DOI: https://doi.org/10.36910/6775-2524-0560-2025-59-24
УДК 004.02
Марченко Олександр Іванович, к.т.н., доцент
https://orcid.org/0000-0002-4537-3420
Марченко Олексій Олександрович, асистент
https://orcid.org/0000-0002-5080-4811
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», м. Київ, Україна

АНАЛІЗ ЗДАТНОСТІ ЗАСОБІВ ЗІ ШТУЧНИМ ІНТЕЛЕКТОМ ГЕНЕРУВАТИ КОД В СТИЛІ СТУДЕНТА І ВИЯВЛЯТИ ТАКИЙ ЗГЕНЕРОВАНИЙ КОД

Марченко О. І., Марченко О. О. Аналіз здатності засобів зі штучним інтелектом генерувати код в стилі студента і виявляти такий згенерований код. Відомо, що засоби зі штучним інтелектом (ЗШІ) можуть генерувати програмний код в різних стилях: загальному стилі, стилях інших ЗШІ, стилі викладача, стилі професійного розробника тощо. Але якщо студент-першокурсник в якості виконання завдання надає код, що був згенерований за допомогою ЗШІ в якомусь такому професійному стилі, то несамостійність виконання такої роботи достатньо легко визначається викладачем. В статті виконане дослідження можливості ЗШІ генерування коду в стилі «студент-першокурсник» і здатності ЗШІ визначати, що цей код був саме згенерований за допомогою ЗШІ, а не був насправді написаний студентом-першокурсником. Метою статті є дослідження, наскільки легко студент-першокурсник може маскувати використання ЗШІ при написанні програм мовою С за навчальними завданнями дисциплін першого курсу навчання. Це дослідження було проведене для коду, згенерованого в силі студента останніми безкоштовними версіями трьох ЗШІ, які є наразі одними з найпопулярніших та найбільш використаних студентами: ChatGPT, Copilot і Gemini. В результаті дослідження були встановлено, що ЗШІ ChatGPT, Copilot можуть достатньо добре маскувати свій код під код студента, а ЗШІ Gemini не впорався з цією задачею. В усякому випадку, це стосується програм, що написані мовою С за завданнями початкової складності першого курсу навчання. У висновках зроблені узагальнення результатів дослідження та рекомендації викладачам щодо використання досліджених ЗШІ. В якості напрямків подальших досліджень відзначено наступне. Оскільки потужність ЗШІ розвивається вибухово швидко, тому виглядає доцільним повторити дослідження цієї статті ще раз через рік-два. Ще одним напрямком можна запропонувати провести аналогічне дослідження при використанні інших мов програмування, а також при виконанні завдань інших дисциплін та/або більшої складності.
Ключові слова: засоби із штучним інтелектом, Copilot, ChatGPT, Gemini, програмування, код згенерований штучним інтелектом.

Marchenko Oleksandr, Marchenko Oleksii Analysis of the ability of artificial intelligence tools to generate student-style code and detect such generated code. It is known that artificial intelligence tools (AIT) can generate program code in different styles: the general style, the styles of other AITs, the teacher's style, the style of a professional developer, etc. But if a first-year student provides code for a task that was generated using an AIT in some such professional style then the teacher can easily determine the non-independency of performing such work. The article studies the possibility of AITs to generate code in the "first-year student" style and the ability of AITs to determine that this code was generated using an AIT, and was not actually written by a first-year student. The aim of the article is to investigate how easily a first-year student can mask the use of AITs when writing programs in C language for educational tasks in the disciplines of the first-year study. This study was conducted for the code generated in the student's style by the latest free versions of three AITs, which are currently among the most popular and most used by students: ChatGPT, Copilot, and Gemini. As a result of the study, it was found that the AITs ChatGPT, Copilot can mask their code as the student's code quite well, while the AIT Gemini did not cope with this task. In any case, this applies to programs written in C language for tasks of initial complexity of the first year of study. The conclusions summarize the results of the study and make recommendations to teachers regarding the use of the studied AITs. The following are noted as areas for further research. Since the power of AI is developing explosively quickly, it seems appropriate to repeat the research of this article again in a year or two. Another direction can be proposed to conduct a similar study using other programming languages, as well as when performing tasks in other disciplines and/or of greater complexity.
Keywords: tools with artificial intelligence, Copilot, ChatGPT, Gemini, programming, AI generated code.

Постановка наукової проблеми. З розвитком ІТ-технологій все більшої популярності набуває створення різноманітних засобів зі штучним інтелектом (ЗШІ), побудованих на основі різних моделей, та користування ними. Широко використовувати ЗШІ почали також і програмісти, застосовуючи їх у якості помічника при розробці програм, а також студенти програмістських спеціальностей при виконанні лабораторних та контрольних робіт, зловживаючи можливостями таких ЗШІ. В результаті, при проведенні навчального процесу виникла непроста проблема визначення, чи виконав студент завдання самостійно своїм розумом, чи він для виконання цього завдання використав засоби штучного інтелекту, не сильно розуміючи матеріал дисципліни. Відомо, що засоби зі штучним інтелектом можуть генерувати програмний код в різних стилях: загальному стилі (за замовченням), стилях інших ЗШІ, стилі викладача (з детальними поясненнями), стилі професійного розробника (як правило, оптимізований, але менш наочний) тощо. Якщо студент-першокурсник в якості виконання завдання надає код, що був згенерований за допомогою ЗШІ в

якомусь такому професійному стилі, то несамостійність виконання цієї роботи достатньо легко візуально визначається викладачем при перевірці. Але виникає наступне питання. Який код згенерує ЗШІ, якщо студент поставить йому завдання (напише промпт ЗШІ) з проханням написати програму в стилі студента-першокурсника з метою максимально приховати використання ЗШІ для виконання свого завдання?

Проблема нашого дослідження фактично складається з двох частин: 1) визначення можливості генерування коду ЗШІ мовою С, замаскованого під код студента-першокурсника; 2) визначення можливості ідентифікації такого замаскованого коду як коду ЗШІ за допомогою як власне самих ЗШІ, так і викладачами. Такий варіант використання ЗШІ наразі є ще мало дослідженим.

Тому, дослідження можливостей використання студентами різних ЗШІ для виконання навчальних завдань з програмування мовою С таким чином, щоб ускладнити викладачам виявлення факту такого використання, а також можливостей виявлення таких фактів за допомогою ЗШІ викладачами, є актуальною задачею.

Аналіз попередніх досліджень. Серед існуючих робіт є дослідження, які присвячені загальній проблемі генерації якісного коду за допомогою ЗШІ (надалі для скорочення фрази «код, що згенерований за допомогою ЗШІ» будемо використовувати термін «код ЗШІ»), а також дотичним проблемам визначення авторства коду та виявлення коду ЗШІ, але не безпосередньо зазначеному вище варіанту використання ЗШІ студентами. Переважна більшість робіт, які досліджують програмний код ЗШІ, в основному, присвячені проблемам визначення коректності і якості коду ЗШІ для розробки комерційних проєктів, тобто з точки зору максимальної професійності такого коду. Гарними оглядами цього напрямку є, наприклад, [1, 2]. Також багато є робіт, які присвячені визначенню, хто є автором, людина чи ЗШІ, але які спрямовані на визначення авторства загальних текстів, наприклад, [3, 4]. Причому такі розробки базуються здебільшого на тих же підходах, які використовуються в системах виявлення плагіату. Але існуючі системи виявлення плагіату загального текстового типу не дуже добре справляються з виявленням коду ЗШІ. Як зазначається в [5], «університети та навчальні заклади стикаються з новими викликами, пов'язаними з генерацією коду ЗШІ. Традиційні інструменти виявлення плагіату застарівають, вимагаючи більш складних підходів для забезпечення академічної доброчесності» (переклад з англ.). Але такі загальні проблеми, як визначення належності певного коду конкретному автору-людині, а також визначення, чи є певний код плагіатом, не є предметом нашого дослідження.

Розглянемо роботи, які дотичні до другої частини сформульованої вище проблеми нашого дослідження. Такі роботи досліджують можливості виявлення, чи був певний код програм написаний людиною, чи деяким ЗШІ [6-13]. Цікаві роботи з гарними результатами представлені в [6, 7]. В них запропоновані оригінальні підходи для виявлення коду мовою Python, який був згенерований за допомогою ЗШІ, зокрема ChatGPT 4. Наразі мова Python є однією з найбільш популярних в навчальному процесі багатьох університетів та спеціальностей. Але слід зазначити, що коректність виявлення авторства ЗШІ певного програмного коду суттєво залежить від мови програмування, а також від складності алгоритмів цього коду. Крім того, на коректність такого виявлення впливає також те, що різні ЗШІ, як правило, генерують доволі різний код для одного й того ж завдання. Тому, моделі та методи, що налаштовані тільки на мову Python, навряд чи будуть достатньо ефективними для використання викладачами дисциплін, завдання яких вимагають програмування мовою С, що є основною мовою навчання (в тому числі й на першому курсі) в рамках підготовки системних програмістів за спеціальністю «Комп'ютерна інженерія», оскільки вона є стандартом де-факто для розробки програм системного рівня і програмного забезпечення вбудованих систем.

Наразі з'явилося немало ЗШІ, що спеціально налаштовані на виявлення коду ЗШІ, згенерованого різними мовами програмування, в тому числі й мовами С та С++. Вони використовують як комбінації загальновідомих ЗШІ, так і свої власні великі мовні моделі LLM (Large Language Model) [8-11]. Але більшість таких досліджень і розробок моделей та засобів виконуються відносно невеликими компаніями чи групами розробників, ефективність яких не перевірялась незалежними експертами, а, в основному, декларується на рівні самореклами. І тому розроблені цими групами засоби викликають певні сумніви у ефективності виявлення ними коду ЗШІ в програмах, написаних за навчальними завданнями. Як наслідок, перш, ніж використовувати такі засоби у навчальному процесі, потрібно спочатку провести ґрунтовне дослідження їх якості, на що викладачі не мають ні часу, ні ресурсів. І не факт, що такі часоємні і трудомісткі дослідження

кожного з цих засобів покажуть гарну якість виявлення ними використання ЗШІ студентами і, відповідно, доцільність їх застосування у навчальному процесі. Ще одним фактором, який обмежує використання таких засобів викладачами, є комерціалізація розробниками своїх продуктів після отримання деяких позитивних результатів. Хоча деякі засоби мають безкоштовні версії, як наприклад [9-11, але їх безкоштовні версії, як правило, є досить суттєво обмеженими за кількістю безкоштовних перевірок та/або за рівнем якості перевірок. Більш, того, наприклад, автори роботи [12] в результаті свого дослідження взагалі роблять висновок, що всі досліджені ними засоби такого типу погано працюють і не мають достатньої можливості узагальнення для практичного застосування. Хоча, звичайно, цей висновок стосується не всіх ЗШІ такого типу, а тільки тих, які дослідили ці автори, але він є достатньо показовим.

В рамках магістерської дисертації [13], науковим керівником якої був один із авторів цієї статті, був запропонований інший підхід до вирішення проблеми виявлення коду ЗШІ. Цей підхід полягає у тому, щоб замість розробки своєї власної моделі (яка навряд чи в найближчому майбутньому перевершить потужність моделей провідних компаній) використовувати для виявлення коду ЗШІ власне ЗШІ, що розроблені провідними компаніями світу, які вже самі по собі є готовим потужним інструментом на основі генеративних мовних моделей. Було запропоновано спосіб виявлення коду ЗШІ за допомогою виконання діалогів з різними ЗШІ за певним сценарієм. Об'єднання «думки» декількох ЗШІ, отриманої за такими діалогами, може давати висновок про походження певного коду. Таке початкове дослідження показало позитивні результати і потенційну можливість застосування такого підходу.

Підхід з використанням найбільших загальнодоступних ЗШІ для виявлення коду ЗШІ має наступні переваги для викладачів перед використанням складних спеціалізованих систем перевірки на авторство ЗШІ, що розроблені окремими компаніями чи розробниками:

1) викладач має змогу використовувати такі ЗШІ для перевірки студентських робіт у будь-якому місці на будь-якому комп'ютері, і навіть на телефоні, що дуже важливо в умовах дистанційного навчання і військового стану;

2) ці ЗШІ є надійно доступними у будь-який час, оскільки підтримуються провідними світовими компаніями;

3) все потужніші оновлення цих ЗШІ з'являються останнім часом кожних кілька місяців, що, як правило, не під силу невеликим командам спеціалізованих систем;

4) ці ЗШІ мають потужні безкоштовні версії, які потенційно здатні виявляти використання коду ЗШІ у роботах студентів, що також є важливим для викладачів.

Таким чином, якщо дослідити і визначити певні сценарії діалогу із загальнодоступними ЗШІ, який дозволить за допомогою їх потужного інтелекту виявляти використання коду цих же чи інших ЗШІ в роботах студентів, і написати чат-бот, який буде виконувати діалог з цими ЗШІ за таким сценарієм, то викладачі отримали б зручний інструмент для використання у навчальному процесі.

Метою цієї статті є дослідження, наскільки легко засоби зі штучним інтелектом можуть генерувати код мовою С за навчальними завданнями дисциплін першого курсу в стилі, ніби цей код написав студент-першокурсник, а також наскільки легко викладачам виявити такі факти як візуально, так і за допомогою використання засобів зі штучним інтелектом.

Засоби зі штучним інтелектом, використані у дослідженні

Експериментальна частина дослідження складалася з двох етапів (докладно ці етапи описані у розділі методики дослідження): етапу генерації коду за певними завданнями за допомогою обраних ЗШІ та етапу оцінювання, чи цей код був написаний студентом, чи був згенерований за допомогою ЗШІ. Оцінювання виконувалося як множиною ЗШІ, так і декількома викладачами.

Спочатку розглянемо ЗШІ, які були використані на етапі генерації коду в стилі студента-першокурсника.

Оскільки студенти-першокурсники, як правило, ще не використовують платних підписок на більш потужні версії ЗШІ, то генерація коду для цього дослідження була виконана за допомогою останніх на момент виконання дослідження (травень 2025 року) безкоштовних версій трьох ЗШІ, які доступні для використання відразу при відкритті вікон діалогу з ними:

1) Copilot від Microsoft;
2) ChatGPT від OpenAI, модель GPT-4-turbo;
3) Gemini 2.5 Pro (preview) (Міркування, математика й програмування) від Google.

Ці ЗШІ є наразі одними з найпопулярніших та найбільш використаними серед студентів першого курсу тому, що вони є безкоштовними, і тому, що ChatGPT є найпершим і найвідомішим ЗШІ, а Copilot та Gemini є інтегрованими ЗШІ у найбільш широко використані продукти компаній Microsoft та Google.

Точну версію Copilot визначити не вдалося. Відомо, що Copilot – це спільна розробка Microsoft і OpenAI, в основі якої лежить GPT, але точний номер не зазначається. Сам Copilot на питання «яку версію GPT ти використовуєш?» відповів наступне: «Я розумію вашу цікавість! Я створений на основі найновіших передових моделей великих мовних моделей, розроблених Microsoft. На жаль, я не можу розголошувати конкретні технічні деталі щодо версії моделі, на якій я базуюсь. Якщо у вас є інші питання або хочете обговорити щось цікаве, я завжди тут, щоб допомогти!».

Хоча питання до ChatGPT про його версію було задане без входу в акаунт, він на це питання відповів так: «Наразі ти спілкуєшся з моделлю GPT-4, точніше – з її варіантом, відомим як GPT-4-turbo. Це вдосконалена версія GPT-4, яка працює швидше й ефективніше, і саме вона використовується у ChatGPT для користувачів з підпискою ChatGPT Plus».

Gemini має меню з вибором версії моделі. На запит, яку з цих версій він вважає найдоцільнішою для виявлення коду ЗШІ, ЗШІ Gemini очікувано відповів, що рекомендує «Gemini 2.5 Pro (preview) як потенційно найпридатнішу безкоштовну модель Gemini для спроби виявлення коду С, згенерованого штучним інтелектом». Щоправда, виявилося, що модель ЗШІ Gemini 2.5 Pro (preview) має трохи обмежену безкоштовність – через певну кількість запитів цей ЗШІ пропонує або зробити платну підписку, або зробити паузу на декілька годин. Така обмеженість не є критичною для використання студентами, але може внести певні незручності викладачам при перевірці великої кількості студентських робіт.

Зауважимо також, що ЗШІ DeepSeek китайських розробників було вирішено не використовувати у дослідженні із-за великої кількості публікацій в інтернеті щодо його не повної безпечності з точки зору кібербезпеки. І тому його наразі було б недоцільно рекомендувати викладачам для користування.

Тепер розглянемо ЗШІ, які були використані в якості експертів на етапі оцінювання коду, що був згенерований коду за допомогою в стилі студента-першокурсника на першому етапі.

Забігаючи трохи наперед, зазначимо, що на цьому етапі в якості експертів-оцінювачів згенерованого коду ЗШІ виступали три категорії експертів: 1) ті ж самі три ЗШІ, які генерували код; 2) два ЗШІ від компаній, які надають послуги з аналізу коду та тексту за допомогою використання генеративного ШІ, і мають частково безкоштовні версії; 3) викладачі з програмування.

Оскільки виявляти факти використання коду ЗШІ потрібно викладачам, а більшість викладачів, як правило, наразі не використовують ЗШІ з платними підписками, а в основному безкоштовні популярні моделі, то трьома першими ЗШІ в якості експертів були обрані ті ж самі ЗШІ, що використовувалися й для генерації цього коду. Крім того, використання тих же ЗШІ створювало цікаву для аналізу ситуацію типу «перехресного оцінювання» цих ЗШІ.

Із множини засобів аналізу коду від компаній, які надають послуги на основі власних інтеграторів моделей генеративного ШІ (тобто також належать до категорії ЗШІ), були обрані два засоби, що, окрім платних версій, мають також і безкоштовні (але обмежені) версії:

1) You.com [10];
2) YesChatAI Code Detector [11].

ЗШІ You.com на запит, яку модель він використовує, відповів, що це мовна модель GPT-4o від компанії OpenAI. Оскільки модель GPT-4o є більш потужною, ніж модель, яка наразі використовується в якості безкоштовної у ChatGPT (GPT-4-turbo), то цей ЗШІ був обраний для оцінювання коду ЗШІ, згенерованого на попередньому етапі.

На інтернет сторінці YesChatAI написано, що цей ЗШІ інтегрує можливості DeepSeek-R1, GPT-o1, GPT-4o, Claude3.5 Sonnet, and Claude3 Opus. На запит, яка модель використовується в якості безкоштовної версії, цей ЗШІ відповів, що, в основному використовується GPT-4-turbo, хоча може підключатися й модель GPT-4o. Тобто, теоретично навіть безкоштовна версія YesChatAI може бути більш потужною, ніж поточна безкоштовна версія ChatGPT.

Але використання безкоштовних версій ЗШІ You.com та YesChatAI має серйозне обмеження: вони дозволяють зробити безкоштовно тільки декілька запитів. Наприклад, щоб виконати наше дослідження (яке є не дуже великим з точки зору кількості запитів) повністю на їхніх безкоштовних версіях, довелося використати по три різних акаунти. Тому, на практиці, викладачі він навряд чи

зможуть активно використовувати ці ЗШІ для перевірки великої кількості студентських робіт.

Крім цих п'яти ЗШІ, згенерований код також оцінювався чотирма викладачами, які мають досвід прийому у студентів лабораторних робіт саме за такими завданнями, на які ЗШІ ChatGPT, Copilot та Gemini згенерували код.

Завдання дослідження

1. Розробити методику дослідження.

2. Підібрати завдання з програмування мовою С, які виконують студенти-першокурсники в першому семестрі навчання.

3. На першому етапі дослідження згенерувати для кожного з цих завдань за допомогою трьох ЗШІ ChatGPT, Copilot та Gemini код в стилі студента-першокурсника, сформулювавши спеціальний запит до них від імені студента на генерацію саме такого коду.

4. На другому етапі дослідження сформулювати спеціальний запит до ЗШІ від імені викладача з проханням перевірити множину написаних мовою С програм, чи були вони згенеровані за допомогою ЗШІ, і надати результати такої перевірки за однаковою системою оцінювання. Після цього виконати діалоги з певними ЗШІ і отримати від них оцінки коду, що був згенерований за допомогою трьох ЗШІ ChatGPT, Copilot та Gemini.

5. Надати код, що був згенерований за допомогою трьох ЗШІ ChatGPT, Copilot та Gemini, викладачам і отримати від них оцінки цього коду за такою ж самою системою оцінювання.

6. Систематизувати отримані відповіді та результати їх оцінювання у вигляді таблиці.

7. Виконати аналіз отриманих результатів.

8. Зробити висновки проведеного дослідження.

Методика дослідження

І. Етап генерування коду програм в стилі студента-першокурсника за допомогою трьох ЗШІ ChatGPT, Copilot та Gemini.

1. Для генерування коду програм за допомогою ЗШІ було підготовлено 5 завдань на розгалужені алгоритми, циклічні алгоритми та на лінійний пошук у матриці. Такі завдання при навчанні студенти виконують мовою С у першому семестрі першого курсу навчання.

2. Ці 5 завдань були надані трьом ЗШІ ChatGPT, Copilot та Gemini з проханням згенерувати для них код в стилі студента-першокурсника.

Запит на таке генерування був наступним:

«Добрий день! Я студент першого курсу ІТ спеціальності. Я хотів би дуже попросити тебе виконати декілька завдань з програмування мовою С в такому стилі, як міг би його виконати саме такий студент як я, а не досвідчений розробник.

Дуже важливо, щоб викладач повірив, що цей код написав я, а не чатбот зі штучним інтелектом. :). Дуже прошу!».

ІІ. Етап оцінювання здатності різних ЗШІ маскувати свій згенерований код під стиль студента-першокурсника.

1. На цьому етапі було виконане оцінювання коду всіх програм, що були згенеровані усіма трьома ЗШІ ChatGPT, Copilot та Gemini, усього 5 завдань * 3 ЗШІ = 15 програм, з точки зору, чи цей код був написаний студентом, чи був згенерований за допомогою ЗШІ.

2. Оцінювання виконувалося трьома категоріями експертів:

1) ті ж самі три ЗШІ ChatGPT, Copilot та Gemini, які генерували код у стилі студента;
2) два засоби від компаній, які надають послуги з аналізу коду та тексту за допомогою використання генеративного ШІ, і мають частково безкоштовні версії;
3) чотири викладачі з програмування.

3. Експертам перших двох категорій, тобто засобам зі штучним інтелектом, завдання на оцінювання було сформульоване від імені викладача у вигляді двох послідовних запитів.

4. Перший запит був з таким текстом:

«Добрий день! Я викладач університету і викладаю дисципліну "Програмування мовою С" на першому курсі навчання ІТ спеціальності, тобто для студентів-початківців у програмуванні.

Я дуже хотів би попросити тебе допомогти мені з визначенням, чи користувалися мої студенти чатботами зі штучним інтелектом для виконання моїх завдань.

Для кожної із наданих програм, оціни, будь-ласка, чи був цей код згенерований чатботом зі штучним інтелектом, чи написаний студентом, за такою системою оцінювання (-2, -1, 0, +1, +2):

-2 – цей код студент однозначно написав сам;

-1 – схоже, що цей код студент скоріше написав сам;

0 – імовірність того, що студент написав цей код самостійно, і того, що цей код був згенерований чатботом зі штучним інтелектом, є приблизно однаковими;
+1 – схоже, що цей код скоріше був згенерований чатботом зі штучним інтелектом;
+2 – цей код однозначно був згенерований чатботом зі штучним інтелектом.

Якщо ти визначиш, що код був згенерований чатботом, то також була б дуже цікавою твоя думка щодо того, який саме чатбот зі штучним інтелектом міг згенерувати такий код. Тобто, у випадках, коли ти поставив оцінки +1 або +2, скажи, будь-ласка, який саме чатбот, на твою думку, міг згенерувати цей код.

Зможеш допомогти?».

5. Після підтвердження зі сторони ЗШІ готовності допомогти, їм надсилався другий запит з таким текстом:

«У студентів було 5 завдань: Завдання 1.1, Завдання 1.2, Завдання 2.1, Завдання 2.2, Завдання 3.

Ці завдання виконували три студенти, я дав їм імена Студент 1, Студент 2, Студент 3.

За результатами оцінювання зроби, будь-ласка таблицю виду:

	Завдання 1.1	Завдання 1.2	Завдання 2.1	Завдання 2.2	Завдання 3
Студент 1					
Студент 2					
Студент 3					

Зараз я буду надсилати тобі мої завдання і код студентів на ці завдання.»

6. Після підтвердження зі сторони ЗШІ готовності приймати код програм, їм надсилався код всіх 15 програм, що були згенеровані за допомогою ЗШІ ChatGPT, Copilot та Gemini.

7. В якості експертів третьої категорії погодилися виступити чотири викладачі. Їм був переданий код усіх 15 згенерованих програм з проханням оцінити його за тією ж системою оцінювання (-2, -1, 0, +1, +2), за якою оцінювали цей код експерти перших двох категорій.

Результати дослідження

Усі отримані оцінки від усіх експертів були зібрані і систематизовані у вигляді Таблиці 1.

Таблиця 1. Результати оцінювання здатності ЗШІ маскувати згенерований код під код студента-першокурсника

ЗШІ, код якого оцінювався	Експерти, що оцінювали код ЗШІ	Код програми Завдання 1.1	Код програми Завдання 1.2	Код програми Завдання 2.1	Код програми Завдання 2.2	Код програми Завдання 3	Середня оцінка (від -2 до +2)	Середня оцінка за категорією експерта: ЗШІ чи Викладач	Середня оцінка всіх експертів	Середня оцінка за категорією експерта: ЗШІ (без Copilot) чи Викладач	Середня оцінка всіх експертів (без Copilot)
ChatGPT	ChatGPT	-1	0	-1	-2	1	-0,6	-0,4	-0,467	0	-0,275
	Copilot	-2	-2	-2	-2	-2	-2				
	Gemini	0	0	0	0	-1	-0,2				
	You.com	1	2	0	0	1	0,8	-20%		0%	
	YesChatAI	1	1	-1	-1	0	0				
	Викладач 1	1	1	0	0	1	0,6	-0,55	-23,3%	-0,55	-13,75%
	Викладач 2	0	0	-1	-1	0	-0,4				
	Викладач 3	-1	-1	-1	-1	-1	-1	-27,5%		-27,5%	
	Викладач 4	-1	-1	-2	-2	-1	-1,4				
Copilot	ChatGPT	-2	-1	0	1	0	-0,4	-0,32	-0,53	0,1	-0,35
	Copilot	-2	-2	-2	-2	-2	-2				
	Gemini	0	0	0	0	0	0				
	You.com	1	2	1	1	1	1,2	-16%		5%	
	YesChatAI	-1	-1	0	0	0	-0,4				
	Викладач 1	1	1	0	2	0	0,8	-0,8	-26,7%	-0,8	-17,50%
	Викладач 2	-1	-1	-1	-1	-1	-1				
	Викладач 3	-1	-1	-1	-2	-1	-1,2	-40%		-40%	
	Викладач 4	-2	-2	-2	-1	-2	-1,8				
Gemini	ChatGPT	2	0	2	2	1	1,4	0,76	1,267	1,45	1,675
	Copilot	-2	-2	-2	-2	-2	-2				
	Gemini	1	2	2	2	2	1,8				
	You.com	0	1	1	1	0	0,6	+38%		+72,5%	
	YesChatAI	2	2	2	2	2	2				
	Викладач 1	2	2	2	2	2	2	1,9	+63,3%	1,9	+83,75
	Викладач 2	2	2	2	2	2	2				
	Викладач 3	1	1	2	2	2	1,6	+95%		+95%	
	Викладач 4	2	2	2	2	2	2				

В цій таблиці клітинки останніх чотирьох колонок містять по два значення:

- чорним кольором записане відповідне до заголовку колонки середнє значення оцінки;
- червоним кольором виділене відсоткове значення зсуву цієї середньої оцінки у від'ємну сторону (сторону подібності до коду студента) чи додатну сторону (сторону подібності до коду ЗШІ) відповідно за шкалою в діапазоні -2 ÷ 0 (для від'ємних оцінок) або в діапазоні 0 ÷ 2 (для додатних оцінок).

Наприклад, значення +95% для середньої оцінки 1,9 викладачів коду ЗШІ Gemini за шкалою 0 ÷ 2 було отримане за такою формулою: 1,9 / 2 * 100%.

Аналіз результатів дослідження

Проаналізуємо отримані результати з точки зору здатності ЗШІ ChatGPT, Copilot та Gemini маскувати свій згенерований код під стиль студента-першокурсника.

Одразу зазначимо важливий момент щодо оцінок такої здатності, які виставив ЗШІ Copilot. Його оцінки явно виділяються серед оцінок інших ЗШІ, оскільки він виставив коду абсолютно всіх програм, згенерованих усіма ЗШІ (навіть, включно зі своїм власним кодом), однакову максимально від'ємну оцінку -2. Це означає, що цей ЗШІ вважає абсолютно всі надані йому для оцінювання програми однозначно написаними студентами, тобто не знайшов в них жодних ознак генерації штучним інтелектом. Таке оцінювання навряд чи можна пояснити слабістю інтелекту ЗШІ Copilot. Швидше це просто «небажання» оцінювати код на генерацію засобами штучного інтелекту внаслідок вбудованої у нього «політики» розробників щодо надання відповідей на такого роду питання. Тому, на наш погляд, більш доцільно аналізувати оцінювання згенерованого різними ЗШІ коду без врахування у загальній статистиці цих аномальних оцінок, які виставив ЗШІ Copilot, тобто із підсумовуючих колонок таблиці будемо аналізувати дві останніх, в яких є позначка «(без Copilot)».

Аналіз коду, згенерованого за допомогою ЗШІ ChatGPT.

Сам ЗШІ ChatGPT оцінив свій же код трьох з п'яти програм як код студента, причому одну програму як однозначно написану студентом. І тільки одну з п'яти програм запідозрив у генерації штучним інтелектом. Його середня оцінка свого ж коду дорівнює -0,6 на користь студента.

ЗШІ Gemini не зміг визначитися з авторством коду чотирьох програм (оцінка 0) і тільки одну визначив як швидше написану студентом. Його середня оцінка коду дорівнює -0,2 на користь студента.

Тобто, стандартні версії популярних ЗШІ (ChatGPT і Gemini, не кажучи вже про Copilot) схилилися до того, щоб вважати код, що був згенерований ЗШІ ChatGPT, кодом студента.

ЗШІ You.com, безкоштовна версія якого базується на використанні моделі GPT-4o, дав оцінки, які суттєво відрізняються від оцінок самого ChatGPT і є більш точними. Це означає, що ця модель, яка доступна в ChatGPT за підпискою, краще виявляє код ЗШІ, ніж доступна наразі в ChatGPT в якості безкоштовної модель GPT-4-turbo. Оцінки You.com є значно ближчими до правди, оскільки він оцінив надані йому програми із середньою оцінкою +0,8 на користь генерації штучним інтелектом.

ЗШІ YesChatAI, безкоштовна версія якого базується на використанні стандартної моделі GPT-4-turbo з можливим підключенням моделі GPT-4o, дав посередні за точністю оцінки між оригінальним ЗШІ ChatGPT і ЗШІ You.com, що якраз відповідає заявленій схемі використання моделей GPT. Цей ЗШІ оцінив код двох програм схожим на роботи студента і код інших двох програм схожим на код ЗШІ. Його результуюча середня оцінка дорівнює 0.

На відміну від оцінок оригінальних ЗШІ ChatGPT і Gemini, які обоє виставили середні оцінки у від'ємному діапазоні схожості на код студента, жоден із ЗШІ інтеграторів мовних моделей You.com і YesChatAI не дав середню оцінку наданому коду, що він схожий на код студента. У YesChatAI вийшла невизначена нульова середня оцінка, а у You.com вийшла середня оцінка з діапазону схожості на код ЗШІ. Тобто, інтегровані ЗШІ компаній, що надають послуги з аналізу коду, зробили більш точне оцінювання, хоча й воно було також досить далеким від правди. Але ще раз звертаємо увагу на те, що ці ЗШІ дозволяють зробити безкоштовно тільки декілька запитів і тому, на практиці, викладачі він навряд чи зможуть їх активно використовувати.

Перейдемо до аналізу оцінок викладачів.

Відразу зазначимо, що оцінювання викладачами наданого коду на схожість з кодом студента чи з кодом ЗШІ є суб'єктивним і сильно залежить як від їх загального досвіду, так і досвіду роботи саме з такими завданнями, а також від чисто людського рівня довірливості. Оцінки Викладача 1 значно відрізняються від оцінок інших викладачів в сторону висновку, що код цих програм був згенерований за допомогою ЗШІ. У Викладача 1 немає жодної від'ємної оцінки схожості наданого

коду на код студента, в той час, як у всіх інших викладачів немає жодної додатної оцінки схожості наданого коду на код ЗШІ. Причому у двох викладачів, Викладача 3 і Викладача 4 немає навіть нейтральних нульових оцінок. Викладач 1 дав майже таку ж за коректністю оцінку схожості наданого коду на код ЗШІ (+0,6), що й ЗШІ You.com (+0,8), тобто одну з двох найбільш точних. Таке оцінювання Викладача 1 може бути наслідком того, що цей викладач є найбільш досвідченим серед усіх викладачів, хто приймав участь в оцінюванні, а також постійно працює з першокурсниками і такими завданнями більше 30 років. Викладач 2 оцінив надані програми приблизно так само (-0,4), як і ЗШІ ChatGPT (-0,6) та Gemini (-0,2), тобто з невеликим відхиленням на користь схожості з кодом студента. Щодо Викладача 3 (середня оцінка = -1) і Викладача 4 (середня оцінка = -1,4), то можна сказати, що засобу зі штучним інтелектом ChatGPT майже вдалося їх переконати, що його код написав студент.

Підведемо підсумки аналізу коду, згенерованого за допомогою ЗШІ ChatGPT, за двома останніми колонками Таблиці 1:

1. Середня оцінка всіх ЗШІ (без Copilot) дорівнює нулю. Тобто, в середньому ЗШІ взагалі не змогли визначитися з походженням коду, який був згенерований за допомогою ЗШІ ChatGPT.
2. Середня оцінка всіх викладачів (завдяки вкладу Викладачів 3 та 4) виявилася достатньо великою на користь студента і дорівнює -0,55. Або ж це складає -27,5% зсуву цієї оцінки у від'ємну сторону (сторону самостійності студента) в діапазоні шкали -2 ÷ 0.
3. Загальна середня оцінка усіх програм усіма експертами (без Copilot) дорівнює -0,275, що у відсотках зсуву цієї оцінки у від'ємну сторону (сторону самостійності студента) за шкалою -2 ÷ 0 складає -13,75%.

Аналіз коду, згенерованого за допомогою ЗШІ Copilot.

Загальна картина оцінювання коду ЗШІ Copilot виявилася схожою на картину оцінювання коду ЗШІ ChatGPT, але з невеликим зсувом середніх оцінок кожного з експертів в ту чи іншу сторону.

ЗШІ ChatGPT, Gemini і You.com, виступаючи в ролі експертів, дали середні оцінки коду ЗШІ Copilot із зсувом у додатну сторону, тобто оцінили його як трохи більш схожий на код штучного інтелекту, ніж вони оцінювали код ЗШІ ChatGPT. Що цікаво, Викладач 1 також дав середню оцінку із зсувом у ту саму сторону. В той час, як ЗШІ YesChatAI і всі інші викладачі оцінили код ЗШІ Copilot як більш схожий на код студента, ніж код ЗШІ ChatGPT.

Підведемо підсумки аналізу коду, згенерованого за допомогою ЗШІ Copilot за двома останніми колонками Таблиці 1:

1. Середня оцінка всіх ЗШІ (без Copilot) не сильно відрізняється від нуля (+0,1). Тобто, в середньому ЗШІ знову майже не змогли визначитися також і з походженням коду, який був згенерований за допомогою ЗШІ Copilot, хоча сумарно й оцінили його як мінімально схожий на код ЗШІ.
2. Середня оцінка всіх викладачів виявилася ще більшою на користь студента, ніж була у ЗШІ ChatGPT, і дорівнює -0,8. Або ж це складає -40% зсуву цієї оцінки у від'ємну сторону (сторону самостійності студента) в діапазоні шкали -2 ÷ 0. Тобто, ЗШІ Copilot своїм згенерованим кодом зумів ввести в оману більшість викладачів.
3. Загальна середня оцінка усіх програм усіма експертами (без Copilot) дорівнює -0,35, що у відсотках зсуву цієї оцінки у від'ємну сторону (сторону самостійності студента) за шкалою -2 ÷ 0 складає -17,5%.

Аналіз коду, згенерованого за допомогою ЗШІ Gemini.

Код, що був згенерований за допомогою ЗШІ Gemini, незважаючи на прохання написати в стилі студента, виявився зовсім не схожим на код студента. В ньому було дуже багато коментарів з текстом, який студенти навряд чи написали б. Тому й оцінки цього коду були зовсім іншими, ніж оцінки коду ЗШІ ChatGPT та Copilot.

Жоден із ЗШІ як експерт не дав жодної від'ємної оцінки коду ЗШІ Gemini. ЗШІ YesChatAI дав однозначну відповідь (всі оцінки дорівнюють +2) на всі програми, що їх код є згенерованим. Сам Gemini досить легко визначив згенерованість свого власного коду в усіх випадках (середня оцінка +1,8). ЗШІ ChatGPT також достатньо впевнено визначив штучність коду ЗШІ Gemini (середня оцінка +1,4). Але найдивнішою виявилась оцінка ЗШІ You.com (середня оцінка +0,6), який оцінив цей явно згенерований код як більш «студентський», ніж він раніше оцінював код, що був згенерований за допомогою ЗШІ ChatGPT (середня оцінка +0,8) і Copilot (середня оцінка +1,2).

Викладачі взагалі виявили завидну одностайність в оцінюванні коду ЗШІ Gemini. З усіх оцінок усіх викладачів коду усіх програм є тільки дві немаксимальні додатні оцінки +1, а всі інші оцінки дорівнюють +2. Тобто, викладачі одностайно і безсумнівно визначили код ЗШІ Gemini як код, що був згенерований засобом зі штучним інтелектом.

Підведемо підсумки аналізу коду, згенерованого за допомогою ЗШІ Gemini, за двома останніми колонками Таблиці 1:

1. Середня оцінка всіх ЗШІ (без Copilot) достатньо однозначно визначає код ЗШІ Gemini, як код штучного інтелекту (+1,45). Або ж це складає +72,5% зсуву цієї оцінки у додатну сторону (сторону коду ЗШІ) в діапазоні шкали $0 \div 2$. Тобто, ЗШІ Gemini своїм згенерованим кодом не зумів ввести в оману інші ЗШІ, що виступали в ролі експертів.
2. Середня оцінка всіх викладачів виявилася взагалі майже максимально додатною і дорівнює +1,9. Або ж це складає +95% зсуву цієї оцінки у додатну сторону (сторону коду ЗШІ) в діапазоні шкали $0 \div 2$. Тобто, викладачі не повірили коду ЗШІ Gemini зовсім.
3. Загальна середня оцінка усіх програм усіма експертами (без Copilot) дорівнює 1,675, що у відсотках зсуву цієї оцінки у додатну сторону (сторону коду ЗШІ) за шкалою $0 \div 2$ складає +83,75%.

Висновки

1. ЗШІ Gemini взагалі не зміг замаскувати свій код під студента-першокурсника. Видається, що викладачу буде неважко обґрунтувати свою думку, чому він вважає, що код, згенерований ЗШІ Gemini, є кодом ЗШІ.

2. Найкраще маскує свій код під студента-першокурсника ЗШІ Copilot. Єдиною ознакою, за якою досвідчений викладач може ідентифікувати цей код як код ЗШІ, є текст тих небагатьох коментарів, які вставив ЗШІ Copilot в код програми. Текст цих коментарів записаний фразами, які є не зовсім характерними для сучасних студентів. Якщо в коді, що згенерував ЗШІ Copilot, студент переформулює коментарі по своєму, або навіть просто видалить всі коментарі, або якщо сам Copilot згенерує код взагалі без коментарів, то цей код не буде мати явних ознак відмінності від коду, який, як правило, пишуть студенти-першокурсники. Але проблема полягає в тому, що формально, навіть з такими коментарями, викладач не має однозначних доводів, щоб стверджувати, що цей код був згенерований за допомогою ЗШІ.

3. ЗШІ ChatGPT також добре маскує свій код під студента-першокурсника, але в середньому все ж таки трохи гірше, ніж Copilot.

4. Навіть, якщо студент-першокурсник згенерує код за допомогою ЗШІ Copilot або ChatGPT, а викладач зуміє ідентифікувати цей код як код ЗШІ, то однаково викладачу буде важко обґрунтувати студенту свою думку, чому він вважає, що цей код є кодом ЗШІ, якщо студент буде наполягати, що він сам написав цей код. І якщо викладач не буде приймати у студентів такі роботи або знижувати оцінку, то це може призвести до конфліктних ситуацій зі студентами, оскільки достатньо доказові ознаки коду ЗШІ в цих програмах відсутні.

5. ЗШІ Copilot, коли виступав в якості експерта, оцінював код на можливість його генерування за допомогою ЗШІ якось аномально і не виявив вміння виявляти такий код. Можливо така його поведінка витікає навіть з політики компанії-розробника. Тому, навряд чи можна рекомендувати викладачам використання ЗШІ Copilot з метою виявлення коду ЗШІ. Хоча це дуже прикро, бо ЗШІ Copilot є необмежено безкоштовним у браузері Edge.

6. Із широкодоступних потужних ЗШІ найбільш точну інформацію, чи є певний код кодом ЗШІ, надає ЗШІ Gemini. Він також надає детальні і достатньо аргументовані пояснення, чому цей код є кодом ЗШІ. Тому, ЗШІ Gemini можна рекомендувати викладачам як помічника у виявленні коду ЗШІ в роботах студентів. Але зазначена в розділі опису використаних в дослідженні ЗШІ обмежена безкоштовність ЗШІ Gemini, відповідно, обмежує й можливості його використання викладачами для перевірки великої кількості студентських програм. Хоча це обмеження є не дуже критичним, якщо часу на перевірку є достатньо багато.

7. ЗШІ ChatGPT, хоча й уникає однозначних оцінок щодо наданого йому коду, але також надає достатньо непогану пояснюючу інформацію щодо властивостей цього коду, яку викладач може використати при оцінюванні робіт студентів. Перевагою ЗШІ ChatGPT є його необмежена безкоштовність, що є дуже важливим фактором для викладачів при перевірці великої кількості робіт.

8. Спеціальні засоби штучного інтелекту від інших компаній, такі як ЗШІ You.com та YesChatAI достатньо непогано виявляють код ЗШІ. Але вони є фактично непридатними для використання викладачами з метою виявлення коду ЗШІ, оскільки надають дуже невелику кількість

безкоштовних запитів для того, щоб ці засоби можна було в реальній практиці використовувати для перевірки великої кількості студентських робіт.

9. Суб'єктивізм викладачів в оцінюванні коду на ознаки коду ЗШІ виявився достатньо великим, в залежності від досвіду роботи з першокурсниками і особистої довірливості викладача. Але тут потрібно зауважити, що значна різниця в оцінюванні коду ЗШІ виявилася тільки у випадку достатньо непогано замаскованого під студента коду ЗШІ ChatGPT і особливо ЗШІ Copilot. А якщо код ЗШІ Gemini має явні ознаки генерації штучним інтелектом, то всі викладачі без проблем виявили штучність створення цього коду і оцінили його однаково.

В якості напрямків подальших досліджень можна запропонувати наступне. Оскільки потужність ЗШІ розвивається вибухово швидко, то виглядає доцільним повторити дослідження цієї статті ще раз через рік-два. Ще одним напрямком можна запропонувати провести аналогічне дослідження при використанні інших мов програмування, а також при виконанні завдань з інших дисциплін та/або більшої складності.

Список бібліографічного опису:

1. Dayu Yang, Tianyang Liu, Daoan Zhang, Antoine Simoulin, Xiaoyi Liu, Yuwei Cao, Zhaopu Teng, Xin Qian, Grey Yang, Jiebo Luo, and Julian McAuley, " Code to Think, Think to Code: A Survey on Code-Enhanced Reasoning and Reasoning-Driven Code Intelligence in LLMs", *arXiv:2502.19411v1 [cs.CL] 26 Feb 2025*. Available: https://arxiv.org/html/2502.19411v1. Accessed on: May 15, 2025.
2. Juyong Jiang, Fan Wang, Jiasi Shen, Sungju Kim, and Sunghun Kim, "A Survey on Large Language Models for Code Generation", *arXiv:2406.00515v2 [cs.CL] 10 Nov 2024*. Available: https://arxiv.org/pdf/2406.00515. Accessed on: May 15, 2025.
3. Janek Bevendorff, Matti Wiegmann, Jussi Karlgren, Luise Dürlich, Evangelia Gogoulou, Aarne Talman, Efstathios Stamatatos, Martin Potthast, and Benno Stein, "Overview of the "Voight-Kampg" Generative AI Authorship Verification Task at PAN and ELOQUENT 2024", *Working Notes of the Conference and Labs of the Evaluation Forum (CLEF 2024),* Grenoble, France, 9-12 September, 2024. pp. 2486-2506, Available: https://ceur-ws.org/Vol-3740/paper-225.pdf. Accessed on: May 15, 2025.
4. Jijie Huang, Yang Chen, Man Luo, and Yonglan Li, "Generative AI Authorship Verification Of Tri-Sentence Analysis Base On The Bert Model", *Working Notes of the Conference and Labs of the Evaluation Forum (CLEF 2024),* Grenoble, France, 9-12 September, 2024. pp. 2632-2637, Available: https://ceur-ws.org/Vol-3740/paper-243.pdf. Accessed on: May 15, 2025.
5. Xuan Ji, "Detecting ai-generated code in C++: A comprehensive guide", BytePlus, Apr 25, 2025. Available: https://www.byteplus.com/en/topic/500341?title=detecting-ai-generated-code-in-c-a-comprehensive-guide. Accessed on: May 15, 2025.
6. Oseremen Joy Idialu, Noble Saji Mathews, Rungroj Maipradit, Joanne M. Atlee, and Mei Nagappan, "Whodunit: Classifying Code as Human Authored or GPT-4 Generated - A case study on CodeChef problems", *MSR '24: Proceedings of the 21st International Conference on Mining Software Repositories*, 2024, pp. 394 – 406. Available: https://doi.org/10.1145/3643991.3644926. Accessed on: May 15, 2025.
7. David Alvarez-Fidalgo and Francisco Ortin, "CLAVE: A deep learning model for source code authorship verification with contrastive learning and transformer encoders", *Information Processing & Management*, Volume 62, Issue 3, May 2025. Available: https://doi.org/10.1016/j.ipm.2024.104005. Accessed on: May 15, 2025.
8. "Top 4 AI Source Code Detector Tools for Enterprises", *BlueOptima*, 11 April 2024, Available: https://www.blueoptima.com/top-4-ai-source-code-detector-tools-for-enterprises/. Accessed on: May 15, 2025.
9. "AI Content Detection in C", *Sampling*, Available: https://sapling.ai/programming-language/ai-content-detector/c. Accessed on: May 15, 2025.
10. "The enterprise AI platform – powered by your data", *You.com*, Available: https://you.com. Accessed on: May 15, 2025.
11. "AI Code Detector – AI-Generated Code Analysis", *YESCHAT AI*, Available: https://surl.li/vbhtoj. Accessed on: May 15, 2025.
12. Hyunjae Suh, Mahan Tafreshipour, Jiawei Li, Adithya Bhattiprolu, Iftekhar Ahmed, "An Empirical Study on Automatically Detecting AI-Generated Source Code: How Far Are We?", *arXiv:2411.04299v1 [cs.SE] 06 Nov 2024*. Available: https://arxiv.org/pdf/2411.04299v1. Accessed on: May 15, 2025.
13. Азиранкулов Є. А. «Спосіб ідентифікації використання чат-ботів зі штучним інтелектом при написанні студентами програм мовою С» : *магістерська дис. : 123 Комп'ютерна інженерія / Азиранкулов Єгор Анатолійович*. – Київ, 2023. – 99 с. Режим доступу до ресурсу: https://ela.kpi.ua/handle/123456789/64438. Дата звернення: 15.05.2025.

References:

1. Dayu Yang, Tianyang Liu, Daoan Zhang, Antoine Simoulin, Xiaoyi Liu, Yuwei Cao, Zhaopu Teng, Xin Qian, Grey Yang, Jiebo Luo, and Julian McAuley, " Code to Think, Think to Code: A Survey on Code-Enhanced Reasoning and Reasoning-Driven Code Intelligence in LLMs", *arXiv:2502.19411v1 [cs.CL] 26 Feb 2025*. Available: https://arxiv.org/html/2502.19411v1. Accessed on: May 15, 2025.
2. Juyong Jiang, Fan Wang, Jiasi Shen, Sungju Kim, and Sunghun Kim, "A Survey on Large Language Models for Code Generation", *arXiv:2406.00515v2 [cs.CL] 10 Nov 2024*. Available: https://arxiv.org/pdf/2406.00515. Accessed

on: May 15, 2025.
3. Janek Bevendorff, Matti Wiegmann, Jussi Karlgren, Luise Dürlich, Evangelia Gogoulou, Aarne Talman, Efstathios Stamatatos, Martin Potthast, and Benno Stein, "Overview of the "Voight-Kampg" Generative AI Authorship Verification Task at PAN and ELOQUENT 2024", *Working Notes of the Conference and Labs of the Evaluation Forum (CLEF 2024),* Grenoble, France, 9-12 September, 2024. pp. 2486-2506, Available: https://ceur-ws.org/Vol-3740/paper-225.pdf. Accessed on: May 15, 2025.
4. Jijie Huang, Yang Chen, Man Luo, and Yonglan Li, "Generative AI Authorship Verification Of Tri-Sentence Analysis Base On The Bert Model", *Working Notes of the Conference and Labs of the Evaluation Forum (CLEF 2024),* Grenoble, France, 9-12 September, 2024. pp. 2632-2637, Available: https://ceur-ws.org/Vol-3740/paper-243.pdf. Accessed on: May 15, 2025.
5. Xuan Ji, "Detecting ai-generated code in C++: A comprehensive guide", BytePlus, Apr 25, 2025. Available: https://www.byteplus.com/en/topic/500341?title=detecting-ai-generated-code-in-c-a-comprehensive-guide. Accessed on: May 15, 2025.
6. Oseremen Joy Idialu, Noble Saji Mathews, Rungroj Maipradit, Joanne M. Atlee, and Mei Nagappan, "Whodunit: Classifying Code as Human Authored or GPT-4 Generated - A case study on CodeChef problems", *MSR '24: Proceedings of the 21st International Conference on Mining Software Repositories*, 2024, pp. 394 – 406. Available: https://doi.org/10.1145/3643991.3644926. Accessed on: May 15, 2025.
7. David Alvarez-Fidalgo and Francisco Ortin, "CLAVE: A deep learning model for source code authorship verification with contrastive learning and transformer encoders", *Information Processing & Management*, Volume 62, Issue 3, May 2025. Available: https://doi.org/10.1016/j.ipm.2024.104005. Accessed on: May 15, 2025.
8. "Top 4 AI Source Code Detector Tools for Enterprises", *BlueOptima*, 11 April 2024, Available: https://www.blueoptima.com/top-4-ai-source-code-detector-tools-for-enterprises/. Accessed on: May 15, 2025.
9. "AI Content Detection in C", *Sampling*, Available: https://sapling.ai/programming-language/ai-content-detector/c. Accessed on: May 15, 2025.
10. "The enterprise AI platform – powered by your data", *You.com*, Available: https://you.com. Accessed on: May 15, 2025.
11. "AI Code Detector – AI-Generated Code Analysis", *YESCHAT AI*, Available: https://surl.li/vbhtoj. Accessed on: May 15, 2025.
12. Hyunjae Suh, Mahan Tafreshipour, Jiawei Li, Adithya Bhattiprolu, Iftekhar Ahmed, "An Empirical Study on Automatically Detecting AI-Generated Source Code: How Far Are We?", *arXiv:2411.04299v1 [cs.SE] 06 Nov 2024*. Available: https://arxiv.org/pdf/2411.04299v1. Accessed on: May 15, 2025.
13. Azyrankulov Ye. A. «Sposib identyfikatsii vykorystannia chat-botiv zi shtuchnym intelektom pry napysanni studentamy prohram movoiu S» : *mahisterska dys. : 123 Kompiuterna inzheneriia / Azyrankulov Yehor Anatoliiovych.* – Kyiv, 2023. – 99 s. Rezhym dostupu do resursu: https://ela.kpi.ua/handle/123456789/64438. Data zvernennia: 15.05.2025.