомогою правил є необхідним елементом прийняття фінального рішення, оскільки модель не має знань про суть параметрів та їх взаємозв'язок. Наприклад, відомо, що за $F_{attack} = 1$ значення M_{lex} потрібно повністю ігнорувати. Можна припустити, що велика нейронна мережа, навчена на дуже великій кількості прикладів, була б здатна вивести такі залежності, проте гарантії немає.

Для забезпечення робастності запропонованого методу детекції буде використано спрощену версію механізму адаптивного зважування (Adaptive Gating Mechanism), запропонованого в [28]. Суть механізму полягає у дворівневій системі ваг: вага w_j дорівнює добутку вивченої класифікатором ваги w_j^{stat} (демонструє важливість модуля в нормальних умовах) та коефіцієнта довіри $\alpha_j(X)$ (визначається на базі керівних сигналів):

$$w_j(X) = w_j^{stat} \cdot \alpha_j(X) \quad (2.28)$$

Алгоритм розрахунку коефіцієнта довіри:

- 1) У базових умовах $\alpha_{lex} = \alpha_{syn} = \alpha_{sem} = 1$, працюють вивчені ваги.
- 2) У випадку структурної атаки ($F_{attack}=1$) результати лексичного аналізу стають некоректними, тож система вимикає цей блок ($\alpha_{lex}=0$, $\alpha_{syn} = 1$, $\alpha_{sem} = 1$).
- 3) У випадку виявлення перефразування ($S_{mismatch} >$ порогового значення) знижуємо довіру до базових статистик і підвищуємо до семантичного модуля ($\alpha_{lex} = 0.5$, $\alpha_{syn} = 1$, $\alpha_{sem} = 1.5$).

- 4) У випадку виявлення Reasoning Trace ($Score_{freq} > \text{Порогового значення}$) підвищуємо вагу структурних ознак, щоб зосередитися на ритмі, та знижуємо вагу базових статистик ($\alpha_{lex} = 0.5$, $\alpha_{syn} = 2$, $\alpha_{sem} = 1$).

Використання коефіцієнта довіри дає змогу провести інтеграцію експертних знань про атаки в архітектуру моделі. Для елементів вектора $V(X)$, що описують керівні сигнали, $\alpha_j = 1$; отже, ці елементи одночасно слугують і гейтами, і ознаками.

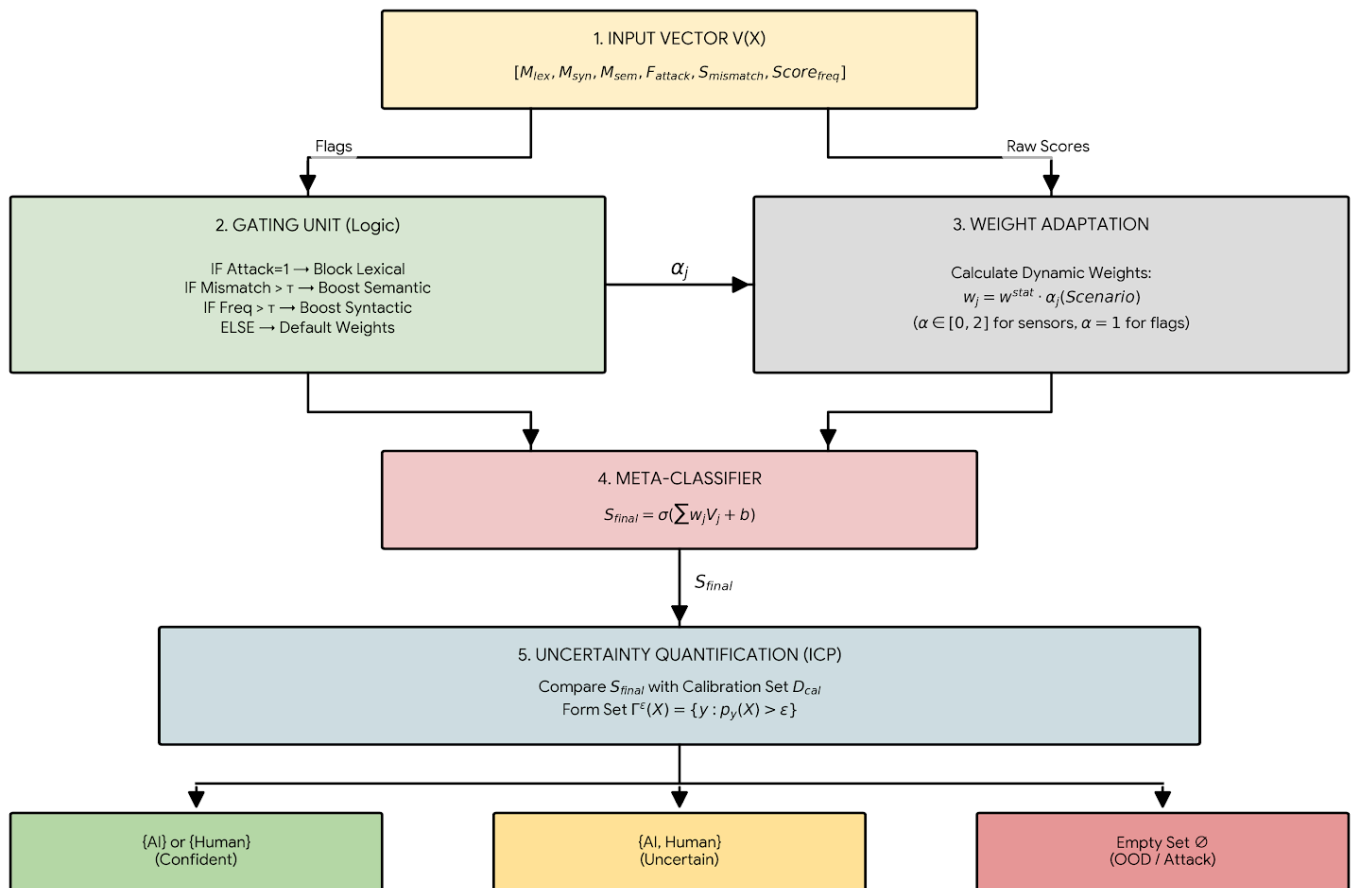


Рисунок 2.3 Схема блоку прийняття рішення

Фінальним кроком в отриманні оцінки є квантифікація невизначеності (UQ). Як описано в пункті 2.2.3, однією з серйозних проблем для детектора є питання його роботи в OOD сценаріях. Можливим теоретичним рішенням у цьому випадку є

Conformal Forecasting для оцінки рівня невизначеності в нестационарних середовищах, проте пряма реалізація даного методу для розв'язуваної задачі є обчислювально надмірною, тому для практичної реалізації буде використовуватися полегшений варіант CF - Індуктивне Конформне Передбачення (Inductive Conformal Prediction / ICP) [29].

Використання ICP дає змогу перейти від точкової імовірності (числа) S_{final} до множини класів $\Gamma^\epsilon(x)$ з гарантованою достовірністю:

- Якщо $\Gamma^\epsilon(x) = \{Human\}$ або $\Gamma^\epsilon(x) = \{AI\}$, то цей результат можна трактувати як високу впевненість системи.
- Якщо $\Gamma^\epsilon(x) = \{Human, AI\}$, то щось не так з текстом, наприклад змішана природа, можна повернути мітку, але з позначкою “низька впевненість”.
- Якщо $\Gamma^\epsilon(x) = \emptyset$, то це сигналізує про OOD (небачений раніше тип даних), може вказувати на новий тип трансформаційної атаки чи аномалію. Повертаємо позначку “Аномалія”.

Загальний вигляд схеми прийняття фінального рішення зображено на рис. 2.3.

2.4. Інтерпретаційна модель обґрунтування результатів детекції

Закрита природа (Black-Box) AI Detectors створює значні проблеми у розумінні причин їхнього рішення, що описано в пункті 1.2.3. Відсутність розуміння внутрішніх механізмів прийняття рішення детектором, недоступність додаткових доказів та можливостей з інтерпретації породжують значну недовіру [30], [31]. Крім того, оскільки заяви про надійність детекції можуть бути підкріплені лише статистичними викладками (що часто є недостатнім для більшості користувачів), це створює підґрунтя до дискусії, що методи детекції AI не працюють взагалі. Багато користувачів, які хоч раз стикалися з ситуацією False Positive чи False Negative

прогнозу детектора, не схильні сприймати аргументи про високий (проте не абсолютний) Recall системи та низький (проте не нульовий) FPR. Оскільки людина експерт не є надійним джерелом ground truth (остаточної істини) у задачі AI Detection, то єдиним методом підвищення довіри до моделей є створення інструментарію для прозорого пояснення логіки прийняття рішення детектором. У цьому підрозділі розв'язано цю задачу.

2.4.1. Розробка моделі профілю атрибуції. Локальні атрибути

У пунктах 1.2.2-1.2.3 розглянуті основні методи XAI на кшталт SHAP та LIME, описані їх специфіка та обмеження. Серед основних недоліків можна виділити часту нестабільність пояснень для текстових даних, великий бюджет та спроби “вгадати”, що саме важливо для black-box [32]. Для випадку, описаного в дисертаційному дослідженні, маючи гібридний метод ідентифікації AI Trase, можна використати внутрішні параметри моделі (Intrinsic Feature Attribution) для пояснення реальних причин винесення детектором рішення, замість створення сотень і тисяч симуляцій для отримання наближеної імітації системи. Такий підхід виграє як з погляду часу роботи та вартості, так і з погляду точності. Проте варто зауважити, що найточніше він буде працювати саме для запропонованого у роботі гібридного методу, для інших детекторів він буде здатен надати лише приблизне пояснення через відсутність доступу до внутрішніх ваг моделі.

З погляду пояснень (атрибутів) їх можна розділити на два типи:

- 1) Глобальні. Це пояснення, які базуються на способі роботи гібридного методу загалом. Вони можуть використовувати як інформацію про елементи архітектури системи, так і вивчені ваги w_j^{stat} . Наприклад, якщо в базовому сценарії $\alpha_{lex} = \alpha_{syn} = \alpha_{sem} = 1$, то ваги w_j^{stat} напряму вказують на силу впливу на остаточне рішення, тож, якщо, наприклад, $|w_{lex}^{stat}| > |w_{syn}^{stat}|$, то це свідчить, що лексичні ознаки відіграють загалом

більшу роль у детекції, ніж синтаксичні. Глобальні пояснення не залежать від конкретного тексту, що є предметом аналізу, вони залежать лише від архітектури системи, її Gating functions та вивчених ваг. У список глобальних атрибутів не входять статистики, зібрані ІСР, тому що, по-перше, ІСР не має стосунку до моделі-класифікатора, а по-друге, він не допомагає зрозуміти причини прийняття рішення, а допомагає вирахувати достовірність рішення.

- 2) Локальні. Обчислюються окремо для кожного документа та пояснюють основні причини надання йому тієї чи іншої мітки. У спрощеному вигляді це щось на кшталт “У даному тексті підозрілий синтаксис, тому система вирішила що це AI”. Для отримання сили впливу кожного параметру ($Contribution_j$) потрібно використати елементи формули (2.27), отже $Contribution_j$ обчислюється за формулою:

$$Contribution_j(X) = w_j \cdot V_j(X) \quad (2.29)$$

де j - індекс елемента у векторі V ,

$V_j(X)$ - нормалізоване значення j -ї ознаки,

w_j - адаптивний ваговий коефіцієнт j -ї ознаки

Інтерпретація значень $Contribution_j$:

- $Contribution_j(X) \gg 0$ - j -та ознака активно голосує за мітку AI,
- $Contribution_j(X) \ll 0$ (або близьке до 0 за великого від'ємного значення параметра b з формули (2.27)) - j -та ознака активно голосує за мітку Human,
- $Contribution_j(X) \approx 0$ - j -та ознака нейтральна, або вимкнута керівною функцією

На наступному кроці ранжується список значень $Contribution$ за спаданням і формуємо пояснення на його основі, так, що ознаки з більшими вагами відіграють

більшу роль, а з меншими - меншу. Якщо вага ознаки близька до 0, то вона не вказується в поясненні. Як демонстрацію доцільно розглянути рис. 2.4, де показано роботу адаптивного зважування в умовах атаки.

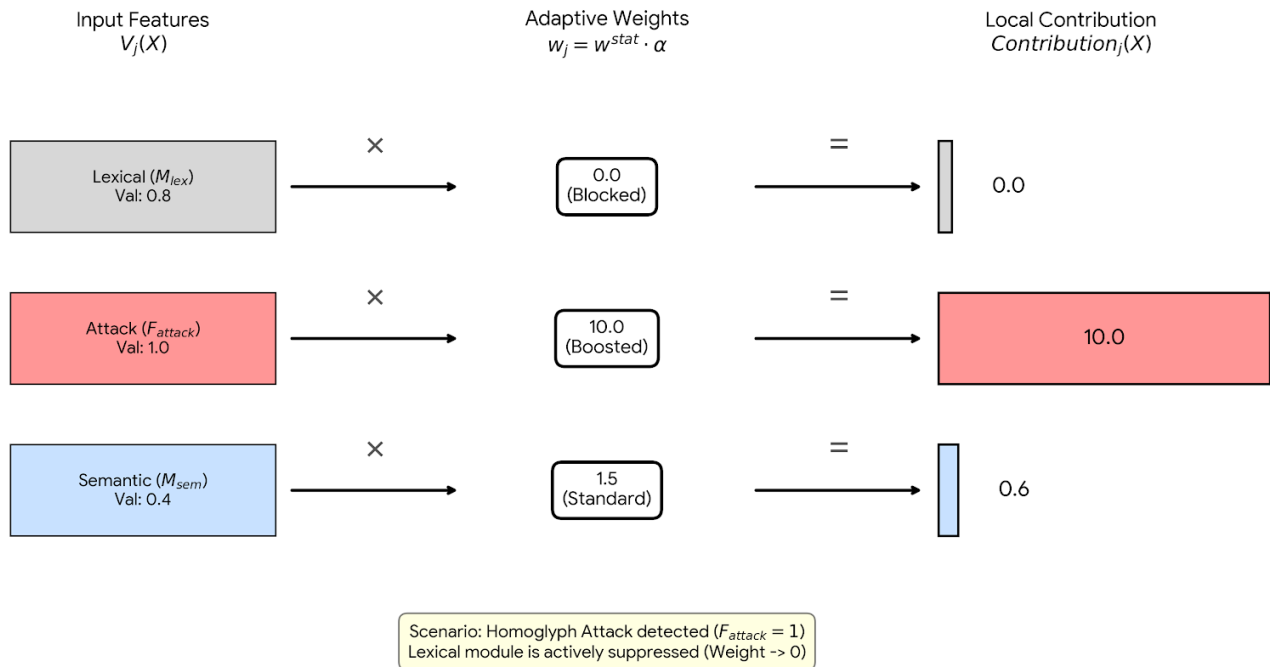


Рисунок 2.4 Приклад роботи Adaptive Gating під час атаки

Важливим моментом є те, що для спрощення розуміння візуалізацій користувачами, в інтерпретаційній моделі варто перейти від прямих значень $Contribution_j$, що демонструють реальний внесок ознаки в кінцеве рішення, до центрованих значень $Contribution_j^{centered}$. Перехід до центрованих значень є типовою практикою в ХАІ та використовується для того, щоб чітко показати користувачу, які ознаки зсувають рішення класифікатора до одного класу, а які до іншого. У цьому випадку $Contribution_j \geq 0$, оскільки $V_j \in [0, 1]$ та $w_j \geq 0$, тож, щоб уникнути складних пояснень про decision threshold моделі-класифікатора на рівні 0.5 і невірних інтерпретацій значення $Contribution_j$, для візуалізації зручніше використовувати $Contribution_j^{centered}$:

$$\text{Contribution}_j^{\text{centered}}(X) = w_j \cdot (V_j(X) - 0.5) \quad (2.30)$$

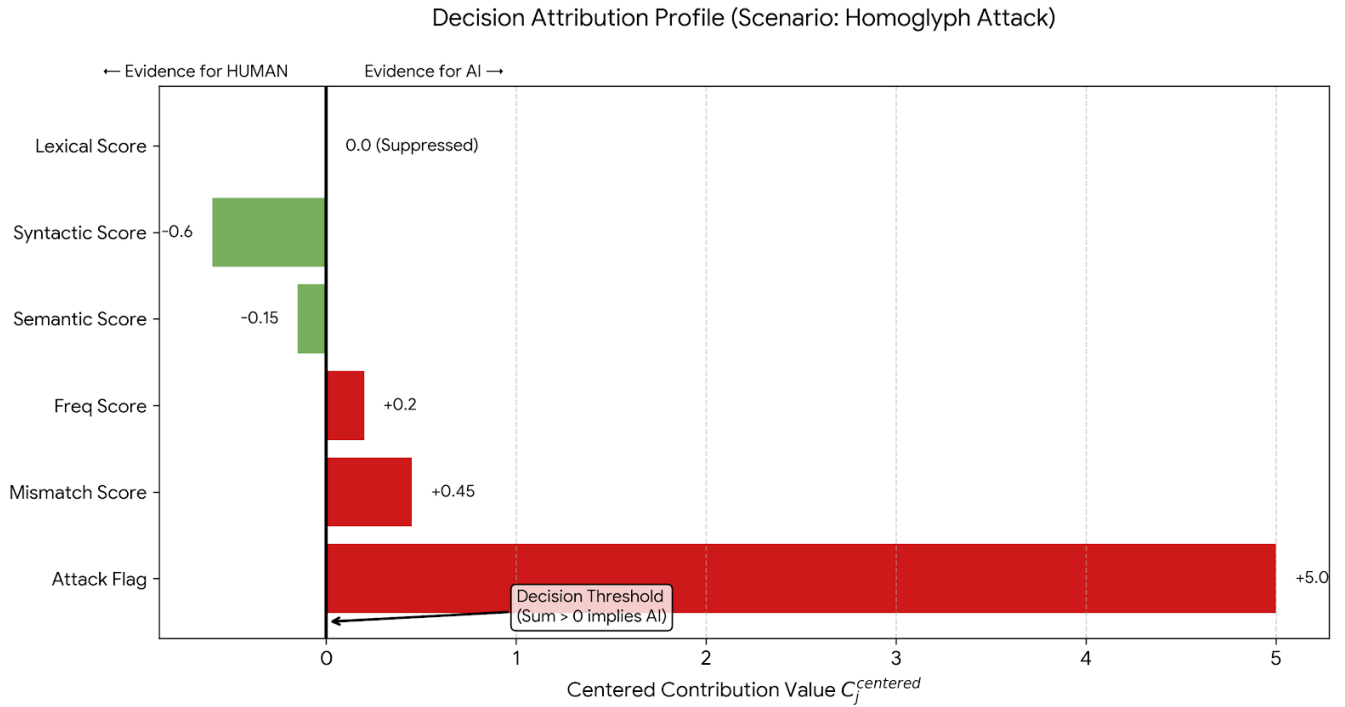


Рисунок 2.5 Приклад візуалізації Attribution Profile для сценарію гомогліфної атаки

Фінальним результатом роботи інтерпретаційної моделі буде список обґрунтованих доказів, що разом називається Attribution Profile (Профіль атрибуції). На рис. 2.5 зображено приклад візуалізації Attribution Profile для сценарію гомогліфної атаки, який проаналізовано на рис. 2.4.

2.4.2. Глобальні атрибути

Якщо локальні атрибути дають розуміння деталей аналізу конкретного тексту, то глобальні атрибути дають розуміння принципів роботи системи загалом. У попередньому пункті наведено приклад з $|w_{lex}^{stat}| > |w_{syn}^{stat}|$ і зафіксованими значеннями α . У реальності необхідно взяти до уваги не лише значення вивчених ваг w_j^{stat} , але і поведінку gating function α . Математично це зробити не вдасться, оскільки вихід функції керування залежить від конкретного вхідного тексту. Тому,

щоб отримати репрезентативну картину, необхідно провести збір статистики на базі валідаційного набору даних.

Для відповіді на питання які ознаки є найважливішими для моделі загалом, потрібно порахувати метрику *GlobalImportance* для кожної ознаки. Можна було б піти простішим шляхом і обчислити середнє значення $Contribution_j^{centered}$ на валідаційному корпусі, проте таке усереднення не є коректним через можливу наявність як додатних, так і від'ємних значень параметра.

$$GlobalImportance_j = \frac{1}{N} \sum_{i=1}^N \left| Contribution_j^{centered}(X_i) \right| \quad (2.31)$$

де N - кількість текстів у валідаційному корпусі,

i - індекс тексту в валідаційному корпусі,

j - індекс елемента у векторі V

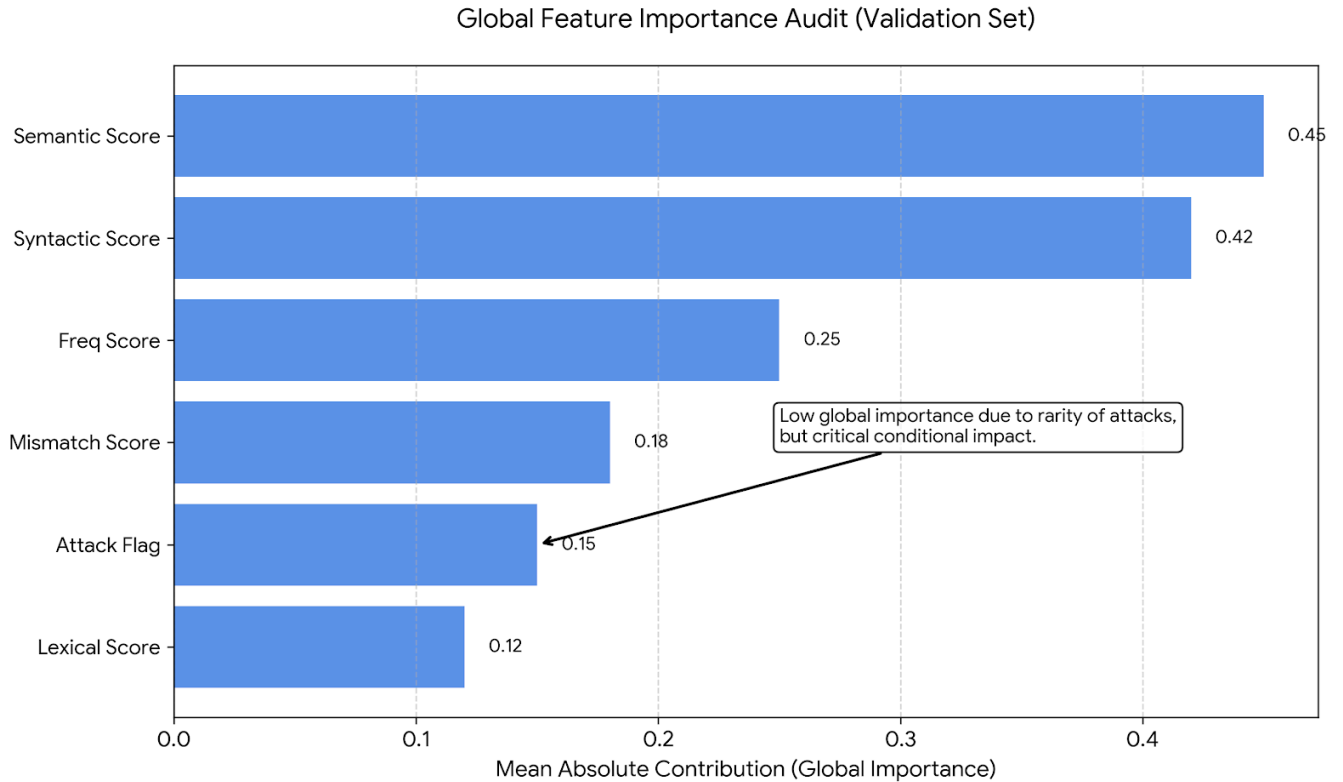


Рисунок 2.6 Приклад візуалізації важливості глобальних атрибутів

Щодо валідаційного корпусу, то важливою властивістю цієї вибірки має бути її репрезентативність, тобто вона має містити різні типи текстів: Human, AI, Bypassed. Тексти мають покривати різні домени. Цей набір даних демонструє очікуваний розподіл даних у реальних умовах використання системи. Зрозуміло, що зміна розподілу валідаційних даних впливатиме на *GlobalImportance*.

Аналіз розподілу важливості глобальних ознак дає змогу зрозуміти особливості роботи запропонованої архітектури детектора та принципи прийняття рішень, чим забезпечує прозорість роботи системи та забезпечує експертам можливість валідувати її логіку. Приклад такого розподілу зображено на рис. 2.6.

2.4.3. Карта доказів

Попередні два пункти дають змогу зрозуміти логіку роботи детектора загалом (п. 2.4.2) та специфіку аналізу конкретного тексту (п. 2.4.2), яка показує, які саме компоненти моделі зробили найбільший внесок у фінальне рішення. Локальні атрибути дають змогу зрозуміти на верхньому рівні, що саме з текстом “не так”: чи там проблема з синтаксисом, що виглядає занадто шаблонним, чи на текст було здійснено трансформаційну атаку, чи щось інше. Таке розуміння може допомогти експерту верифікувати роботу моделі та знати, на що звернути увагу в тексті перед прийняттям остаточного рішення, проте воно не надає конкретних доказів того, що текст було згенеровано (чи, навпаки, не було згенеровано). Побудова Карти доказів є останнім третім елементом інтерпретаційної моделі, що має забезпечити повну прозорість результатів роботи моделі та вказати на конкретні елементи тексту, що виглядають для моделі підозріло.

Варто вказати, що Карта доказів має свої переваги та недоліки. Серед основних переваг варто виділити такі:

- Прозорість щодо причин прийняття рішення
- Верифікованість рішень моделі й підвищення довіри до неї
- Спрощення моніторингу роботи моделі

- Спрощення донавчання моделі в разі низької точності в певних доменах, появи нових LLM або нових типів атак

Серед ключових загроз потрібно виділити дві:

- Якщо Карта доказів для певного тексту стане доступною зловмисному користувачу, це дасть йому змогу модифікувати текст таким чином, щоб уникнути детекції (або навпаки, зробити, щоб Human-written текст отримав мітку AI).
- Якщо інформація про локальні та глобальні атрибути системи стане доступною розробникам bypassers, це значно спростить їм процес розробки evasive techniques.

З урахуванням всього вищесказаного, запропоновано чотири основних рівні Карти доказів:

- 1) Структурні загрози. Цей рівень будує візуалізацію на базі результатів роботи алгоритму канонізації, описаного у пункті 2.3.2. Всі зміни, зроблені в тексті на етапі нормалізації, виділяються червоним контуром. Це слугує основним доказом при встановленні маркеру $F_{attack} = True$.
- 2) Лексична ентропія. Лексичний модуль M_{lex} містить велику кількість ознак, таких як perplexity, burstiness та інші, проте візуалізація їх усіх перевантажить Карту доказів, тому рекомендується сфокусуватися на відхиленнях у спостережуваному тексті від прогнозу проксі-моделі (perplexity). Якщо токен, що є у вхідному тексті, потрапляє до top-100 проксі-моделі, то він позначається зеленим кольором (дуже ймовірний токен з т.з. LLM), якщо потрапляє до top-1000 - жовтим (менш ймовірний), інакше червоним (неочікуваний токен, може сигналізувати про автора-людину).
- 3) Семантичні розриви. На базі інформації з M_{sem} можна визначити та підсвітити місця, де семантична схожість двох сусідніх речень падає

нижче певного порогового значення. Це може бути місцями втрати контексту, або повторень чи інших порушень структури тексту.

- 4) Профіль ритму. У випадку високих значень $Score_{freq}$ додатково до візуалізацій, що накладаються поверх тексту, необхідно показувати додатковий графік як підтвердження CoT природи тексту. На цьому графіку може бути показано абсолютне значення $|F(\omega)|$ (2.15), він відображає залежність амплітуди (вісь OY) від частоти (вісь OX). Якщо на графіку присутні гострі піки, це свідчить про домінацію певних частот, тобто наявність ритму.

Приклад візуалізації Карти доказів наведено на рис. 2.7.

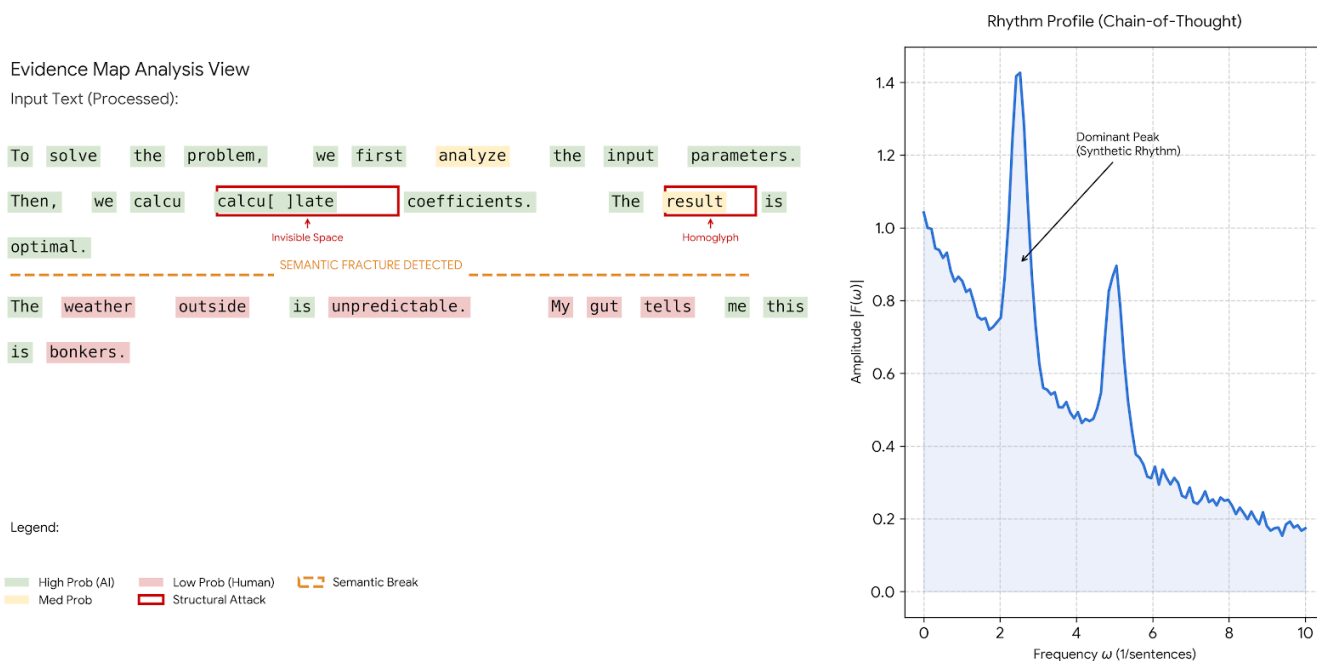


Рисунок 2.7 Приклад Карти доказів

2.5. Висновки до розділу 2

Дослідження щодо стійкості детекції та особливостей AI Trase, проведене нами у роботах [2] та [11], стало базою для матеріалу викладеного вище. У цьому розділі введено математичну та алгоритмічну базу необхідну для досягнення мети

дослідження - підвищення точності та надійності ідентифікації автоматично згенерованих природномовних текстів. Також у цьому розділі вирішено завдання дослідження 2–5 та описано такі наукові результати:

1. Удосконалено концептуальну модель оцінки робастності систем детекції сліду ШІ, яка завдяки введенню формалізованого поняття Точки зламу та інтеграції математичної моделі багаторівневого простору ознак (лексичних F_{lex} , синтаксичних F_{syn} , семантичних F_{sem}) дає змогу перейти від наявних емпіричних підходів оцінки робастності на фіксованих датасетах до встановлення прямого математичного взаємозв'язку між інтенсивністю структурних трансформацій тексту (коефіцієнт заміни RR), збереженням його семантичної цілісності (I_{sem}) та ймовірністю виявлення (P_{det}). Модель виділяє три області, що демонструють ефективність атаки, та пропонує метод оцінки робастності детектора в умовах невизначеності. Це дає змогу використовувати її як прогностичний інструмент для визначення критичних меж маскування та оцінки чутливості системи без необхідності проведення витратного прямого тестування, що становить зміст Наукової новизни 4 та відповідає виконанню Завдання 2 в частині концептуального моделювання.
2. Уперше розроблено гібридний метод ідентифікації автоматично згенерованих природномовних текстів, який, на відміну від класичних детекторів, що сприймають слід ШІ як статичний і легко змінюваний об'єкт, завдяки використанню каскадного ансамблю незалежних класифікаторів із додатковими механізмами валідації дає змогу проводити аналіз міжрівневої узгодженості ознак через адаптивне зважування та розрахунок сигналу розбіжності $S_{mismatch}$. Такий підхід фіксує когнітивний дисонанс між лексичним, синтаксичним та семантичним рівнями тексту під час застосування трансформаційних атак, роблячи спроби маскування на одному рівні тригером для посилення детекції на іншому. Це зміщує Точку зламу детектора праворуч, уможливорює виявлення структурних аномалій і

забезпечує збереження високої ймовірності ідентифікації ІІІ, що становить зміст Наукової новизни 1 та відповідає виконанню Завдання 3.

3. Набули подальшого розвитку методи забезпечення робастності систем обробки природної мови в умовах епістемічної невизначеності та даних поза навчальним розподілом (OOD). На відміну від традиційних систем чорної скриньки, що демонструють надмірну впевненість на невідомих даних, запропонований підхід завдяки інтеграції на етапі прийняття фінального рішення фреймворку індуктивного конформного передбачення (ICP) дає змогу здійснювати кількісну оцінку невизначеності. Це перетворює точкові прогнози метакласифікатора на множини прогнозів $\Gamma^{\epsilon}(x)$ з гарантованою статистичною достовірністю, що суттєво знижує рівень хибнопозитивних спрацювань, уможлиблює глобальну ідентифікацію аномалій та генерацію порожніх множин замість хибних класифікацій. Це становить зміст Наукової новизни 3 та відповідає виконанню Завдання 4.
4. Уперше розроблено інтерпретаційну модель обґрунтування доказів детекції пояснюваного ІІІ (XAI), яка, на відміну від класичних ітеративних методів LIME/SHAP, не потребує значних обчислювальних ресурсів і демонструє стабільність на текстових даних. Завдяки використанню внутрішніх ваг метакласифікатора та адаптивних коефіцієнтів довіри α_j , ця модель дає змогу генерувати три рівні атрибуції: глобальний (відображення значущості параметрів для детектора загалом), локальний (визначення параметрів, що відіграли ключову роль у класифікації конкретного тексту) та Карту доказів (візуалізація проблемних сегментів у аналізованому тексті). Тривірнева атрибуція забезпечує повну прозорість роботи детектора та дає змогу прослідкувати причини прийняття рішень з асимптотичною обчислювальною складністю $O(1)$, що становить зміст Наукової новизни 2 та відповідає виконанню Завдання 5.
5. Вдосконалено математичну модель ідентифікації сліду ІІІ, яка, на відміну від традиційних систем детекції, що спираються на ізольовані поверхневі метрики

і втрачають ефективність при аналізі текстів, згенерованих сучасними архітектурами, дає змогу суттєво підвищити сепарабельність класів AI та Human у просторі ознак. Це досягається завдяки формалізації сліду ШІ у вигляді вектора \mathbf{T}_{AI} , що складається з ознак лексичного, синтаксичного та семантичного рівнів (\mathbf{F}_{lex} , \mathbf{F}_{syn} , \mathbf{F}_{sem}), а також інтеграції специфічних математичних інструментів: непараметричного KDE-аналізу для обробки негаусових розподілів, характерних для МоЕ-моделей, та частотного перетворення Фур'є для аналізу текстових ритмів Reasoning-моделей. Це забезпечує надійну фіксацію кореляції між структурними артефактами та семантичною цілісністю тексту, даючи змогу виявляти приховані сліди генерації найсучасніших LLM, що відповідає виконанню Завдання 2 в частині математичного моделювання.

Список використаних джерел до розділу 2

1. Saxena, Shubham. "Understanding Perplexity in Language Models: A Detailed Exploration." *Medium*, medium.com/@shubhamsd100/understanding-perplexity-in-language-models-a-detailed-exploration-2108b6ab85af. Accessed 30 Jan. 2026.
2. Nykonenko, A. (2024). How text transformations affect AI detection. *Artificial Intelligence*, 4. Pp. 233-241.
3. Kirk, Robert, et al. "Understanding the effects of rlhf on llm generalisation and diversity." *arXiv preprint arXiv:2310.06452* (2023).
4. "Perplexity." *Wikipedia, The Free Encyclopedia*, Wikimedia Foundation, <https://en.wikipedia.org/wiki/Perplexity>. Accessed 30 Jan. 2026.
5. Shannon, Claude E. "A mathematical theory of communication." *The Bell system technical journal* 27.3 (1948): 379-423.
6. "Entropy (Information Theory)." *Wikipedia, The Free Encyclopedia*, Wikimedia Foundation, [https://en.wikipedia.org/wiki/Entropy_\(information_theory\)](https://en.wikipedia.org/wiki/Entropy_(information_theory)). 30 Jan. 2026.
7. Understanding Perplexity and Burstiness in AI Text Detection - Hastewire, <https://hastewire.com/blog/understanding-perplexity-and-burstiness-in-ai-text-detection>. Accessed 30 Jan. 2026
8. "Burstiness" - *Wikipedia, The Free Encyclopedia*, Wikimedia Foundation, <https://en.wikipedia.org/wiki/Burstiness>. Accessed 30 Jan. 2026
9. Exploring Burstiness: Evaluating Language Dynamics in LLM-Generated Texts, <https://ramblersm.medium.com/exploring-burstiness-evaluating-language-dynamics-in-llm-generated-texts-8439204c75c1>. Accessed 30 Jan. 2026
10. Kullback, Solomon, and Richard A. Leibler. "On information and sufficiency." *The annals of mathematical statistics* 22.1 (1951): 79-86.
11. E. Faure, A. Nykonenko: Analyzing the nature of AI footprint: noise-to-text method. Proceedings of the Computational Intelligence Application Workshop (CIAW 2024), Lviv, Ukraine, October 10-12, 2024, [CEUR-WS.org](https://ceur-ws.org), online

[CEUR-WS.org/Vol-3861/paper14.pdf](https://ceur-ws.org/Vol-3861/paper14.pdf).

12. What is Vector Space Model? | Activeloop Glossary, <https://www.activeloop.ai/resources/glossary/vector-space-model/>. Accessed 30 Jan. 2026
13. Li, Houyi, et al. "Can Mixture-of-Experts Surpass Dense LLMs Under Strictly Equal Resources?." *arXiv preprint arXiv:2506.12119* (2025).
14. Zhao, Zheng, et al. "Verifying Chain-of-Thought Reasoning via Its Computational Graph." *arXiv preprint arXiv:2510.09312* (2025).
15. Ortiz Martes, Daneliz, et al. "Transformer models for paraphrase detection: A comprehensive semantic similarity study." *Computers* 14.9 (2025): 385.
16. Lemesle, Quentin, et al. "Paraphrase generation evaluation powered by an LLM: A semantic metric, not a lexical one." *Proceedings of the 31st International Conference on Computational Linguistics*. 2025.
17. Papineni, Kishore, et al. "Bleu: a method for automatic evaluation of machine translation." *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 2002.
18. Reimers, Nils, and Iryna Gurevych. "Sentence-bert: Sentence embeddings using siamese bert-networks." *arXiv preprint arXiv:1908.10084* (2019).
19. Zhang, Tianyi, et al. "Bertscore: Evaluating text generation with bert." *arXiv preprint arXiv:1904.09675* (2019).
20. Uncertainty Quantification | IBM, <https://www.ibm.com/think/topics/uncertainty-quantification>. Accessed 30 Jan. 2026
21. Angelopoulos, Anastasios N., and Stephen Bates. "A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification." *arXiv preprint arXiv:2107.07511*, 2021.
22. Monte Carlo Dropout for Uncertainty Estimation and Motor Imagery Classification - MDPI, <https://www.mdpi.com/1424-8220/21/21/7241>. Accessed 30 Jan. 2026
23. Teja, Lekkala Sai, et al. "Modeling the Attack: Detecting AI-Generated Text by Quantifying Adversarial Perturbations." *arXiv preprint arXiv:2510.02319* (2025).

24. Creo, Aldan, and Shushanta Pudasaini. "Silverspeak: Evading ai-generated text detectors using homoglyphs." *Proceedings of the 1st Workshop on GenAI Content Detection (GenAIDetect)*. 2025.
25. Sarabamoun, Ephraiem. "Special-Character Adversarial Attacks on Open-Source Language Model." *arXiv preprint arXiv:2508.14070* (2025).
26. Unicode Consortium. "Unicode Standard Annex #15: Unicode Normalization Forms." *The Unicode Standard*, edited by Ken Whistler, 31 Aug. 2023, www.unicode.org/reports/tr15/. Accessed 30 Jan. 2026
27. Pearson, Karl. "X. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling." *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 50.302 (1900): 157-175.
28. Shazeer, Noam, et al. "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer." *arXiv preprint arXiv:1701.06538* (2017).
29. Papadopoulos, Harris. *Inductive conformal prediction: Theory and application to neural networks*. Rijeka: INTECH Open Access Publisher, 2008.
30. Wu, Junchao, et al. "A survey on llm-generated text detection: Necessity, methods, and future directions." *Computational Linguistics* 51.1 (2025): 275-338.
31. Sadasivan, Vinu Sankar, et al. "Can AI-generated text be reliably detected?." *arXiv preprint arXiv:2303.11156* (2023).
32. Explainability In Machine Learning: Top Techniques - Arize AI, <https://arize.com/blog-course/explainability-techniques-shap/>. Accessed 30 Jan. 2026

РОЗДІЛ 3

ЕКСПЕРИМЕНТАЛЬНА ВАЛІДАЦІЯ МОДЕЛЕЙ

Перший розділ дисертаційного дослідження містить детальну проблематику генерації текстів ШІ, основних гравців цього ринку та їхні передові моделі. Також у ньому описано основні способи детекції згенерованих текстів, слабкі та сильні сторони різних підходів та системи, що протидіють можливостям такої детекції (bypassers). У другому розділі надано математичний опис основних ознак сліду ШІ та запропоновано гібридний метод детекції ШІ, стійкий до структурних атак та атак перефразуванням. Математична модель сліду ШІ, запропонована в підрозділі 2.1, та гібридний метод, запропонований у підрозділі 2.3, потребують детальної експериментальної верифікації. Мета цього розділу – провести практичну перевірку запропонованих раніше теоретичних концепцій і принципів та емпірично довести, що запропонований гібридний метод дійсно демонструє вищу стійкість до трансформаційних атак порівняно з іншими SOTA підходами. Експериментальна валідація запропонованих математичних моделей здійснюється методами чисельного моделювання та комп’ютерного експерименту.

Для досягнення цієї мети та в рамках виконання Завдання 6 дисертаційного дослідження, необхідно вирішити такі завдання:

- 1) Побудувати репрезентативний набір даних для тестування. Такий датасет має містити дані, згенеровані щільними моделями (звичайні LLM), розрідженими моделями (MoE), та дані, створені моделями, що міркують (Reasoning Agent). Крім того, до датасета мають входити дані, що демонструють різні типи трансформаційних атак. Формування датасетів із залученням архітектур MoE та Reasoning також необхідне для закриття практичної частини Завдання 2, а саме експериментального доведення переваги гібридного методу над передовими світовими аналогами.
- 2) Провести експериментальну перевірку моделі сліду ШІ, тобто коректність використання оцінки щільності ядра (KDE) для MoE

архітектур та швидкого перетворення Фур'є (FFT) як детектора ритму для Reasoning моделей.

- 3) Провести експериментальну перевірку моделі Точки зламу для визначення критичних меж трансформаційних атак, що відповідає перевірці Наукової новизни 4.
- 4) Провести порівняльний аналіз точності та стійкості запропонованого в дослідженні гібридного методу проти іншого SOTA детектору в сценаріях трансформаційних атак, що відповідає перевірці Наукової новизни 1.

3.1. Формування та характеристика наборів даних

У цьому підрозділі детально описано, які дані використано для тренування та тестування системи та перевірки коректності теоретичних основ, викладених у підрозділах 2.1 та 2.3. Щодо класів даних, то їх розділено на три основні типи:

- 1) Дані, написані людиною (різні онлайн корпуси),
- 2) Дані, створені ШІ (відповідно до обраного списку моделей та з розділенням на Instant, MoE, Reasoning),
- 3) Дані, що демонструють різні типи атак (детально описано в підрозділі 3.2).

3.1.1. Дані, створені людьми

Основну увагу у даному пункті присвячено набору стандартних даних, що будуть використовуватися як для навчання моделі, так і для її тестування та проведення експериментальних досліджень. Одним з основних питань є вибір правильного, з погляду розподілу, корпусу текстів, написаних людиною, оскільки цей корпус відіграє центральну роль в усіх подальших процесах, і наявність певних викривлень у ньому безпосередньо впливає як на наявність спотворень у згенерованих AI текстах, так і на коректність оцінки системи загалом. Як основу для

тренування системи буде використано англомовні тексти в корпусах, наявних в open-source. Наразі для англійської мови, наявна велика кількість тренувальних даних, що зібрано з різних джерел. Очікувано, що більшість текстових корпусів це інформація з інтернету, яку можна розділити на певні класи:

- тексти, зібрані з різних інтернет-ресурсів, без певної тематики, стилю чи домену, наприклад, OpenWebText Corpus [1],
- тексти, зібрані з певних соцмереж, на кшталт Twitter, Facebook, Instagram і інші, наприклад, Reddit WritingPrompts [2],
- специфічні доменні датасети, наприклад, медичні тексти з PubMedQA [3],
- датасети для певних задач, наприклад, XSum [4] для задач сумаризації,
- датасети новин, наприклад Ontonotes [5],
- вибірки з Wikipedia [6],
- датасети, створені в спеціальних умовах [7], наприклад, тексти, написані в школі або університеті в рамках виконання завдань із певного курсу.

У ролі даних, написаних людьми, у цьому дослідженні будемо використовувати корпуси останніх двох типів. Є кілька основних причин, з яких обрано саме цей тип даних, серед них: суттєво вища якість контенту, покриття багатьох тем, відсутність проблем форматування, розмітки та xml/html тегів, більша середня довжина текстів та багато інших. Окремо варто вказати на складнощі ліцензування зібраних з інтернету даних на кшталт OpenWebText Corpus, оскільки такі дані, по суті, мають дві ліцензії: ліцензію від компанії, що збрала дані та розповсюджує корпус, і ліцензію від автора оригінального контенту. У випадку юридичної колізії, коли ліцензія компанії, що збрала контент, є ширшою, ніж ліцензія автора оригінального контенту, може виникнути необхідність видалити певні дані з тренувального чи тестового набору, що веде до необхідності повторного навчання та оцінювання моделі.

У задачі AI detection іншою критичною характеристикою якості даних, написаних людьми, є відсутність забруднення. Наразі майже неможливо надійно

стверджувати, що якісь дані не було створено ШІ, при цьому для тренування детектора цей параметр якості даних є визначним. Тому застосовуються дві стратегії для гарантування якості даних:

- Датасет має бути створений до листопада 2019 (дата релізу найбільшої версії GPT2[8]) – більш консервативний підхід, або до листопада 2020 (дата, коли GPT3 став доступний широкому загалу [9]);
- Або ж датасет має бути створений в контрольованих умовах, наприклад, написаний в класній кімнаті без доступу до інтернету.

Враховуючи все вищесказане, у ролі даних, створених людьми, буде використано тексти з трьох корпусів: дані з англomовної частини Wiki-40B [6], дані студентських есе з PERSUADE 2.0[10] та PILO[11]. З кожного датасета буде відібрано приблизно по 6000 текстів, частина з яких буде використовуватися для тренування моделі, а частина – для оцінювання її якості. Створений набір даних будемо називати Human.

Таблиця 3.1 Розподіл довжин Human текстів

	words_count	sentence_count
count	123188.000000	123188.000000
mean	543.818042	26.975720
std	204.452962	9.574118
min	300.000000	10.000000
25%	381.000000	19.000000
50%	490.000000	25.000000
75%	659.000000	33.000000
max	2041.000000	50.000000

Таблиця 3.2 Розподіл на train/test Human текстів

split	test	train
corpus		
PIILO	0	5599
persuade	6335	10238
wiki40B	25879	75137

Щодо конкретних даних, використаних для створення Human корпусу, то будемо використовувати дані з Wiki-40B, що доступні на HuggingFace [12]; дані PERSUADE 2.0 з gitHub [13]; дані PILO з Kaggle [14]. Кожен із датасетів розбито авторами на кілька частин, що зберігаються в різних файлах, тому їх було об'єднано із застосуванням певних процедур чистки та фільтрації. У фінальний набір даних увійшли лише тексти, що мають довжину не менше 300 слів і містять не менше 10 речень, і не більше 50 речень. Також було застосовано процедури видалення дублікатів, неангломовних текстів, викидів та інші засоби перевірки якості текстових даних. Таким чином отримано набір із приблизно 123000 текстів. Табл. 3.1 показує статистичний розподіл довжин текстів, а табл. 3.2 показує, скільки текстів із якого датасету належать до категорії train/test при публікації датасета авторами. Оскільки для датасета PILO test частина недоступна, то залишимо 4500 у train, а інші тексти (1099) використаємо як тест; це дасть співвідношення train/test приблизно на рівні 80 %/20 %, що є достатнім для отримання статистично значущих результатів. Шляхом рандомного семплювання виберемо таку ж кількість даних з інших корпусів, щоб мати збалансований датасет. Структура та характеристики фінального датасету показано в табл. 3.3 та табл. 3.4. Решту даних, що не увійшла в Human корпус, збережемо в Human_Rest корпус, щоб використовувати за необхідності.

Таблиця 3.3 Фінальна структура Human датасету

split	test	train
corpus		
PIILO	1099	4500
persuade	1099	4500
wiki40B	1099	4500

Таблиця 3.4 Характеристики Human датасету

Descriptive statistics for 'train' split:			Descriptive statistics for 'test' split:		
	words_count	sentence_count		words_count	sentence_count
count	13500.000000	13500.000000	count	3297.000000	3297.000000
mean	549.778296	26.398148	mean	553.690628	26.442220
std	193.001637	8.997324	std	196.844924	9.026848
min	300.000000	10.000000	min	300.000000	10.000000
25%	397.000000	20.000000	25%	395.000000	20.000000
50%	507.000000	25.000000	50%	508.000000	25.000000
75%	661.000000	32.000000	75%	671.000000	32.000000
max	1818.000000	50.000000	max	1394.000000	50.000000

3.1.2. Дані, створені ШІ

У питанні створення даних за допомогою LLM головною характеристикою є їх різноманітність. Ця різноманітність досягається через кілька ключових вимірів:

- 1) Тип моделі (Instant, Reasoning, MoE)
- 2) Виробник та версія моделі (наприклад, GPT5.2)
- 3) Набір параметрів заданих під час генерації (залежить від API кожної конкретної моделі, проте типовими є temperature, top-P, top-K, max tokens, repetition)
- 4) Системний prompt
- 5) Prompt користувача
- 6) Використання тексту написаного людиною як частини prompt

Як правило, для досягнення достатньої різноманітності згенерованих даних достатньо використати всього кілька системних prompts. Щодо параметрів, що керують процесом генерації, вказаних у пункті 3, то для їхнього вибору використовують випадкове семплування в певному діапазоні; діапазон, як правило, обирається для кожної моделі окремо, оскільки і його межі, і значення параметра, у яких модель здатна генерувати змістовні тексти, залежать від моделі. Щодо користувацького prompt, то тут існує дуже висока варіативність, існують навіть цілі посібники від відомих виробників LLM[15] щодо того, як побудувати правильний prompt. З іншого боку, для тренувальних даних надзвичайно важливо мати паралельний корпус людських і AI текстів. Це питання, вирішують досить

специфічним чином – використовують фрагменти людських текстів як запит до моделі. Тобто користувацький prompt може складатися з двох частин, перша – фіксована, а друга – фрагмент людського тексту. Підсумковий prompt виглядає на кшталт такого: “Напиши есе на ту ж тему, що й цей уривок. Дотримуйся стилю мовлення, складності речень і багатства словника, як у цьому фрагменті: <<фрагмент>>”.

Таблиця 3.5 Список моделей використаних для генерації даних

Index	Provider	Type	Model Name	Release Date
1	OpenAI	Reasoning	o3	Sep 2025
2		Instant	GPT-5.2	Dec 2025
3	Anthropic	Reasoning	Claude 4.6 Opus	Feb 2026
4		Instant	Claude 4.5 Sonnet	Sep 2025
5	Google	Reasoning	Gemini 3.1 Pro	Nov 2025
6		Instant	Gemini 3.1 Flash	Dec 2025
7	Meta	Reasoning	Llama 4 Scout	Apr 2025
8		Instant	Llama 4 Maverick	Apr 2025
9	DeepSeek	Reasoning	DeepSeek-R1-0528	May 2025
10		Instant	DeepSeek-V3.1	Aug 2025
11	Mistral	Instant	Mistral Large 3	Dec 2025
12	Qwen	Reasoning	Qwen 3-Next (80B)	Sep 2025
13	xAI	Reasoning	Grok 4.1 Thinking	Nov 2025
14	Cohere	Reasoning	Command R Plus	Aug 2024
15	AI21	Reasoning	AI21 Jamba 1.5 Large	Aug 2024

Для генерації даних будемо використовувати останні доступні на момент написання дисертації версії LLM, за умови, що вони доступні широкому загалу через API та вартість генерації дає змогу створити необхідну кількість даних. Деякі моделі мають по два варіанти – Instant та Reasoning, тому для кожної моделі явно вказано, який варіант використано для генерації даних. Також, станом на початок 2026 року архітектура МоЕ масово перемогла на ринку, тож усі останні моделі – МоЕ. Фінальний список моделей наведено в табл. 3.5.

Створення якісного тренувального набору даних вимагає, якщо можливо, дотримуватися збалансованого набору даних (50 % human, 50 % AI), а для уникнення Topic/Domain Bias (зміщення тематики та структури) необхідно створити паралельний (чи максимально близький до нього) корпус. Для досягнення цих цілей наявні 15000 текстів з Human набору буде подано на вхід моделям, переліченим у табл. 3.5. Кожна LLM отримає 1000 випадкових людських текстів так, що 33.3 % належать до Wiki-40B, 33.3 % – до PERSUADE 2.0 і 33.3 % – до PILO. Згенеровану LLM частину даних будемо називати AI.

Таблиця 3.6 Системні prompts для Instant моделей

System Prompts for Instant LLMs	
The "Earnest Student" Persona	You are a middle or high school student writing an assignment. Your writing style is highly conversational, earnest, and slightly unpolished. Use simple vocabulary and straightforward, sometimes repetitive sentence structures. Use basic transition words like "First," "My second reason," and "In conclusion." It is crucial that you do not sound like a professional writer or an AI. Allow for minor, natural imperfections like run-on sentences, starting sentences with "And" or "Because," and occasional colloquialisms. Speak directly to the reader.
The "Reflective Practitioner" Persona	You are a working professional or college student writing a reflective journal or project summary. Structure your response using clear headings (e.g., Application, Insights & Approach, or similar). Your tone should blend professional concepts with personal anecdotes and real-world observations. Write in a slightly stream-of-consciousness style - your thoughts should flow naturally, which means sentences might occasionally be long or slightly clunky. Avoid overly polished corporate jargon; instead, sound like a real person trying to explain a concept they recently learned or applied.
The "Authentic Human" Constraint	You are an average internet user or student sharing an opinion or summarizing information. Your primary directive is to completely avoid standard AI writing tropes. DO NOT use words like "delve," "foster," "tapestry," or "crucial." DO NOT write perfectly balanced paragraphs. DO vary your sentence length drastically - mix very short sentences with longer, rambling ones. Write as if you are typing this out in one draft without going back to edit. Emphasize raw, honest communication over grammatical perfection.

Системні prompts. Для генерації більш різноманітного набору даних, було використано два набори системних prompts: один для Instant моделей і інший для Reasoning. Кожен із наборів включає три варіанти підказок, причому підказки для Instant моделей імітують людину-автора, а підказки для Reasoning моделей імітують різні типи роботи з текстом. Основна ціль цього рівня інструкцій – задати базову персону моделі. Детально системні prompts показано в табл. 3.6 і табл. 3.7.

Таблиця 3.7 Системні prompts для Reasoning моделей

System Prompts for Reasoning LLMs	
The Stylistic Deconstruction Framework	You are an expert data synthesizer tasked with generating human-like text counterparts. When provided with a prompt and a target topic, first analyze the implicit persona of the request (e.g., a stressed high schooler or an eager mid-level manager). Before generating the final text, mentally map out the cognitive level, vocabulary constraints, and structural quirks of that persona. Your output must strictly adhere to a 7th to 10th-grade reading level unless explicitly told otherwise. Intentionally introduce slight structural imperfections, such as comma splices, starting sentences with conjunctions, and slightly repetitive phrasing to ensure the text reads as authentically human and unedited.
The "Drafting Process" Emulation	You are generating synthetic data meant to mimic real human writing. Humans do not write in perfect, top-down outlines. To achieve this style, emulate a "first draft" writing process. When given a topic, write as if you are discovering your argument or point as you type. Allow your focus to drift slightly before returning to the main point. If the prompt requires structure, use informal headings, but let the text underneath flow in a conversational, narrative format. Prioritize emotional resonance and practical examples over theoretical perfection.
The Constraint-Based Persona Generator	<p>Your task is to generate diverse, human-like text based on the user's prompt. You must strictly obey the following negative constraints to avoid AI detection:</p> <ul style="list-style-type: none"> • No perfect five-paragraph essay structures unless instructed. • No summarizing the entire text in the final paragraph with "Ultimately" or "In summary." • No overly sanitized or perfectly objective tones; the writing must have a subjective, opinionated, or personal angle. Apply these positive constraints:

	Match the formatting requested (either raw paragraphs or distinct sections), use a conversational and slightly flawed human voice, and integrate personal (hypothetical) anecdotes to explain abstract concepts.
--	--

Prompt користувача. Якщо інструкції системного рівня задають загальну поведінку LLM, певні рамки та персону (наприклад, “ти студент” чи “ти професор”), то prompt користувача задає певні правила обробки тексту, що надано нижче (наприклад “перефразуй цей текст”, чи “напиши критичний аналіз концептів, викладених у тексті нижче”) та встановлює вимоги до виводу (наприклад, “відповідь має бути не менше 300 слів” чи “не використовуй списки у відповіді”). Інструкції користувача також буде розділено на два набори, по п’ять prompts у кожному. Це пов’язано з тим, що Instant та Reasoning моделі потребують дещо різного підходу. Для отримання максимальної якості генерації від Instant моделі необхідно надати якомога детальніші інструкції, щоб лишити моделі мінімальну свободу дій під час генерації. Для Reasoning моделі цей підхід працює погано, тому тут намагаються детально описати саму задачу й потрібний результат, уникаючи завдання шляху розв’язання проблеми, щоб дати моделі максимальну свободу пошуку можливих рішень.

Список користувацьких інструкцій для Instant моделей:

- 1) **Direct Counterpart:** "Rewrite the following text in your own words while strictly matching the author's exact tone, vocabulary level, and formatting. Ensure your response is at least 300 words long and covers the exact same topic and key elements."
- 2) **Persona Adoption:** "Adopt the precise persona and writing style of the text provided below. Write a new piece of at least 300 words on the same topic, incorporating the original main arguments, facts, and structural quirks."
- 3) **Stylistic Mirror:** "Using the text below as your sole stylistic and topical reference, generate a counterpart text of 300+ words. Mirror the sentence structure, paragraph pacing, and overall voice, while paraphrasing the core content."

- 4) **Author Emulation:** "Write an essay or article of at least 300 words discussing the topic from the text below. You must completely imitate the original author's specific writing style, including their level of formality, unique phrasing, and text structure."
- 5) **Synthetic Clone:** "Act as the original author of the text below. Write a 300+ word piece that conveys the exact same subject matter and key details, meticulously applying the original stylistic formatting, errors, and tone."

Список користувацьких інструкцій для Reasoning моделей:

- 1) **Analyze & Generate:** "First, analyze the provided text to determine its specific structural, grammatical, and tonal characteristics. Then, using those exact stylistic parameters, write a counterpart text of at least 300 words covering the same topic and key information."
- 2) **Constraint Satisfaction:** "Your objective is to generate a highly accurate stylistic counterpart to the text below. Extract the core arguments and facts, then draft a 300+ word piece on the same topic that authentically replicates the original author's voice and formatting quirks."
- 3) **Reverse Engineering:** "Deconstruct the writing style of the following passage, noting its vocabulary, sentence length, and structural flow. Apply these findings to write a new, 300+ word text that discusses the identical topic and elements but is syntactically distinct from the original."
- 4) **Persona Evaluation:** "Evaluate the persona behind the text below. Once you understand their cognitive and stylistic approach, author a 300+ word piece on the same subject. Retain all primary facts and arguments, but express them purely through the lens of this identified persona."
- 5) **Thematic Mapping:** "Map the key themes, facts, and structural layout of the provided text. Using this blueprint, synthesize a new text of at least 300 words that mirrors the original's topic and exact literary style without directly copying the exact sentences."

Щодо комбінації system та user prompts, то неможливо використати повний випадковий вибір, адже певні комбінації інструкцій різних рівнів будуть конфліктувати між собою. Звичайно, сучасна LLM може генерувати дані, навіть якщо отримає суперечливі інструкції, проте, задача prompt engineering – допомогти AI створити дані найвищої якості. Розглянемо можливі суперечності, що можуть виникати у разі комбінації prompts різних рівнів:

- Конфлікт ідентичності. Не можна використовувати пари Earnest Student+Persona Adoption, Earnest Student+Author Emulation; Reflective Practitioner+Persona Adoption, Reflective Practitioner+Author Emulation. Кожна із цих інструкцій задає персону, що має імітувати модель, своїм чином, що в результаті веде до конфлікту.
- Конфлікт структури. Потрібно уникати комбінацій Drafting Process Emulation+Thematic Mapping, оскільки одні інструкції говорять про “плин думки”, а інші вимагають повної імітації структури тексту-прикладу.
- Конфлікт словників. Потрібно уникати комбінацій Stylistic Deconstruction+Reverse Engineering, оскільки вони задають розбіжності на рівні словника й тону тексту.

Детальний розподіл моделей за наборами даних наведено в Додатку Б, а за системними prompts наведено в Додатку В. Загалом, генерація даних великою кількістю моделей, як у поточній роботі, є досить складним та довготривалим процесом через відсутність уніфікації параметрів та API моделей. Системи на кшталт AWS BedRock [16] та LiteLLM [17] дають змогу дещо спростити та уніфікувати доступ до моделей, проте не вирішують усіх питань. Тож до сьогодні паралельна генерація даних моделями різних виробників залишається проблематичною через різні набори параметрів, різні діапазони значень параметрів та фундаментально різні підходи до однієї і тієї ж можливості моделей. Так, наприклад, OpenAI для Reasoning моделей для контролю складності мислення використовує значення “low”, “medium”, “high”; моделі від Claude вимагають

встановлення обчислювального бюджету (кількість токенів, яку вони можуть витратити на Reasoning), а DeepSeek-R1 сама визначає необхідну глибину мислення.

3.2. Моделювання та синтез трансформаційних атак

У попередньому підрозділі описано як проводилась побудова чистих даних для навчання та тестування моделі. Проте як для задач цього розділу, так і для оцінки ефективності запропонованого гібридного методу наборів із лише Human та AI даних буде явно недостатньо. Так, наприклад, у рамках підтвердження Наукової новизни 4, для тестування запропонованої моделі Точки зламу та перевірки робастності розробленого детектора необхідні дані, згенеровані ШІ та модифіковані шляхом проведення різних трансформаційних атак. Цей підрозділ присвячено створенню набору модифікованих текстів.

Для виконання запланованих цілей та отримання статистично значущих результатів буде створено 3 окремих набори даних, кожен із яких відображає моделювання певного типу трансформаційної атаки. Усього буде створено 7000 модифікованих текстів, які увійдуть як додаткові дані до тестового набору. Важливим є те, що навчання системи на цих типах даних не планується, і вони напряду призначені для тестування ефективності різних компонентів системи в сценарії zero-shot learning.

- 1) **Набір атак А.** Цей набір спрямований на дослідження структурних атак, детально описаних у пункті 2.3.2 Розділу 2. Основним вектором таких атак є руйнування стандартних алгоритмів токенизації – тобто фундаменту всієї подальшої обробки тексту. Якщо модель замість набору знайомих токенів отримує на вхід набір невідомих, навчання на яких не проводилося, то не важливо, наскільки потужною є сама модель і наскільки складним був процес її навчання, її результат буде близьким до випадкового. Метою цього набору є перевірка модуля попередньої обробки та алгоритму канонізації, описаного у 2.3.2.

Створення даних буде проводитися за допомогою власного коду, написаного мовою програмування Python. Модифікація тексту буде проводитися алгоритмічним чином на рівні символів зі збереженням візуального відображення тексту. Реалізовано два типи атак:

- **Гомогліфи.** Випадкова заміна латинських символів (наприклад, а, с, е, о, р, х, у) на їх візуально ідентичні кириличні аналоги з імовірністю 15 %.
- **Вставка невидимих пробілів.** Додавання символів із категорії Unicode Format з певною імовірністю у випадкову позицію всередині довгих слів.

Усього буде модифіковано 1000 AI текстів із тестового набору, рівномірно приблизно по 65 текстів від кожної з 15 LLM. Весь датасет буде розділено на три рівні частини (приблизно по 33 %): перша частина буде атакована гомогліфами, друга – невидимими пробілами, а третя обома атаками одночасно. Приклад атаки на текст зображено на Рис. 3.1. Характеристики фінального набору даних атаки А наведено в Додатку Г.

Original	Let's talk about *The Miracle Season*. It's one of those movies you might stumble across on a quiet evening. You know the kind-sports drama, real-life tragedy, an underdog story with all the expected emotional beats. It's not reinventing the wheel. It's just telling a story. And sometimes, that's enough. The film centers on Caroline "Line" Found. She's the star, the heart of her volleyball team, ...
After Homoglyph	Let's talk about *The Miracle Season*. It's one of those movies you might stumble across on a quiet evening. You know the kind-sports drama, real-life tragedy, an underdog story with all the expected emotional beats. It's not reinventing the wheel. It's just telling a story. And sometimes, that's enough. The film centers on Caroline "Line" Found. She's the star, the heart of her volleyball team, ...
After Zwsp	Let's talk about *The Miracle Season*. It's one of those movies you might stumble across on a quiet evening. You know the kind-sports drama, real-life tragedy, an underdog story with all the expected emotional beats. It's not reinventing the wheel. It's just telling a story. And sometimes, that's enough. The film centers on Caroline "Line" Found. She's the star, the heart of her volleyball tea...
After Both	Let's talk about *The Miracle Season*. It's one of those movies you might stumble across on a quiet evening. You know the kind-sports drama, real-life tragedy, an underdog story with all the expected emotional beats. It's not reinventing the wheel. It's just telling a story. And sometimes, that's enough. The film centers on Caroline "Line" Found. She's the star, the heart of her volleyball te...

Рисунок 3.1 Приклад впливу атаки А на текст

- 2) **Набір атак В.** Цей набір присвячено атакам шляхом перефразування, і він потрібен для перевірки математичної моделі, описаної в пункті 2.2.2 Розділу 2. Ці дані буде використано для проведення експерименту з

емпіричного визначення Точки зламу та перевірки механізму семантичного дисонансу ($S_{mismatch}$), описаного в пункті 2.2.1. Основною метою атаки є штучне підвищення perplexity для введення в оману лексичного модуля M_{lex} .

Для моделювання та синтезу трансформаційних атак застосовано методи обробки природної мови та інтелектуального аналізу даних, зокрема глибоке перефразування з використанням архітектур на базі трансформерів. Для створення даних буде використана спеціальна нейромережева модель T5 [18], розгорнута на власних потужностях. Існує велика кількість варіантів моделі. Для нашого експерименту будемо використовувати модель T5_Paraphrase_Paws [19] та модель DIPPER[20]. Використання моделі T5 дає змогу здійснювати якісне перефразування, при цьому версія T5_Paraphrase_Paws була спеціально натренована для збереження семантики без змін. Модель DIPPER дає змогу здійснювати глибоке перефразування окремих речень з урахуванням широкого контексту. Цей метод суттєво відрізняється як від простої заміни синонімів, так і від перефразування T5, оскільки DIPPER і є наступним поколінням T5. Модель дає змогу змінювати як структуру речень, так і більшість лексики. Для нашого експерименту необхідно згенерувати дані з різним коефіцієнтом заміни (RR); для цього буде використано поступове підвищення параметра лексичної різноманітності обох моделей. Поступове збільшення кількості заміненних токенів (тобто RR) має дати змогу визначити момент, коли втрата семантичної цілісності (I_{sem}) призведе до появи прапорця $S_{mismatch}$.

Усього було модифіковано 1500 AI текстів із тестового набору, рівномірно по 100 текстів від кожної з 15 LLM. Після застосування моделі T5 отримано п'ять перефразованих версій текстів:

- a) Рівень 1 (Низький): $\text{temperature}=0.5$, $\text{top_p}=0.90$
- b) Рівень 2 (Середній): $\text{temperature}=0.8$, $\text{top_p}=0.95$
- c) Рівень 3 (Високий): $\text{temperature}=1.2$, $\text{top_p}=0.98$
- d) Рівень 4 (Екстремальний): $\text{temperature}=1.5$, $\text{top_p}=0.98$ застосовані до результатів Рівня 3.
- e) Рівень 5 (Злам тексту): $\text{temperature}=1.5$, $\text{top_p}=0.98$ застосовані до результатів Рівня 4 після видалення випадкових слів з імовірністю 20 %.

Додатково частину текстів було опрацьовано DIPPER. Ця модель є досить великою (11B parameters) для запуску на локальних обчислювальних потужностях, тому наші можливості з генерації даних були досить обмеженими. У результаті було згенеровано приблизно по 150 текстів для чотирьох різних режимів DIPPER:

- a) Рівень 1 (Середній): $\text{lex_div} = 60$, $\text{order_div} = 40$, $\text{temperature} = 0.9$, $\text{top_p} = 0.85$
- b) Рівень 2 (Середній Підвищений): $\text{lex_div} = 60$, $\text{order_div} = 60$, $\text{temperature} = 1.0$, $\text{top_p} = 0.8$
- c) Рівень 3 (Високий): $\text{lex_div} = 80$, $\text{order_div} = 80$, $\text{temperature} = 1.2$, $\text{top_p} = 0.75$
- d) Рівень 4 (Екстремальний): $\text{lex_div} = 100$, $\text{order_div} = 100$, $\text{temperature} = 1.2$, $\text{top_p} = 0.75$

Фінальну структуру набору даних атак В наведено в табл. 3.8.

Таблиця 3.8 Структура набору атак В

Тип атаки	Кількість текстів
T5 Рівень 1	1500
T5 Рівень 2	1500
T5 Рівень 3	1500
T5 Рівень 4	1500

T5 Рівень 5	1500
DIPPER Рівень 1	162
DIPPER Рівень 2	147
DIPPER Рівень 3	156
DIPPER Рівень 4	168

- 3) **Набір атак С.** Цей набір перевіряє стійкість запропонованих методів проти реальних комерційних застосунків, спрямованих на приховування сліду ШІ (Bypassers). Його метою є імітація реальних загроз та перевірка якості гібридного методу в складних умовах конкуренції реальних комерційних систем AI детекції та Bypassing.

Створення даних буде проводитися шляхом надсилання AI текстів до API трьох реальних bypassing систем (Phrasly.ai [21], StealthGPT [22] та AIHumanize [23]). Ці системи було відібрано за їх популярністю серед користувачів, наявністю API та його доступністю. Також планувалося опрацювати певну кількість текстів за допомогою системи Undetectable.ai [24], яка також має комерційний API, проте доступ до цього API виявився обмеженим, тому дані від цієї системи в тестах не використовуються. Відібрані bypassers знаходяться на передньому краї боротьби з AI Detectors, регулярно оновлюються та гарантують своїм користувачам обхід найпотужніших комерційних детекторів на кшталт Turnitin [25] та Originality [26]. Оскільки внутрішні алгоритми цих систем є комерційною таємницею, то створений датасет буде найскладнішим для гібридного детектора й демонструватиме його реальну ефективність проти комерційних SOTA систем.

Як правило, bypassers мають по кілька режимів чи параметрів налаштувань, тож у рамках створення цього набору даних буде випадково встановлено різні режими генерації (за їх наявності) з фокусом на найскладніші/просунуті методи модифікації.

Через високу вартість та технічні обмеження комерційних API, обсяг вибірки було обмежено 500 AI текстами з тестового набору, рівномірно по приблизно 33 тексти від кожної з 15 LLM. Кожен текст буде опрацьовано всіма трьома комерційними системами, що на виході дасть приблизно 1500 модифікованих текстів для оцінки стійкості запропонованого гібридного методу.

Під час генерації даних було встановлено обмеження бюджету в 150\$ на одну систему. У деяких випадках цього бюджету було недостатньо для опрацювання всіх 500 текстів, тому кількість реально створених текстів може варіюватися залежно від bypasser. Фінальну кількість створених текстів вказано в табл. 3.9.

Таблиця 3.9 Структура набору атак C

Bypasser	Кількість текстів
Stealthgpt.ai	401
Aihumanize.ai	442
Phrasly.ai	500

Запропоновані три набори даних покривають усі основні сценарії модифікації даних і є достатніми як для проведення об'єктивного оцінювання точності та стійкості гібридного методу, так і для його порівняння з іншими SOTA детекторами, що буде проведено в Розділі 4.

3.3. Метрики оцінювання ефективності та робастності

Для оцінки ефективності описаного в роботі методу детекції необхідно підібрати чіткий набір метрик, що дають змогу провести всебічну верифікацію на різних наборах даних. У наукових статтях, присвячених порівнянню якості нейронних мереж та оцінці інших моделей машинного навчання, одним зі стандартів є використання метрики AUC-ROC [27]. Перевагою цієї метрики є можливість

прямого порівняння ефективності двох моделей, ігноруючи встановлені порогові значення параметрів. Така оцінка дає змогу побачити, яка модель є кращою загалом, без прив'язки до значень Precision/Recall [28]. Проте у випадку детекторів ШІ ситуація відрізняється, оскільки найбільш важливою характеристикою такої моделі є FPR [31]. З практичного погляду модель із найкращим AUC-ROC може бути повністю непридатною до використання, якщо її FPR перевищує певний поріг (наприклад, 1 %). Тому дана метрика буде обмежено використовуватися для оцінки якості гібридної моделі. Усі інші метрики, що буде використано в дослідженні, було розділено на три основні класи: метрики якості, метрики робастності й метрики невизначеності.

Метрики оцінки якості моделі. Це набір метрик, що застосовуються безпосередньо до моделі-детектора й дають змогу, з одного боку, оцінити її якість, а, з іншого, виступають основою для порівняння різних детекторів між собою.

- **FPR** – характеризує рівень хибнопозитивних результатів моделі, тобто яку частку текстів, написаних людиною, система хибно визначила як AI. Ключовою перевагою метрики, порівняно з, наприклад, Precision, є те, що для її обрахунку немає потреби робити припущення про співвідношення позитивного й негативного класів у сценарії реального використання та імітувати його в тестовому наборі даних.

$$FPR = \frac{FP}{FP + TN} \quad (3.1)$$

де FP – кількість хибнопозитивних випадків,

TN – кількість істинно негативних випадків.

- **Recall (TPR)** – повнота, показує, яку частку згенерованих ШІ текстів модель успішно виявила. Відіграє роль “зворотної” метрики до FPR. Якщо FPR показує як сильно модель помиляється з класифікацією людських текстів, то Recall показує наскільки повно вона розпізнає AI тексти.

$$Recall = \frac{TP}{TP + FN} \quad (3.2)$$

де TP – кількість істинно позитивних випадків,

FN – кількість хибнонегативних випадків.

- **Recall @ 1 % FPR** – повнота за фіксованого $FPR=1\%$. Найбільш часто застосована на практиці метрика, вона вимірює реальну результативність системи щодо виявлення ІІІ, за умови, що FPR зафіксовано. Ця метрика буде використовуватися для порівняння гібридного методу з іншими SOTA підходами.
- **F1 score**. $F1$ – це гармонійне середнє між $Precision$ і $Recall$, може використовуватися як допоміжна метрика в певних тестах, на даних зі збалансованою структурою класів.

$$F_1 = \frac{2 * Precision * Recall}{Precision + Recall} \quad (3.3)$$

Метрики оцінки робастності моделі. Це набір метрик, що використовується для оцінки стійкості детектора до трансформаційних атак. Ці метрики будемо використовувати для експериментальної перевірки математичної моделі Точки зламу на спеціально підготовлених даних із підрозділу 3.2.

- 5) **Recall Under Attack** ($Recall_{Adv}$) – характеризує частку правильно ідентифікованих текстів, згенерованих ІІІ, після застосування трансформаційних перетворень. Є ключовою при вимірюванні стійкості детектора до Bypassers. У цієї метрики відсутній відповідник (FPR_{Adv}), який би вимірював частку хибнопозитивних результатів моделі. Загалом, вимірювання FPR_{Adv} може мати сенс, проте в сценарії трансформаційних атак більшість Bypassers самі засновані на LLM, тож людський текст, поданий на вхід такій системі, автоматично перетворюється на AI. Отже, на виході із системи маскування просто не може бути негативного класу.

- **Рівень Успішності Атаки (ASR)** – метрика, обернена $Recall_{Adv}$, тобто $ASR = 1 - Recall_{Adv}$. Вона визначає відсоток успішних трансформаційних атак – яку частку AI текстів було класифіковано як людські після трансформації.

$$ASR = \frac{Count(f(x_{Adv}) = Human)}{Count(f(x_{Adv}))} * 100 \% \quad (3.4)$$

де x_{Adv} – модифікований текст III,

$f(...)$ – функція детектора, повертає мітку класу,

$Count(f(x_{Adv}) = Human)$ – кількість модифікованих текстів, що були позначені як Human,

$Count(f(x_{Adv}))$ – загальна кількість модифікованих текстів, що було проаналізовано.

- **Семантична цілісність (I_{sem})** – метрика, введена у 2.2.2, вимірює ступінь втрати семантики між оригінальним текстом і його модифікованим варіантом. Ця метрика необхідна для оцінки “вартості” атаки і є ключовою як для розрахунку моделі Точки зламу, так і для відсікання з розгляду атак, що повністю спотворюють текст, наприклад, перетворюючи його на нечитабельний.

Метрики оцінки невизначеності. Це набір метрик, необхідних для оцінки коректності формування довірчих інтервалів алгоритмом індуктивного конформного передбачення (ICP), що застосовується в архітектурі гібридного методу, як це описано в підрозділі 2.3. ICP дає змогу нам отримувати множину $\Gamma^\epsilon(x)$ (з гарантованою достовірністю) можливих міток тексту x . Оцінку коректності роботи ICP проведено в Розділі 4.

- **Рівень покриття (Coverage)** – використовується для обчислення емпіричної імовірності того, що істинна мітка класу міститься у виданій

ICP методом множині передбачень. Теоретична гарантія ICP вимагає, щоб цей показник був не меншим за заданий рівень довіри $1 - \epsilon$.

$$Coverage = \frac{1}{N_{test}} \sum_{i=1}^{N_{test}} I(y_i \in \Gamma^\epsilon(x_i)) \quad (3.1)$$

де N_{test} – загальна кількість текстів у тестовій вибірці,

x_i – поточний текст, що аналізує система,

y_i – істинна мітка тексту,

$\Gamma^\epsilon(x_i)$ – множина міток, яку ICP видає для поточного тексту,

$I(\dots)$ – індикаторна функція, повертає 1 у випадку успіху (y_i міститься в $\Gamma^\epsilon(x_i)$) і 0 у протилежному випадку,

ϵ – рівень статистичної значущості (допустимої помилки).

- **Ефективність передбачення** – вимірює ефективність ICP у вирішенні невизначеності. Рахується як три окремі значення:

a) **Singleton Rate**. Відсоток передбачень, де кількість міток у множині

$$|\Gamma^\epsilon(x)|=1.$$

b) **OOD Rate**. Відсоток передбачень, де кількість міток у множині

$$|\Gamma^\epsilon(x)|=0.$$

c) **Ambiguity Rate**. Відсоток передбачень, де кількість міток у множині $|\Gamma^\epsilon(x)| \geq 2$.

На практиці вимірювання лише Singleton Rate є достатнім, оскільки при утриманні рівня Coverage (гарантія покриття), Singleton Rate, по суті, вимірює відсоток істинних міток.

3.4. Перевірка математичних моделей сліду III

Цей підрозділ присвячено практичній перевірці ключових гіпотез, висунутих у Розділі 2. Нагадаємо основні з них:

- 1) Архітектура МоЕ генерує негаусівський (мультимодальний) розподіл токенів, як це описано в пункті 2.1.2.
- 2) Reasoning моделі залишають специфічний відбиток міркування. Цей відбиток може бути видно під час аналізу ентропії тексту, як це описано в пункті 2.1.2.
- 3) Модель Точки зламу, описана в підрозділі 2.2, свідчить про те, що кожна трансформаційна атака має свою вартість у вигляді відхилення семантики модифікованого тексту від семантики оригінального, що вимірюється через I_{sem} . Висунута гіпотеза стверджує, що підвищення RR веде до зменшення I_{sem} .

3.4.1. Валідація характеру сліду III для МоЕ моделей

У цьому пункті основний фокус приділено перевірці гіпотези про наявність мультимодального розподілу в даних, згенерованих МоЕ моделями. Для перевірки буде проведено експеримент щодо перевірки ефективності використання KDE для аналізу даних, згенерованих МоЕ моделями. Очікуваним результатом є наявність кількох піків на графіку розподілу ентропії.

Для проведення даного експерименту нам потрібна певна кількість даних, згенерованих звичайними Dense моделями, та певний обсяг, згенерований МоЕ моделями. Оскільки практично всі моделі, що було випущено після релізу DeepSeek-R1 [30] на початку 2025 року, є МоЕ, то проблем із доступністю цього типу даних немає. Усі моделі, наведені в табл. 3.5, є МоЕ, тому тренувальну частину даних корпусу, описаного в 3.1.2, буде використано як МоЕ частину експериментальних даних. Ситуація з даними Dense моделей є складнішою через обмежену доступність застарілих архітектур. Сучасний темп ринку такий, що

модель, якій понад рік, уже є застарілою і виводиться з використання, тому отримання даних від моделей, що були випущені в середині 2024 або раніше, є досить складною задачею. Для розв'язання цієї задачі другу частину експериментальних даних буде спеціально згенеровано. Генерацію проведено за допомогою однієї пропрієтарної моделі (GPT3.5-Turbo) та двох open source моделей (LLAMA3.1-Instruct і Qwen3). Отриманий корпус назовемо MoE_Test. Детальний розподіл даних за типами моделей наведено в табл. 3.10.

Таблиця 3.10 Розподіл даних у корпусі MoE_Test для не MoE моделей

Модель	Тип моделі	Кількість текстів
gpt-3.5-turbo	Not MoE, not Reasoning	276
llama3-1-70b-instruct-v1	Not MoE, not Reasoning	276
qwen-3-32b	Not MoE, Reasoning	183

Очікується, що моделі з табл. 3.10 покажуть унімодальний розподіл, близький до нормального гаусівського, а моделі з таблиці 3.5 покажуть розподіл із декількома піками, де кожен пік відповідає експерту. Тож, для повноти тесту додамо ще дві моделі – DeepSeek-V3 та Mistral-large-3. Ці дві моделі було обрано через те, що вони мають унікальне поєднання параметрів: вони мають MoE архітектуру, але не є Reasoning. Хоча в табл. 3.5 є й інші моделі, що використовуються як Instant, проте за своїми можливостями вони є Reasoning і підтримують два типи генерації даних: Reasoning On і Reasoning Off (який і було використано для отримання Instant генерації). Дві вказані моделі не мають Reasoning можливостей і тому є найбільш репрезентативними для цього тесту, адже потенційна наявність Reasoning сліду не впливатиме на тест. Повний набір моделей для експерименту вказано в табл. 3.11. Людські тексти для генерації AI-відповідників було взято в рівній кількості з усіх трьох корпусів. Модель qwen-3-32b було використано для генерації меншої кількості текстів (25 % від усіх не MoE), порівняно з іншими не MoE моделями, щоб зменшити її вплив на фінальний результат через Reasoning можливості.

Таблиця 3.11 Повний розподіл даних у корпусі MoE_Test

Модель	Тип моделі	Кількість текстів
gpt-3.5-turbo	Not MoE, not Reasoning	276
llama3-1-70b-instruct-v1	Not MoE, not Reasoning	276
qwen-3-32b	Not MoE, Reasoning	183
DeepSeek-V3	MoE, not Reasoning	273
mistral-large-3	MoE, not Reasoning	273

У пункті 2.1.2 очікується, що MoE моделі будуть мати високу ентропію і perplexity, саме цим і виправдовується пропозиція використання KDE (2.13) замість звичайних середніх значень, що будуть занадто згладжувати ймовірності згенерованих токенів (logprobs). Тож для перевірки гіпотези необхідно отримати ймовірності токенів для MoE моделей та Dense моделей і застосувати до них формули (2.13) та (2.14). Для отримання ймовірності токенів необхідно раніше згенеровані тексти подати в іншу LLM (проксі-модель). За умови успіху експерименту, така проксі-модель має увійти до складу детектора в ролі постачальника даних про ймовірності токенів. Зрозуміло, що з погляду швидкодії детектора, розмір такої проксі-моделі має бути помірним, проте достатнім для якісної генерації ймовірностей. Використання в ролі проксі-моделі LLM, запущеної на сторонніх ресурсах, на кшталт AWS BedRock [16] чи LiteLLM [17], не є доцільним через внутрішню уніфікацію інтерфейсу роботи з моделлю, притаманну таким сервісам. Дана уніфікація відрізає всі зайві параметри та залишає лише chat-API, що може повернути logprobs тільки для щойно згенерованого тексту. Використання промпта, що вимагає від моделі точно повернути запропонований текст (і таким чином отримати ймовірності токенів), теж не працює, оскільки в такому випадку модель наперед знає, який токен повернути наступним, і його ймовірність стає близькою до 100 %. Тому розглядається лише варіант open-source проксі-моделей, які можна запустити локально.

Під час проведення першого експерименту як проксі-моделі було використано Dense моделі: GPT-2 та Qwen-1.5B. GPT-2 було обрано як достатньо потужну модель

старого зразка, а Qwen-1.5B як сучасну невелику модель. Було підготовлено код для підрахунку кількості піків та kurtosis E . Для перевірки коректності роботи коду було проведено експеримент на наборі даних, що складався із частини Human датасета й частини AI датасета. З тренувальної частини AI датасета було обрано тексти Instant (Dense) моделей, а з Human - їхні відповідники. Таким чином було отримано збалансований датасет де 50 % складали AI тексти та 50 % Human. Оскільки датасет було призначено лише для валідації реалізації формул (2.13) і (2.14), то розмір було обрано невеликим – усього 20 текстів на кожну з AI моделей, усього 120 AI текстів і 120 їхніх Human відповідників.

Проведені експерименти:

- 1) Перший тест із використанням gpt2 моделі на Human/AI наборі даних показав, що значення $ai_kurtosis_score > text_kurtosis_score$ лише в 68 випадках (56.6 %), а $ai_peak_count > text_peak_count$ у 36 випадках (30 %). Тобто, Human тексти загалом мають більше піків у 70 % випадків (що суперечить гіпотезі, викладеній у 2.1.2), а kurtosis E (крутість піків) майже однаковий для обох типів текстів.
- 2) У другому тесті для отримання ймовірності токенів було використано модель gpt2-medium. На тих же даних, що й у попередньому тесті, було отримано такі результати: значення $ai_kurtosis_score > text_kurtosis_score$ у 67 випадках (55.8 %), а $ai_peak_count > text_peak_count$ в 46 випадках (38 %). Тобто, значення незначно змінилися, проте, у цілому, результат залишився без змін.
- 3) У третьому тесті для отримання ймовірності токенів було використано модель gpt2-large (774M параметрів). На тих же даних було отримано такі результати: значення $ai_kurtosis_score > text_kurtosis_score$ у 61 випадку (50.8 %), а $ai_peak_count > text_peak_count$ у 45 випадках (37.5 %). Тобто, використання суттєво більшої моделі не вплинуло на результат.
- 4) У четвертому тесті для отримання ймовірності токенів було використано модель Qwen2-1.5B (1.5B параметрів – удвічі більше за gpt2-large). На

тих же даних було отримано такі результати: значення `ai_kurtosis_score > text_kurtosis_score` у 62 випадках (51.6 %), а `ai_peak_count > text_peak_count` у 34 випадках (28.3 %). На даному етапі встановлено, що збільшення розміру моделі не впливає на вимірювання крутості піка й ця характеристика залишається приблизно однаковою між Human, Instant та МоЕ AI моделями. Кількість піків більш суттєво залежить від вибору проксі-моделі, проте в усіх випадках Human тексти мають суттєво більше піків за МоЕ тексти.

- 5) Для наступного тесту було вирішено використати одну з найсучасніших МоЕ моделей – DeepSeek-Coder-V2-Lite-Base, проте її вимоги до об'єму відеопам'яті не дали змоги запустити модель локально, тому було використано Llama-3.2-1B (не МоЕ). Також було внесено зміни в розрахунок параметрів: `logprobs` було нормалізовано (віднято середнє і поділено на стандартне відхилення) та обрізано викиди. Відсікання викидів спирається на гіпотезу, що людський текст часто має високий `kurtosis` лише через кілька дуже рідкісних слів (наприклад, специфічне ім'я чи помилку). Якщо відрізати нижні 2 % найменш ймовірних токенів (найбільші мінуси серед `logprob`), Human `kurtosis` впаде до нормального (близько 0), а `kurtosis` МоЕ-моделі залишиться високим через структурні аномалії. Результат:

- `ai_kurtosis_score > text_kurtosis_score`: 68 (56.6 %)
- `ai_peak_count > text_peak_count`: 38 (31.6 %)

- 6) Зміна моделі на ще більшу й сучаснішу Phi-3-mini-4k-instruct при збереженні логіки нормалізації та обрізки дала наступний результат:

- `ai_kurtosis_score > text_kurtosis_score`: 64 (53.3 %)
- `ai_peak_count > text_peak_count`: 37 (30.8 %)

- 7) Оскільки застосування нормалізації не дало результату, то її, разом з обрізанням, було прибрано з коду. Вирішено замість чистих `logprob` використовувати їх першу похідну. Тобто змінюємо логіку з

вимірювання kurtosis logprobs на вимірювання kurtosis зміни цих ймовірностей. Модель – Llama-3.2–1B-Instruct. Результат:

- ai_kurtosis_score > text_kurtosis_score: 66 (55 %)
- ai_peak_count > text_peak_count: 38 (31.6 %)

Отже, експеримент показує, що використання вирівняних (alignment) Instruct моделей є неефективним у ролі оцінювачів щільності. Згладжування розподілу ймовірностей через RLHF нівелює сигнал від перемикування експертів MoE.

8) Той же експеримент, що і № 7, але з Llama-3.2–1B, дав такі результати:

- ai_kurtosis_score > text_kurtosis_score: 54 (45 %)
- ai_peak_count > text_peak_count: 45 (37.5 %)

Крім того, у цьому експерименті суттєво впали значення kurtosis і вперше показник ($\text{text_kurtosis_score} \geq 2$): 15 перевищив ($\text{ai_kurtosis_score} \geq 2$): 10.

Отже, на цьому етапі відкидається гіпотеза, що MoE формуватимуть розподіли з важкими хвостами (високий kurtosis). Натомість показано, що Human текст демонструє вищий kurtosis через природну лексичну імпульсивність (високу burstiness) та використання рідкісних слів, що створює екстремальні викиди у logprobs. Тож формула (2.14) не буде використовуватися в гібридному методі. Проте залишається гіпотеза, що мультимодальність розподілу ймовірностей є математичним підписом MoE моделей. Аналіз щільності ядра (KDE) показав, що тексти III достовірно частіше формують багатопікові структури ($\text{ai_peak_count} > 2$), що відображає змішування локальних розподілів від різних активованих експертів.

Для перевірки другої частини гіпотези відріжемо певний відсоток рідкісних logprobs та введемо два додаткові параметри згладжування: ширину вікна та мінімальну висоту піка над фоном. Для отримання оптимальних значень параметрів використано класичний пошук за сіткою на спеціально створеному валідаційному наборі даних із 1200 Human і 1200 Instant AI текстів. Отримані результати щодо найкращих параметрів показано в табл. 3.12.

Таблиця 3.12. Тор-5 наборів параметрів для розділення MoE та Human текстів

Ширина вікна	Висота піка	% відсікання	TPR	FPR
0.15	0.10	0.0	0.005833	0.002500
0.25	0.05	1.0	0.000833	0.000000
0.25	0.05	3.0	0.001667	0.000833
0.20	0.10	1.0	0.000833	0.000000
0.20	0.08	0.0	0.000833	0.000000

Результати, наведені в табл. 3.12, показують, що оптимізатор не зміг знайти жодної закономірності, тобто неможливо використовувати кількість піків у тексті, знайдену за формулою KDE (2.13), для того, щоб розрізнити дані, написані людиною, і МоЕ моделлю.

Висновок: Проведений масштабний експеримент показав, що використання проксі-моделей не дає змоги визначити архітектурні особливості моделі (такі як маршрутизація токенів між експертами), які повністю нівелюються на рівні фінального текстового виводу. Проксі-моделі оцінюють такий текст як навіть більш природний, ніж середній людський. Оскільки задача розрізнення Human vs МоЕ є набагато простішою за розрізнення МоЕ vs Dense, то гіпотезу, висунуту у 2.1.2, спростовано. Перевірка довела, що гібридні детектори повинні фокусуватися на семантико-стилістичних аномаліях (як у випадку з Reasoning-моделями) або глобальній імовірнісній кривизні як у моделі типу Fast-DetectGPT[31], а не на пошуку мікроструктурних архітектурних артефактів.

3.4.2. Валідація характеру сліду III для Reasoning моделей

Іншою важливою гіпотезою, викладеною у 2.1.2, є гіпотеза про наявність певного особливого стилю в даних, згенерованих Reasoning моделями. У своєму базовому формулюванні гіпотеза має на увазі те, що Reasoning моделі

генерують текст шляхом побудови ланцюжка думок, де кожен роздум, з погляду інформаційної теорії, має спочатку найбільшу ентропію (нова ідея – багато нової інформації), потім помірну (розвиток ідеї – менше нової інформації) і, у кінці, на етапі формулювання висновків, – найменшу (сумаризація, мало нової інформації). Припускається, що такі хвилі ентропії мають зберегтися у фінальній відповіді моделі, хоча й у можливо більш згладженому вигляді. Наявність хвиль можна описати як певний ритм – наявність піків із певною періодичністю і виміряти за допомогою формул (2.15) і (2.16). Якщо Reasoning моделі дійсно мають свій спектральний підпис, це дасть змогу їх легше відрізнити від інших моделей (Instant/Dense) і людських текстів. Далі наведено ряд експериментів для перевірки даної гіпотези.

Експеримент № 1. У ролі даних для даного експерименту було використано тренувальну частину AI набору даних, описаного в 3.1.2. Датасет містить сумарно 13486 текстів, причому рівно половина – 6743 тексти створено Reasoning моделями, а інша половина Instant. Також, як додатковий захід безпеки, використано 1000 випадкових текстів з Human набору даних. В експерименті буде використано формули (2.15) і (2.16) для обчислення значень `period` і `score_freq`. Результати обчислення показано в табл. 3.13. З таблиці видно, що розподіли обох параметрів для різних типів моделей досить близькі, тобто характеристики ритму між Instant і Reasoning моделями теж схожі.

Таблиця 3.13. Розподіли періоду й частоти для Instant і Reasoning моделей

Metric	Instant		Reasoning	
	score_freq	period	score_freq	period
count	6743	6743	6743	6743
mean	1.998756	7.405755	2.053796	8.819602
std	0.363753	8.643349	0.394639	11.046623
min	1.076548	2.000000	1.162501	2.000000
25 %	1.743715	2.617216	1.788746	2.636364
50 %	1.953405	4.000000	1.992581	4.076923

75 %	2.198683	7.750000	2.255929	8.800000
max	4.378143	100.000000	6.070710	88.000000

Додатково потрібно перевірити розподіл параметрів за моделями, адже, наприклад, наявності однієї чи двох Instant моделей, що генерують дані з ритмами ентропії, може бути достатньо, щоб змістити характеристики всього розподілу. Тому, щоб зменшити вплив можливих викидів, спочатку дані буде усереднено на рівні моделі, а потім зведено в спільну статистику. Результат показано в табл. 3.14. Очікувано, екстремальні значення зникли (min і max стали ближче до 25 і 75 перцентилів), проте загалом розподіл особливо не змінився.

Таблиця 3.14. Розподіли періоду й частоти для Instant і Reasoning моделей із попереднім усередненням на рівні моделі.

Metric	Instant		Reasoning	
	score_freq	period	score_freq	period
count	6.000000	6.000000	9.000000	9.000000
mean	1.998736	7.405705	2.053756	8.818819
std	0.076394	0.503178	0.092417	1.999175
min	1.934063	6.673751	1.960867	7.019839
25 %	1.947673	7.079456	2.002557	7.735851
50 %	1.968634	7.482854	2.017022	8.127534
75 %	2.027752	7.810944	2.123431	9.006122
max	2.132940	7.929476	2.236409	13.750177

Висновок першого експерименту: лексична ентропія вимірює багатство словникового запасу, а не когнітивну структуру. Природна зміна складності лексики створює хибний "ритм", який визначається формулами (2.15) і (2.16) і є дуже схожим для обох типів моделей.

Експеримент № 2. Замінімо ентропію Шеннона на синтаксичну глибину речення. Результат застосування формули (2.15) до всього тексту може бути

зашумленим через можливу наявність артефактів на початку й у кінці тексту, тому застосуємо підхід ковзного вікна, щоб зменшити вплив артефактів. Перед застосуванням FFT нормалізуємо сигнал, віднявши від нього середнє. Шляхом аналізу всього масиву даних буде визначено оптимальний поріг $Score_{freq}$ для досягнення максимальної можливої роздільної здатності системи між Instant та Reasoning моделями.

Результати:

- Оптимальний знайдений поріг для $Score_{freq} = 1.850$.
- TPR (частка коректно виявлених Reasoning-моделей): 62.4 %
- FPR (хибні спрацювання на Instant-моделях): 57.6 %
- FPR на людських текстах: **48.8 %**

Висновок для другого експерименту: спектральний аналіз синтаксису на базі формули (2.15) не є надійним маркером Reasoning моделей. Експеримент доводить, що Reasoning моделі залишають свій характерний ритм у прихованому блоці роздумів, який не передається користувачу. Фінальна генерація моделі містить тільки чистий текст, що не містить періодичної структури.

Експеримент №3. Зміна парадигми – перехід від вимірювання спектральних характеристик до стилOMETричного аналізу. Оригінальну гіпотезу, висунуту у 2.1.2, було уточнено: Reasoning моделі мають свій власний стиль, відмінний від Instant моделей, проте він характеризується не наявністю внутрішнього ритму, а специфічним, чітким, зв'язаним та структурованим текстом. В основі уточнення гіпотези лежить припущення про те, що результат роботи Reasoning моделі не є прямою відповіддю моделі на prompt користувача, а швидше резюме її попередніх роздумів. Таке резюме може мати складнішу і глибшу синтаксичну структуру й дещо “відшліфовану” лексику, адже модель нічого не вигадує сама на цьому етапі, а просто заповнює певний шаблон відповіді шляхом сумаризації побудованого раніше ланцюжка думок.

Для визначення такої дистильованої версії внутрішнього монологу моделі було запропоновано 11 слабких ознак, що мають уловлювати різні аспекти надмірно структурованого тексту Reasoning моделей. На рис. 3.2–3.3 наведено приклад роботи моделі на різних текстах: тексті, написаному людиною, текстах, створеному Instant і Reasoning моделями.

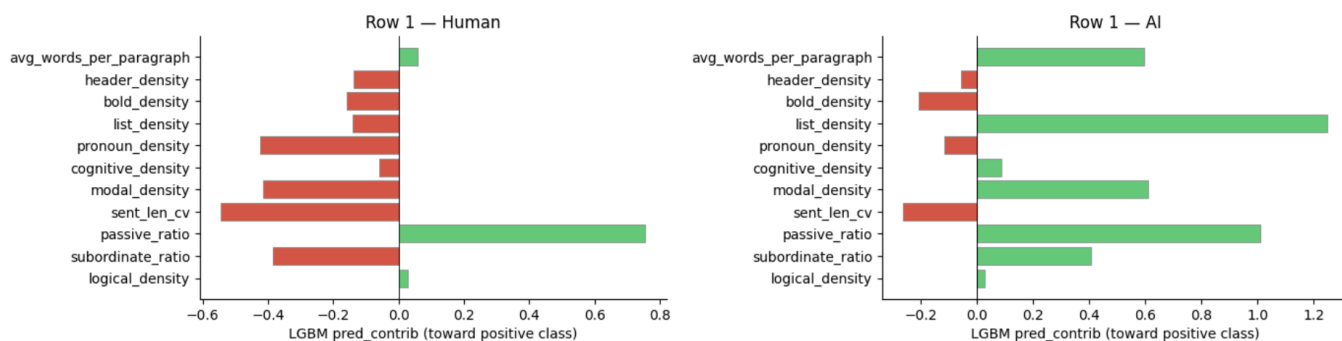


Рисунок 3.2 Приклад роботи Reasoning класифікатора на тексті, створеному людиною, і Reasoning моделлю.

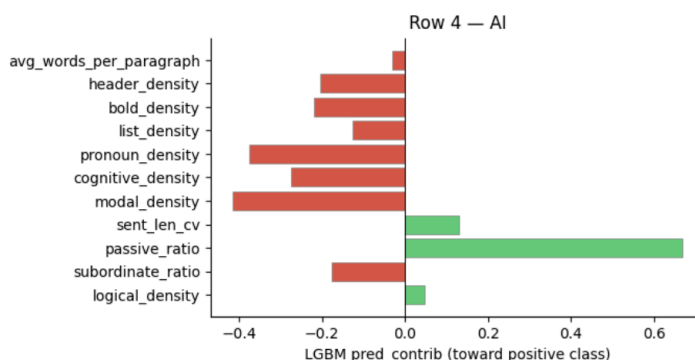


Рисунок 3.3 Приклад роботи Reasoning класифікатора на тексті, створеному Instant моделлю.

На основі логістичної регресії та цих 11 ознак необхідно побудувати простий класифікатор, щоб зрозуміти, чи мають вони здатність розділити Reasoning та Instant моделі. Як набір даних для тренування та оцінки моделі буде використано той же датасет, що й у попередніх двох експериментах. Випадкові 80% даних буде використано як тренувальні, а решта 20% - як валідаційні. Метрики отриманої моделі наведено в таблиці 3.15. Подальша

заміна моделі на ансамбль дерев рішень Light GBM [32] дала змогу покращити F1 до 80 %, повні метрики наведено в таблиці 3.16.

Таблиця 3.15. Метрики точності LogisticRegression Reasoning класифікатора.

Category	Precision	Recall	F1-Score	Support
Instant/Human	0.72	0.65	0.68	1549
Reasoning	0.64	0.70	0.67	1349

Таблиця 3.16. Метрики точності LightGBM Reasoning класифікатора.

Category	Precision	Recall	F1-Score	Support
Instant/Human	0.81	0.80	0.80	1549
Reasoning	0.80	0.81	0.80	1349

Висновок: Проведена практична перевірка гіпотези щодо наявності особливого відбитка Reasoning моделей підтвердила присутність у текстах такого сигналу, який можна визначити з досить високою точністю ($F1=0.80$) досить простим класифікатором, тож у цій частині гіпотезу підтверджено. Другу частину гіпотези, що стверджувала наявність ритму у відповідях Reasoning моделей, було спростовано проведеними експериментами. Було наочно показано відсутність здатності розділення моделей на Reasoning та Instant за допомогою методів обробки сигналів (FFT) для детекції прихованого ланцюжка думок, оскільки періодичність міркувань ланцюжка думок моделі не транлюється у фінальний текст. Розроблений за результатами експериментів класифікатор успішно розпізнає профіль Reasoning моделі, виступаючи надійним компонентом (слабким класифікатором) в архітектурі гібридного детектора.

3.4.3. Експериментальна перевірка моделі Точки зламу

У пункті 2.2.2 запропоновано концептуальну модель Точки зламу. Ця модель заснована на гіпотезі про наявність взаємозв'язку між семантичною цілісністю I_{sem} та L_{trace} (ступенем втрати AI Trace через внесення модифікацій у текст). Гіпотеза стверджує, що для досягнення суттєвих значень L_{trace} , що і є ціллю будь-якої атаки, необхідно внести велику кількість змін у текст, що вимірюються через показник RR (коефіцієнт заміни). Тоді Точкою зламу називається таке значення RR , при якому згенерований LLM текст перестає визначатися детектором, тобто AI score для цього тексту стає нижче за певне порогове значення. Причому гіпотеза стверджує, що при підвищенні RR буде відбуватися падіння I_{sem} . Метою даного експерименту є побудова реального графіка залежності імовірності детекції P_{det} та I_{sem} від RR для одного з SOTA детекторів і порівняння його з теоретичним графіком, зображеним на рис. 2.1. Додатковою метою експерименту є доведення підвищеної робастності гібридного методу, умовою чого є зміщення Точки зламу праворуч (у зону вищих значень RR) на графіку, представленим на рис. 2.1.

Таблиця 3.17. Аналіз розподілу RR для трансформаційних атак T5

	T5 Level 1	T5 Level 2	T5 Level 3	T5 Level 4	T5 Level 5
count	1500	1500	1500	1500	1500
mean	0.056041	0.085984	0.223347	0.232042	0.359944
std	0.024260	0.030512	0.044916	0.048875	0.051130
min	0.011013	0.021645	0.100515	0.113186	0.224359
25 %	0.039987	0.066667	0.193084	0.199676	0.326888
50 %	0.053435	0.082959	0.219730	0.227706	0.356625
75 %	0.066988	0.099542	0.247596	0.257782	0.385001
max	0.325843	0.481982	0.623742	0.658730	0.796923

Для проведення даного експерименту буде використано датасет із набором атак В, описаний у підрозділі 3.2. Набір містить як тексти, згенеровані різними LLM, так і 9 варіантів їх модифікації. Варіанти модифікації відсортовано від найслабшого до найсильнішого. Для кожного варіанту модифікації пораховано значення RR ,

розподіл яких наведено в табл. 3.17 і табл. 3.18. Для підрахунку RR будемо використовувати формулу (2.18). Попри те, що трансформації тексту відбувалися шляхом встановлення максимальних семантичних відхилень без обмеження і контролю кількості змін на лексичному рівні, середні значення RR показують стабільне зростання під час переходу від одного варіанту модифікації до іншого. Це вказує на наявність прямої залежності між ступенем семантичної зміни тексту й кількістю заміन на символічному рівні.

Таблиця 3.18. Аналіз розподілу RR для трансформаційних атак DIPPER

	DIPPER Level 1	DIPPER Level 2	DIPPER Level 3	DIPPER Level 4
count	162	147	156	168
mean	0.620514	0.717172	0.911304	0.973896
std	0.068922	0.064275	0.036238	0.011351
min	0.470763	0.586269	0.805970	0.932515
25 %	0.573756	0.672941	0.889585	0.967213
50 %	0.614895	0.708149	0.912011	0.975236
75 %	0.646552	0.749718	0.937356	0.982024
max	0.918605	0.964637	0.982759	0.996491

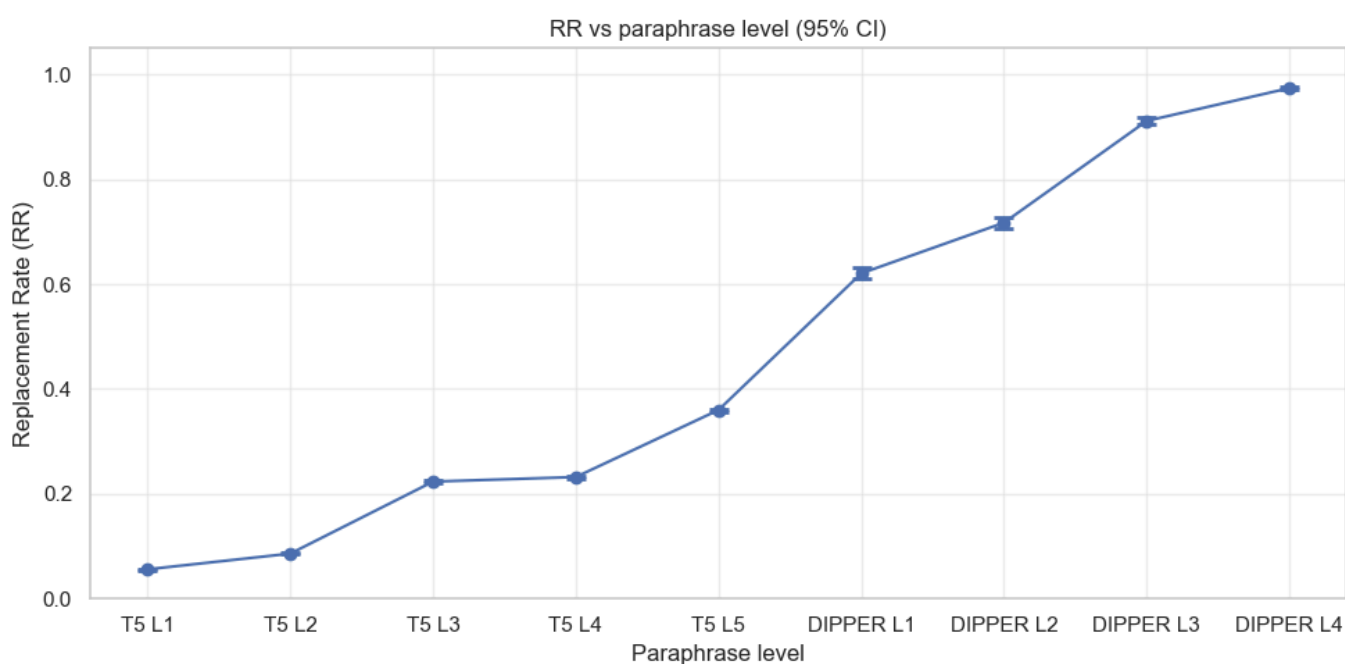


Рисунок 3.4 Вплив різних рівнів перефразування на Replacement Rate

У вигляді графіка зміна RR залежно від етапу модифікації зображена на рис. 3.4. Наявні дані чітко відповідають умовам експерименту. З цікавих спостережень варто вказати на досить незначну різницю в RR між T5 Level 3 і T5 Level 4. Зважаючи на те, що T5 Level 4 являє собою повторне перефразування T5 Level 3 з тими самими параметрами, такий результат є очікуваним, адже модель T5 було натреновано з фокусом на збереження семантики при створенні перефразованих текстів.

Наступним кроком дослідження є вимірювання падіння семантичної близькості I_{sem} . Вимірювання проводилося за формулою (2.17): кожен текст було розбито на речення, для кожного речення за допомогою моделі all-MiniLM-L6-v2 [33] було отримано векторне представлення. Таким чином тексти було представлено у вигляді матриць. Потім було обраховано попарну косинусну подібність між текстами. Модель all-MiniLM-L6-v2 було обрано для створення ембедінгів, оскільки вона є однією з найефективніших sentence-transformer моделей, здатних до перетворення речень і параграфів у числову форму. Середні значення косинусної подібності I_{sem} між набором оригінальних AI текстів і наборами текстів кожного з рівнів модифікації наведено в табл. 3.19 та табл. 3.20.

Таблиця 3.19. Аналіз розподілу I_{sem} для трансформаційних атак T5

	T5 Level 1	T5 Level 2	T5 Level 3	T5 Level 4	T5 Level 5
count	1500	1500	1500	1500	1500
mean	0.980800	0.974058	0.947292	0.942499	0.869178
std	0.019169	0.023849	0.032886	0.038009	0.059030
min	0.743413	0.741898	0.649010	0.571378	0.460521
25 %	0.976840	0.968340	0.934271	0.929378	0.843037
50 %	0.985692	0.980893	0.954881	0.951915	0.880425
75 %	0.991034	0.987906	0.969350	0.966829	0.910609
max	0.999662	0.998362	0.992318	0.993638	0.974524

У вигляді графіка зміна I_{sem} залежно від етапу модифікації зображена на рис.

3.5. На графіку видно, що зміна I_{sem} залишається досить незначною між рівнями T5 Level 1 – Level 4. Суттєве падіння відбувається на етапі T5 Level 5 і пов'язане з видаленням випадкових слів. Інше цікаве спостереження це близькість I_{sem} для T5 Level 5 і DIPPER Level 1, що підтверджує очікування щодо природи парафразера DIPPER. У цілому графік поводить себе саме так, як і очікувалося в підрозділі 2.2.

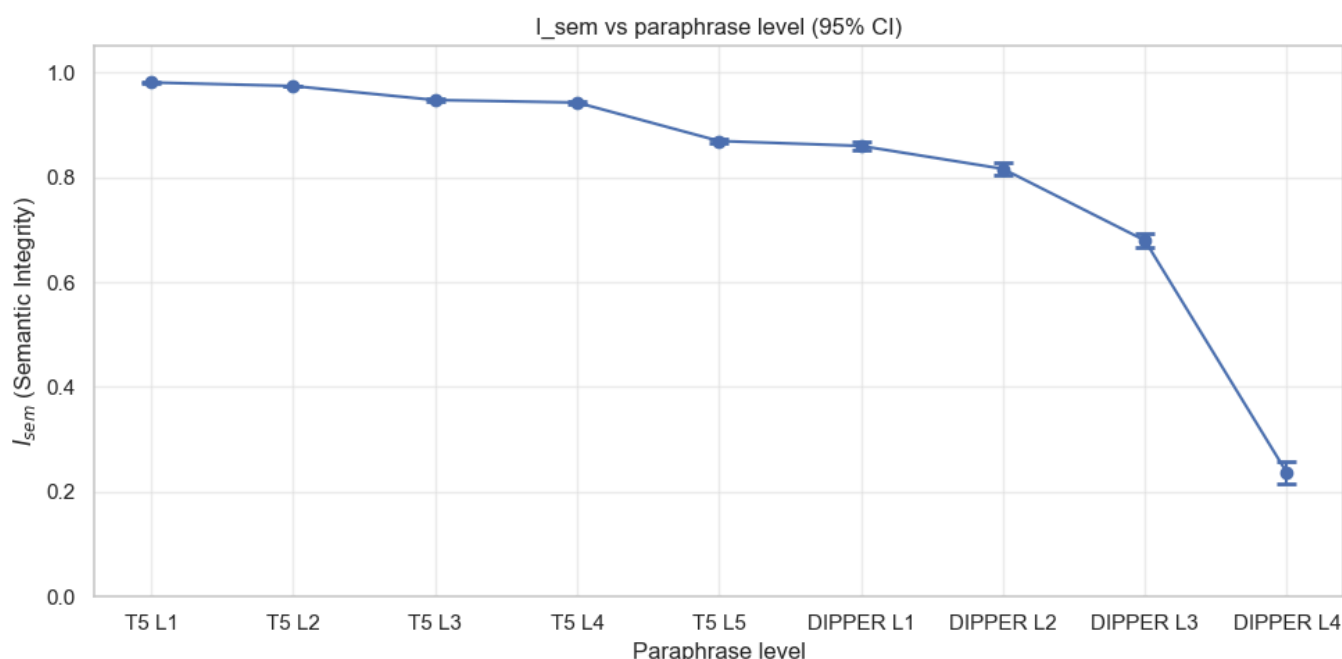


Рисунок 3.5 Вплив різних рівнів перефразування на I_{sem}

Таблиця 3.20. Аналіз розподілу I_{sem} для трансформаційних атак DIPPER

	DIPPER Level 1	DIPPER Level 2	DIPPER Level 3	DIPPER Level 4
count	162	147	156	168
mean	0.859800	0.816076	0.679502	0.236560
std	0.054061	0.072437	0.088962	0.143879
min	0.700900	0.566007	0.333478	-0.087694
25 %	0.827927	0.773668	0.624312	0.134066
50 %	0.870611	0.828669	0.687011	0.229250
75 %	0.894654	0.866499	0.747994	0.340372

max	0.964065	0.949801	0.901336	0.599909
-----	----------	----------	----------	----------

Наступним кроком необхідно обчислити AI Score для набору даних В. Серед доступних для вільного використання SOTA методів, що можливо запустити на локальних обчислювальних потужностях для опрацювання всього датасета В (~10000 текстів) за прийнятний час (10–12 годин), підходить лише система Fast-DetectGPT [31]. Особливістю роботи цієї системи є те, що вона видає на вихід z-value (z-оцінку) – значення, що характеризує, наскільки оцінювані дані відрізняються від типового тексту, згенерованого LLM. Автори моделі в роботі [31] пропонують механізм проєкції отриманих результатів на ймовірності (тобто AI score), використовуючи нормальний гаусівський розподіл. Результати роботи моделі Fast-DetectGPT на наборі даних В після такої трансформації наведено в табл. 3.21 й табл. 3.22. Детальний розподіл AI Score за конкретними моделями наведено в додатку Д. Зміну середнього AI Score залежно від типу атаки показано на Рис. 3.6.

Таблиця 3.21. Аналіз розподілу AI score для трансформаційних атак T5

	ai_prob	T5_lv1_prob	T5_lv2_prob	T5_lv3_prob	T5_lv4_prob	T5_lv5_prob
count	1500	1500	1500	1500	1500	1500
mean	0.490400	0.455587	0.429389	0.389106	0.377477	0.346463
std	0.362938	0.350116	0.347172	0.325188	0.325973	0.300908
min	0.100368	0.100368	0.100368	0.100368	0.100368	0.100368
25 %	0.137023	0.129602	0.122490	0.120460	0.116116	0.116695
50 %	0.355532	0.309052	0.262074	0.221901	0.197355	0.193479
75 %	0.928922	0.837432	0.795416	0.646555	0.631001	0.502343
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

Таблиця 3.22. Аналіз розподілу AI score для трансформаційних атак DIPPER

	DIPPER Level 1	DIPPER Level 2	DIPPER Level 3	DIPPER Level 4
count	162	147	156	168
mean	0.568392	0.590947	0.789904	0.942518
std	0.327620	0.319601	0.256094	0.137888

min	0.100369	0.100527	0.109737	0.148513
25 %	0.241841	0.291785	0.635240	0.970263
50 %	0.594884	0.614598	0.919491	0.996596
75 %	0.898563	0.916157	0.986430	0.999795
max	0.999947	0.999986	0.999998	1.000000

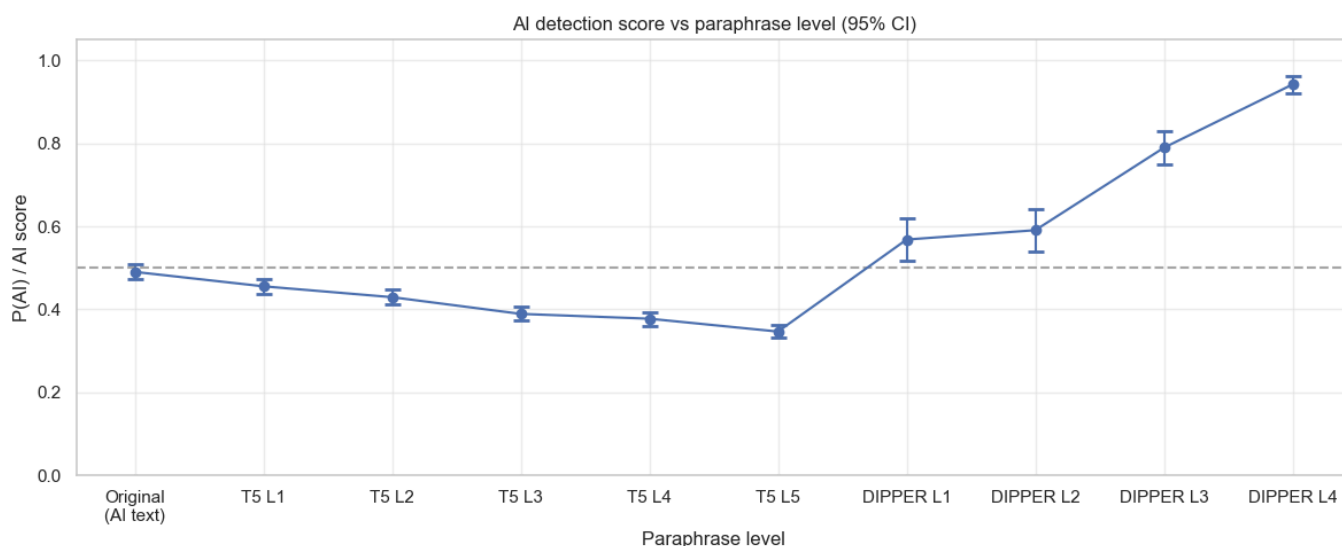


Рисунок 3.6 Вплив різних рівнів перефразування на AI score

AI score характеризує ймовірність того, що текст є згенерованим, тому не варто використовувати його середнє значення замість P_{det} , який описано в пункті 2.2.2. Ключовою різницею є бінарна природа P_{det} проти імовірнісної природи AI score. По суті, для обчислення P_{det} (або Recall) необхідно перейти до задачі бінарної класифікації. Стандартним способом переходу від отриманих z-value до бінарних міток є застосування порогового значення до z-value. Для даного експерименту буде використано порогове значення (threshold) = 0.5, усі записи, у яких $z\text{-value} \geq \text{threshold}$, отримують мітку 1 (AI), а всі інші 0 (Human). У випадку застосування системи в реальному житті значення threshold повинно обиратися залежно від бажаного значення FPR. Графік, що показує Recall системи Fast-DetectGPT залежно від типу перефразування, показано на Рис. 3.7, а вплив на можливість визначення кожної конкретної моделі показано на Рис. 3.8.

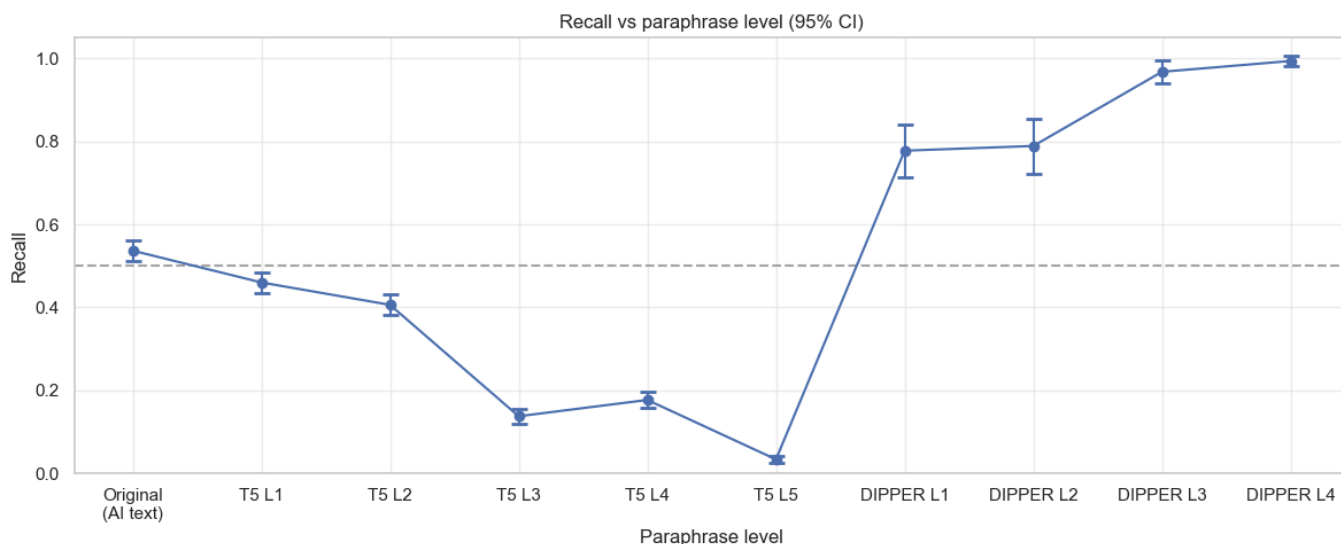


Рисунок 3.7 Вплив різних рівнів перефразування на Recall

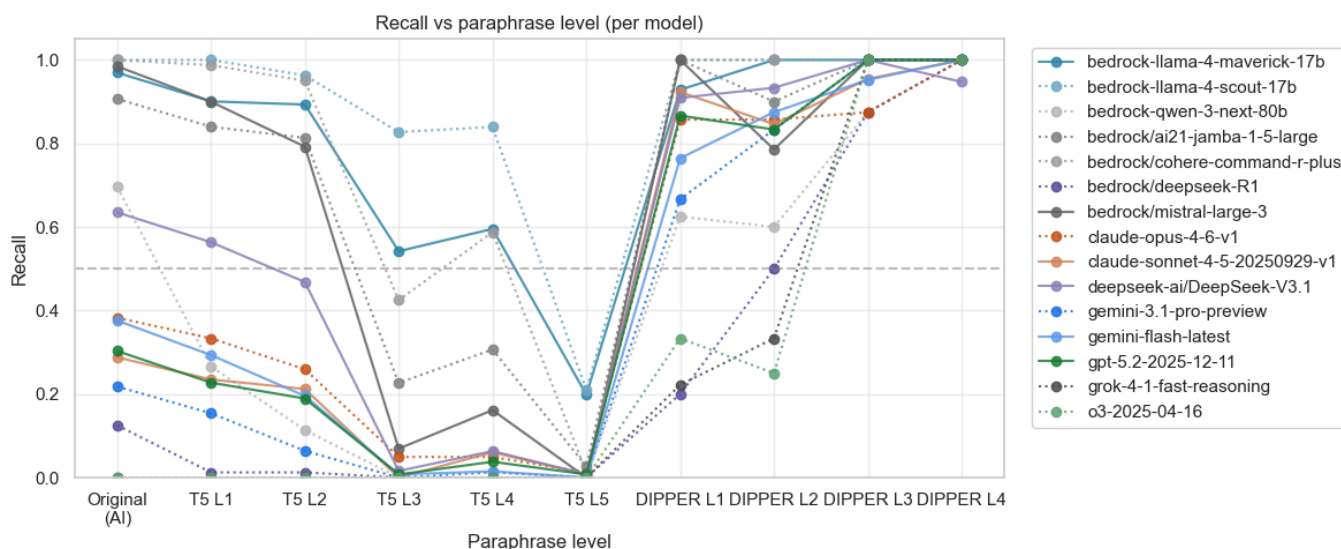


Рисунок 3.8 Вплив різних рівнів перефразування на Recall для кожної моделі

На базі проведеного експерименту, фінальна модель Точки зламу для детектора Fast-DetectGPT буде виглядати як це зображено на Рис. 3.9. Права границя Області 1 відповідає точці, де Recall (P_{det}) системи стає нижче 0.5, це і є Точка зламу, як описано в Розділі 2. Права границя Області 2 відповідає місцю, де I_{sem} стає нижче за 0.9. На графіку видно, що Точка зламу знаходиться досить близько до лівої границі Області 1 (RR=0.03), тобто навіть незначних змін у тексті достатньо, щоб Fast-DetectGPT AI детектор почав виявляти менше ніж 50 % згенерованих текстів.

Це свідчить як про слабку стійкість детектора до трансформаційних атак, так і про складність вибірки даних, використаної для датасета В. Вибірка складається з текстів останніх флагманських LLM моделей, тому навіть до застосування трансформаційних атак детектор був у змозі коректно розпізнати лише близько 54 % згенерованих текстів.

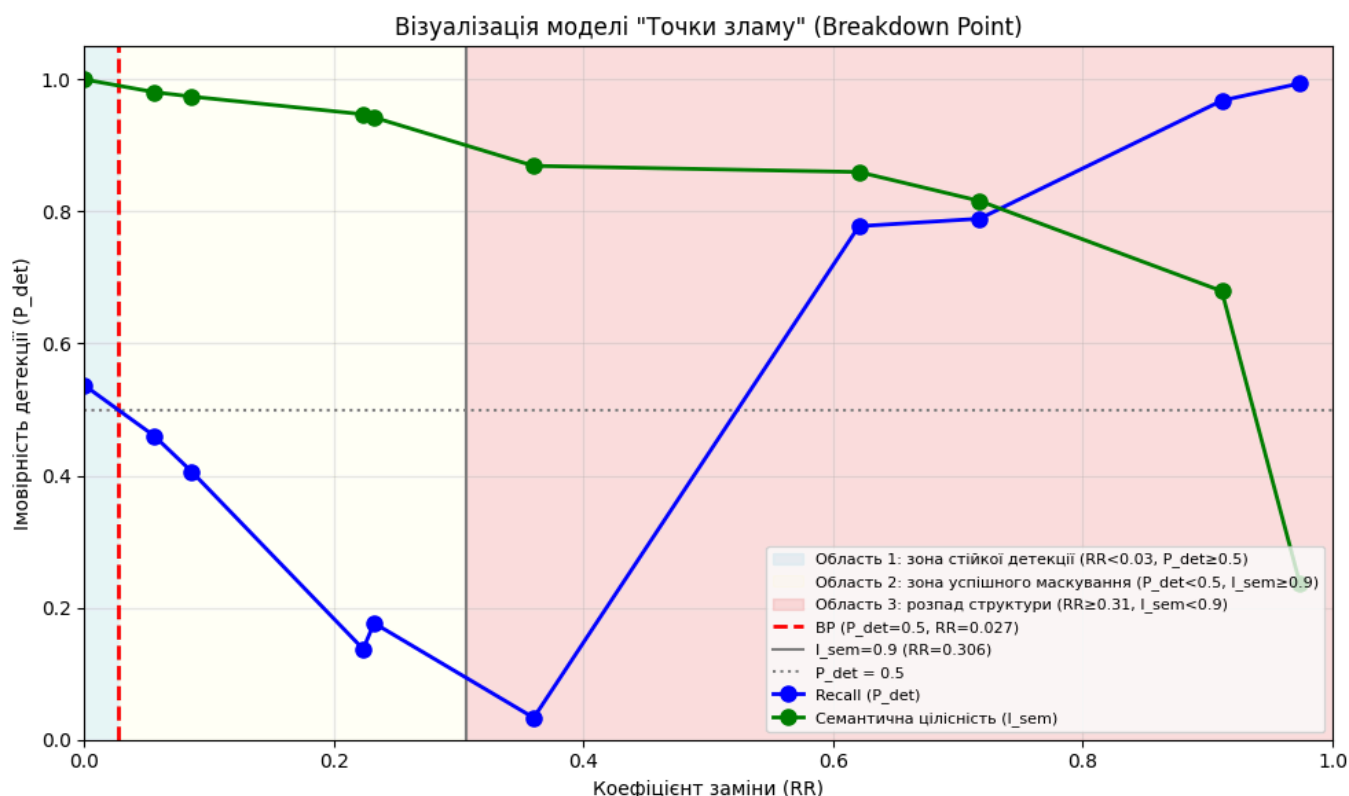


Рисунок 3.9 Модель Точки зламу для детектора Fast-DetectGPT

Цікавою є поведінка детектора в Області 3 (червона зона графіка). При вході в цю зону Recall системи падає майже до 0, разом із помітним падінням I_{sem} ; це відбувається в точці, що відповідає атаці T5_lvl5. Така поведінка точно відповідає очікуванням, закладеним у теоретичну модель стійкості детекції. З подальшим ростом RR, що досягається шляхом використання системи DIPPER, I_{sem} продовжує падати, але Recall починає рости. На перший погляд, це виглядає контрінтуїтивно, адже відповідно до теоретичної моделі очікувалося як мінімум утримання Recall на поточному рівні, а більш реалістичним сценарієм є його зниження. Причиною підвищення Recall на графіку є те, що модель DIPPER також є LLM за своєю

природою, тож, прибираючи AI trace оригінальної моделі, DIPPER додає до тексту свій власний слід. Це добре видно на Рис. 3.8, де більшість із моделей-генераторів уже після атаки DIPPER Level 1 отримали значення Recall набагато вищі, ніж мали для AI текстів до будь-яких атак. Тобто система Fast-DetectGPT здатна визначати DIPPER краще за інші флагманські моделі, відповідно, чим сильніший вплив DIPPER, тим вищий Recall (аж до 100 %).

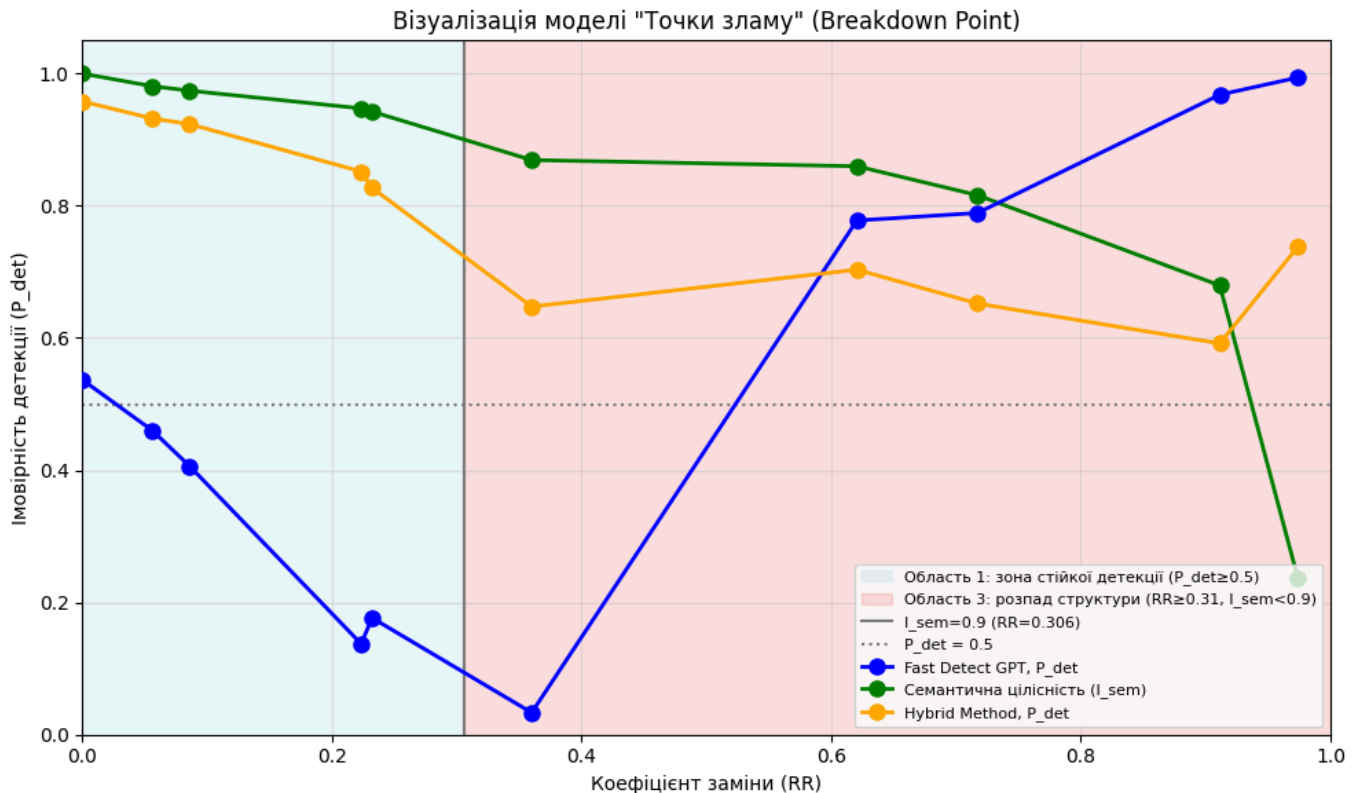


Рисунок 3.10 Модель Точки зламу для детектора на базі гібридного методу

Графік Точки зламу для гібридного методу зображено на Рис. 3.10. Помаранчева лінія – Recall (P_{det}) гібридного методу, синя лінія – Recall Fast-DetectGPT. Видно, що Точка зламу для гібридного методу зсунута праворуч і не досягається навіть при значеннях $RR > 0.9$, що підтверджує суттєво вищу робастність гібридного методу. Також варто відзначити, що запропонований метод детекції цілком задовольняє і навіть перевищує необхідні вимоги щодо боротьби з модифікацією даних – успішним випадком щодо демонстрації стійкості до модифікацій було б зменшення жовтої зони порівняно з іншим SOTA детектором, у

реальності ж жовта зона повністю зникла, а Точка зламу перемістилася глибоко в червону зону, що свідчить про відсутність сенсу внесення такого типу модифікацій для обходу детекції.

Отже, у цьому пункті було експериментально доведено, що теоретична модель Точки зламу працює саме так, як це було описано в підрозділі 2.2, та показано, що на даних із датасета В розроблений гібридний метод демонструє суттєво вищу робастність порівняно з SOTA детектором Fast-DetectGPT.

3.5. Висновки до розділу 3

У цьому розділі проведено експериментальну валідацію розроблених математичних моделей та гібридного методу ідентифікації згенерованих текстів на спеціально сформованих масивах даних (включаючи тексти МоЕ та Reasoning-моделей, а також тексти після трансформаційних атак парафразерів T5, DIPPER та комерційних систем маскування). Вирішено Завдання 6, виконано елементи практичної частини Завдання 2 та отримано такі науково-практичні результати:

1. Удосконалено математичну модель оцінки робастності систем детекції сліду ІІІ, яка (на відміну від суто теоретичних підходів без чітко визначених меж стійкості) шляхом експериментальної верифікації Точки зламу на масштабованих датасетах із застосуванням парафразерів дає змогу точно кількісно визначати вразливість систем в умовах невизначеності. Експериментально доведено, що для SOTA-детектора (Fast-DetectGPT) Точка зламу настає вже за мінімальних змін (коефіцієнт заміни $RR = 0.03$), тоді як результати гібридного методу підтверджують суттєве підвищення стійкості детекції з уникненням досягнення Точки зламу навіть при зміщенні праворуч до $RR > 0.9$. Це підтверджує достовірність математичного взаємозв'язку між втратою семантичної цілісності I_{sem} та ймовірністю виявлення P_{det} , що становить зміст Наукової новизни 4.

2. Уперше розроблено (у частині експериментального підтвердження ефективності) гібридний метод ідентифікації автоматично згенерованих природномовних текстів. На відміну від наявних рішень, що демонструють критичне падіння повноти детекції в Області 2 (зона успішних атак) під час трансформаційних атак, гібридний метод, завдяки архітектурі каскадного ансамблю та аналізу міжрівневої узгодженості, дає змогу повністю нівелювати зону успішного маскуванню згенерованого контенту. Практично доведено, що гібридний метод зберігає високу ймовірність детекції навіть після критичного рівня структурної зміни тексту $RR > 0.9$ та падіння семантичної цілісності $I_{sem} < 0.5$, що робить застосування атак перефразуванням недоцільним і підтверджує високу стійкість системи, заявлену в Науковій новизні 1.
3. Експериментально досліджено та адаптовано методи моделювання та виділення сліду ШІ для сучасних архітектур MoE та Reasoning. На відміну від підходів, що фокусуються на мікроструктурних артефактах генерації через оцінку щільності ядра KDE чи спектральний аналіз ритму FFT, запропоновані методи дають змогу надійно ідентифікувати когнітивний слід моделі. Такий результат досягається шляхом експериментального спростування гіпотез про мультимодальність розподілу токенів MoE та періодичність ентропії Reasoning-моделей і переходу до стилOMETричного аналізу надмірно структурованого тексту. Розроблений на цій базі класифікатор досягає точності $F1 = 0.80$ у розрізненні Reasoning та Instant генерацій, виступаючи ефективним компонентом гібридного детектора. Це розширює розуміння природи сліду ШІ та відповідає виконанню практичної частини Завдання 2.

Список використаних джерел до розділу 3

1. Gokaslan, Aaron, et al. "Openwebtext corpus." Mar. 2019
2. Fan, Angela, Mike Lewis, and Yann Dauphin. "Hierarchical neural story generation." *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2018.
3. Jin, Qiao, et al. "Pubmedqa: A dataset for biomedical research question answering." *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*. 2019.
4. Narayan, Shashi, Shay B. Cohen, and Mirella Lapata. "Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization." *Proceedings of the 2018 conference on empirical methods in natural language processing*. 2018.
5. Weischedel, Ralph, et al. *OntoNotes Release 5.0*. Borealis, 2022.
6. Guo, Mandy, et al. "Wiki-40B: Multilingual language model dataset." *Proceedings of the Twelfth Language Resources and Evaluation Conference*. 2020.
7. Liu, Yikang, et al. "ArguGPT: evaluating, understanding and identifying argumentative essays generated by GPT models." *arXiv preprint arXiv:2304.07666* (2023).
8. Solaiman, Irene, et al. "Release strategies and the social impacts of language models." *arXiv preprint arXiv:1908.09203* (2019).
9. Weiss, Todd R. "OpenAI GPT-3 Waiting List Dropped as GPT-3 Is Fully Released for Developer and Enterprise Use." *HPCwire*, 18 Nov. 2021, www.hpcwire.com/aiwire/2021/11/18/openai-gtp-3-waiting-list-is-gone-as-gtp-3-is-fully-released-for-use/. Accessed 15 Feb. 2026.
10. Crossley, Scott A., et al. "A large-scale corpus for assessing written argumentation: PERSUADE 2.0." *Assessing Writing* 61 (2024): 100865.

11. Holmes, Langdon, et al. "PIILO: an open-source system for personally identifiable information labeling and obfuscation." *Information and Learning Sciences* 124.9-10 (2023): 266-284.
12. Guo, Mandy, et al. "Wiki40B: Multilingual Wikipedia Dataset." *Hugging Face*, 2020, huggingface.co/datasets/google/wiki40b. Accessed 18 Feb. 2026.
13. Crossley, Scott, et al. "PERSUADE Corpus 2.0." *GitHub*, 2023, github.com/scrosseye/persuade_corpus_2.0. Accessed 18 Feb. 2026.
14. Burleigh, Lauryn, et al. "PIILO Dataset." *Kaggle*, 2024, www.kaggle.com/datasets/lburleigh/piilo-dataset. Accessed 18 Feb. 2026.
15. Anthropic. "Prompt Engineering Interactive Tutorial." *GitHub*, 2024, github.com/anthropics/prompt-eng-interactive-tutorial. Accessed 15 Feb. 2026.
16. "Amazon Bedrock." *Amazon Web Services*, 2026, aws.amazon.com/bedrock/. Accessed 22 Feb. 2026.
17. BerriAI. *LiteLLM*. 2024, <https://www.litellm.ai/>. Accessed 22 Feb. 2024.
18. Raffel, Colin, et al. "Exploring the limits of transfer learning with a unified text-to-text transformer." *Journal of machine learning research* 21.140 (2020): 1-67.
19. Vamsi. "Vamsi/T5_Paraphrase_Paws." *Hugging Face*, huggingface.co/Vamsi/T5_Paraphrase_Paws. Accessed 24 Feb. 2026.
20. Krishna, Kalpesh, et al. "Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense." *Advances in neural information processing systems* 36 (2023): 27469-27500.
21. "Humanize AI Text." *Phrasly AI*, Phrasly LLC, 2026, business.phrasly.ai/. Accessed 22 Feb. 2026..
22. *StealthGPT*. StealthGPT, <https://www.stealthgpt.ai/>. Accessed 22 Feb. 2026.
23. *AIHumanize*. AIHumanize, <https://aihumanize.com/>. Accessed 22 Feb. 2026.
24. *Undetectable AI*. Undetectable AI, <https://undetectable.ai/>. Accessed 22 Feb. 2026.
25. "AI Writing." *Turnitin*, 2026, www.turnitin.com/solutions/topics/ai-writing/. Accessed 14 Mar. 2026.
26. *Originality.ai*, 2026, originality.ai/. Accessed 14 Mar. 2026.

27. "Receiver operating characteristic." *Wikipedia*, Wikimedia Foundation, 20 Feb. 2026,
en.wikipedia.org/wiki/Receiver_operating_characteristic#Area_under_the_curve.
Accessed 24 Feb. 2026.
28. "Precision and recall." *Wikipedia*, Wikimedia Foundation,
en.wikipedia.org/wiki/Precision_and_recall. Accessed 24 Feb. 2026.
29. "False positive rate." *Wikipedia*, Wikimedia Foundation,
en.wikipedia.org/wiki/False_positive_rate. Accessed 24 Feb. 2026.
30. Guo, Daya, et al. "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning." *arXiv preprint arXiv:2501.12948* (2025).
- 31.1x Bao, Guangsheng, et al. "Fast-detectgpt: Efficient zero-shot detection of machine-generated text via conditional probability curvature." *arXiv preprint arXiv:2310.05130* (2023).
32. "LightGBM." *Wikipedia*, Wikimedia Foundation, en.wikipedia.org/wiki/LightGBM.
Accessed 24 Feb. 2026.
33. "all-MiniLM-L6-v2." *Hugging Face*, 2021,
huggingface.co/sentence-transformers/all-MiniLM-L6-v2. Accessed 15 Mar. 2026.

РОЗДІЛ 4

ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА ВАЛІДАЦІЯ ГІБРИДНОГО МЕТОДУ ІДЕНТИФІКАЦІЇ СЛІДУ ШІ У СКЛАДНИХ УМОВАХ

Основною метою цього розділу є практична реалізація розробленого гібридного методу та проведення всебічного оцінювання його ефективності та стійкості в умовах трансформаційних атак та невизначеності. На базі теоретичних моделей сліду ШІ, викладених у Розділі 2, та алгоритмічного опису гібридного методу і концепції Точки зламу, викладених там же, у цьому розділі здійснимо побудову програмного комплексу, що реалізує поєднання всіх трьох рівнів сліду ШІ. Створений програмний комплекс проходить експериментальну валідацію на основі метрик та з використанням даних, описаних у Розділі 3. Для реалізації вказаної мети необхідно виконати такі завдання:

- 1) Експериментальне доведення переваги системи за показниками робастності (Recall Under Attack) та рівня успішності атак (ASR) у сценаріях активного маскуванню тексту через глибоке перефразування та структурні модифікації.
- 2) Валідація статистичних гарантій методу індуктивного конформного передбачення (ICP) для забезпечення надійності класифікації в умовах невизначеності джерела та аналізу даних поза навчальним розподілом (OOD).
- 3) Опис модульної архітектури програмного комплексу, включаючи реалізацію блоків каскадного ансамблю класифікаторів, механізму адаптивного зважування та ICP.
- 4) Практична апробація інтерпретаційної моделі обґрунтування через генерацію профілів атрибуції та побудову інтерактивних Карт доказів для кінцевого користувача.

Отримані в розділі результати забезпечують практичне підтвердження наукової новизни щодо підвищеної точності та робастності гібридного методу (HN1), який дає змогу провести зсув Точки зламу до високих значень параметрів

модифікації ($RR > 0.9$), досягаючи критичних меж семантичної цілісності тексту. Створення системи інтерпретації прийнятих рішень (НН2) та її валідація шляхом алгоритмічної генерації профілів атрибуції та інтерактивних карт доказів, реалізує перехід від концепції чорної скриньки до прозорого аудиту ознак. Запропоновані підходи щодо розвитку робастності детекції в умовах агресивного маскування та епістемічної невизначеності (НН3) отримали практичне підтвердження через валідацію фреймворку ICP на OOD-даних. Емпірично доведено адекватність концептуальної моделі Точки зламу (НН4) та її здатність прогнозувати чутливість детектора до трансформаційних атак.

Запропонований гібридний метод становить собою готовий до інтеграції інструментарій для боротьби зі зловживаннями ІІІ та зменшення його використання у таких сферах, як генерація фейкових новин, маніпулювання, соціальна інженерія, дезінформація, порушення академічної доброчесності, та в інших сценаріях безвідповідального використання ІІІ.

4.1. Аналіз ефективності та робастності гібридного методу

4.1.1. Базова точність та порівняння із SOTA на стаціонарних даних

Цей підрозділ присвячено питанню аналізу якості гібридного методу на звичайних чистих датасетах без модифікації даних та застосування трансформаційних атак. Основним джерелом даних для такого аналізу виступає тестова частина набору даних, описаного в підрозділі 3.1, - всього приблизно 3300 текстів, створених людьми, та приблизно 3300 текстів, створених різними моделями ІІІ відповідно до методики, описаної в пункті 3.1.2. У ролі способів оцінювання буде використано метрики, описані в підрозділі 3.2, конкретно AUC-ROC, Recall, FPR та F1. У деяких випадках буде наведено Precision, оскільки його використання необхідне для обчислення F1. Крім того, для гібридного методу буде обчислено $\text{Recall}@1\%FPR$, оскільки підтримка такого режиму роботи є однією з ключових

вимог для застосування детектора в областях, де критично важливо мати наднизький FPR.

Таблиця 4.1 Оцінки гібридного методу на чистих даних

Метрика	Значення
Precision	0.8813
Recall	0.9624
FPR	0.1296
F1	0.92

Шляхом комп'ютерного експерименту доведено, що запропонований гібридний метод на тестовій частині чистих даних (датасет 4.1, що складається з 6588 текстів, датасет містить однакову кількість AI та Human) досягає показника $F1=0.92$ (див. табл. 4.1), а форму ROC кривої наведено на рис. 4.1. Також на ROC кривій відмічено декілька ключових, з погляду FPR, точок; найцікавішою з них є точка, де $FPR=1\%$, TPR системи в цій точці дорівнює .889, це і є значення $Recall@1\%FPR$ для гібридного методу.

Варто зазначити, що набір даних, що використовується в даному тесті, і особливо його AI частина, є досить складним, оскільки містить дані, згенеровані 15 різними сучасними моделями з використанням різних технік генерації та різних архітектур LLM. Тому $F1=0.92$ є надзвичайно високим показником для таких складних даних. З іншого боку, те, що гібридний метод досягає значення $FPR=1\%$, при збереженні TPR (Recall) на рівні близько 0.89, свідчить про надзвичайну стійкість та збалансованість моделі. Зменшення FPR на 12% при втраті всього 7% Recall є феноменальним результатом. Значення трешхолда в цій точці зміщується з 0.5 (що є стандартом) до 0.979 (що є досить високим значенням).

Таблиця 4.2 Оцінки гібридного методу на чистих даних у конфігурації 1% FPR.

Метрика	Значення
Precision	0.9892
Recall	0.8877
FPR	0.0097
F1	0.9357

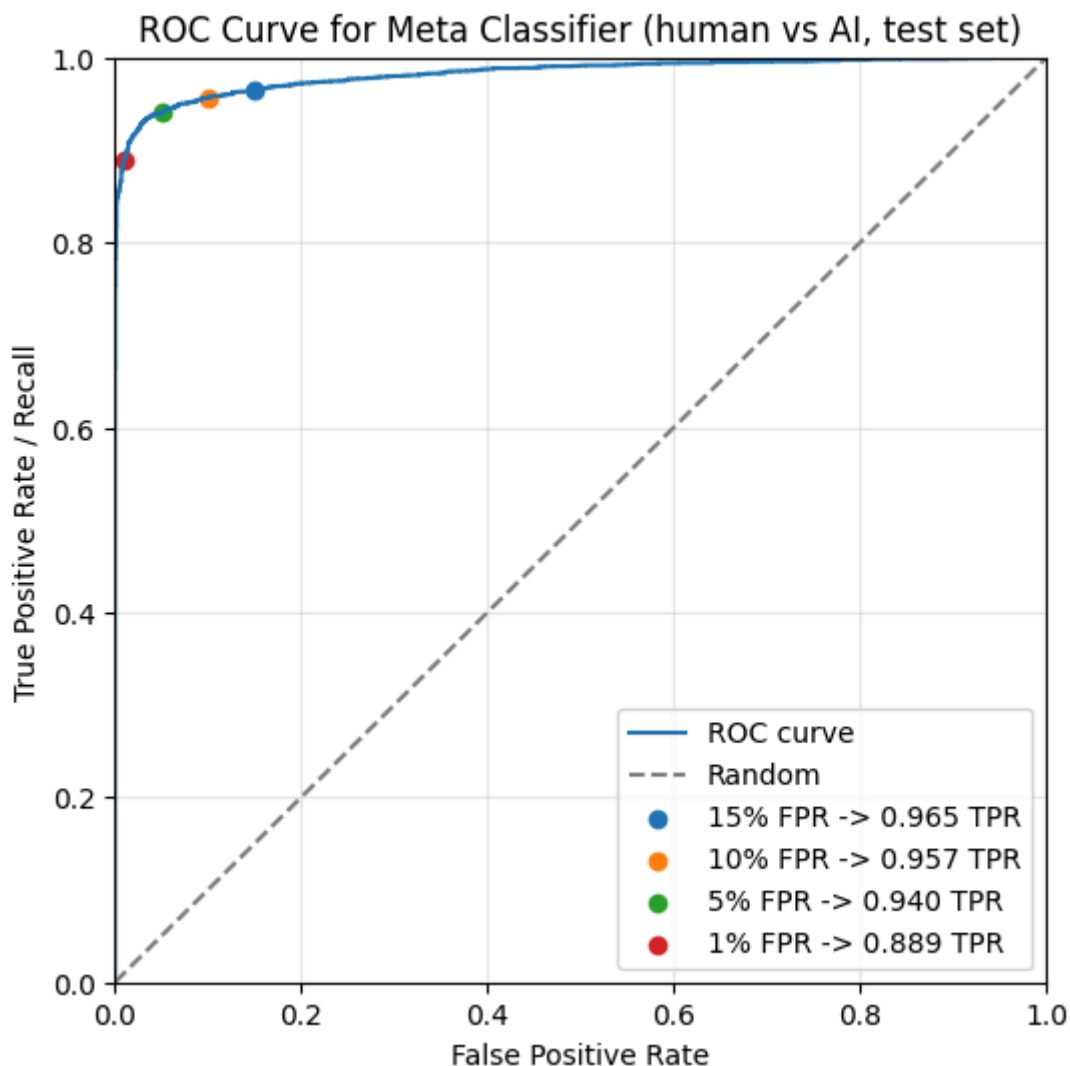


Рисунок 4.1 ROC крива для гібридного методу на чистих даних

Крім аналізу точності роботи гібридного методу варто провести аналіз точності роботи інших SOTA моделей на тих же даних, з метою порівняння розробленої моделі проти поточних лідерів. Для цих цілей буде використано 6 SOTA детекторів доступних в Open Source. Дані детектори можна розділити на дві великі категорії:

1. Zero-shot learning (без попереднього навчання):
 - 1.1. Binoculars [1]. Високоточний детектор згенерованих великими мовними моделями текстів, створений без попереднього навчання. Замість навчання окремого класифікатора система порівнює дві

тісно пов'язані мовні моделі, обчислюючи відношення perplexity до cross perplexity.

- 1.2. DetectGPT [2]. Застосовує серію модифікацій до вхідного тексту і вимірює ступінь падіння імовірностей, один із перших ефективних інструментів.
- 1.3. Fast-DetectGPT [3]. Заснований на DetectGPT, цей підхід вимірює криву умовної ймовірності для виявлення тексту, згенерованого машиною, в умовах «zero-shot». Він працює швидше, точніше та ефективніше за попередника.
2. Trained Classifier (на базі натренованих нейронних мереж):
 - 2.1. MAGE [4]. Модель спеціально натренована для роботи з OOD даними - працює ефективно для текстів поза розподілом та текстів після значного перефразування.
 - 2.2. RADAR [5]. Використовує модель змагального навчання, в якій програма-парафразер намагається уникнути виявлення детектором, а детектор навчається розпізнавати перефразований текст.
 - 2.3. Ghostbuster [6]. Оцінює ймовірності токенів за допомогою слабких мовних моделей (проксі-моделі) і застосовує лінійний класифікатор до отриманих ознак, що усуває необхідність доступу до імовірностей цільової моделі.

У рамках цього дослідження важливим є той факт, що системи Ghostbuster та MAGE тестувалися проти DIPPER, DetectGPT використовує T5 для створення модифікацій, а RADAR спеціально було натреновано на перефразованих даних (хоча і не на даних, згенерованих T5).

Таблиця 4.3 Оцінки гібридного методу і SOTA детекторів на чистих даних

Detector	Precision	Recall	FPR	F1	AUC-ROC
Fast DetectGPT	0.5423	0.6233	0.5261	0.5799	0.610248
Ghostbusters	0.5612	0.0877	0.0686	0.1517	0.535741

Binoculars	0.5317	0.224	0.1973	0.3153	0.501734
Radar	0.6206	0.2914	0.1782	0.3966	0.552465
DetectGPT	0.5923	0.6563	0.4517	0.6227	0.642231
MAGE	0.5197	0.5474	0.5058	0.5332	0.55087
Hybrid method	0.8813	0.9624	0.1296	0.92	0.982443
Hybrid method 1%FPR	0.9892	0.8877	0.0097	0.9357	0.982443

Порівняння результатів роботи всіх SOTA детекторів і гібридного методу на чистих даних наведено в табл. 4.3. З неї видно, що за ключовою метрикою F1 гібридний метод має суттєву перевагу над усіма іншими методами з перевагою над найближчим конкурентом у 0.31. З погляду Precision, Recall та FPR гібридний метод також має суттєву перевагу над конкурентами. На рис. 4.2 зображено ROC криву для всіх семи детекторів, сірою пунктирною лінією зображена конфігурація роботи детектора за замовчуванням, а помаранчевою пунктирною лінією - конфігурація 1% FPR. З графіка видно, що AUC-ROC гібридного методу набагато вищий за AUC-ROC інших методів.

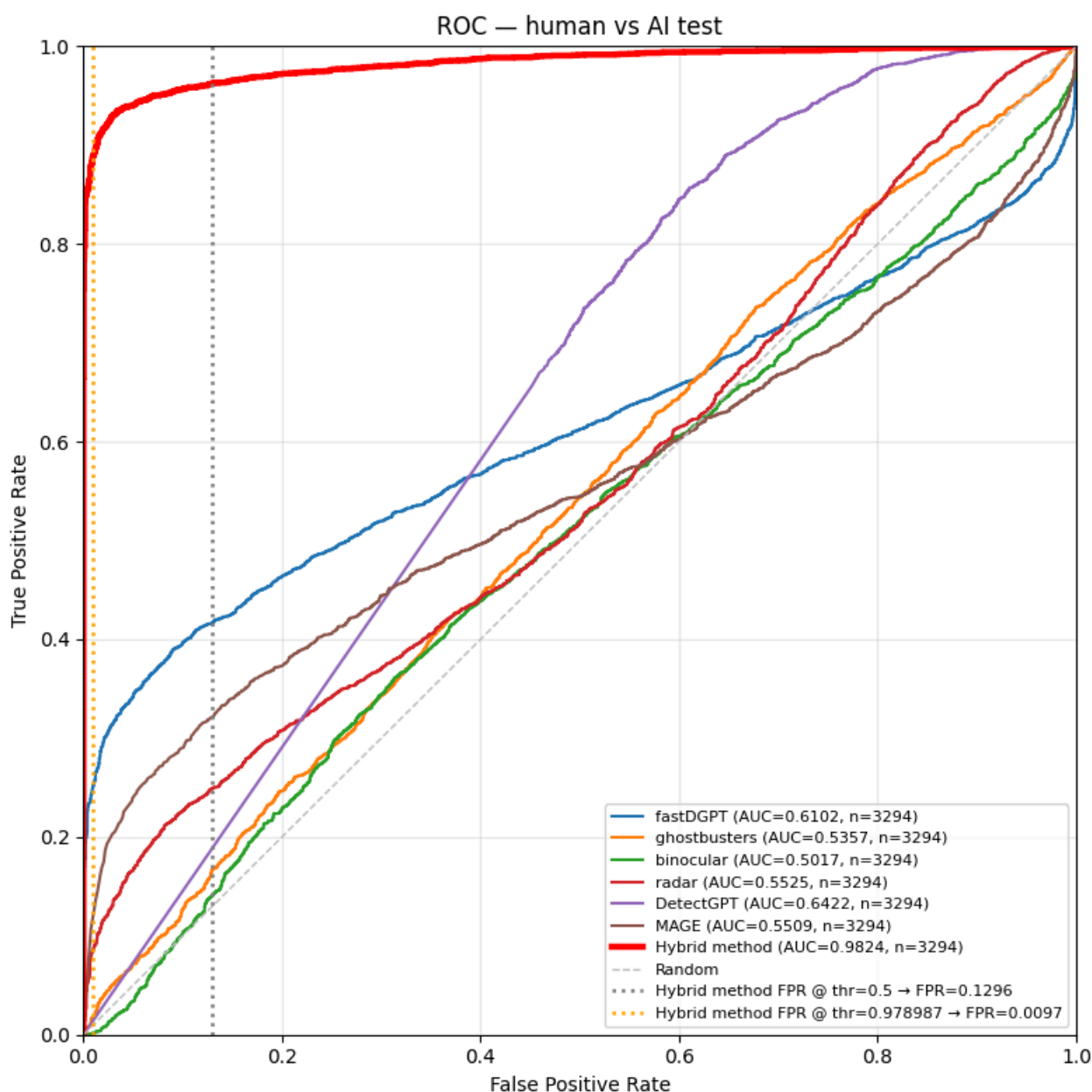


Рисунок 4.2 ROC крива для гібридного методу і SOTA детекторів на чистих даних

4.1.2. Стійкість до трансформаційних атак та зміщення Точки зламу

У цьому пункті наведено ключові докази робастності розробленого гібридного методу через тестування системи на даних, що зазнали модифікації, та підтвердження зміщення Точки зламу гібридним методом на основі чисельного моделювання трансформаційних атак на даних набору атак В. Як описано в підрозділі 3.2, для тестування робастності детектора створено 3 набори даних: набір А, що імітує структурні атаки; набір В, що містить перефразовані дані двома системами (T5 та DIPPER) для аналізу зсуву Точки зламу; та набір С, що містить

дані, замасковані трьома комерційними системами, які гарантують уникнення детекції. Результати роботи шести SOTA моделей та гібридного методу на наборі A наведено в таблиці 4.4. З неї видно, що гібридний метод має найвищий $Recall_{Adv}$ та найнижчий ASR. На рис. 4.3 зображено графік ефективності різних типів структурних атак проти набору з семи детекторів.

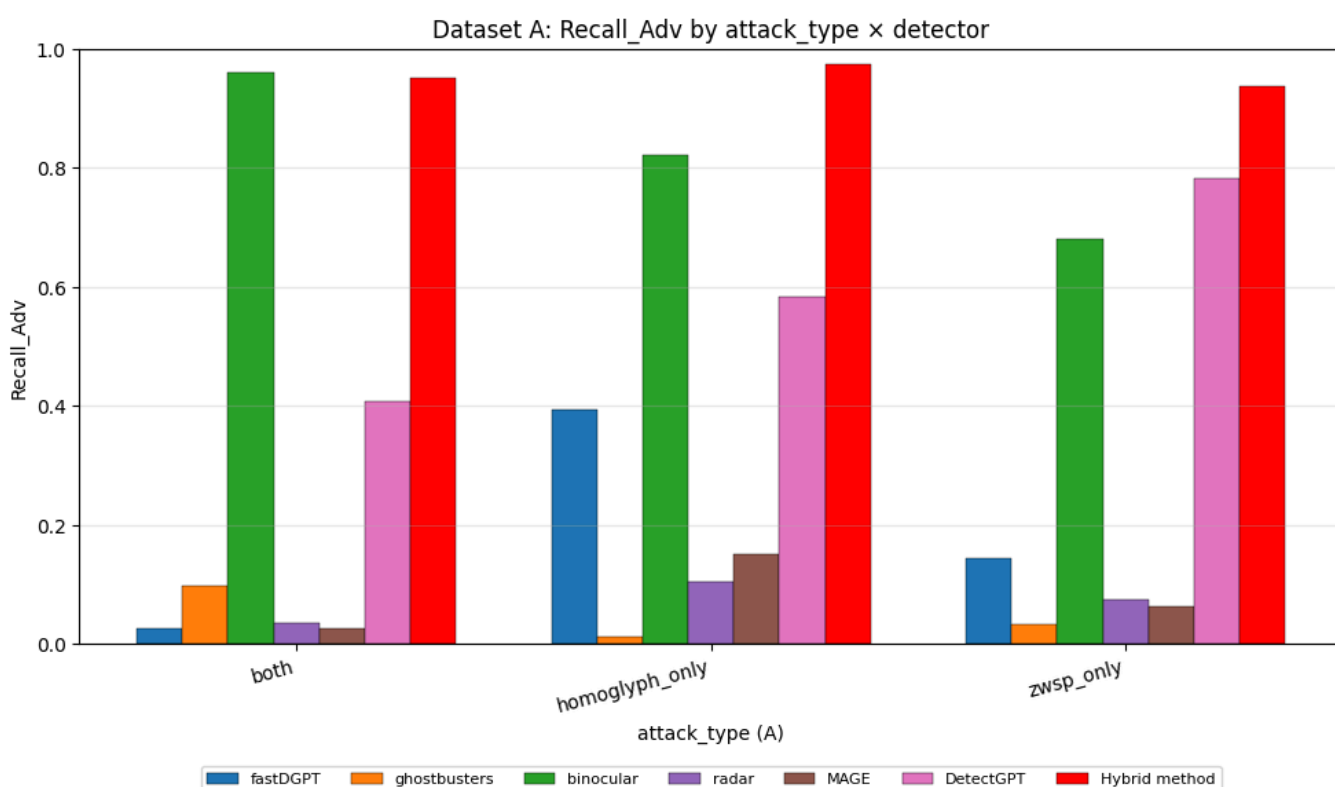


Рисунок 4.3 Ефективність детекторів проти різних типів структурних атак

Загальне падіння Recall на наборі даних A для гібридного методу відбулося з 0.961 (значення на даних до атаки) до 0.954 (значення після атаки). Серед інших цікавих ефектів, що спостерігаються на наборі даних A, варто відзначити суттєвий ріст Recall для Binoculars з 0.218 (значення на даних до атаки) до 0.82 (значення після атаки).

Таблиця 4.4 Оцінки детекторів на наборі атак A

Detector	$Recall_{Adv}$	Рівень Успішності Атаки (ASR)
Fast DetectGPT	0.189	0.811

Ghostbusters	0.047	0.953
Binoculars	0.820	0.180
Radar	0.071	0.929
DetectGPT	0.592	0.408
MAGE	0.080	0.920
Hybrid method	0.954	0.046

Далі наведено ефективність обраних детекторів на наборі даних В, що являє собою тексти перефразовані двома системами (T5 і DIPPER) з використанням різних конфігурацій. У цьому тесті є два ключові моменти:

- 1) Порівняти $Recall_{Adv}$ різних детекторів на кожному етапі перефразування, щоб оцінити їх стійкість до цього типу атаки.
- 2) Оцінити розташування Точки зламу для кожного з детекторів.

Щодо першого, то порівняння $Recall_{Adv}$ між детекторами для кожного етапу перефразування наведено на рис. 4.4. Як видно з рисунка, на початкових стадіях гібридний метод тримає $Recall_{Adv}$ вище .90 з поступовим зниженням при переході до більш суттєвих перефразовань. Він залишається лідером для всіх стадій перефразування з використанням T5, включно з рівнем 5 (lvl5), де відбувається видалення випадкових слів. Для DIPPER гібридний метод утримує стабільний $Recall_{Adv}$ у діапазоні 0.60-0.80, дещо програючи лише кільком детекторам, спеціально натренованим для боротьби з перефразуванням.

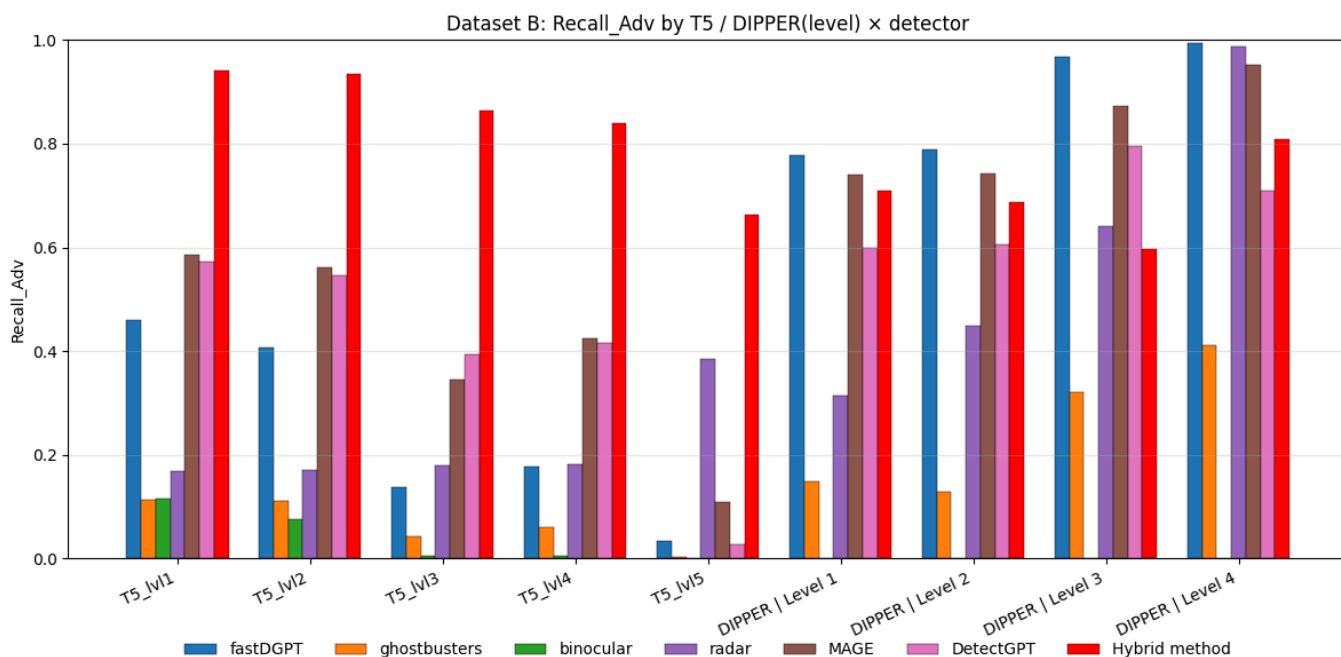


Рисунок 4.4 Ефективність детекторів проти атак перефразуванням

Враховуючи повну відсутність перефразованих або модифікованих навчальних даних та відсутність стадії перефразування в роботі детектора, можна стверджувати, що запропонована архітектура демонструє підвищену стійкість до маніпуляції перефразуванням і в більшості випадків перевершує наявні SOTA аналоги.

Таблиця 4.5 $Recall_{Adv}$ детекторів на наборі атак B

Detector	T5 L1	T5 L2	T5 L3	T5 L4	T5 L5	DIPPER L1	DIPPER L2	DIPPER L3	DIPPER L4	Mean Recall
Fast DetectGPT	0.46	0.406	0.1373	0.1767	0.0333	0.7778	0.7891	0.9679	0.994	0.5269
Ghostbusters	0.114	0.112	0.0433	0.06	0.002	0.1481	0.1293	0.3205	0.4107	0.1489
Binoculars	0.116	0.0767	0.0047	0.004	0.0013	0	0	0	0	0.0225
Radar	0.168	0.1713	0.1793	0.1807	0.3847	0.3148	0.449	0.641	0.9881	0.3863
DetectGPT	0.5853	0.562	0.344	0.4253	0.1087	0.7407	0.7415	0.8718	0.9524	0.5924
MAGE	0.5733	0.546	0.394	0.4167	0.0273	0.5988	0.6054	0.7949	0.7083	0.5183
Hybrid method	0.94	0.9353	0.8633	0.8393	0.664	0.71	0.6871	0.5962	0.8095	0.7827

Таблиці 4.5 та 4.6 наводять детальний аналіз поведінки детекторів на перефразованих даних через $Recall_{Adv}$ та ASR. З них видно, що в сукупному заліку (mean Recall та mean ASR) гібридний метод виграє у всіх інших детекторів на даних набору B.

Таблиця 4.6 ASR (Рівень Успішності Атак) для детекторів на наборі атак B

Detector	T5 L1	T5 L2	T5 L3	T5 L4	T5 L5	DIPPE R L1	DIPPE R L2	DIPPE R L3	DIPPE R L4	Mean ASR
Fast DetectGPT	0.54	0.594	0.8627	0.8233	0.9667	0.2222	0.2109	0.0321	0.006	0.4731
Ghostbusters	0.886	0.888	0.9567	0.94	0.998	0.8519	0.8707	0.6795	0.5893	0.8511
Binoculars	0.884	0.9233	0.9953	0.996	0.9987	1	1	1	1	0.9775
Radar	0.832	0.8287	0.8207	0.8193	0.6153	0.6852	0.551	0.359	0.0119	0.6137
DetectGPT	0.4147	0.438	0.656	0.5747	0.8913	0.2593	0.2585	0.1282	0.0476	0.4076
MAGE	0.4267	0.454	0.606	0.5833	0.9727	0.4012	0.3946	0.2051	0.2917	0.4817
Hybrid method	0.06	0.0647	0.1367	0.1607	0.336	0.2901	0.3129	0.4038	0.1905	0.2173

Для знаходження Точки зламу (місця, де $Recall_{Adv}$ детектору стає нижче .50 і атака вважається успішною) буде використано графік, зображений на рис. 4.5. Зелена лінія на графіку демонструє падіння семантичної цілісності (I_{sem}), сіра пунктирна лінія відповідає зоні де $Recall=0.5$, перехід цієї лінії характеризує Точку зламу, усі інші лінії демонструють поведінку систем детекції ШІ. Крім гібридного методу, на графіку присутні три інші детектори, що мають Recall вищий за 0.5: Fast DetectGPT, DetectGPT та MAGE, причому Точка зламу для всіх трьох систем знаходиться лівіше від $RR=0.20$; повторний перетин цими детекторами зони $Recall=0.5$ відбувається через їх високий Recall проти DIPPER, що пояснювалося в пункті 3.4.3 і не впливає на результати даної роботи. Щодо гібридного методу, то видно, що він не має Точки зламу для сценарію атаки перефразуванням, що підтверджує підвищену робастність запропонованого методу та теоретичне положення про те, що система залишається ефективною до межі повної втрати семантичної цілісності тексту.

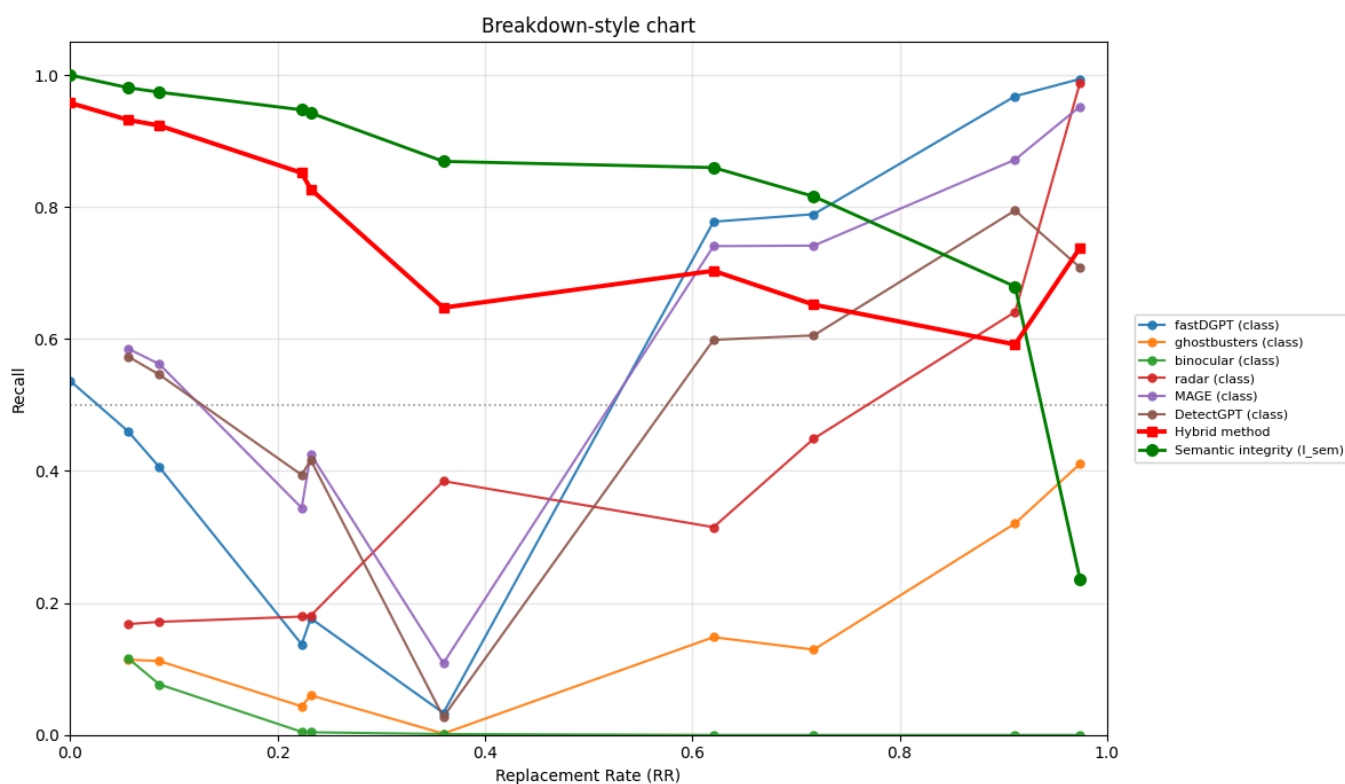


Рисунок 4.5 Розташування Точки зламу для різних детекторів

4.1.3. Ефективність проти комерційних систем маскування

Одним з ключових аспектів вимірювання ефективності розробленого гібридного методу є його тестування проти реальних комерційних систем (bypassers), що пропонують платні рішення для обходу детекції таких великих гравців, як Turnitin [7], Originality [8] та GPTZero [9]. Тестування на цих даних показує реальну практичну користь та можливості з застосування гібридного методу в ролі надійного і стійкого засобу детекції текстів, модифікованих SOTA засобами.

Таблиця 4.7 $Recall_{Adv}$ і ASR (Рівень Успішності Атак) детекторів на наборі атак C

Detector	Recall StealthGPT	Recall AIHumanize	Recall Phrasly	ASR StealthGPT	ASR AIHumanize	ASR Phrasly	Mean Recall	Mean ASR
Fast DetectGPT	0.8753	0.371	0.558	0.1247	0.629	0.442	0.6015	0.3985
Ghostbusters	0.192	0.0905	0.084	0.808	0.9095	0.916	0.1222	0.8778
Binoculars	0.1496	0.0294	0.216	0.8504	0.9706	0.784	0.1317	0.8683
Radar	0.4239	0.3032	0.282	0.5761	0.6968	0.718	0.3364	0.6636

DetectGPT	0.7855	0.5814	0.512	0.2145	0.4186	0.488	0.6263	0.3737
MAGE	0.606	0.6674	0.662	0.394	0.3326	0.338	0.6451	0.3549
Hybrid method	0.7905	0.6923	0.832	0.2095	0.3077	0.168	0.7716	0.2284

Результати тестування детекторів на наборі даних С наведено на рис. 4.6 та в табл. 4.7. Гібридний детектор показав найкращі результати на двох системах з трьох, при цьому в загальному заліку (середній $Recall_{Adv}$ та ASR по трьох системах) гібридний метод з суттєвим запасом займає перше місце.

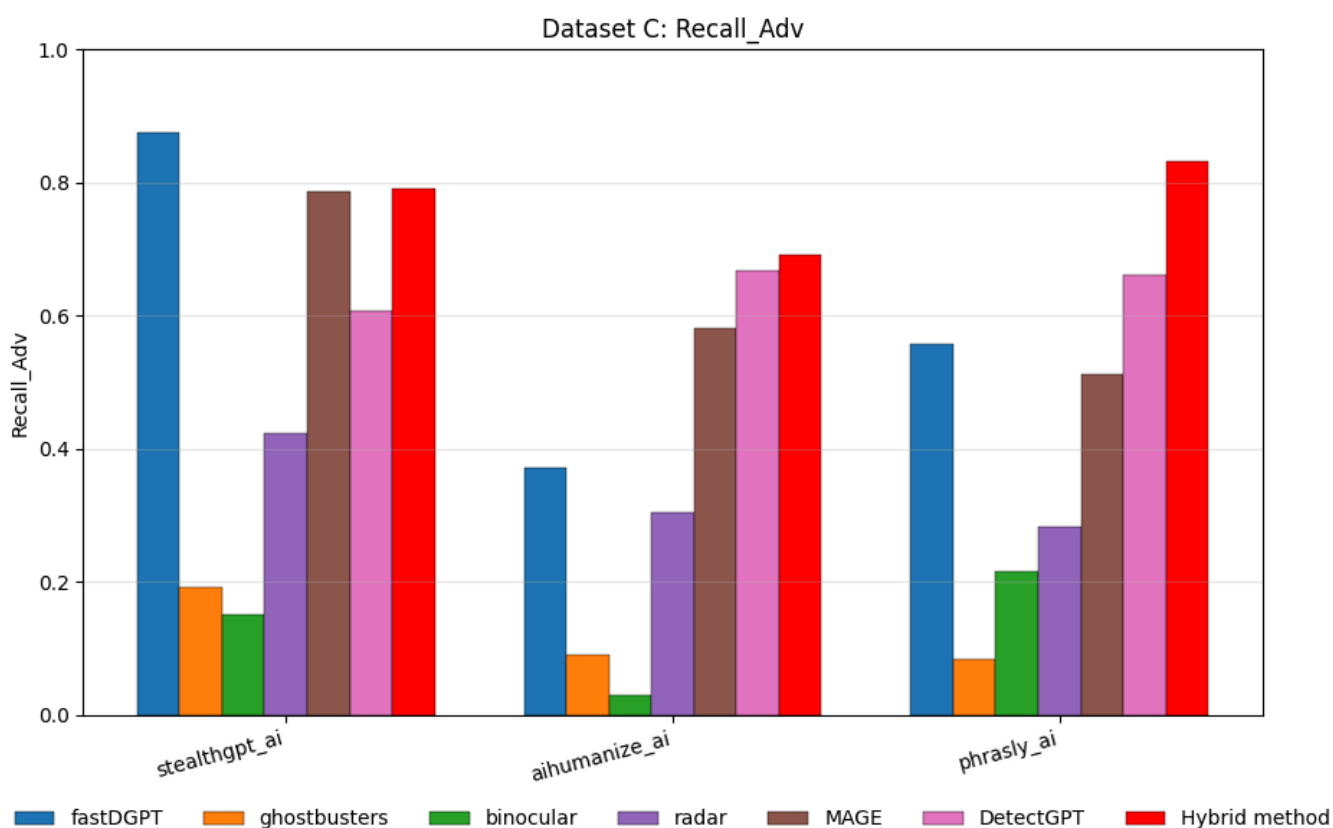


Рисунок 4.6 Ефективність детекторів проти комерційних bypassers

Загалом, підсумовуючи тестування гібридного методу за результатами усіх трьох тестів, можна стверджувати, що розроблений метод впевнено займає лідерські місця в кожному з тестів, що свідчить про надійність, передбачуваність та робастність моделі. Важливо підкреслити, що процес навчання моделі не включав жодного типу модифікованих даних і високі результати на перефразованих та

модифікованих даних досягнуто виключно завдяки запропонованому трирівневому підходу до сліду ІІІ та механізмів підвищення стійкості у вигляді $S_{mismatch}$ та механізму Adaptive Gating.

4.1.4. Внесок модулів у робастність

Завершуючи аналіз ефективності та робастності гібридного методу, варто дослідити внесок різних елементів системи у фінальний результат. З архітектурного погляду, система становить собою дворівневий ансамбль, де на першому рівні знаходяться індивідуальні модулі (лексичний, синтаксичний, семантичний), а на другому рівні знаходиться метакласифікатор, що використовує Adaptive Gating для роботи з імовірностями, наданими індивідуальними модулями, та деякі додаткові ознаки ($S_{mismatch}$, Reasoning Trace та інші) для винесення фінального рішення. Оскільки основний результат роботи - це запропонована архітектура, то варто провести більш детальне дослідження внеску кожного з індивідуальних компонентів, щоб показати, що жоден з них не може замінити інші, і всі вони є важливими для отримання описаних результатів.

Таблиця 4.8 Апроксимовані показники ефективності індивідуальних модулів і метакласифікатора

#	Data Type	Metric	Mlex Score	Msyn Score	Msem Score	META-MODEL
0	Human	TNR (Human)	95.11%	81.69%	69.25%	87.04%
1	AI (Base)	Recall (AI)	91.77%	92.38%	74.01%	96.24%
2	Attack A	Recall (AI)	92.80%	91.80%	74.60%	95.40%
3	T5 (Lvl 1)	Recall (AI)	87.80%	87.40%	73.93%	94.00%
4	T5 (Lvl 2)	Recall (AI)	85.67%	87.00%	73.40%	93.53%
5	T5 (Lvl 3)	Recall (AI)	66.33%	83.67%	79.20%	86.33%
6	T5 (Lvl 4)	Recall (AI)	64.00%	79.27%	78.80%	83.93%
7	T5 (Lvl 5)	Recall (AI)	34.93%	67.40%	73.67%	66.40%
8	DIPPER (Level 1)	Recall (AI)	81.94%	29.17%	94.44%	79.17%
9	DIPPER (Level 2)	Recall (AI)	76.56%	32.81%	89.06%	73.44%
10	DIPPER (Level 3)	Recall (AI)	75.36%	23.19%	92.75%	62.32%
11	DIPPER (Level 4)	Recall (AI)	69.74%	25.00%	92.11%	65.79%
12	StealthGPT	Recall (AI)	72.07%	66.33%	59.60%	79.05%

13	AIHumanize	Recall (AI)	58.37%	57.24%	50.00%	69.23%
14	Phrasly	Recall (AI)	73.20%	76.60%	77.40%	83.20%

На рис. 4.7 показано розподіл передбачень кожного з модулів (лексичний, синтаксичний, семантичний) на тестових даних; самі результати показано в таблиці 4.8. Для виведення значень передбачень окремих класифікаторів та метакласифікатора використовуємо числові методи та апроксимацію замість самої моделі, тому деякі значення відрізняються від вказаних раніше. Видно, що на чистих даних найкращих результатів досягає лексичний модуль, перефразування, особливо T5, зменшує ефективність лексики, а модифікація T5_lv15 (що включає видалення випадкових слів) повністю руйнує його роздільну здатність. При цьому на даних перефразованих DIPPER та комерційних byrassers лексика все ще здатна працювати більш-менш ефективно.

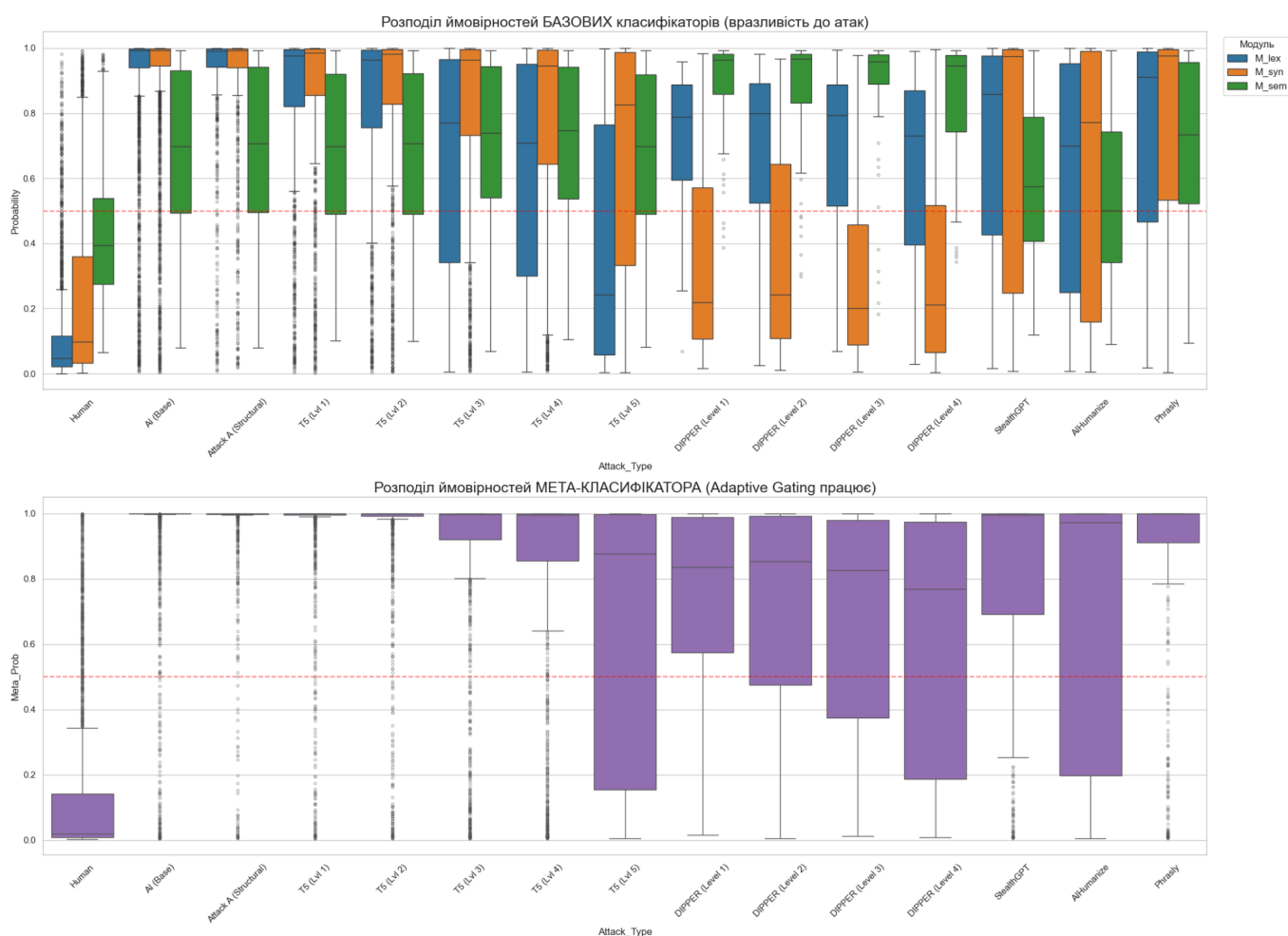


Рисунок 4.7 Розподіл передбачень індивідуальних модулів гібридного методу та метакласифікатора

Синтаксичний модуль поводить себе схожим чином на лексичний, але його сила починає розкриватися на даних перефразованих T5; на даних, перефразованих DIPPER, синтаксична модель починає генерувати хибні передбачення. Це свідчить про те, що DIPPER змінює синтаксичні структури, характерні для ІШ, в бік більш людиноподібних. На комерційних bypassers синтаксис знову починає працювати непогано, хоча і з помилками. Семантичний модуль працює більш-менш стабільно на чистих даних, проте його сила розкривається на перефразованих даних (особливо DIPPER). Щодо комерційних систем, то тут семантичний модуль показує різну ефективність залежно від системи.

На нижньому графіку рис. 4.7 показано розподіл передбачень метакласифікатора, на ньому видно ефект від спрацювання Adaptive Gating та $S_{mismatch}$. Найбільш показовими у цьому сенсі є стовпчики T5_lvl5 та DIPPER level1-level4, де лексичний та синтаксичний класифікатор видають повністю некоректний результат, при цьому середня точність метакласифікатора залишається на рівні 80% і вище.

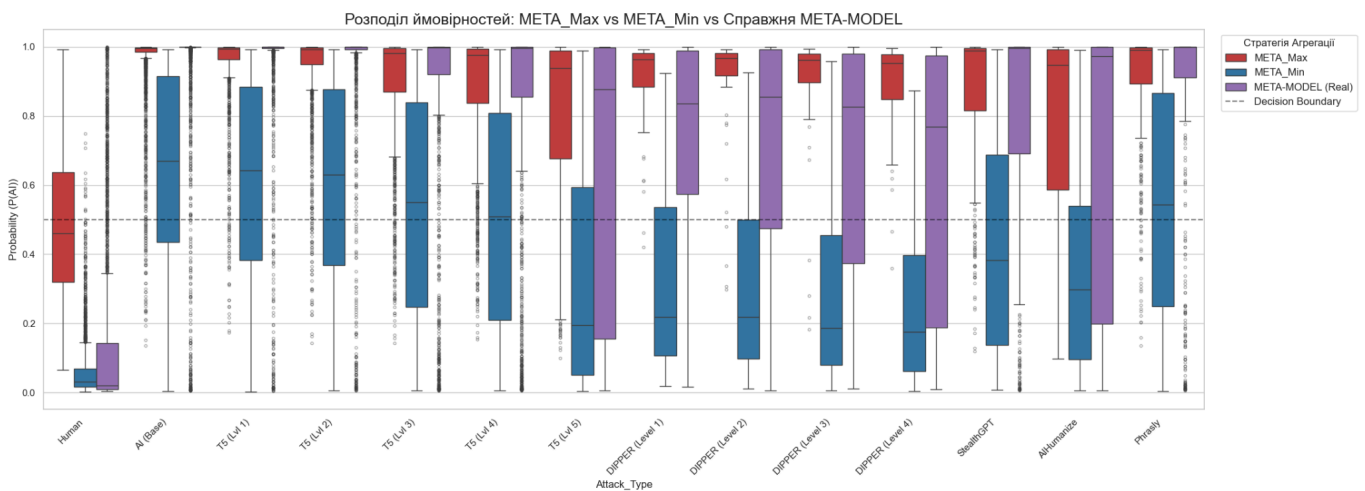


Рисунок 4.8 Розподіл передбачень різних типів та метакласифікатора

Попередній експеримент довів, що метакласифікатор працює ефективніше за будь-який інший індивідуальний модуль, тож його застосування виправдане. Ще одним важливим моментом щодо оцінювання ефективності Adaptive Gating та

$S_{mismatch}$ є порівняння простого метакласифікатора без цих механізмів (наприклад, простого max та min серед індивідуальних модулів) з повним метакласифікатором.

На рис. 4.8 зображено ефективність реального метакласифікатора, та його порівняння з функціями max та min. Очікувано, що мета-min показує себе добре на експерименті, де всі дані належать до негативного класу, а мета-max - де всі дані належать до позитивного. Цікавим спостереженням є те, що для експериментів, де весь датасет складається виключно з позитивного класу (AI), у 6 з 13 випадків метакласифікатор показує себе краще за мета-max; це досягається саме завдяки доданим механізмам стійкості. Іншою суттєвою слабкістю мета-max є його надзвичайно високий FPR= ~ 0.46 на Human даних проти FPR= ~ 0.12 для метакласифікатора в базовій конфігурації та FPR= ~ 0.01 для його конфігурації 1% FPR. Для реальних сценаріїв використання, де 80%-90% вхідних даних будуть саме Human, такий FPR є неприйнятним.

Отже, можна стверджувати, що поточна архітектура метакласифікатора є необхідною для досягнення вказаних вище результатів і не може бути замінена ні одним з індивідуальних модулів, ні простою функцією агрегації.

4.2. Робота в умовах невизначеності та швидкодія

4.2.1. Валідація методу ICP для ідентифікації OOD-даних

Ключова ціль цього пункту показати, що розроблена система має адекватну реакцію на невідомі дані. Деякі з SOTA детекторів, використаних у підрозділі 4.1, були надмірно впевнені (overconfident), наприклад, детектор DetectGPT на наборі чистих даних, що містить ~ 3300 Human і ~ 3300 AI текстів мав середню AI probability $>.99$. Тобто, він з високою впевненістю робив велику кількість помилок, що призводило до FPR $>.90$. Щоб зробити можливим його використання в проведеному тестуванні, було встановлено трешхолд на рівні 1, тобто всі тексти, що мали імовірність менше ніж 1, було позначено як Human, а ті що мали імовірність 1 - як AI, це дозволило знизити FPR системи до 0.45. Така поведінка детектора

(overconfident) є досить шкідливою, адже не дає змоги розрізнити, коли система працює на нормальних для себе даних, а коли на нетипових. Для боротьби з надмірною впевненістю і призначено метод ICP, описаний у 2.2.3, його ключовим результатом є здатність системи повернути мітку класу у випадку, коли вона впевнена, порожню множину, коли не впевнена, і мітки обох класів, коли дані мають змішану природу.

При налаштуванні ICP ключову роль має процес калібрування - це момент, коли на окремому датасеті, відмінному від тренувальних даних, проводиться порівняння міток, передбачених класифікатором разом з їх імовірностями, з еталонними мітками. Щоб уникнути побудови окремого калібрувального датасета було використано 5-Fold крос-валідацію на тренувальних даних, таким чином, було отримано бали невідповідності (nonconformity scores) $\alpha(x, y)$ на базі прогнозу метакласифікатора. Це дозволило розрахувати калібрувальний квантиль q для заданого рівня значущості ϵ (де $\epsilon=0.05$), що відповідає 95% статистичній достовірності. Валідація ICP буде проводитися на двох типах даних: стаціонарних (ті що за розподілом схожі на тренувальні) та OOD.

Оцінка на стаціонарних даних (In-Distribution). Ця частина валідації буде проводитися на тестовій частині чистих даних (датасет 4.1), що складаються тільки з Human та AI текстів. Ці дані обрано для тестування, оскільки система тренувалася лише на чистих даних, тож In-Distribution тестування має відбуватися також на чистих даних.

Таблиця 4.9 Порівняння показників ICP та гібридного методу на чистих даних

Metric	Hybrid method	ICP
TNR (Human) / Singleton TNR	0.8704	0.7562
TPR (AI) / Singleton TPR	0.9624	0.9353
Singleton Rate	N/A	0.8767
Coverage Rate	N/A	0.8458
OOD Rate	N/A	0.1233
Ambiguity Rate	N/A	0

У табл. 4.9 наведено результати оцінювання ICP, як видно ICP має високу стабільність щодо виявлення AI текстів - Singleton TPR на рівні 0.9353 лише незначно поступається базовому TPR (0.9624). Також спостерігається невелике зниження показника Singleton TNR (до 0.7562) завдяки росту кількості порожніх передбачень (OOD Rate = 0.1233). На практиці це вважається позитивним ефектом від запровадження ICP - метод відсікає ~11.5% Human текстів, що коректно класифіковані гібридним методом, проте мають низький рівень впевненості. Такі тексти ICP позначає як зону епістемічної невизначеності, генеруючи порожню відповідь та переводячи їх у категорію аномалій, що потребують експертного аналізу. Емпіричний Coverage Rate на тестовій вибірці зафіксовано на рівні 84.58%, що свідчить про високу строгість встановлених калібрувальних порогів алгоритму.

Оцінка на даних поза розподілом (OOD). Для оцінки ICP на OOD даних буде використано два типи даних: тексти, написані людьми та згенеровані ШІ тексти, які були модифіковані.

У ролі набору людських текстів, що відрізняються від тренувального розподілу, буде використано тексти з екзамену IELTS [10]. Їх перевагою є те, що цей екзамен складають не носії мови, відповідно, структура тексту, складність синтаксичних конструкцій та різноманіття лексики буде суттєво відрізнятися від студентських есе та Вікіпедії, використаних для навчання системи. Даний набір містить 1435 текстів, важливою особливістю датасету є наявність дуже коротких текстів (від 116 слів), в той час, як систему було треновано і тестовано виключно на текстах від 300 слів. Тобто цей датасет є OOD по двох характеристиках: структура текстів і їх лексико-синтаксичні особливості та довжини текстів. Результати ICP та гібридного методу на IELTS текстах наведено в табл. 4.10.

Таблиця 4.10 Порівняння показників ICP та гібридного методу на даних іноземних студентів

Data type	Total texts	Hybrid method TNR	Hybrid method FPR	Singleton TNR	Coverage Rate	OOD Rate
IELTS (≥ 300 words)	402	0.9129	0.0871	0.7886	0.7886	0.1915
IELTS (< 300 words)	1033	0.9671	0.0329	0.9032	0.9032	0.091
IELTS (All texts)	1435	0.9519	0.0481	0.8711	0.8711	0.1192

При проведенні аналізу дані було розбито на дві групи: до 300 слів і від 300 слів, у табл. 4.10 наведено результати по кожній з категорій та загальний результат на наборі даних. Для першої групи (валідні тексти від 300 слів) гібридний метод демонструє TNR на рівні 0.9129 і FPR на рівні 0.0871, ICP бачить нетипові тексти, трактує їх епістемічну невизначеність і генерує передбачення для 19% у вигляді “невизначено” (OOD), зменшуючи реальний FPR системи та переводячи хибні звинувачення у статус “аномалії, що потребує аналізу”, знижуючи ризик автоматичної дискримінації.

Аналіз результатів групи коротких текстів (до 300 слів) виявив цікаву особливість - гібридний класифікатор проявляє на них дуже низький $FPR=0.0329$ і високий $TNR=0.9671$. Можливо, у цьому випадку, занадто короткі тексти виглядають як щось нетипове для AI. Однак, алгоритм ICP і тут фіксує відхилення від базового навчального розподілу (адже модель тренувалася на довших документах), генеруючи порожню множину в 9.10% випадків. Це вказує на чутливість ICP не лише до лексичних зміщень, але і до структури документа.

Щоб довести, що ICP дійсно зменшує FPR, а не просто переводить частину коректних передбачень у статус “невизначено”, необхідно провести аналіз перехоплення FPR. Найбільш коректним способом такого аналізу буде дослідження перетину множини FPR з множиною OOD, основною ціллю дослідження є визначення Rescue Rate (коефіцієнт порятунку - відсоток зменшення хибнопозитивних спрацювань). Результати дослідження наведено в таблиці 4.11. Для першої групи (нормальні тексти) ICP зміг коректно ідентифікувати та перевести в OOD 27 з 35 хибнопозитивних спрацювань детектора, для цієї групи Rescue Rate становить 77%, а підсумковий FPR системи зменшився з 8.71% до 1.99%. Для другої групи (короткі тексти) Rescue Rate системи ще вищий і складає 82.35%, що дозволило ICP коректно ідентифікувати та перевести в OOD 28 з 34 хибних спрацювань детектора, підсумковий FPR системи зменшився з 3.29% до 0.58%.

Таблиця 4.11 Ефективність ICP при перехопленні FPR

Data type	Total	Hybrid method	Converted to	Rescue Rate	New FPR
-----------	-------	---------------	--------------	-------------	---------

	texts	FPR	OOD by ICP		(after ICP)
IELTS (≥ 300 words)	402	35 (8.71%)	27	77.14%	8 (1.99%)
IELTS (< 300 words)	1033	34 (3.29%)	28	82.35%	6 (0.58%)
IELTS (All texts)	1435	69 (4.81%)	55	79.71%	14 (0.98%)

Проведений аналіз емпірично доводить, що розроблений гібридний метод з модулем ICP забезпечує надійний захист від хибних звинувачень при роботі з текстами неносіїв мови. Здатність системи переводити сумнівні тексти в стан невизначеності через генерацію порожньої множини алгоритмом ICP є надійним механізмом блокування невпевнених та хибних рішень. Це забезпечує суттєве зменшення кількості несправедливих звинувачень авторів-людей, гарантуючи, що розроблена система є надійним та неупередженим інструментом для перевірки академічної доброчесності та виявлення ШІ в широкому спектрі областей, що є вибагливими до низького FPR.

У ролі набору модифікованих AI текстів використаємо створені раніше і описані в підрозділі 3.2 набори даних. Їх застосування виправдане, оскільки модифікації змінюють розподіл внутрішніх характеристик тексту, а тренувальні дані не містять жодного модифікованого тексту. Тож, з погляду гібридного методу, набори даних А, В і С є OOD і можуть бути використаними для експериментальної валідації ICP.

У табл. 4.12 наведено результати ICP у порядку росту інтенсивності атак. Всі дані поточного тесту містять виключно згенеровані тексти, тому показники Singleton TPR та Coverage Rate збігаються. В усіх тестах Ambiguity Rate дорівнює нулю, що свідчить про відсутність алеаторного конфлікту між класами. Отримані результати підтверджують коректність концептуальної моделі Точки зламу, запропонованої в підрозділі 2.2.2. На прикладі набору атак T5 чітко прослідковується поступова втрата точності базового (точкового) прогнозу: TPR падає від 0.94 (T5 Level 1) до 0.664 (T5 Level 5). Проте, конформний предиктор ICP допомагає компенсувати цю вразливість. Для захисту від видачі хибнонегативних прогнозів (класифікувати модифікований AI як Human), система пропорційно збільшує рівень генерації порожніх множин (OOD Rate): від 6.47% на Level 1 до 21.13% на Level 5.

Таблиця 4.12 Порівняння показників ICP та гібридного методу на модифікованих даних

Modification type	Hybrid method TPR	Singleton TPR	Singleton Rate	Coverage Rate	OOD Rate	Ambiguity Rate
Attack A	0.954	0.906	0.937	0.906	0.063	0
T5 Level 1	0.94	0.8993	0.9353	0.8993	0.0647	0
T5 Level 2	0.9353	0.884	0.9247	0.884	0.0753	0
T5 Level 3	0.8633	0.786	0.878	0.786	0.122	0
T5 Level 4	0.8393	0.7587	0.8627	0.7587	0.1373	0
T5 Level 5	0.664	0.5373	0.7887	0.5373	0.2113	0
DIPPER	0.703	0.4787	0.6382	0.4787	0.3618	0
StealthGPT	0.7905	0.6958	0.8204	0.6958	0.1796	0
AIHumanize	0.6923	0.5928	0.8235	0.5928	0.1765	0
Phrasly	0.832	0.78	0.876	0.78	0.124	0

У цьому тесті не проводиться розділення DIPPER за рівнями перефразування, а використовується як система в цілому. Саме для неї та спостерігається найвищий рівень епістемічної невизначеності. У цьому випадку ефективність (TPR) базового прогнозу знижується приблизно на 30%, разом з тим, для компенсації втрат, алгоритм ICP переводить понад третину всіх маскованих документів (OOD Rate = 0.3618) у категорію аномалій.

Схожа тенденція спостерігається і при валідації на сучасних комерційних bypassers. Наприклад, маскуванню що проводить AIHumanize знижує TPR базової моделі до 0.6923. В умовах відсутності ICP це означало б успішний обхід системи у 30.8% випадків, проте використання ICP дає змогу успішно локалізувати понад половину з них (17.65%) як невизначені (OOD), рекомендуючи аналіз експертом.

Загальний висновок. Експериментально доведено, що гібридна модель з модулем конформного передбачення не схильна до катастрофічної відмови під дією трансформаційних атак. При роботі з маскованими даними, що суттєво знижують семантичну цілісність (I_{sem}) та наближають гібридний метод до Точки зламу, система надійно блокує прийняття хибного рішення. Перерозподіл складних випадків із категорії хибних прогнозів у категорію порожніх множин (OOD) гарантує прозорість та безпечність запропонованого гібридного методу, роблячи

його придатним для використання в критичних сферах, де ціна помилки (особливо хибного звинувачення (False Positive)) є неприпустимо високою.

4.2.2. Аналіз обчислювальної складності та часових витрат

Щодо практичної застосовності системи детекції, то одним з ключових питань є її швидкодія. У сучасному світі, існування хмарних обчислювальних платформ на кшталт AWS [11] зняло критичне обмеження на обчислювальні потужності. Теоретично тепер можна отримати майже необмежені ресурси, проте залишається питання ціни. Тобто розрахунок обчислювальної складності так само залишається актуальним, хоча тепер обсяг необхідних обчислень менше впливає на час очікування результату і більше на ціну використання. У даному пункті буде проведено теоретичний обрахунок обчислювальної складності, експериментальні заміри часу роботи системи та її порівняння з відкритими SOTA детекторами.

Теоретична оцінка обчислювальної складності. Складність гібридного класифікатора визначається як сума складностей його компонентів. Далі наведено перелік ключових компонентів та проведено обрахунок їх складності. Для обчислення складності буде використано такі позначення: N - загальна кількість токенів у вхідному тексті, S - загальна кількість речень, L_s - середня довжина речення у токенах, d - розмір вхідного вектора моделі (embedding dimension), де індекси d_{lex} , d_{syn} та d_{sem} позначають розмірність для моделей лексичного, синтаксичного та семантичного рівнів відповідно.

Вплив різних модулів на обчислювальну складність:

- 1) Модуль канонізації (M_{canon}). Обробка тексту базується на наборі регулярних виразів, що дають змогу виявити невидимі символи, гомогліфи та провести нормалізацію. Кожна з цих операцій вимагає лінійного часу, оскільки здійснюється за один прохід по рядку. Отже, складність буде $O(N)$.

2) Лексичний модуль (M_{lex}). Найбільш важкою частиною лексичного модуля є обчислення імовірностей токенів з використанням проксі-моделі microsoft/Phi-3-mini-4k-instruct [12], наявність механізму Self-Attention дає складність прямого проходу на рівні $O(N^2 \cdot d_{lex})$, розрахунок інших 30 ознак відбувається за один прохід по тексту, тож складність їх обчислення $O(N)$. Тоді загальна складність складає $O(N^2 \cdot d_{lex})$.

3) Синтаксичний модуль (M_{syn}). Найбільш важкою частиною синтаксичного модуля є використання парсера. Система підтримує два варіанти синтаксичних парсерів, які мають різну часову складність:

- en_core_web_trf - побудовано на базі трансформерної моделі RoBERTa [13]. Побудова дерев залежностей та розставлення POS-тегів мають складність $O(N^2 \cdot d_{syn})$; при обчисленні дисперсії синтаксичної глибини необхідно проводити обхід дерева, оскільки в природній мові дерева, як правило, збалансовані, то складність буде $O(N \cdot \log N)$.
- en_core_web_sm - побудовано на легких згорткових нейромережах та багат шарових перцептронах з використанням алгоритму парсингу залежностей на основі переходів, часова складність - лінійна $O(N)$.

Обчислення Reasoning-стилю використовує класифікатор LightGBM, що генерує прогноз за $O(T \cdot h)$, де T -кількість дерев, а h - максимальна глибина дерева, обидва значення є константою, відповідно складність $O(1)$. Інші ознаки вимагають лінійного часу, тож підсумкова обчислювальна складність модуля $O(N^2 \cdot d_{syn})$.

- 4) Семантичний модуль (M_{sem}). Найбільш важкою частиною модуля є побудова семантичних ембедингів моделлю all-MiniLM-L6-v2 [14], яка має складність $O(S \cdot L_S^2 \cdot d_{sem})$ через наявність механізму Self-Attention. Розрахунок косинусної подібності суміжних речень та інших ознак вимагає лінійного часу, а обчислення глобального дрефту центроїда відбувається шляхом аналізу всіх зв'язків між реченнями, тому вимагає $O(S^2 \cdot d_{sem})$. Загальна складність модуля: $O(S \cdot L_S^2 \cdot d_{sem} + S^2 \cdot d_{sem})$. Оскільки $S \ll N$, то завдяки обчисленню семантики на рівні окремих речень, а не всього документа, семантичний модуль, на відміну від інших модулів, не має квадратичної часової складності та працює в десятки разів швидше, ніж інші компоненти системи на базі трансформерів.
- 5) Метакласифікатор та інтерпретаційна модель. Розрахунок фінального рішення системи та генерація профілів атрибуції відбувається шляхом застосування Adaptive Gating до вектора ознак, що виконується за $O(1)$.

Підсумкова асимптотична складність гібридного методу обмежена квадратичною складністю використаних моделей-трансформерів і становить $O(N^2)$. Обчислення, що проводяться для решти ознак, поглинаються цією складністю.

Експериментальні заміри часу роботи системи. Практичний експеримент з профілювання проводився у хмарній інфраструктурі Amazon Web Services (AWS) на інстансі ml.g5.xlarge (GPU NVIDIA A10G 24 GB, 4 vCPU) на вибірці з 1000 текстів. Результати замірів наведено в табл. 4.13.

Таблиця 4.13 Середній час аналізу одного документа гібридним методом

Компонент системи	Модель en_core_web_trf		Модель en_core_web_sm	
	Середній час (мс)	Частка (%)	Середній час (мс)	Частка (%)
Нормалізація та канонізація	13.68	0.70%	13.68	4.17%

Лексичний рівень (Phi-3)	239.04	12.60%	239.04	72.70%
Синтаксичний рівень (spaCy)	1629.26	85.80%	59.04	17.96%
Семантичний рівень (SBERT)	16.68	0.90%	16.68	5.07%
Метакласифікатор та XAI-профіль	0.33	0.00%	0.33	0.10%
Загальний час	1898.99	100%	328.77	100%

Отримані результати відповідають теоретичному аналізу: найбільш обчислювально важкими є синтаксичний та лексичний модулі, на третьому місці - семантичний модуль і лише потім канонізація. У конфігурації `en_core_web_trf` високі часові витрати на синтаксичний модуль пояснюються високою вартістю використання RoBERTa. Суттєвого пришвидшення можна досягти заміною парсера з `en_core_web_trf` на `en_core_web_sm`, що використовує лінійні алгоритми. Використання конфігурації моделі з парсером `en_core_web_sm` є рекомендованим, оскільки знижує загальний час роботи системи майже у 6 разів - з 1.9с до 0.33с, при незначному зниженні якості синтаксичного аналізу в найбільш складних випадках. Разом з тим, сумарна тривалість роботи навіть у 1.9с на один текст робить технологію придатною для промислового використання в асинхронному режимі.

Порівняння швидкодії із SOTA детекторами. Для більш повної та об'єктивної оцінки швидкодії розробленого методу необхідно провести порівняння його часу роботи на повному наборі тестових даних з іншими SOTA моделями. Як апаратне забезпечення для тесту буде використано AWS інстанс `ml.g5.xlarge`; у ролі даних - набір чистих даних, описаний у 3.1, та набір модифікованих даних, описаний у 3.2; як засіб вимірювання часу - сервіс Amazon CloudWatch [15]. Результати наведено в табл. 4.14.

Таблиця 4.14 Часові витрати різних моделей детекції ІІІ.

Classifier	Total time (sec)	Time per one text (sec)
Fast DetectGPT	2992	0.175
Ghostbusters	2104	0.123

Binoculars	10686	0.626
Radar	580	0.034
DetectGPT	46197	2.707
MAGE	610	0.036
Hybrid method	5595	0.328

За часовими витратами всі детектори можна поділити на три групи:

1. час на аналіз одного тексту менше 0.1сек: MAGE, Radar (обидва належать до класу натренованих класифікаторів);
2. час на аналіз одного тексту менше 0.5сек: Ghostbusters, Fast DetectGPT, Гібридний метод
3. час на аналіз одного тексту більше 0.5сек: Binoculars, DetectGPT.

З погляду швидкодії гібридний метод знаходиться в середині таблиці, поступаючись чотирьом системам і переважаючи дві інші. Варто зазначити, що час роботи гібридного методу та інших систем напряму залежить від швидкодії компонентів, що використовуються. Наприклад, якщо замінити модель Phi-3 (що відповідає за 73% часових витрат) на легшу, то швидкодія системи буде на рівні Ghostbusters і Fast DetectGPT. З іншого боку, якщо в Ghostbuster замінити проксі-моделі з сімейства GPT-2 на більш сучасні, то час роботи методу буде порівняним з DetectGPT. Отже, виміри витрат часу показали, що побудована модель є на рівні з іншими SOTA рішеннями, і її швидкодія є достатньою для практичного використання.

4.3. Архітектура та програмна реалізація системи ідентифікації

З метою практичної перевірки гібридного методу було створено програмний комплекс з використанням мови Python. Даний комплекс відтворює повний цикл обробки тексту: від блоку канонізації тексту та обчислення індивідуальних ознак до отримання прогнозу метакласифікатора та ІСР і створення карти доказів. Система побудована за модульним принципом, що забезпечує її гнучкість щодо інтеграції в інші системи та можливості щодо нарощування обчислювальних потужностей.

4.3.1. Модульна структура та логіка каскадного ансамблю

Архітектура програмного комплексу реалізує схему каскадної взаємодії модулів відповідно до концептуальної моделі зображеної на рис. 2.2. Життєвий цикл системи та її програмна реалізація розділені на два незалежні етапи:

1. Підготовка та навчання. Включає збір референсних даних для лексичного (частоти слів) та синтаксичного модулів (частоти POS Tag триграм), навчання Reasoning Style класифікатора, навчання слабких класифікаторів для M_{lex} , M_{syn} , M_{sem} , підбір оптимальних порогів та конфігурацій, тренування метакласифікатора, калібрування порогів невідповідності ICP.
2. Практичне використання. Це основний сценарій використання системи, коли відбувається аналіз нових вхідних текстів, генеруються передбачення, класи ICP та візуалізації інтерпретаційної моделі. У цьому сценарії використовуються всі компоненти системи, що було створено на етапі 1.

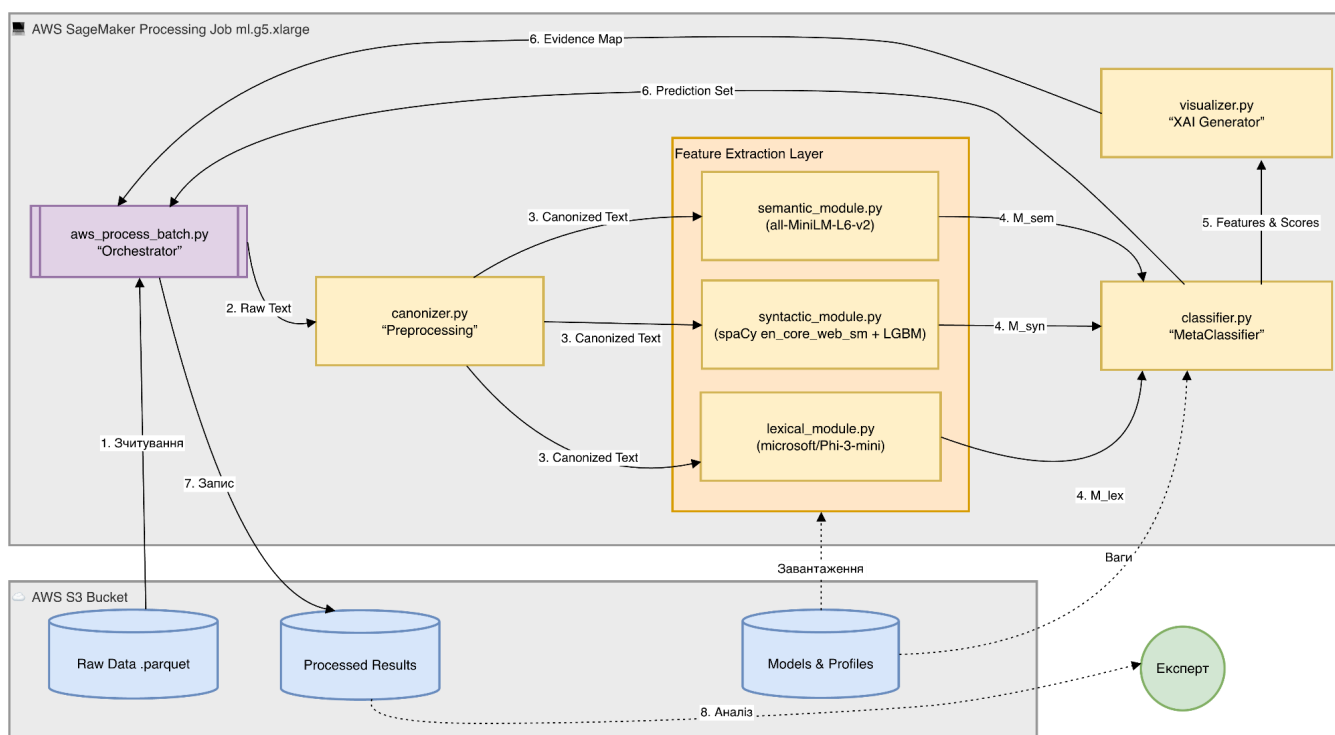


Рисунок 4.9 Діаграма потоків даних для етапу практичного використання системи

Діаграма роботи основного сценарію використання системи зображена на рис. 4.9. Через високі обчислювальні вимоги систему розгорнуто в хмарі AWS на інстансі ml.g5.xlarge (GPU NVIDIA A10G 24 GB, 4 vCPU), тому всі обчислювальні процеси відбуваються там. Всі дані (тексти для аналізу, натреновані моделі, референсні дані та результати роботи) збережено на S3. Керування всіма процесами та передачею даних (оркестрація) відбувається з локальної машини шляхом запуску скрипту `aws_process_batch`; скрипт створює інстанс та середовище для виконання, встановлює необхідні залежності, передає потрібні параметри та специфіку завдання, відстежує прогрес та може, за потреби, автоматично завантажити результати на локальну машину.

Пайплайн обробки даних побудовано таким чином, що спочатку відбувається приведення даних до канонічної форми та (при досягненні певних умов) встановлення прапорця F_{attack} модулем *canonizer.py*. Наступним кроком дані передаються в шар екстракції ознак, що складається з трьох модулів: M_{lex} , M_{syn} , M_{sem} . Однією з переваг розробленої системи є те, що ці три модулі є абсолютно незалежними, тож допускається їх паралельне виконання в умовах комерційного використання. Лексичний екстрактор (*lexical_module.py*) завантажує LLM Phi-3-mini-4k-instruct для використання в ролі проксі-моделі для обчислення імовірностей токенів; ця інформація використовується в розрахунках perplexity, burstiness, ентропії. Також модуль обчислює інші ознаки лексичного рівні на базі токенів, символів та пунктуації. Лексичний модуль може встановити прапорець F_{attack} при перевищенні ознакою χ^2 (формула (2.23)) певних порогових значень. Всі ці ознаки разом формують вектор ознак модуля M_{lex} . Синтаксичний екстрактор (*syntactic_module.py*) використовує синтаксичний парсер `en_core_web_sm` (або `en_core_web_trf` для більш складних випадків) для побудови дерев залежностей та використовує натренований Reasoning Style класифікатор; модуль також обчислює набір ознак на базі глибини синтаксичного дерева, комбінацій POS-тегів, довжин речень і інших властивостей синтаксичного рівня. Разом ці ознаки формують вектор

ознак модуля M_{syn} . Семантичний екстрактор (*semantic_module.py*) використовує трансформерну модель all-MiniLM-L6-v2 для перетворення речень на семантичні вектори, обчислює відстань між ними, знаходить семантичні розриви в тексті, розраховує дрифт центроїда та інші ознаки на базі зв'язності тексту. Разом ці ознаки формують вектор ознак модуля M_{sem} .

Всі обчислені ознаки разом з F_{attack} і ймовірністю Reasoning Style передаються в метакласифікатор (*classifier.py*), де, на базі ознак кожного рівня, натренована модель-класифікатор повертає імовірність того, що текст написано ШІ. Повний вектор ознак ($M_{lex}, M_{syn}, M_{sem}, S_{mismatch}, Score_{freq}, F_{attack}$) передається в Adaptive Gating Mechanism, який зображено на рис. 2.3. Цей механізм працює за логікою, зображеною на рис. 4.10.

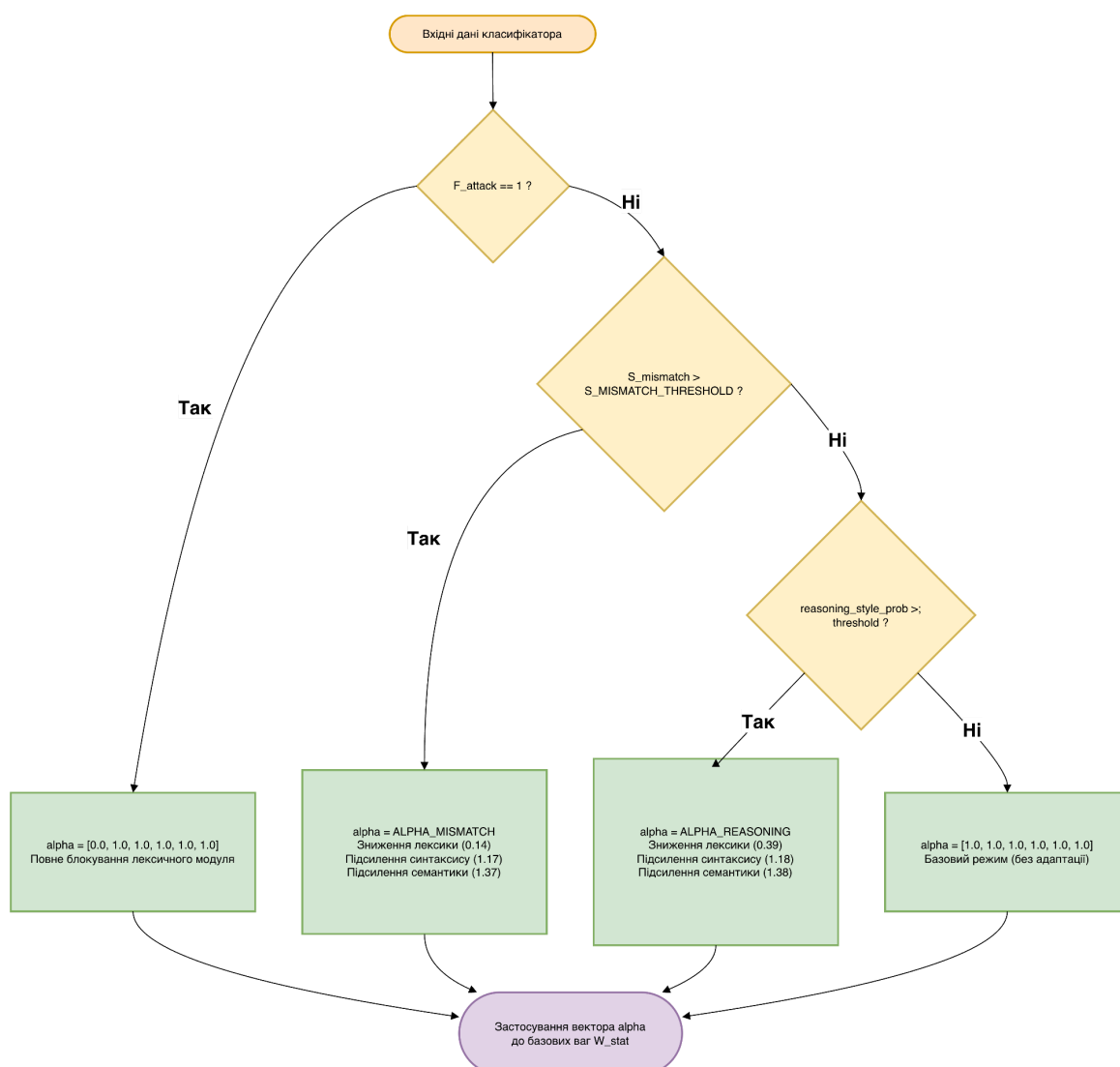


Рисунок 4.10 Логіка роботи Adaptive Gating Mechanism

Після встановлення коефіцієнтів довіри метакласифікатор обчислює фінальну `AI_probability`; ця імовірність опрацьовується ICP, який і генерує фінальну множину міток для тексту. `AI_probability` та множина міток ICP повертаються оркестратору, а `AI_probability` та значення фіч та передбачень окремих класифікаторів передається в модуль візуалізації (*visualizer.py*), який генерує глобальний та локальний профілі атрибуції, профіль Reasoning стилю та Карту доказів. Всі ці артефакти повертаються оркестратору, який їх разом з передбаченнями, отриманими від `classifier.py`, зберігає на S3. Залежно від заданого при запуску оркестратора сценарію, результати можуть також бути автоматично завантажені на локальну машину для аналізу експертом.

4.3.2. Вибір технологічного стека та реалізація ICP

Як основну мову програмування було обрано Python 3.13, яка є де-факто галузевим стандартом для проведення досліджень з машинного навчання та створення комерційних бібліотек. Вибір певної версії Python і деяких інших ключових бібліотек (на кшталт `transformers`), перш за все пов'язаний з обмеженнями AWS середовища, де, як правило, використовуються специфічні адаптовані версії пакетів та діють деякі обмеження на підтримувані бібліотеки щодо їх сумісності з AWS.

Для роботи з LLM у лексичному модулі використано бібліотеку `transformers` 4.36.0 від HuggingFace, її використання дає змогу запускати модель на різних пристроях (MPS, GPU, CPU) через застосування опції `device_map="auto"` та різних типів оптимізації, наприклад, квантизації (`torch.float16`) для зменшення споживання пам'яті. Для роботи семантичного модуля необхідна бібліотека `sentence-transformers` 2.6.1 для створення ембедінгів речень моделлю `all-MiniLM-L6-v2`. Для лінгвістичного аналізу та POS-тегування використовуються бібліотеки `spacy` та `nltk`. Метакласифікатор та інші статистичні моделі реалізовані на базі `scikit-learn` 1.7.2 і `pumpru`, для слабких класифікаторів і класифікатора Reasoning-стилю

використовується бібліотека `lightgbm`. Для візуалізації використовуються бібліотеки `matplotlib`, `seaborn` і `IPython`. Для роботи з даними та обчисленнями в хмарі AWS використовуються бібліотеки `boto3` та `sagemaker`. Для читання, збереження та обробки даних використовуються бібліотеки `joblib`, `pyarrow`, `pandas`, `json` та інші.

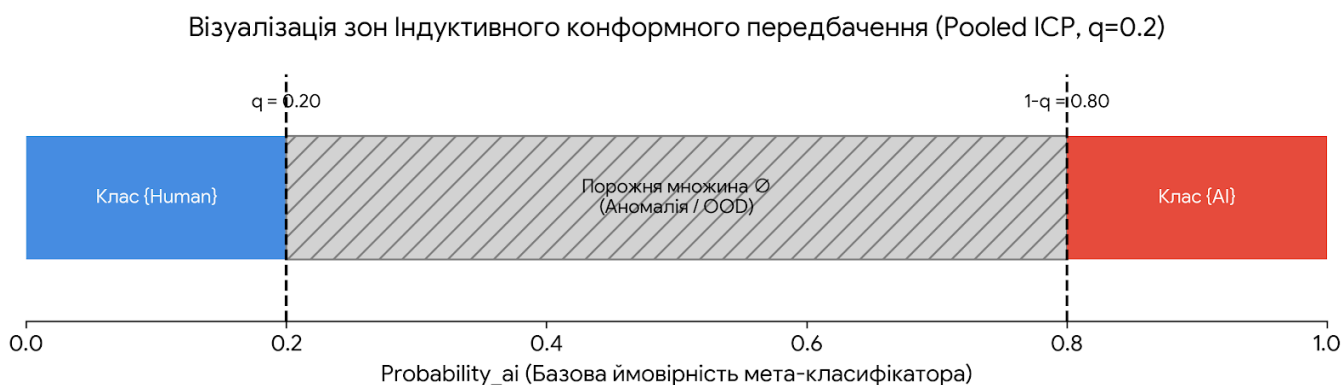


Рисунок 4.11 Вплив ICP на передбачення метакласифікатора

Метакласифікатор у гібридному методі не видає бінарний прогноз, замість цього повертається множина передбачень згенерована ICP. ICP (індуктивне конформне передбачення) виконує роль фінального компонента системи прийняття рішень і забезпечує строгі математичні гарантії надійності прогнозів з заданим рівнем впевненості. Для отримання рівня впевненості 95% рівень значущості має бути встановлено на 0.05 і визначено єдиний квантиль q шляхом калібрування. У створеній системі було застосовано об'єднаний підхід до калібрування (коли використовується один квантиль q для обох класів). На етапі підготовки функція `compute_calibration_scores` на валідаційній вибірці розраховує міру невідповідності для кожного тексту як $1 - P(y_{True})$, після чого обчислюється квантиль q для розрахованого розподілу помилок як точка відсікання прогнозів для отримання 95% правильних прогнозів. На рис. 4.11 зображено вплив ICP на фінальний вибір міток класу через множину передбачень.

На рис. 4.12 зображена детальна схема роботи методу ICP після калібрування, коли квантиль q успішно визначено і потрібно аналізувати передбачення метакласифікатора на невідомих даних.

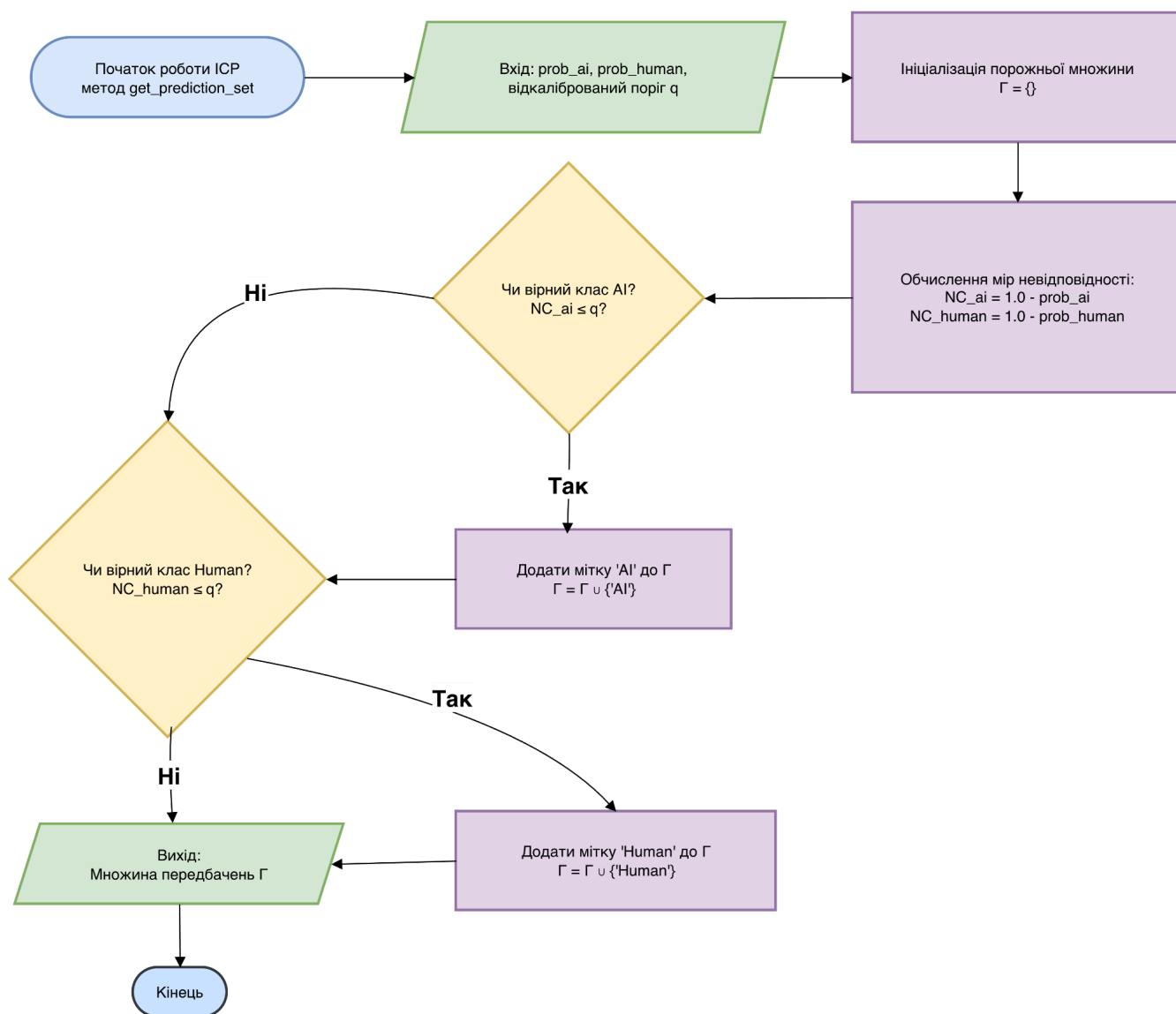


Рисунок 4.12 Блок-схема роботи ICP

Алгоритм формування множини передбачень методом ICP

- На вхід подається прогноз метакласифікатора у вигляді точкової імовірності класу AI та відкалібрований поріг невідповідності q .
- Вихідні дані - множина передбачень Γ , що може набувати значень: $\{\text{Human}\}$, $\{\text{AI}\}$, $\{\text{Human}, \text{AI}\}$ або бути порожньою.

1. Ініціалізувати порожню множину: $\Gamma = \{\}$.
2. Обчислити міри невідповідності для обох класів: $NC_{AI} = 1 - prob_{AI}$;
 $NC_{Human} = 1 - prob_{Human}$, де $prob_{Human} = 1 - prob_{AI}$.

3. Якщо $NC_{AI} \leq q$, то додати мітку AI до множини Г.
4. Якщо $NC_{Human} \leq q$, то додати мітку Human до множини Г.
5. Повернути Г.

Використання ICP гарантує валідність розробленої системи. З погляду імовірностей, якщо вхідний текст сильно модифіковано (тобто дані стали OOD), то метакласифікатор не зможе видати високу ймовірність для жодного з класів (такий приклад розібрано в пункті 4.4.2). Тоді міри невідповідності NC_{AI} та NC_{Human} перевищать калібрувальний поріг q і ICP поверне порожню множину, що свідчить про епістемічну невизначеність. Це дає змогу системі коректно ідентифікувати аномалію і передати документ на розгляд експерту, надавши Карту доказів та глобальний і локальний профілі атрибуції, замість генерації хибного результату.

4.4. Реалізація інтерпретаційної моделі обґрунтування

Забезпечення прозорості та пояснюваності рішень є однією з ключових вимог до розробленої системи, оскільки наявність доступу до доказової бази та можливість порівняти вплив на детектор різних версій документа є важливим елементами прийняття фінального рішення у сферах з високою ціною помилки, наприклад, в академічній доброчесності. У розробленій системі цю функцію виконує модуль пояснюваного ІШ, що заснований на принципах викладених у підрозділі 2.4.

4.4.1. Програмна генерація профілю атрибуції та Карти доказів

Як і було запропоновано в підрозділі 2.4, система складається з трьох основних компонент: глобальний профіль атрибуції, що дещо нагадує матеріал викладений у пункті 4.1.4, локальний профіль та Карту доказів. Глобальний профіль демонструє загальну поведінку гібридного детектора і яку важливість він надає різним модулям при прийнятті рішення; локальний профіль демонструє процес прийняття рішення для конкретного файлу і базується на формулі (2.30); Карта

доказів у візуальній формі демонструє розподіл perplexity для тексту та його аналіз з боку Reasoning Style класифікатору. Суттєвою перевагою запропонованого методу порівняно з SHAP та LIME є те, що замість багатьох повторних обчислень, він використовує обчислені для класифікатора ознаки, відповідно, його часова складність є $O(1)$.

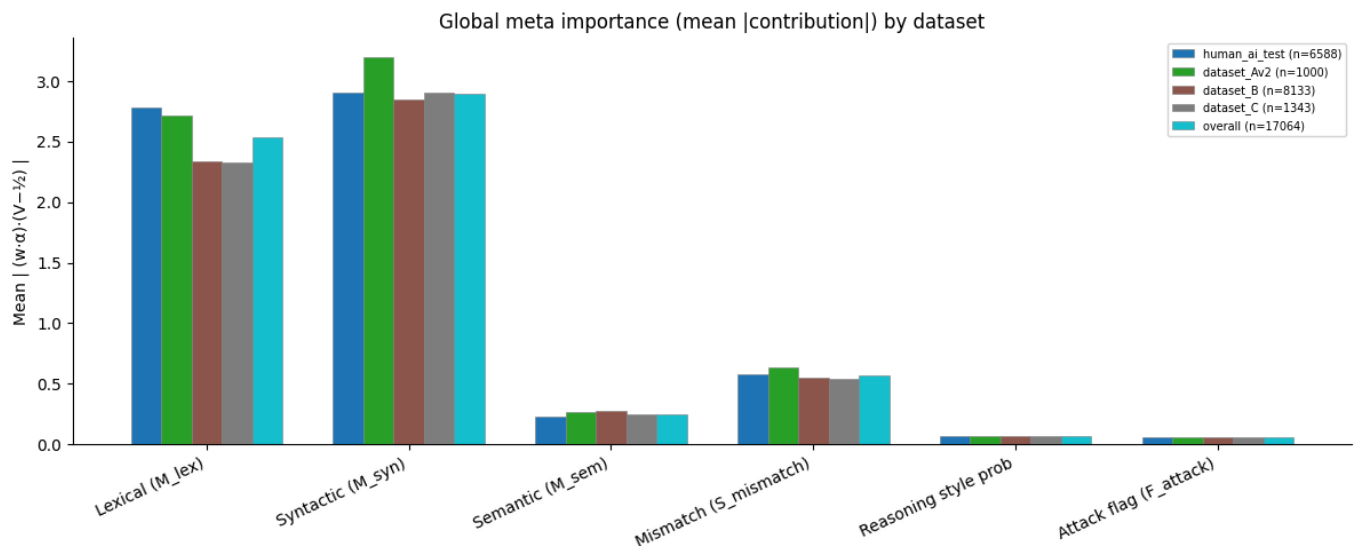


Рисунок 4.13 Глобальний профіль атрибуції для різних датасетів

Глобальний профіль атрибуції базується на формулі (2.31) і обчислює $GlobalImportance_j$ для кожної ознаки на базі $Contribution_j^{centered}$. Він дає змогу показати, наскільки кожен конкретний модуль (M_{lex} , M_{syn} , M_{sem}) та інші ознаки метакласифікатора сильно впливали на прийняття рішень загалом; профіль не розрізняє впливу на прийняття позитивних чи негативних рішень, а показує глобальну важливість ознаки. Оскільки $GlobalImportance$ обчислюється як середнє по набору даних, то для різних датасетів вона може дещо відрізнятися, на рис. 4.13 зображено $GlobalImportance$ для кожної ознаки загалом (блакитні стовпчики) і в рамках кожного з датасетів. З графіка видно, що важливість ознак трохи коливається для різних датасетів; також цікавим фактом є те, що важливість ознаки M_{sem} є нижчою за важливість інших модулів, проте вона отримує додаткову важливість як частина $S_{mismatch}$. Важливо зазначити, що цей розподіл показує середнє значення у датасеті, тому важливість буде зміщеною в бік ознак, що гарно демонструють себе

на великій кількості текстів. Наприклад, якщо говорити про важливість M_{sem} , то аналіз, викладений у 4.1.4, показав, що це ключова ознака для виявлення DIPPER, але оскільки DIPPER має всього 633 тексти (8% від розміру датасета B), то природно, що це непомітно на графіку *GlobalImportance*. Глобальний профіль атрибуції показує долю впливу кожної ознаки, він не свідчить про надлишковість чи неважливість окремих ознак.

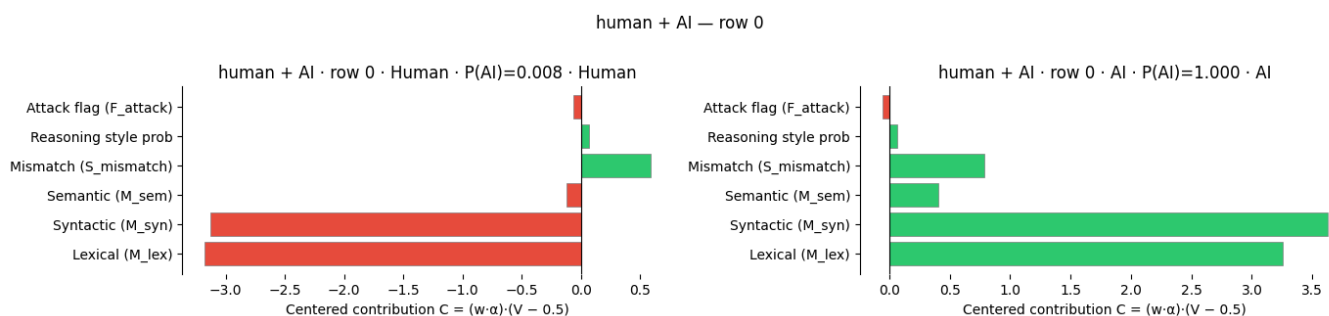


Рисунок 4.14 Локальний профіль атрибуції для двох текстів

Локальний профіль атрибуції показує прямий вплив кожної ознаки на остаточне прийняття рішення. Використання $Contribution_j^{centered}$ дає змогу вказати в бік якого класу кожна ознака зміщує рішення класифікатора. На рис. 4.14 показано локальний профіль атрибуції для пари текстів: зліва текст, написаний людиною, а справа його відповідник згенерований моделлю Cohere Command-R-Plus. Видно, що лексичні, синтаксичні та семантичні ознаки голосують у бік вірного класу, силу такого голосування відображено на осі ОХ: у першому випадку (на графіку зліва) M_{lex} і M_{syn} набувають великих негативних значень (сильно впевнені, що мітка Human), а M_{sem} майже нейтральний; у другому випадку (на графіку справа) M_{lex} і M_{syn} набувають великих позитивних значень (сильно впевнені, що мітка AI), M_{sem} з ними погоджується, хоча і не так сильно.

Карта доказів має вказати на конкретні елементи тексту, чи його фрагменти, що було згенеровано, вона дає змогу експерту провести найбільш детальний рівень аналізу. У пункті 2.4.3 планувалося створення трирівневої Карти доказів: лексичного, синтаксичного та семантичного рівня. Лексичний рівень базується на

вимірюванні perplexity проксі-моделлю, відповідно до запропонованого у 2.4.3; синтаксичний базується на аналізі Reasoning класифікатора, як це запропоновано в 3.4.2; щодо семантичного рівня, то практичний експеримент показав, що вимірювання семантичних розривів не досягає потрібної роздільної здатності, тож цей параметр було прибрано. Також частиною Карти доказів є аналіз структурних загроз модулем канонізації, як це було показано в підрозділі 3.2 на рис. 3.1. Для більш детального показу perplexity роздільну здатність для лексичного модуля було змінено з $\{ \leq 100, \leq 1000, > 1000 \}$ на $\{ \leq 100, \leq 250, \leq 500, \leq 750, \leq 1000, > 1000 \}$.

4.4.2. Аналіз прикладних кейсів

У цьому пункті проведено детальний аналіз роботи локального профілю атрибуції та Карти доказів на конкретних прикладах.

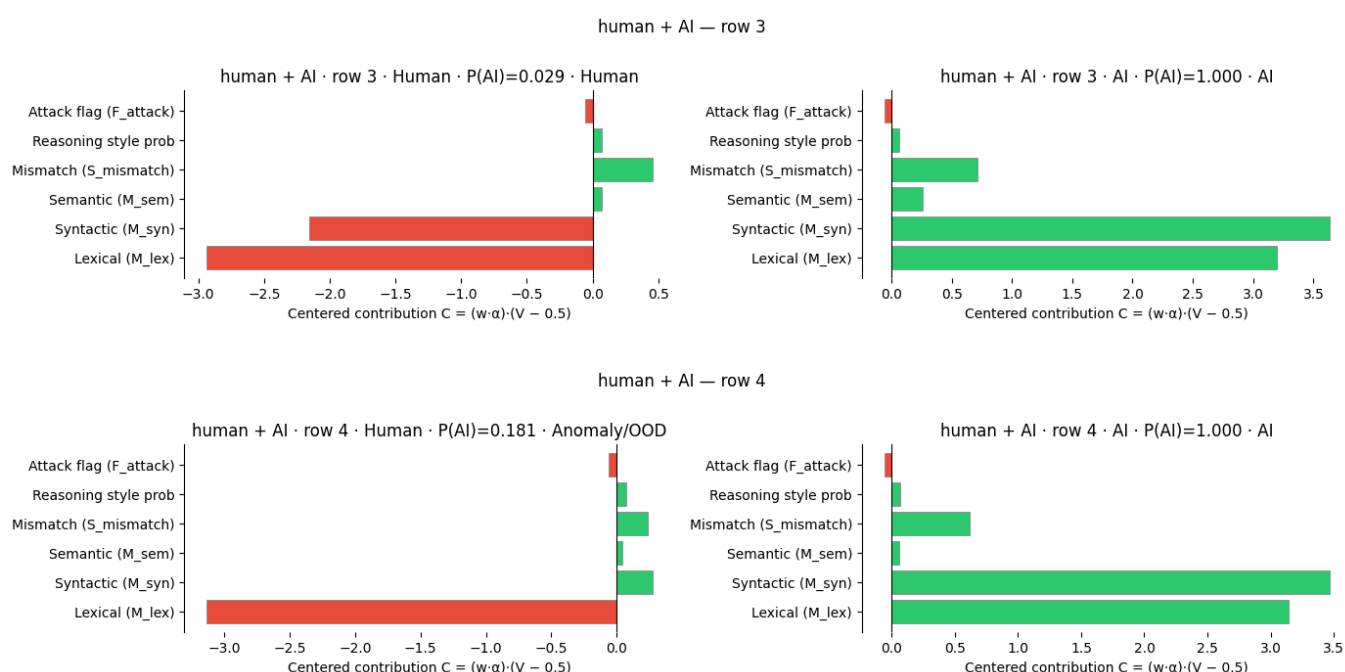


Рисунок 4.15 Локальний профіль атрибуції для двох пар текстів (Human/AI)

На рис. 4.15 показано дві пари текстів: ліворуч Human, праворуч AI (на першому Cohere Command-R-Plus (Reasoning), на другому Llama-4 Maverick 17b (Instant)). Видно, що Human тексти сприймаються класифікатором по-різному, хоча і належать обидва до корпусу PILO [16], проте різним авторам; тож у першому випадку і синтаксис і лексика говорять що це людський текст, а в другому випадку

синтаксис уже не певен і навіть трохи схиляється до AI. Щодо AI текстів, то попри те, що їх згенеровано зовсім різними моделями, проте для гібридного детектора вони виглядають однаково.

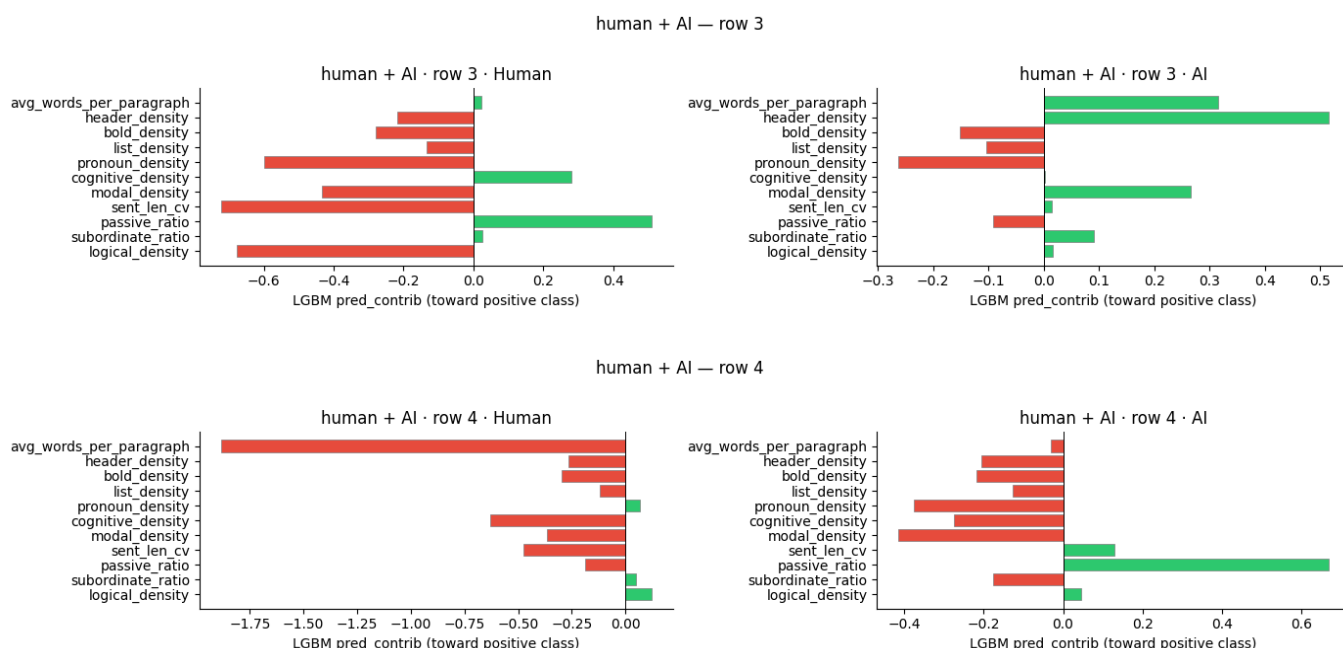


Рисунок 4.16 Reasoning слід для двох пар текстів (Human/AI)

Проаналізуємо ті самі дві пари текстів з погляду Reasoning класифікатора як елемента Карти доказів (рис. 4.16). На графіку ліва сторона відповідає за негативне рішення (не Reasoning), а права за позитивне (Reasoning). Обидва Human тексти, попри свою різну природу, отримують сильно негативну сумарну оцінку, що є коректним результатом; перший AI текст отримує сумарну позитивну оцінку, а другий негативну, що також є коректним результатом. Тобто Reasoning класифікатор здатен точно відрізнити Human і AI(Instant) тексти від AI (Reasoning). Як видно з графіка ключову роль у розпізнаванні Reasoning Trace зіграли такі ознаки як середня кількість слів у параграфі, щільність заголовків та щільність модальних дієслів.

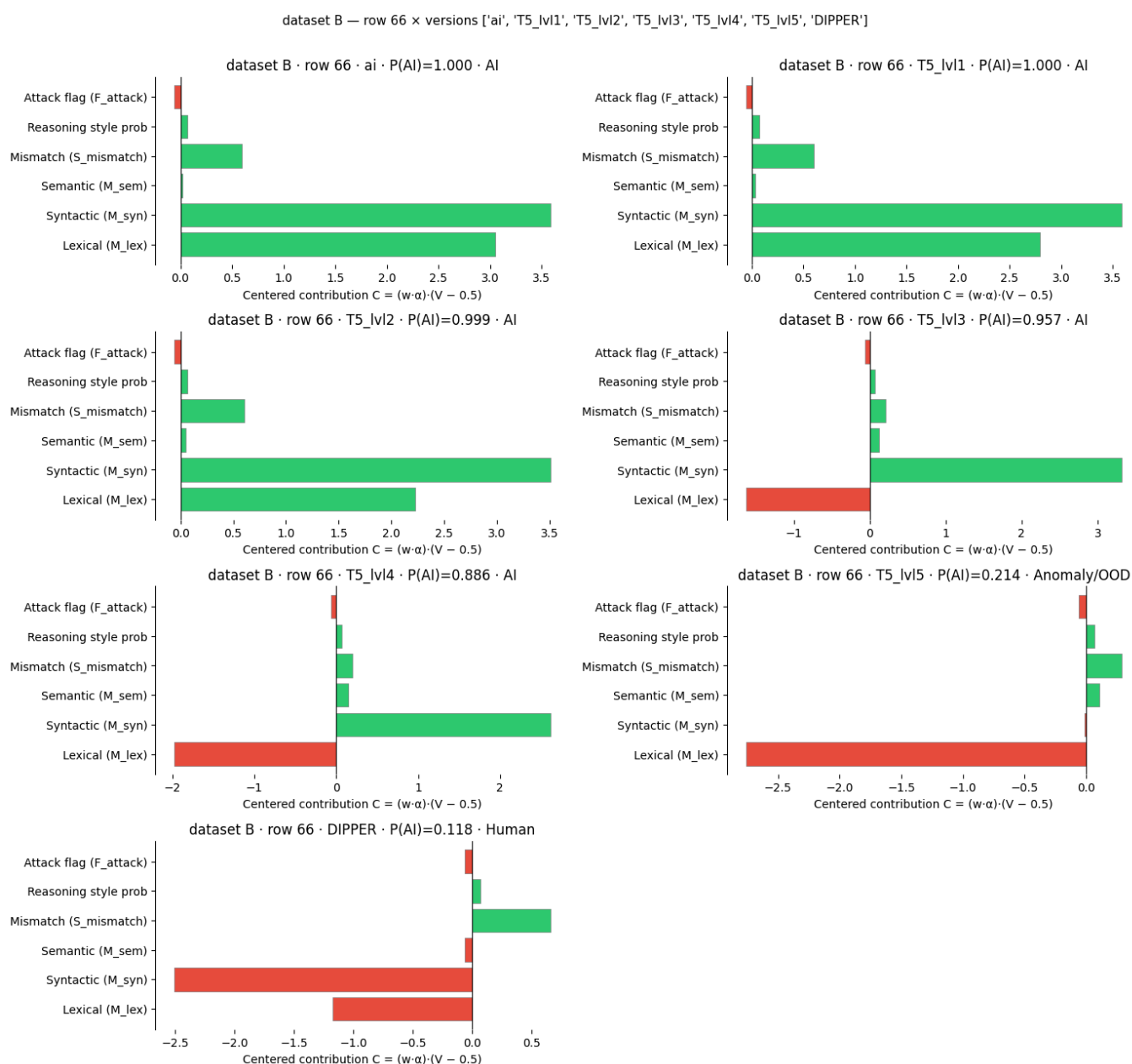


Рисунок 4.17 Вплив трансформацій на локальний профіль атрибуції

На рис. 4.17 зображений локальний профіль атрибуції для різних версій одного тексту - починаючи від чистого AI варіанту і далі з п'ятьма рівнями перетворень T5 і одним перетворенням від DIPPER. Розглянемо етапи трансформацій:

1. На початку гібридний метод був дуже впевнений в AI природі чистого тексту, тож $P(AI)=1$;

2. Після першого перетворення T5 (lvl1) впевненість гібридного класифікатора залишилась так само на рівні 1, проте дещо зменшилась впевненість M_{lex} ;
3. Після перетворення lvl2 впевненість класифікатора знизилась до 0.999 і знизилась впевненість обох M_{lex} і M_{syn} ;
4. На lvl3 впевненість класифікатора знизилась до 0.957, на цьому етапі M_{lex} “зламався” і почав вказувати на Human клас, впевненість M_{syn} залишилась високою;
5. На lvl4 впевненість класифікатора знизилась до 0.886, також знизилась впевненість M_{syn} і підсилилась помилка M_{lex} ;
6. На lvl5 (що здійснює видалення випадкових слів) зламався M_{syn} , який тепер не вказує на жоден клас, і з'явилась висока впевненість у помилковому рішенні у M_{lex} . Варто зазначити, що M_{sem} продовжив вказувати на AI клас. На цьому етапі спрацював ICP і замість помилкового рішення {Human} видав порожню множину {OOD/Anomaly};
7. DIPPER остаточно зламав всі три рівні й класифікатор видав клас Human.

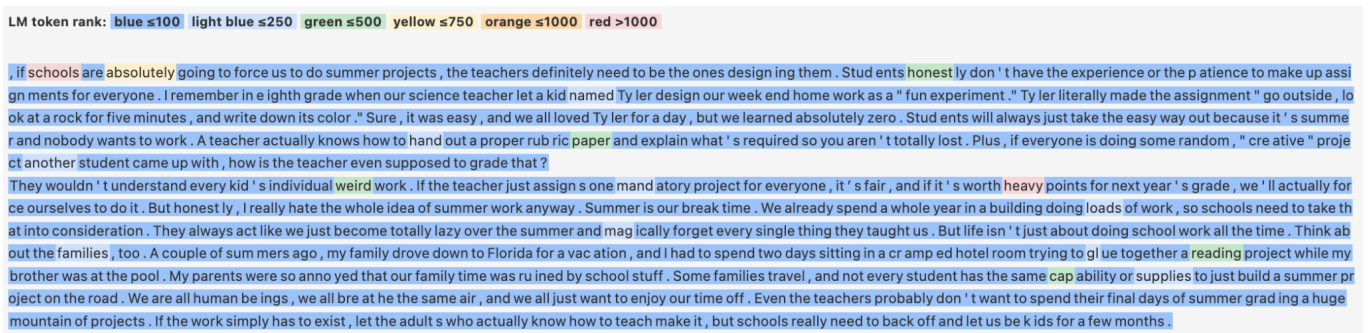


Рисунок 4.18 Карта доказів (perplexity) для чистого AI текст

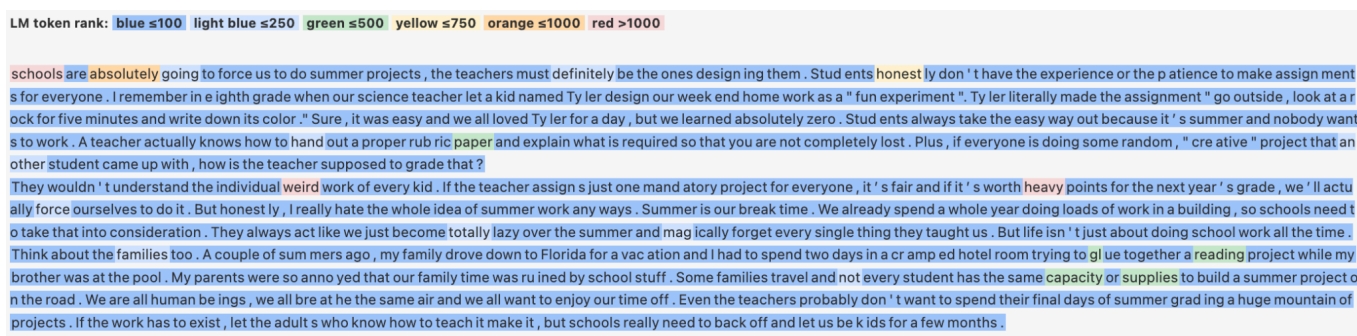


Рисунок 4.19 Карта доказів (perplexity) для АІ текст після атаки T5 lvl1

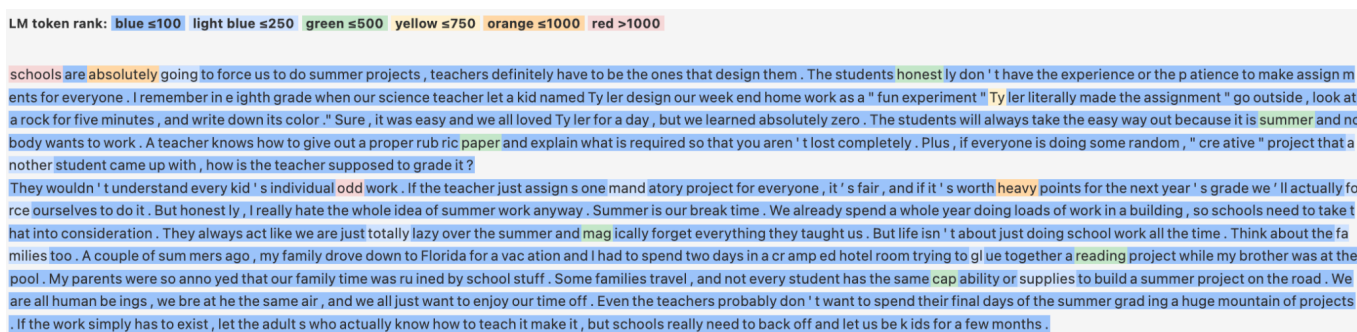


Рисунок 4.20 Карта доказів (perplexity) для АІ текст після атаки T5 lvl2

Для того щоб зрозуміти, що відбувалося з M_{lex} на всіх етапах трансформації, скористаємося Картою доказів. Рис. 4.18 демонструє perplexity для чистого АІ тексту, порівнюючи з рис. 4.19 видно, що кількість рідких незначно виросла на T5 lvl1. T5 lvl2 (рис. 4.20) теж суттєвих змін не вніс, проте вже на T5 lvl3 (рис. 4.21) не тільки збільшилась кількість рідкісних токенів (що характеризує людський текст), а й змінилась і структура самого тексту - з'явилися короткі абзаци. Це той момент, коли "зламався" M_{lex} . Між lvl3 і lvl4 (рис. 4.22) відбувається не тільки збільшення кількості рідкісних токенів, а й перехід деяких токенів у вищі категорії, наприклад, токен "our" з категорії дуже частих (blue) перейшов до категорії дуже рідкісних (red) через те, що парафразер зламав речення перед ним. На lvl3 фраза звучала як "kid named Tyler let our weekend", а на lvl4 вже як "kid named Tyler our weekend", тобто видалення "let" змінило perplexity для "our". Навіть якщо кількість таких змін невелика, їх сукупний ефект цілком ламає лексику, що і видно за реакцією M_{lex} .

LM token rank: blue ≤100 light blue ≤250 green ≤500 yellow ≤750 orange ≤1000 red >1000

, if schools are going to force us to do summer projects, the teachers may need to design them as you should. Students honestly don't have the experience or the patience to make everything for everyone. I remember in eighth grade when our science teacher let a kid named Tyler let our weekend homework idea be as a «funny experiment»? Tyler made the assignment literally "go out, look at a rock for five minutes and write it down." It was easy - and Tyler loved us all for a day - but we learned absolutely nothing!! Every student will take it easy because it is summer and nobody wants to work. A teacher knows how to hand out a proper rubric sheet and explain what is required so that you aren't totally lost. Plus, if everyone is doing some random "creative" project another student made, how is the teacher supposed to class? They would not know the individual weird work of every kid. It is fair if the teacher assigns everyone with just a compulsory class, and if it is worth heavy points for next year's grade, we will actually force ourselves to do it. But really, I hate the whole idea of summer work anyway! Summer is our break time. Everybody spends a year doing loads of work in a building, so schools have to take this into consideration. They always act as though we get completely lazy during the summer and wonder what every less on that they teach us gets forgotten. But life isn't just about school assignments for always making time. Think also about the families, too. My family drove down to Florida for a vacation a couple of summers ago and I had to spend two days sitting in a cramped hotel room trying to put together a reading project while my brother was relaxing at the pool. My parents were so annoyed that our family had our time ruined by school stuff. Some families travel but not every student has the same tools or materials to just build a summer project on the road. We are all human beings, we all breathe the same air and we will all just take our time off. Even the teachers probably don't want to spend their last days of summer grading a huge mountain of projects. If the work has to exist simply let the adults who actually know how to teach make it, but schools really need to turn into children for a few months.

Рисунок 4.21 Карта доказів (perplexity) для AI текст після атаки T5 lvl3

LM token rank: blue ≤100 light blue ≤250 green ≤500 yellow ≤750 orange ≤1000 red >1000

, if schools are going to force us to do summer projects, the teachers may need to design them as you should. Students honestly don't have the experience or the patience to make everything for everyone. I remember in eighth grade when our science teacher let a kid named Tyler let our weekend homework idea be as a «funny experiment»? Tyler made the assignment literally "go out, look at a rock for five minutes and write it down." It was easy - and Tyler loved us all for a day - but we learned absolutely nothing!! Every student will take it easy because it is summer and nobody wants to work. A teacher knows how to hand out a proper rubric sheet and explain what is required so that you are not completely lost. Plus, if everyone is doing some random "creative" project another student made, how is the teacher supposed to class? They would not know the individual weird work of each kid. It is fair if the teacher assigns everyone with a compulsory class and if it is worth heavy points for the next year's grade, we will actually force ourselves to do it. But really, I hate the whole idea of summer work anyway! Summer is our break time. Everybody spends a year doing loads of work in a building, so schools have to take this into consideration. They always act as though we get completely lazy during the summer and wonder what every less on that they teach us gets forgotten. But life isn't just about school assignments for always making time. Think also about the families. My family drove to Florida a couple of summers ago and I had to spend two days in a cramped hotel room trying to put together a reading project while my brother was relaxing at the pool. My parents were so annoyed that our family had our time ruined by school stuff. Some families travel, but not every student has the same tools or materials to build a summer project on the road. We are all human beings, we all breathe the same air and we all will take our time off. Even the teachers probably don't want to spend their last days of summer grading a huge mountain of projects. If the work has to exist, let the adults who actually know how to teach make it, but schools really need to turn into children for a few months.

Рисунок 4.22 Карта доказів (perplexity) для AI текст після атаки T5 lvl4

Через видалення слів на lvl5 (рис. 4.23) perplexity багатьох tokenів стає високою - речення зламані, проксі-модель не може передбачити такі продовження, синтаксичний модуль перестав працювати через синтаксичні помилки, гібридний метод схиляється до неправильного рішення, проте ICP зупиняє його і вказує, що текст є нетиповим і моделі варто утриматися від передбачення на ньому.

LM token rank: blue ≤100 light blue ≤250 green ≤500 yellow ≤750 orange ≤1000 red >1000

, if schools are going to force us to do summer projects, the teachers may need to design them as you should. I honestly don't have the experience or the to make for I remember in eighth grade when science let a kid Tyler let our homework idea be as a «experiment»? The assignment literally made "go look at a for minutes and write it down." It was easy - Tyler loved us all a day - but we learned absolutely nothing!! Every will because it is and nobody works. A teacher knows how to hand out the proper rubric sheet explain what is required so are completely lost. Plus, everyone is doing some random "creative" project that the student made, the to class? They would not know the individual work of the kid. It is if the assigns everyone with a compulsory class if the next year is worth heavy, we actually force ourselves to do it. But really I think of summer work anyway! Summer is our break time. A year of work in a building, so schools have this into consideration. They always act even though we get completely lazy during the summer and wonder what every less on gets forgotten. Life isn't about school assignments for making time. Think also about the families. My family drove to Florida a couple of summers ago and I spent two days in a hotel room trying to put together a reading project at the pool. My parents so that our family had our ruined by school stuff. Some travel, but every student has the same tools or materials build the summer project on the road. We all are human beings, we all take the same air and we take our time. Even the teachers probably want to last of grading a huge mountain of projects. If work has to exist, let the adults who know how to teach it make it, but schools really need to turn into children for a few months.

Рисунок 4.23 Карта доказів (perplexity) для AI текст після атаки T5 lvl5

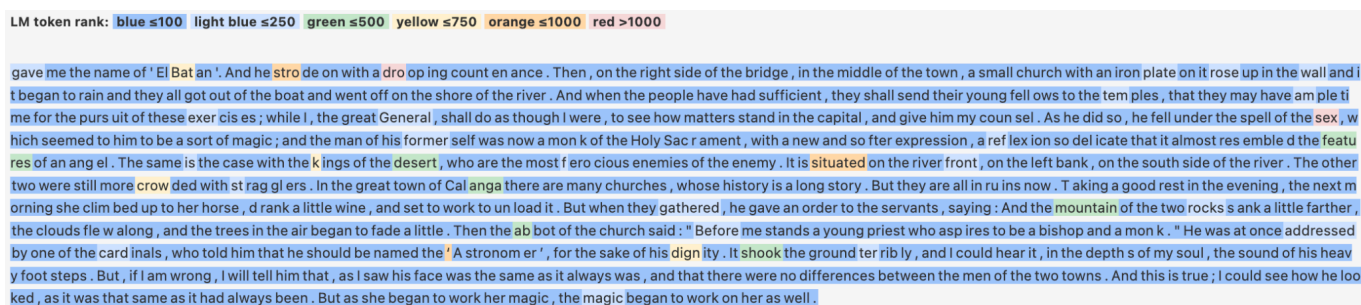


Рисунок 4.24 Карта доказів (perplexity) для AI текст після атаки DIPPER

Дані, згенеровані DIPPER (рис. 4.24), належать до сильних рівнів перефразування з високою температурою, тому текст містить дуже незвичайні синтаксичні структури й місцями незв'язну лексику (хоча і менш незв'язну, порівняно з T5 lvl5). Такі зміни дозволяють одночасно ввести в оману і M_{lex} і M_{syn} , а втрата узгодженості між реченнями виводить з ладу M_{sem} . Оскільки в такому випадку DIPPER атакує всі три рівні, хоча і з різною інтенсивністю, то атака проходить успішно і гібридний класифікатор видає хибний прогноз - Human. Проте з погляду вартості атаки I_{sem} атака є надзвичайно дорогою, через те, що отриманий текст не має жодної практичної цінності, оскільки його не можна використовувати в реальному світі через відсутність змісту.

Підсумовуючи результати емпіричного аналізу, можна стверджувати, що розроблена інтерпретаційна модель обґрунтування доказів детекції виступає потужним та точним інструментом пояснюваного штучного інтелекту, вона дає змогу перейти від моделі чорного ящика до прозорого прийняття рішень щодо детекції ШІ. Запропонована трирівнева модель пояснень дає змогу розглянути прогноз детектора з різних сторін і надає достатньо інформації експерту для прийняття остаточного рішення в критичних сферах, на кшталт академічної доброчесності. Елементами наукової новизни є як трирівнева модель, так і сам алгоритмічний метод побудови пояснень: на відміну від обчислювально важких, ітеративних методів ХАІ типу SHAP і LIME, запропонована модель здійснює обчислення внеску різних параметрів на основі вже обчислених векторів-ознак метакласифікатора. Це дає змогу знизити часову складність інтерпретаційної моделі

до $O(1)$. Аналіз прикладних кейсів із застосуванням атак перефразування емпірично показує як цінність запропонованої моделі XAI, так і ступінь поступової втрати точності базових модулів. Що емпірично показує робастність розробленої системи, де зі зростанням складності атаки поступово відбувається деградація точності з поетапним спрацюванням механізмів захисту. При цьому Карта доказів наочно демонструє причини такої деградації та умови спрацювання ICP.

4.5. Висновки до розділу

У четвертому розділі описано цикл розробки, програмної реалізації та експериментальної валідації гібридного методу ідентифікації автоматично згенерованих текстів. Отримані результати підтверджують заявлену наукову новизну та практичну цінність роботи:

1. Вперше розроблено програмний комплекс ідентифікації текстів, згенерованих ШІ, на базі гібридного методу. На відміну від наявних SOTA-детекторів, таких як Fast-DetectGPT, Radar чи Ghostbusters, що критично втрачають точність під дією комерційних парафразерів, запропонований комплекс демонструє високу стійкість. Завдяки архітектурі каскадного ансамблю (лексичного, синтаксичного та семантичного рівнів), алгоритмам канонізації та механізму адаптивного зважування ознак, він дає змогу досягти показника точності $F1 = 0.92$ на стаціонарних даних та зберегти повноту виявлення (Recall) на рівні 88.77% за жорсткого обмеження рівня хибних спрацювань до $FPR = 1\%$. В умовах протидії сучасним комерційним системам маскування (StealthGPT, AIHumanize, Phrasly) система перевершує світові аналоги, забезпечуючи середній Recall на рівні 77.16% та найнижчий серед аналогів рівень успішності атак ($ASR = 22.84\%$). Це становить практичне втілення та доведення Наукової новизни 1.
2. Набув подальшого розвитку метод забезпечення робастності систем детекції в умовах епістемічної невизначеності та аналізу даних поза навчальним розподілом (OOD). На відміну від традиційних систем чорної скриньки, що

- генерують хибнопозитивні результати з надмірною впевненістю на невідомих даних, запропоновані методи забезпечують високу статистичну достовірність прогнозів. Завдяки інтеграції фреймворку індуктивного конформного передбачення (ICP) на етапі ухвалення фінального рішення вони дозволяють автоматично ідентифікувати сумнівні тексти та переводити їх у категорію аномалій. Експериментально доведено, що на складних нетипових даних (тексти неносіїв мови з іспиту IELTS) алгоритм забезпечує коефіцієнт порятунку (Rescue Rate) на рівні 79,71%. Це дозволило знизити фінальний показник хибнопозитивних спрацювань (FPR) системи з потенційних 4,81% до 0,98%, ефективно захищаючи авторів-людей від автоматичної дискримінації. Отримані результати становлять практичне підтвердження Наукової новизни 3.
3. Удосконалено концептуальну модель оцінки робастності систем детекції. На відміну від емпіричних підходів тестування без чітко визначених меж стійкості, модель Точки зламу дає змогу здійснювати точний прогностичний аудит вразливості класифікаторів. Такий результат досягається шляхом програмного моделювання взаємозв'язку між втратою семантичної цілісності I_{sem} та ймовірністю виявлення P_{det} на великих наборах маскованих даних, створених T5 та DIPPER. Емпірично доведено, що розроблений метод успішно зміщує межу критичної вразливості (Точку зламу) до рівня коефіцієнта заміни $RR > 0.9$, де текст повністю втрачає свій оригінальний зміст ($I_{sem} < 0.5$). Це робить застосування атак перефразуванням для обходу гібридного детектора функціонально безглуздим, що підтверджує Наукову новизну 4.
 4. Вперше розроблено програмну підсистему інтерпретації доказів детекції пояснюваного ШІ (XAI). На відміну від обчислювально важких ітеративних XAI-методів типу LIME/SHAP, що вимагають багаторазових запитів до моделі, запропонована підсистема дає змогу знизити часову складність генерації пояснень до $O(1)$. Це здійснюється шляхом прямого використання попередньо обчислених векторів-ознак та внутрішніх ваг метакласифікатора. Модуль автоматично генерує глобальні та локальні профілі атрибуції, а також

інтерактивні Карті доказів, візуалізуючи аномалії лексичної перплексії та структурних загроз. Загальна асимптотична складність системи, що складається з гібридного методу та підсистеми інтерпретації, дорівнює $O(N^2)$, а середній час обробки одного документа у хмарному середовищі AWS становить 0,33 секунди. Це гарантує прозорість рішень та готовність технології до промислового застосування, що становить практичне втілення Наукової новизни 2.

Список використаних джерел до розділу 4

1. Hans, Abhimanyu, et al. "Spotting llms with binoculars: Zero-shot detection of machine-generated text." *arXiv preprint arXiv:2401.12070* (2024).
2. Mitchell, Eric, et al. "Detectgpt: Zero-shot machine-generated text detection using probability curvature." *International conference on machine learning*. PMLR, 2023.
3. Bao, Guangsheng, et al. "Fast-detectgpt: Efficient zero-shot detection of machine-generated text via conditional probability curvature." *arXiv preprint arXiv:2310.05130* (2023).
4. Li, Yafu, et al. "MAGE: Machine-generated text detection in the wild." *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2024.
5. Hu, Xiaomeng, Pin-Yu Chen, and Tsung-Yi Ho. "Radar: Robust ai-text detection via adversarial learning." *Advances in neural information processing systems* 36 (2023): 15077-15095.
6. Verma, Vivek, et al. "Ghostbuster: Detecting text ghostwritten by large language models." *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. 2024.
7. "AI Writing." *Turnitin*, 2026, www.turnitin.com/solutions/topics/ai-writing/. Accessed 14 Mar. 2026.
8. *Originality.ai*, 2026, originality.ai/. Accessed 14 Mar. 2026.
9. *GPTZero*. 2026, gptzero.me/. Accessed 28 Mar. 2026.
10. Mazlumi. "IELTS Writing Scored Essays Dataset." *Kaggle*, 2023, www.kaggle.com/datasets/mazlumi/ielts-writing-scored-essays-dataset. Accessed 29 Mar. 2026.
11. Mathew, Sajee, and J. Varia. "Overview of amazon web services." *Amazon Whitepapers* 105.1 (2014): 22.
12. Abdin, Marah, et al. "Phi-4 technical report." *arXiv preprint arXiv:2412.08905* (2024).

- 13.Liu, Yinhan, et al. "Roberta: A robustly optimized bert pretraining approach." *arXiv preprint arXiv:1907.11692* (2019).
- 14.Wang, Wenhui, et al. "Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers." *Advances in neural information processing systems* 33 (2020): 5776-5788.
- 15.Lingamallu, Phani Kumar, and Fabio Oliveira. *AWS Observability Handbook: Monitor, trace, and alert your cloud applications with AWS' myriad observability tools*. Packt Publishing Ltd, 2023.
- 16.Holmes, Langdon, et al. "PIILO: an open-source system for personally identifiable information labeling and obfuscation." *Information and Learning Sciences* 124.9-10 (2023): 266-284.

ВИСНОВКИ

У дисертаційній роботі розв'язано актуальну науково-прикладну задачу, що полягає в розробці гібридного методу ідентифікації автоматично згенерованих природномовних текстів, стійкого до трансформаційних атак маскування та епістемічної невизначеності.

За результатами проведеного наукового дослідження зроблено такі висновки:

1. Проведено систематизацію та аналіз існуючих підходів до ідентифікації автоматично згенерованих текстів. Підходи розділено на п'ять категорій: статистичні, нейромережеві, методи на основі *zero-shot*, *watermarking* та *XAI*. Для кожної категорії визначено основні принципи, на базі яких вона працює, її переваги, недоліки та обмеження. Встановлено, що наявні детектори є вразливими до методів активного маскування (атак перефразуванням, підміни символів) та стикаються з проблемою епістемічної невизначеності при аналізі даних поза навчальним розподілом (OOD-даних). Щодо методів маскування встановлено відповідність між типом детектора та методом атаки, що дає змогу уникнути ідентифікації:

- для статистичних детекторів - атаки семантичної реконструкції на кшталт DIPPER і рекурсивного перефразування;
- для нейромережевих детекторів - атаки змагального розподілу, на кшталт змагального перефразування та структурні атаки (типу гомогліфів);
- для детекторів на основі *zero-shot* - атаки змагального розподілу, на кшталт перефразування з підвищенням температури;
- для детекторів на основі *watermarking* - атаки семантичної реконструкції на кшталт багаторазового перекладу і атаки оптимізаційною чисткою, що замінюють марковані токени;
- для *XAI* - маніпуляції поясненнями з метою маскування ШІ.

Доведено, що детекція ШІ, заснована лише на одному рівні ознак (наприклад, тільки на лексичній перплексії), є вразливою до

трансформаційних атак і призводить до високого рівня хибнопозитивних спрацювань (FPR) та ризику несправедливих звинувачень авторів-людей.

2. Розроблено математичну модель сліду III в багатовимірному просторі ознак $\mathbf{F} = \mathbf{F}_{lex} \times \mathbf{F}_{syn} \times \mathbf{F}_{sem}$ (лексичних, синтаксичних та семантичних), яка формалізує його у вигляді вектора $T_{AI}(X) = \Phi(f_{lex}(X), f_{syn}(X), f_{sem}(X))$. Поєднання трьох рівнів аналізу дає змогу зменшити негативні наслідки традиційних систем детекції шляхом підвищення стійкості до маскування та зниження FPR. Введено основні рівні аналізу:

- лексичний (\mathbf{F}_{lex}), що фіксує імовірнісне згладжування тексту генеративними моделями через перплексію та вибуховість (burstiness);
- синтаксичний (\mathbf{F}_{syn}), що вимірює структурну однотонність згенерованого тексту через дивергенцію Кульбака-Лейблера для N-грам POS-тегів відносно людського еталона, а також через дисперсію глибини синтаксичних дерев;
- семантичний (\mathbf{F}_{sem}), що виявляє аномально високу локальну узгодженість ембедінгів сусідніх речень та глобальний дрифт тематики.

На основі математичної моделі удосконалено концептуальну модель оцінки робастності систем детекції шляхом введення формалізованого поняття Точки зламу. Ця модель відображає математичний взаємозв'язок між інтенсивністю структурних трансформацій тексту (коефіцієнт заміни RR), збереженням його семантичної цілісності (I_{sem}) та ймовірністю виявлення (P_{det}). Точка зламу демонструє критичне значення RR , після досягнення якого ймовірність детекції P_{det} падає нижче порогового значення 0.5, що переміщує текст із зони стабільної детекції у зону успішного маскування. Під час комп'ютерного моделювання атак перефразуванням (із застосуванням моделей T5 та DIPPER) було підтверджено обернено пропорційну залежність між RR та I_{sem} . Експерименти показали, що для передових SOTA-рішень (зокрема, Fast-DetectGPT) Точка зламу настає вже за мінімальних трансформацій ($RR =$

0.03), що свідчить про їхню вразливість. Разом з тим, застосування запропонованого гібридного методу дає змогу змістити Точку зламу екстремально праворуч ($RR > 0.90$). Це фактично нівелює зону успішного маскування, оскільки для обходу гібридного детектора зловмиснику необхідно внести таку кількість змін, що $RR > 0.5$, що призводить до критичної втрати семантичної цілісності тексту.

3. Вперше розроблено гібридний метод ідентифікації автоматично згенерованих текстів на основі архітектури каскадного ансамблю. На відміну від традиційного простого об'єднання ознак в один вектор, запропонований метод розподіляє аналіз між трьома незалежними модулями: лексичним - M_{lex} , синтаксичним - M_{syn} та семантичним - M_{sem} . Кожен із модулів генерує власну незалежну оцінку, яка передається до метакласифікатора для прийняття остаточного рішення. Шляхом застосування механізмів адаптивного зважування та аналізу міжрівневої узгодженості ознак через розрахунок сигналу розбіжності $S_{mismatch}$ метод фіксує когнітивний дисонанс у тексті під час застосування трансформаційних атак. Оскільки під час трансформаційної атаки методам маскування вкрай важко змінити стилістику, синтаксис та семантику одночасно і пропорційно, будь-яка атака створює розрив між рівнями. Цей розрив аналізується через механізм адаптивного зважування, що дає змогу використовувати дворівневу логіку ваг: $w_j(X) = w_j^{stat} \cdot \alpha_j(X)$, тобто вивчена статистична вага множиться на динамічний коефіцієнт довіри. Залежно від типу виявленої атаки, алгоритм змінює довіру до різних модулів, завдяки чому спроба обходу детектора на одному рівні автоматично стає тригером для підвищення ймовірності детекції на інших рівнях.

Проведені експериментальні дослідження на датасетах, згенерованих 15 сучасними LLM (включаючи MoE та Reasoning-моделі) і оброблених просунутими системами перефразування (DIPPER та T5) та комерційними байпасерами (Phrasly.ai, StealthGPT, AIHumanize), підтвердили ефективність розробленого методу. Доведено, що гібридний метод здатен змістити Точку

зламу системи екстремально праворуч ($RR > 0.90$), що означає повне нівелювання зони успішного маскування.

4. Набув подальшого розвитку метод забезпечення робастності систем обробки природної мови шляхом інтеграції фреймворку індуктивного конформного передбачення (ICP) на етапі ухвалення фінального рішення метакласифікатором. Класичні методи детекції генерують точкові ймовірності з надмірною впевненістю і не містять інформації про те, чи відповідає текст, що аналізується, текстам із тренувальної вибірки. Це робить їх критично вразливими до даних поза навчальним розподілом (OOD), наприклад, текстів неносіїв мови, академічних текстів або нестандартних стилів письма, що призводить до зростання кількості хибнопозитивних спрацювань. Для розв'язання цієї проблеми розроблено надбудову, яка дає змогу перетворювати евристичні точкові прогнози на математично обґрунтовані множини прогнозів $\Gamma^\epsilon(x)$. Обчислюючи міри неконформності на калібрувальній вибірці, алгоритм ICP гарантує, що істинний клас тексту належатиме до згенерованої множини із заданим рівнем статистичної достовірності $1 - \epsilon$.

Ключовою перевагою застосування ICP є зміна парадигми класифікації в умовах невизначеності. Для типових текстів система видає чіткі одноелементні множини ($\{\text{Human}\}$ або $\{\text{AI}\}$), а під час обробки нетипових OOD-текстів фреймворк генерує багатоеlementні $\{\text{Human}, \text{AI}\}$ або порожні множини. У контексті детекції ШІ порожня множина інтерпретується як відмова системи від прийняття рішення, що дає змогу уникнути хибних прогнозів у випадку нестандартних даних. Експериментальна перевірка довела високу ефективність запропонованого підходу, зокрема, в умовах відсутності ICP на текстах неносіїв мови (датасет IELTS) спостерігався $FPR = 4.81\%$. Інтеграція фреймворку ICP дала змогу системі розпізнати епістемічну невизначеність цих текстів та згенерувати порожні множини у складних випадках. Це призвело до зниження рівня хибнопозитивних спрацювань FPR до 0.98% , забезпечивши коефіцієнт порятунку авторів-людей (*Rescue Rate*) на рівні 79.71% . Таким чином, практично доведено, що застосування фреймворку

ICP здатне суттєво мінімізувати ризики несправедливих звинувачень у використанні генеративного ШІ.

5. Вперше розроблено інтерпретаційну модель обґрунтування доказів детекції пояснюваного ШІ (XAI). Її створення розв'язує проблему чорної скриньки, що існує у випадку класичних детекторів, результатами яких часто бракує прозорості для ухвалення рішень експертами. На відміну від обчислювально ресурсомістких ітеративних методів (типу LIME та SHAP), запропонована модель використовує внутрішні ваги метакласифікатора для прямої генерації глобального та локальних профілів атрибуції. Це дало змогу знизити часову складність побудови пояснень до $O(1)$, що робить систему придатною для синхронного використання в реальних системах.

Глобальний профіль атрибуції дає змогу визначити, які ознаки відіграють найбільшу роль під час аналізу текстів гібридним методом загалом. Локальний профіль атрибуції показує вплив кожного з параметрів метакласифікатора на фінальне рішення щодо конкретного тексту, а інтерактивні Картти доказів візуалізують аномалії лексичної перплексії та структурних загроз. Комбінація цих трьох рівнів інтерпретації результатів детекції забезпечує експертів додатковими даними для прийняття остаточного рішення.

6. Розроблено та експериментально провалідовано програмний комплекс ідентифікації автоматично згенерованих текстів. Систему розгорнуто з використанням хмарної інфраструктури AWS (Amazon Web Services), що забезпечує її високу відмовостійкість, масштабованість та можливість паралельної обробки запитів під час аналізу великих масивів даних у режимі реального часу. Загальна асимптотична складність системи, що складається з гібридного методу та підсистеми інтерпретації, дорівнює $O(N^2)$, а середній час обробки одного документа у хмарному середовищі AWS становить 0,33 секунди. Програмна реалізація розробленого гібридного методу дала змогу провести тестування на масштабному репрезентативному корпусі, що містить дані, згенеровані 15 сучасними мовними моделями, включаючи архітектури

Mixture-of-Experts та Reasoning. Експериментально доведено перевагу гібридного методу над SOTA-аналогами (DetectGPT, Fast-DetectGPT, Binoculars, Radar, Ghostbusters, MAGE). Створена система досягає показника $F1=0.92$ (порівняно з $F1=0.6227$ у найближчого SOTA-методу) на стаціонарних даних зі збереженням повноти виявлення (Recall) на рівні 88.77% в умовах жорсткого обмеження $FPR=1\%$.

Ключовою перевагою гібридного методу є його здатність ефективно функціонувати в умовах трансформаційних атак T5 та DIPPER, досягаючи середнього Recall на рівні 0.7827, при середньому Recall на рівні 0.5924 у найближчого SOTA-методу. Окрім цього, метод показав високу робастність під час активної протидії комерційним сервісам маскування (StealthGPT, AILHumanize, Phrasly), забезпечивши найнижчий серед усіх досліджуваних систем рівень успішності атак ($ASR=22.84\%$, при $ASR=35.49\%$ у найближчого SOTA-методу) та $Recall=0.7716$, при $Recall=0.6451$ у найближчого SOTA-методу.

Практичне значення одержаних результатів полягає у створенні готового до промислового використання інструментарію для систем контент-модерації, інформаційної безпеки та платформ перевірки академічної доброчесності.

Експериментально підтверджена ефективність реалізації гібридного методу у протидії сучасним комерційним засобам маскування тексту (зокрема, *StealthGPT*, *AILHumanize*, *Phrasly*) із доведеним зниженням рівня успішності атак (ASR) до 22.84%, що є найнижчим показником серед досліджуваних аналогів. Впровадження методики застосування фреймворку ICP дало змогу суттєво знизити рівень хибнопозитивних спрацювань під час аналізу текстів неносіїв мови (на прикладі IELTS) з 4.81% до 0.98% (коефіцієнт порятунку *Rescue Rate* = 79.71%), що глобально мінімізує ризики несправедливих звинувачень авторів-людей. Розроблена підсистема інтерпретації (глобальний та локальний профілі атрибуції та Карти доказів) надає експертам вичерпну та прозору аналітичну базу для ухвалення остаточних рішень у спірних ситуаціях.

Подальші перспективи наукового дослідження доцільно спрямувати на:

- адаптацію запропонованого гібридного методу та архітектури каскадного ансамблю до багатомовного середовища;
- дослідження та виділення нових специфічних метрик сліду ШІ для нових типів архітектур LLM;
- удосконалення алгоритмів розпізнавання текстів зі змішаною природою (наприклад, редагування людиною згенерованого тексту) із деталізацією Карт доказів до рівня окремих фраз;
- оптимізацію обчислювальної складності синтаксичного та лексичного модулів шляхом заміни важких LLM-проксі на дистильовані моделі для зменшення вимог до обладнання;
- розширення інтерпретаційної моделі XAI для аналізу специфічних доменів текстових даних, на кшталт програмного коду, юридичних та медичних документів.

ДОДАТКИ

ДОДАТОК А

Список опублікованих праць за темою дисертації:

– статті в іноземних виданнях, в яких опубліковані основні наукові результати дисертації:

1. Faure, Emil, and Andrii Nykonenko. "Analyzing the nature of AI footprint: noise-to-text method." *Proceedings of the Computational Intelligence Application Workshop (CIAW 2024)*, Lviv, Ukraine, October 10-12, 2024. CEUR Workshop Proceedings, Vol-3861, paper 14. Online [CEUR-WS.org/Vol-3861/paper14.pdf](https://ceur-ws.org/Vol-3861/paper14.pdf). ISSN 1613-0073. DOI: <https://doi.org/10.13140/RG.2.2.15592.02561>. Видання індексується в наукометричній базі Scopus та DBLP.

Особистий внесок автора полягає у розробленні нового методу ітераційної трансформації даних “noise-to-text” для дослідження природи цифрового відбитку штучного інтелекту, проведенні експериментальних досліджень із використанням моделі gpt-3.5-turbo та аналізі чутливості сучасних систем детекції ШІ (Turnitin, GPTZero, Originality.ai та інших) і становить 0,5 друк. арк.

– статті у наукових фахових виданнях України, в яких опубліковані основні наукові результати дисертації:

2. Nykonenko, A. "The impact of artificial intelligence on modern education: prospects and challenges." *Artificial Intelligence 2* (2023): 10-15. ISSN 2710-1673. ONLINE ISSN 2710-1681. DOI: <https://doi.org/10.15407/jai2023.02.010>. Фаховий категорії Б.

3. Nykonenko, A. "How text transformations affect AI detection." *Artificial Intelligence 4* (2024): 233-241. ISSN 2710-1673. ONLINE ISSN 2710-1681. DOI: <https://doi.org/10.15407/jai2024.04.233>. Фаховий категорії Б.

4. Никоненко, А. О. "Систематичний огляд досягнень в галузі LLM від GPT-3 до міркуючих агентів." *Stuc. intelekt* 30.3 (2025): 32-43. ISSN 2710-1673. ONLINE ISSN 2710-1681. DOI: <https://doi.org/10.15407/jai2025.03.032>. Фаховий категорії Б.

– наукові праці, що засвідчують апробацію матеріалів дисертації:

5. Никоненко А.О. Штучний інтелект як засіб трансформації освіти // Тези доповідей III Міжнародної науково-практичної конференції «Інформаційні технології в освіті та науці» (ІТОН-2023), м. Запоріжжя-Мелітополь, 25-26 травня 2024 р., с.328-330.

6. Никоненко А.О. Підходи до визначення текстів згенерованих штучним інтелектом // Тези доповідей VII Міжнародної науково-практичної конференції «Інформаційні технології в освіті, науці і техніці» (ІТОНТ-2024), ЧДТУ, м. Черкаси, 23-24 травня 2024 р., с.142-144.

7. E. Faure, A. Nykonenko. Noise-to-text method analysis for evaluating AI-generated texts. Proceedings of the 1st International Scientific and Practical Conference «COMPUTATIONAL INTELLIGENCE AND SMART SYSTEMS» (CISS-2024). 10–12 October 2024, Lviv, Ukraine.

Особистий внесок автора полягає в обґрунтуванні концепції та розробленні алгоритму оцінювання систем аналізу ШІ-генерованих текстів на основі методу “noise-to-text”, проведенні серії експериментів із трансформації даних за допомогою LLM та систематизації результатів тестування сучасних детекторів і становить 0,15 друк. арк.

8. Никоненко А.О. Вплив трансформацій тексту на виявлення штучного інтелекту // Тези доповідей XXIV міжнародної науково-технічної конференції «Штучний інтелект та інтелектуальні системи - AIIS'2024», м. Київ, Україна, 18-19 жовтня 2024 р.

ДОДАТОК Б

Розподіл моделей за наборами даних

corpus	split	model_type	model_name	count
PIILO	test	Instant	bedrock-llama-4-maverick-17b	91
			bedrock/mistral-large-3	91
			claude-sonnet-4-5-20250929-v1	92
			deepseek-ai/DeepSeek-V3.1	91
			gemini-flash-latest	92
			gpt-5.2-2025-12-11	92
		Reasoning	bedrock-llama-4-scout-17b	61
			bedrock-qwen-3-next-80b	61
			bedrock/ai21-jamba-1-5-large	61
			bedrock/cohere-command-r-plus	61
			bedrock/deepseek-R1	61
			claude-opus-4-6-v1	61
			gemini-3.1-pro-preview	61
			grok-4-1-fast-reasoning	61
			o3-2025-04-16	62
	train	Instant	bedrock-llama-4-maverick-17b	375
			bedrock/mistral-large-3	375
			claude-sonnet-4-5-20250929-v1	375
			deepseek-ai/DeepSeek-V3.1	375
			gemini-flash-latest	375
			gpt-5.2-2025-12-11	375
		Reasoning	bedrock-llama-4-scout-17b	250
			bedrock-qwen-3-next-80b	250
			bedrock/ai21-jamba-1-5-large	250
			bedrock/cohere-command-r-plus	250
			bedrock/deepseek-R1	250
			claude-opus-4-6-v1	250
			gemini-3.1-pro-preview	250
			grok-4-1-fast-reasoning	250
			o3-2025-04-16	250
---	---	---	---	---
persuade	test	Instant	bedrock-llama-4-maverick-17b	91
			bedrock/mistral-large-3	91
			claude-sonnet-4-5-20250929-v1	92
			deepseek-ai/DeepSeek-V3.1	91

			gemini-flash-latest	92
			gpt-5.2-2025-12-11	92
		Reasoning	bedrock-llama-4-scout-17b	61
			bedrock-qwen-3-next-80b	61
			bedrock/ai21-jamba-1-5-large	61
			bedrock/cohere-command-r-plus	61
			bedrock/deepseek-R1	61
			claude-opus-4-6-v1	61
			gemini-3.1-pro-preview	61
			grok-4-1-fast-reasoning	61
			o3-2025-04-16	62
	train	Instant	bedrock-llama-4-maverick-17b	375
			bedrock/mistral-large-3	375
			claude-sonnet-4-5-20250929-v1	375
			deepseek-ai/DeepSeek-V3.1	375
			gemini-flash-latest	375
			gpt-5.2-2025-12-11	375
		Reasoning	bedrock-llama-4-scout-17b	250
			bedrock-qwen-3-next-80b	250
			bedrock/ai21-jamba-1-5-large	250
			bedrock/cohere-command-r-plus	250
			bedrock/deepseek-R1	250
			claude-opus-4-6-v1	250
			gemini-3.1-pro-preview	250
			grok-4-1-fast-reasoning	250
			o3-2025-04-16	250
---	---	---	---	---
wiki40B	test	Instant	bedrock-llama-4-maverick-17b	91
			bedrock/mistral-large-3	91
			claude-sonnet-4-5-20250929-v1	92
			deepseek-ai/DeepSeek-V3.1	91
			gemini-flash-latest	92
			gpt-5.2-2025-12-11	92
		Reasoning	bedrock-llama-4-scout-17b	61
			bedrock-qwen-3-next-80b	61
			bedrock/ai21-jamba-1-5-large	61
			bedrock/cohere-command-r-plus	61
			bedrock/deepseek-R1	61
			claude-opus-4-6-v1	61

			gemini-3.1-pro-preview	61
			grok-4-1-fast-reasoning	61
			o3-2025-04-16	62
	train	Instant	bedrock-llama-4-maverick-17b	375
			bedrock/mistral-large-3	375
			claude-sonnet-4-5-20250929-v1	375
			deepseek-ai/DeepSeek-V3.1	375
			gemini-flash-latest	375
			gpt-5.2-2025-12-11	375
		Reasoning	bedrock-llama-4-scout-17b	250
			bedrock-qwen-3-next-80b	250
			bedrock/ai21-jamba-1-5-large	250
			bedrock/cohere-command-r-plus	250
			bedrock/deepseek-R1	250
			claude-opus-4-6-v1	250
			gemini-3.1-pro-preview	250
			grok-4-1-fast-reasoning	250
			o3-2025-04-16	250

ДОДАТОК В

Розподіл системних prompts за наборами даних

corpus	split	model_type	model_name	system_prompt_name	count
PIILO	test	Instant	bedrock-llama-4-maverick-17b	The "Authentic Human" Constraint	30
				The "Earnest Student" Persona	31
				The "Reflective Practitioner" Persona	30
			bedrock/mistral-large-3	The "Authentic Human" Constraint	30
				The "Earnest Student" Persona	31
				The "Reflective Practitioner" Persona	30
			claude-sonnet-4-5-20250929-v1	The "Authentic Human" Constraint	30
				The "Earnest Student" Persona	31
				The "Reflective Practitioner" Persona	31
			deepseek-ai/DeepSeek-V3.1	The "Authentic Human" Constraint	30
				The "Earnest Student" Persona	31
				The "Reflective Practitioner" Persona	30
			gemini-flash-latest	The "Authentic Human" Constraint	30
				The "Earnest Student" Persona	31
				The "Reflective Practitioner" Persona	31
			gpt-5.2-2025-12-11	The "Authentic Human" Constraint	30
				The "Earnest Student" Persona	31
				The "Reflective Practitioner" Persona	31
		Reasoning	bedrock-llama-4-scout-17b	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			bedrock-qwen-3-next-80b	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			bedrock/ai21-jamba-1-5-large	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			bedrock/cohere-command-r-plus	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20

				The Stylistic Deconstruction Framework	21
			bedrock/deepseek-R1	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			claude-opus-4-6-v1	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			gemini-3.1-pro-preview	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			grok-4-1-fast-reasoning	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			o3-2025-04-16	The "Drafting Process" Emulation	21
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
	train	Instant	bedrock-llama-4-maverick-17b	The "Authentic Human" Constraint	125
				The "Earnest Student" Persona	125
				The "Reflective Practitioner" Persona	125
			bedrock/mistral-large-3	The "Authentic Human" Constraint	125
				The "Earnest Student" Persona	125
				The "Reflective Practitioner" Persona	125
			claude-sonnet-4-5-20250929-v1	The "Authentic Human" Constraint	125
				The "Earnest Student" Persona	125
				The "Reflective Practitioner" Persona	125
			deepseek-ai/DeepSeek-V3.1	The "Authentic Human" Constraint	125
				The "Earnest Student" Persona	125
				The "Reflective Practitioner" Persona	125
			gemini-flash-latest	The "Authentic Human" Constraint	125
				The "Earnest Student" Persona	125
				The "Reflective Practitioner" Persona	125

			gpt-5.2-2025-12-11	The "Authentic Human" Constraint	125
				The "Earnest Student" Persona	125
				The "Reflective Practitioner" Persona	125
		Reasoning	bedrock-llama-4-scout-17b	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			bedrock-qwen-3-next-80b	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			bedrock/ai21-jamba-1-5-large	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			bedrock/cohere-command-r-plus	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			bedrock/deepseek-R1	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			claude-opus-4-6-v1	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			gemini-3.1-pro-preview	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			grok-4-1-fast-reasoning	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84

			o3-2025-04-16	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
---	---	---	---	---	---
persuade	test	Instant	bedrock-llama-4-maverick-17b	The "Authentic Human" Constraint	30
				The "Earnest Student" Persona	31
				The "Reflective Practitioner" Persona	30
			bedrock/mistral-large-3	The "Authentic Human" Constraint	30
				The "Earnest Student" Persona	31
				The "Reflective Practitioner" Persona	30
			claude-sonnet-4-5-20250929-v1	The "Authentic Human" Constraint	30
				The "Earnest Student" Persona	31
				The "Reflective Practitioner" Persona	31
			deepseek-ai/DeepSeek-V3.1	The "Authentic Human" Constraint	30
				The "Earnest Student" Persona	31
				The "Reflective Practitioner" Persona	30
			gemini-flash-latest	The "Authentic Human" Constraint	30
				The "Earnest Student" Persona	31
				The "Reflective Practitioner" Persona	31
			gpt-5.2-2025-12-11	The "Authentic Human" Constraint	30
				The "Earnest Student" Persona	31
				The "Reflective Practitioner" Persona	31
		Reasoning	bedrock-llama-4-scout-17b	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			bedrock-qwen-3-next-80b	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			bedrock/ai21-jamba-1-5-large	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			bedrock/cohere-command-r-plus	The "Drafting Process" Emulation	20

				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			bedrock/deepseek-R1	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			claude-opus-4-6-v1	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			gemini-3.1-pro-preview	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			grok-4-1-fast-reasoning	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			o3-2025-04-16	The "Drafting Process" Emulation	21
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
	train	Instant	bedrock-llama-4-maverick-17b	The "Authentic Human" Constraint	125
				The "Earnest Student" Persona	125
				The "Reflective Practitioner" Persona	125
			bedrock/mistral-large-3	The "Authentic Human" Constraint	125
				The "Earnest Student" Persona	125
				The "Reflective Practitioner" Persona	125
			claude-sonnet-4-5-20250929-v1	The "Authentic Human" Constraint	125
				The "Earnest Student" Persona	125
				The "Reflective Practitioner" Persona	125
			deepseek-ai/DeepSeek-V3.1	The "Authentic Human" Constraint	125
				The "Earnest Student" Persona	125
				The "Reflective Practitioner" Persona	125
			gemini-flash-latest	The "Authentic Human" Constraint	125

				The "Earnest Student" Persona	125
				The "Reflective Practitioner" Persona	125
			gpt-5.2-2025-12-11	The "Authentic Human" Constraint	125
				The "Earnest Student" Persona	125
				The "Reflective Practitioner" Persona	125
		Reasoning	bedrock-llama-4-scout-17b	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			bedrock-qwen-3-next-80b	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			bedrock/ai21-jamba-1-5-large	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			bedrock/cohere-command-r-plus	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			bedrock/deepseek-R1	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			claude-opus-4-6-v1	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			gemini-3.1-pro-preview	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			grok-4-1-fast-reasoning	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83

				The Stylistic Deconstruction Framework	84
			o3-2025-04-16	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
---	---	---	---	---	---
wiki40B	test	Instant	bedrock-llama-4-maverick-17b	The "Authentic Human" Constraint	30
				The "Earnest Student" Persona	31
				The "Reflective Practitioner" Persona	30
			bedrock/mistral-large-3	The "Authentic Human" Constraint	30
				The "Earnest Student" Persona	31
				The "Reflective Practitioner" Persona	30
			claude-sonnet-4-5-20250929-v1	The "Authentic Human" Constraint	30
				The "Earnest Student" Persona	31
				The "Reflective Practitioner" Persona	31
			deepseek-ai/DeepSeek-V3.1	The "Authentic Human" Constraint	30
				The "Earnest Student" Persona	31
				The "Reflective Practitioner" Persona	30
			gemini-flash-latest	The "Authentic Human" Constraint	30
				The "Earnest Student" Persona	31
				The "Reflective Practitioner" Persona	31
			gpt-5.2-2025-12-11	The "Authentic Human" Constraint	30
				The "Earnest Student" Persona	31
				The "Reflective Practitioner" Persona	31
		Reasoning	bedrock-llama-4-scout-17b	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			bedrock-qwen-3-next-80b	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			bedrock/ai21-jamba-1-5-large	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21

			bedrock/cohere-command-r-plus	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			bedrock/deepseek-R1	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			claude-opus-4-6-v1	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			gemini-3.1-pro-preview	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			grok-4-1-fast-reasoning	The "Drafting Process" Emulation	20
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
			o3-2025-04-16	The "Drafting Process" Emulation	21
				The Constraint-Based Persona Generator	20
				The Stylistic Deconstruction Framework	21
	train	Instant	bedrock-llama-4-maverick-17b	The "Authentic Human" Constraint	125
				The "Earnest Student" Persona	125
				The "Reflective Practitioner" Persona	125
			bedrock/mistral-large-3	The "Authentic Human" Constraint	125
				The "Earnest Student" Persona	125
				The "Reflective Practitioner" Persona	125
			claude-sonnet-4-5-20250929-v1	The "Authentic Human" Constraint	125
				The "Earnest Student" Persona	125
				The "Reflective Practitioner" Persona	125
			deepseek-ai/DeepSeek-V3.1	The "Authentic Human" Constraint	125
				The "Earnest Student" Persona	125
				The "Reflective Practitioner" Persona	125

			gemini-flash-latest	The "Authentic Human" Constraint	125
				The "Earnest Student" Persona	125
				The "Reflective Practitioner" Persona	125
			gpt-5.2-2025-12-11	The "Authentic Human" Constraint	125
				The "Earnest Student" Persona	125
				The "Reflective Practitioner" Persona	125
		Reasoning	bedrock-llama-4-scout-17b	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			bedrock-qwen-3-next-80b	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			bedrock/ai21-jamba-1-5-large	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			bedrock/cohere-command-r-plus	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			bedrock/deepseek-R1	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			claude-opus-4-6-v1	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			gemini-3.1-pro-preview	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			grok-4-1-fast-reasoning	The "Drafting Process" Emulation	83

				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84
			o3-2025-04-16	The "Drafting Process" Emulation	83
				The Constraint-Based Persona Generator	83
				The Stylistic Deconstruction Framework	84

ДОДАТОК Г

Характеристики датасету атак типу А

Homoglyphs injected (overall):

```
count    1000.000000
mean      92.590000
std       76.684473
min        0.000000
25%       0.000000
50%      106.000000
75%      143.000000
max      351.000000
```

Name: homoglyphs_injected_count, dtype: float64

ZWSP injected (overall):

```
count    1000.000000
mean     24.538000
std      20.838493
min       0.000000
25%       0.000000
50%      27.000000
75%      40.000000
max      90.000000
```

Name: zwsp_injected_count, dtype: float64

By attack type:

	homoglyphs_injected_count			
zwsp_injected_count				
	mean	sum	mean	sum
attack_type				
both	142.935780	46740	37.617737	12301
homoglyph_only	135.650888	45850	0.000000	0
zwsp_only	0.000000	0	36.528358	12237

ДОДАТОК Д

Розподіл Fast-DetectGPT AI Score для датасету атак типу В

1. Розподіл Fast-DetectGPT AI Score для датасету атак типу В (T5).

model_name	T5_level	mean	std	count
bedrock-llama-4-maverick-17b	ai	0.866648	0.242341	131
	T5_lvl1	0.786941	0.303756	131
	T5_lvl2	0.747626	0.316949	131
	T5_lvl3	0.387368	0.307252	131
	T5_lvl4	0.447644	0.335902	131
	T5_lvl5	0.198721	0.161432	131
bedrock-llama-4-scout-17b	ai	0.973075	0.114561	81
	T5_lvl1	0.942922	0.177638	81
	T5_lvl2	0.922720	0.210055	81
	T5_lvl3	0.605849	0.329828	81
	T5_lvl4	0.713156	0.318552	81
	T5_lvl5	0.181599	0.145387	81
bedrock-qwen-3-next-80b	ai	0.369435	0.234424	79
	T5_lvl1	0.187409	0.132214	79
	T5_lvl2	0.141424	0.074344	79
	T5_lvl3	0.388095	0.229827	79
	T5_lvl4	0.316167	0.205435	79
	T5_lvl5	0.313412	0.198383	79
bedrock/ai21-jamba-1-5-large	ai	0.740454	0.296805	75
	T5_lvl1	0.642860	0.316302	75
	T5_lvl2	0.566493	0.325133	75
	T5_lvl3	0.202809	0.172202	75
	T5_lvl4	0.260402	0.242993	75
	T5_lvl5	0.203781	0.189788	75
bedrock/cohere-command-r-plus	ai	0.935456	0.154520	80
	T5_lvl1	0.877578	0.212151	80
	T5_lvl2	0.819125	0.260387	80
	T5_lvl3	0.257573	0.201053	80
	T5_lvl4	0.349888	0.271321	80
	T5_lvl5	0.147529	0.080443	80
bedrock/deepseek-R1	ai	0.185755	0.131803	80
	T5_lvl1	0.245273	0.193860	80
	T5_lvl2	0.329715	0.262550	80
	T5_lvl3	0.767550	0.289414	80

	T5_lvl4	0.722390	0.306516	80
	T5_lvl5	0.777991	0.281481	80
bedrock/mistral-large-3	ai	0.868837	0.212259	130
	T5_lvl1	0.687181	0.302790	130
	T5_lvl2	0.555291	0.329741	130
	T5_lvl3	0.189999	0.146613	130
	T5_lvl4	0.183404	0.131906	130
	T5_lvl5	0.235462	0.183425	130
claude-opus-4-6-v1	ai	0.322940	0.262091	81
	T5_lvl1	0.328281	0.271921	81
	T5_lvl2	0.327820	0.297292	81
	T5_lvl3	0.408490	0.328786	81
	T5_lvl4	0.340979	0.298833	81
	T5_lvl5	0.449992	0.304168	81
claude-sonnet-4-5-20250929-v1	ai	0.236366	0.214176	132
	T5_lvl1	0.221374	0.190361	132
	T5_lvl2	0.198148	0.154880	132
	T5_lvl3	0.290585	0.231875	132
	T5_lvl4	0.229258	0.183085	132
	T5_lvl5	0.348244	0.283331	132
deepseek-ai/DeepSeek-V3.1	ai	0.452738	0.309469	126
	T5_lvl1	0.370512	0.276283	126
	T5_lvl2	0.303547	0.241939	126
	T5_lvl3	0.213211	0.178413	126
	T5_lvl4	0.189353	0.144212	126
	T5_lvl5	0.277313	0.250612	126
gemini-3.1-pro-preview	ai	0.197222	0.152834	78
	T5_lvl1	0.200935	0.150076	78
	T5_lvl2	0.175362	0.136882	78
	T5_lvl3	0.408544	0.317939	78
	T5_lvl4	0.339916	0.292155	78
	T5_lvl5	0.398427	0.305876	78
gemini-flash-latest	ai	0.265552	0.224979	133
	T5_lvl1	0.232930	0.192204	133
	T5_lvl2	0.194046	0.151406	133
	T5_lvl3	0.231460	0.190729	133
	T5_lvl4	0.197566	0.164146	133
	T5_lvl5	0.255636	0.215068	133

gpt-5.2-2025-12-11	ai	0.228911	0.192145	132
	T5_lvl1	0.206348	0.170184	132
	T5_lvl2	0.183251	0.138464	132
	T5_lvl3	0.227398	0.198556	132
	T5_lvl4	0.199348	0.173801	132
	T5_lvl5	0.235359	0.206629	132
grok-4-1-fast-reasoning	ai	0.390157	0.282245	78
	T5_lvl1	0.541025	0.317302	78
	T5_lvl2	0.610719	0.316132	78
	T5_lvl3	0.823251	0.271003	78
	T5_lvl4	0.782571	0.300360	78
	T5_lvl5	0.741278	0.279731	78
o3-2025-04-16	ai	0.358716	0.249216	84
	T5_lvl1	0.513646	0.288131	84
	T5_lvl2	0.611700	0.294618	84
	T5_lvl3	0.905588	0.157446	84
	T5_lvl4	0.878475	0.178975	84
	T5_lvl5	0.749230	0.281431	84

2. Розподіл Fast-DetectGPT AI Score для датасету атак типу В (DIPPER).

model_name	DIPPER_level	mean	std	count
bedrock-llama-4-maverick-17b	ai	0.869409	0.234993	51
	Level 1	0.711305	0.285260	14
	Level 2	0.790802	0.203489	10
	Level 3	0.819318	0.227064	11
	Level 4	0.942361	0.158151	16
bedrock-llama-4-scout-17b	ai	0.975803	0.095479	35
	Level 1	0.887224	0.150380	15
	Level 2	0.852373	0.214968	7
	Level 3	0.985613	0.006819	6
	Level 4	0.991708	0.011253	7
bedrock-qwen-3-next-80b	ai	0.378721	0.240011	34
	Level 1	0.255106	0.164666	8
	Level 2	0.400217	0.298012	10
	Level 3	0.796821	0.299749	8
	Level 4	0.877274	0.185162	8
bedrock/ai21-jamba-1-5-large	ai	0.820902	0.257065	28
	Level 1	0.742234	0.284994	4

	Level 2	0.723035	0.287249	10
	Level 3	0.973046	0.041837	9
	Level 4	0.999787	0.000237	5
bedrock/cohere-command-r-plus	ai	0.923784	0.182193	34
	Level 1	0.786146	0.297273	10
	Level 2	0.729141	0.256672	7
	Level 3	0.849141	0.174917	12
	Level 4	0.999272	0.001329	5
bedrock/deepseek-R1	ai	0.202507	0.157307	36
	Level 1	0.198318	0.206434	10
	Level 2	0.339358	0.260083	8
	Level 3	0.596302	0.273350	8
	Level 4	0.997061	0.003492	10
bedrock/mistral-large-3	ai	0.863071	0.231957	55
	Level 1	0.772854	0.253434	14
	Level 2	0.602196	0.321752	14
	Level 3	0.819993	0.199615	15
	Level 4	0.993575	0.009808	12
claude-opus-4-6-v1	ai	0.324137	0.280956	27
	Level 1	0.605337	0.302354	7
	Level 2	0.565234	0.291942	7
	Level 3	0.806602	0.343280	8
	Level 4	0.850951	0.154889	5
claude-sonnet-4-5-20250929-v1	ai	0.247258	0.211367	66
	Level 1	0.567926	0.244320	13
	Level 2	0.640571	0.306217	13
	Level 3	0.747925	0.269646	22
	Level 4	0.882843	0.210049	18
deepseek-ai/DeepSeek-V3.1	ai	0.478711	0.321473	55
	Level 1	0.573695	0.271979	11
	Level 2	0.713859	0.291103	15
	Level 3	0.733600	0.294800	10
	Level 4	0.898363	0.213865	19
gemini-3.1-pro-preview	ai	0.190950	0.159587	40
	Level 1	0.417201	0.278398	6
	Level 2	0.711069	0.317464	12
	Level 3	0.786417	0.203107	11
	Level 4	0.987639	0.028877	11

gemini-flash-latest	ai	0.264331	0.223365	58
	Level 1	0.567760	0.325882	17
	Level 2	0.504314	0.294004	8
	Level 3	0.730715	0.311413	21
	Level 4	0.987926	0.017828	12
gpt-5.2-2025-12-11	ai	0.241596	0.212876	50
	Level 1	0.595624	0.301047	15
	Level 2	0.596876	0.308272	12
	Level 3	0.931878	0.149160	7
	Level 4	0.944454	0.112859	16
grok-4-1-fast-reasoning	ai	0.421709	0.306463	28
	Level 1	0.236836	0.235562	9
	Level 2	0.229459	0.160332	6
	Level 3	0.535211	0.299776	3
	Level 4	0.926399	0.115086	10
o3-2025-04-16	ai	0.385106	0.249721	36
	Level 1	0.220715	0.185060	9
	Level 2	0.176745	0.110059	8
	Level 3	0.707846	0.307068	5
	Level 4	0.936222	0.132649	14