DOI 10.31891/2307-5732-2024-345-6-36
УДК 004.023

КРАВЧУК ОЛЬГА
Хмельницький національний університет
https://orcid.org/0000-0001-6937-5001
e-mail: kravchukoa2@gmail.com

ШТУЧНИЙ ІНТЕЛЕКТ У ПРОГРАМУВАННІ: ЯК ШІ ЗМІНЮЄ ПІДХІД ДО РОЗРОБКИ ТА АВТОМАТИЗАЦІЇ КОДУ

У статті розглядається вплив штучного інтелекту на процеси програмування, зокрема автоматизацію коду, оптимізацію алгоритмів та тестування. Аналізуються сучасні дослідження у сфері ШІ-програмування, основні виклики, що постають перед розробниками, та перспективи подальшого розвитку технології. Основна увага приділяється питанням ефективності, безпеки та етичності використання штучного інтелекту у розробці програмного забезпечення. Робота містить аналіз актуальних наукових публікацій та визначає можливі напрями майбутніх досліджень.

Ключові слова: штучний інтелект, програмування, автоматизація коду, машинне навчання, тестування, оптимізація алгоритмів, етика штучного інтелекту, кібербезпека.

KRAVCHUK OLHA
Khmelnytsky National University

ARTIFICIAL INTELLIGENCE IN PROGRAMMING: HOW ШІ IS CHANGING THE APPROACH TO CODE DEVELOPMENT AND AUTOMATION

The article discusses the impact of artificial intelligence on programming processes, including code automation, algorithm optimisation, and testing. The article analyses current research in the field of AI programming, the main challenges faced by developers, and the prospects for further development of the technology. The main attention is paid to the efficiency, security and ethics of using artificial intelligence in software development. The paper contains an analysis of relevant scientific publications and identifies possible areas for future research.

Artificial intelligence has long ceased to be a science fiction concept and has become an important part of many industries, including programming. It helps to automate complex processes, optimise code, and increase developer productivity. In recent years, AI tools have become an integral part of the developer workflow, helping to write code faster, detect errors at early stages, and even create new algorithms. This not only speeds up the work but also changes the very approach to software development.

Thanks to artificial intelligence technologies, programming becomes more accessible and efficient, which opens up new opportunities for both experienced developers and beginners. In this article, we will consider how artificial intelligence changes programming, what tools exist, and what challenges developers face in the world of automated development.

The introduction of artificial intelligence into programming brings both significant benefits and challenges. The main problem is the balance between automation and maintaining human control. Automated code writing and optimisation can reduce development time, but at the same time raise questions of security, ethics, and trust in machine solutions.

From a scientific point of view, research in AI programming is focused on improving machine learning models, reducing error rates, improving integration with modern programming languages, and developing self-learning algorithms. On the practical side, companies and developers seek to minimise the risks associated with the use of artificial intelligence, avoid code duplication, and create more efficient automated processes.

Thus, the study of artificial intelligence in programming is critical for the development of modern technologies, as it allows not only to increase productivity but also to ensure the safe and ethical use of artificial intelligence in software engineering.

Keywords: artificial intelligence, programming, code automation, machine learning, testing, algorithm optimisation, ШІ ethics, cybersecurity.

Постановка проблеми у загальному вигляді та її зв'язок із важливими науковими чи практичними завданнями

Штучний інтелект (ШІ) вже давно перестав бути лише концепцією з наукової фантастики і став важливою частиною багатьох галузей, зокрема програмування. З його допомогою автоматизуються складні процеси, оптимізується код і підвищується продуктивність розробників. В останні роки ШІ-інструменти стали невід'ємною частиною робочого процесу розробників, допомагаючи писати код швидше, виявляти помилки на ранніх етапах та навіть створювати нові алгоритми. Це не лише прискорює роботу, але й змінює сам підхід до розробки програмного забезпечення.

Завдяки технологіям штучного інтелекту, програмування стає більш доступним і ефективним, що відкриває нові можливості як для досвідчених розробників, так і для новачків. У цій статті ми розглянемо, як штучний інтелект змінює програмування, які існують інструменти та які виклики стоять перед розробниками у світі автоматизованої розробки.

Впровадження штучного інтелекту у програмування несе як значні переваги, так і суттєві виклики. Основна проблема полягає у балансі між автоматизацією та збереженням

контролю з боку людини. Автоматичне написання коду та його оптимізація можуть скоротити час розробки, але водночас ставлять питання безпеки, етичності та довіри до машинних рішень.

З наукової точки зору, дослідження у сфері ШІ-програмування зосереджені на вдосконаленні моделей машинного навчання, зниженні рівня помилок, покращенні інтеграції з сучасними мовами програмування та розробці самонавчальних алгоритмів. З практичного боку, компанії та розробники прагнуть мінімізувати ризики, пов'язані з використанням штучного інтелекту, уникати дублювання коду та створювати більш ефективні автоматизовані процеси.

Таким чином, дослідження штучного інтелекту у програмуванні є критично важливим для розвитку сучасних технологій, адже дозволяє не лише підвищити продуктивність, а й забезпечити безпечне та етичне використання штучного інтелекту в програмній інженерії.

Аналіз останніх досліджень і публікацій

Останні дослідження у сфері штучного інтелекту в програмуванні демонструють стрімкий розвиток технологій автоматизованого написання коду, тестування та оптимізації алгоритмів. У наукових роботах розглядаються методи підвищення точності ШІ-моделей, їх адаптація до різних мов програмування та вплив на продуктивність розробників [1].

Основні напрями досліджень:

- генерація коду та його оптимізація – останні наукові роботи фокусуються на вдосконаленні мовних моделей (LLM) для точнішого передбачення коду та зниження рівня помилок.

- автоматизоване тестування – дослідження вказують на зростаючу ефективність штучного інтелекту у створенні тестових сценаріїв і виявленні багів;

- етичні аспекти ШІ у програмуванні – вчені розглядають питання авторського права, відповідальності за згенерований код і потенційні упередження в алгоритмах [1];

- штучний інтелект у кібербезпеці – аналізуються способи використання машинного навчання для запобігання кібератакам та покращення захисту програмного забезпечення [4];

- зокрема, дослідження, опубліковані в журналах IEEE Transactions on Software Engineering, ACM Computing Surveys та Nature Machine Intelligence, підтверджують значний прогрес у розробці ШІ-інструментів для програмування. Вони вказують, що ефективне використання ШІ може підвищити продуктивність розробників на 30-50% та значно скоротити час на тестування програмного забезпечення.

Формулювання цілей статті

У цій статті розглянемо, як штучний інтелект впливає на процес розробки програмного забезпечення та які інструменти вже доступні для автоматизації.

Виклад основного матеріалу

Розглянемо використання штучного інтелекту у написанні коду. Сучасні ШІ-моделі, такі як GitHub Copilot, ChatGPT, Tabnine та CodeWhisperer, можуть генерувати фрагменти коду, допомагати виправляти помилки та навіть пропонувати оптимальні алгоритми. Вони використовують глибоке навчання на великих наборах даних коду, щоб розуміти контекст і допомагати розробникам.

Перевагами штучного інтелекту, щодо допомоги для розробників є:

- прискорення роботи – штучний інтелект може писати стандартний код, дозволяючи програмістам фокусуватися на логіці;

- зменшення кількості помилок – автоматизоване тестування та аналіз коду допомагають виявити помилки ще на ранньому етапі;

- оптимізація алгоритмів, тобто штучний інтелект може створювати коментарі до коду, пояснюючи його функціональність, що полегшує підтримку та розширення програмного забезпечення, а також штучний інтелект може пропонувати покращені варіанти реалізації завдань;

- допомога в навчанні – новачки можуть використовувати штучний інтелект для швидкого освоєння мов програмування.
- генерація тестів – штучний інтелект допомагає створювати юніт-тести та інтеграційні тести, що значно підвищує якість програмного забезпечення та спрощує процес тестування.

Таким чином, використання штучного інтелекту у написанні коду дозволяє розробникам зосередитися на складніших ідейних завданнях, залишаючи рутинні аспекти роботи машинним алгоритмам.

Штучний інтелект та автоматизація процесів розробки. Автоматизація розробки – це використання інструментів, скриптів і систем, які дозволяють спростити, прискорити або повністю замінити людську працю в процесі створення програмного забезпечення. Це охоплює: автоматичну генерацію коду; CI/CD (неперервна інтеграція та доставка); тестування; моніторинг продуктивності; розгортання. ШІ не лише допомагає писати код, а й активно використовується для автоматизації розгортання, тестування та моніторингу систем [5].

Основні напрямки автоматизації:

- CI/CD (Continuous Integration / Continuous Deployment) – штучний інтелект допомагає аналізує зміни в коді та прогнозує можливі збої;
- автоматизоване тестування – сервіси, такі як Selenium або Test. ШІ, використовують ШІ для створення тестових сценаріїв;
- оптимізація продуктивності – штучний інтелект допомагає знаходити вузькі місця в продуктивності програмного забезпечення;
- кібербезпека – алгоритми машинного навчання можуть виявляти підозрілі патерни та попереджати про потенційні атаки.

Штучний інтелект виводить автоматизацію на новий рівень завдяки здатності: аналізувати код (інструменти на базі ШІ можуть виявляти помилки, покращувати структуру коду, пропонувати рефакторинг); генерувати код (моделі на кшталт ChatGPT або GitHub Copilot допомагають створювати фрагменти коду на основі опису завдання); писати тести (ШІ може автоматично створювати юніт-тести, що значно прискорює процес тестування); прогнозувати помилки (за допомогою аналізу історії проєкту ШІ може передбачити, де найвірогідніше виникнуть баги); працювати з документацією (автоматичне генерування та оновлення документації проєкту).

Виклики та майбутнє штучного інтелекту у програмуванні. Попри численні переваги, штучний інтелект у програмуванні має свої виклики [2].

По-перше, це залежність від штучного інтелекту, тобто надмірне використання ШІ може знизити рівень навичок розробників. Отже, залежність – це стан, коли програміст постійно покладається на допомогу штучного інтелекту, замість того щоб самостійно вирішувати задачі; шукати й аналізувати помилки; розвивати аналітичне мислення; вдосконалювати власні навички. Проявами залежності є те, що деякі розробники відчувають "психологічну потребу" постійно отримувати підказки, а також втрачення навичок читання документації або самостійного вивчення нових технологій та відсутність глибокого розуміння коду, що ускладнює роботу в команді, дебагінг і масштабування проєктів. Чому це небезпечно? Тому що це призводить до зниження якості розробників (без практики — немає розвитку); до поверхневого мислення (програміст більше "копіює", ніж "створює"); до ризику некоректного коду (тучний інтелект не завжди дає оптимальні або безпечні рішення) та до проблеми з працевлаштуванням (роботодавці шукають спеціалістів із глибокими знаннями, а не лише користувачів штучного інтелекту). Але цю залежність можна зменшити, використовучи ШІ як помічника, а не замість себе; ставити запитання, а потім перевіряй відповіді; не довіряй сліпо, тобто писати код самостійно хоча б частину часу; розбирайся в тому, що робить ШІ - це допоможе навчитися новому та вчися – не лише за допомогою штучного інтелекту, а й із книг, курсів, документації.

По-друге, це безпека коду – автоматично згенерований код може містити уразливості [3,4,6]. Основні ризики для безпеки це: вразливості у згенерованому коді (ШІ

може створити код з XSS, SQL-ін'єкціями, незахищеними API-запитами та часто пропускається обробка помилок, перевірка прав доступу, шифрування тощо); вставка потенційно небезпечних бібліотек (ШІ може порадити бібліотеку або фреймворк без урахування останніх вразливостей або ліцензійних обмежень; плагіат або порушення авторського права (ШІ може випадково згенерувати фрагменти коду, що були взяті з відкритих репозиторіїв без належного зазначення авторства); витік конфіденційної інформації (при роботі з хмарними ШІ існує ризик передавання фрагментів приватного коду або секретних ключів на зовнішні сервери). Тому необхідно забезпечити безпеку при використанні ШІ в програмуванні, тобто перевіряти згенерований код вручну, обов'язково виконувати аудит коду та застосовувати статичний аналізатор коду (наприклад, SonarQube, ESLint, Pylint); вчися в ШІ, але не покладайся сліпо, використовувати код як приклад, але писати остаточну версію самостійно;, уникати передавання чутливих даних, тобіш не надсилати у ШІ-контекст API-ключі, паролі, або фрагменти приватного коду без дозволу; використовувати sandbox або тестове середовище, тобто перед тим як застосовувати згенерований код у продуктові, потрібно перевірити його у безпечному середовищі; оновлювати залежності, перевіряти, що всі бібліотеки та інструменти, запропоновані ШІ, актуальні та безпечні. Отже, штучний інтелект – це потужний союзник, але не заміна людській відповідальності за безпеку. Вивчай, використовуй, експериментуй, але завжди перевіряй.

Програмування – це творчість і логіка, а не лише автозаповнення. Щоб зберегти навички та зростати як спеціаліст, важливо не тільки користуватись інструментами, а й залишатися розробником.

У майбутньому штучний інтелект стане ще більш інтегрованим у розробницькі процеси, можливо, навіть зможе створювати повністю автономні програмні рішення. Проте роль програміста залишиться важливою, оскільки контроль, креативність та стратегічне мислення неможливо повністю автоматизувати.

Таким чином, штучний інтелект змінює підхід до програмування, роблячи його швидшим і ефективнішим. Використання ШІ-інструментів дає змогу автоматизувати багато процесів, але важливо зберігати баланс між автоматизацією та людським контролем. Штучний інтелект – це не заміна програмістів, а потужний інструмент для підвищення їхньої продуктивності та ефективності.

Висновки з даного дослідження і перспективи подальших розвідок у даному напрямі

Дослідження показують, що штучний інтелект має величезний потенціал у сфері програмування. Він не тільки спрощує процес розробки, але й дозволяє автоматизувати рутинні завдання, покращуючи ефективність програмістів. Використання ШІ-інструментів у генерації коду, тестуванні, оптимізації алгоритмів та кібербезпеці відкриває нові горизонти для інновацій.

Однак залишаються важливі питання, що потребують подальших досліджень, зокрема: підвищення точності та надійності ШІ-інструментів (покращення моделей, щоб зменшити кількість помилок у згенерованому коді; етичні аспекти та авторське право (дослідження можливостей створення регулюючих норм щодо використання штучного інтелекту у програмуванні); інтеграція штучного інтелекту в різні мови програмування (оптимізація моделей для роботи з широким спектром технологій); автономні ШІ-розробники (перспективи створення повністю автономних ШІ-систем, що можуть виконувати складні розробницькі завдання).

Отже, штучний інтелект вже зараз змінює програмування і ця тенденція лише посилюватиметься. У майбутньому його роль у програмній інженерії стане ще важливішою, але ключовим залишиться баланс між автоматизацією та людським контролем. Але, майбутнє штучного інтелекту – це не лише технічна революція, а й зміна всієї парадигми взаємодії з інформацією та навколишнім світом починаючи з інтеграція ШІ у повсякденне життя (смарт-помічники, автопілоти, персональні рекомендаційні системи стануть ще точнішими й адаптивнішими та відбудеться глибша інтеграція ШІ в побут, медицину,

освіту, правосуддя) і закінчуючи квантовим ШІ (комбінація квантових обчислень із ШІ відкриє нові горизонти в обробці складних задач: моделювання молекул, прогнозування клімату, криптографія).

Література

1. Ethical guidelines on the use of artificial intelligence (AI) and data in teaching and learning for educators. European Commission. 2022. URL: https://op.europa.eu/en/publication-detail/-/publication/d81a0d54-5348-11ed-92ed-01aa75ed71a1/language-en.
2. These 100 companies are leading the AI revolution /The World Economic Forum. URL: https://www.weforum.org/agenda/2018/01/these-100-companies-are-leading-the-world-in-artificial-intelligence (дата звернення 04.04.2025)
3. Schwalbe N, Wahl B. Artificial intelligence and the future of global health. Lancet. 2020 May 16. URL: https://pubmed.ncbi.nlm.nih.gov/32416782/(дата звернення 04.00.2025)
4. 31 Best AI Coding Tools in 2024. URL: https://www.scaler.com/blog/coding-ai-tools/
5. AI That Writes Code: 7 Best & Popular AI Coding Tools 2024. URL: https://www.bytracyjackson.com/ai-that-writes-code/
6. Best AI Coding Tools in 2024. URL: https://www.qodo.ai/blog/best-ai-coding-assistant-tools/

References

1. Ethical guidelines on the use of artificial intelligence (AI) and data in teaching and learning for educators. European Commission. 2022. URL: https://op.europa.eu/en/publication-detail/-/publication/d81a0d54-5348-11ed-92ed-01aa75ed71a1/language-en.
2. The World Economic Forum (2018), "These 100 companies are leading the AI revolution", URL: https://www.weforum.org/agenda/2018/01/these-100-companies-are-leading-the-world-in-artificial-intelligence (Accessed 04.04.2025).
3. Schwalbe N, Wahl B. Artificial intelligence and the future of global health. Lancet. 2020 May 16. URL: https://pubmed.ncbi.nlm.nih.gov/32416782/(Accessed 04.00.2025)
4. 31 Best AI Coding Tools in 2024. Available: https://www.scaler.com/blog/coding-ai-tools/
5. AI That Writes Code: 7 Best & Popular AI Coding Tools 2024. URL: https://www.bytracyjackson.com/ai-that-writes-code/
6. Best AI Coding Tools in 2024. URL: https://www.qodo.ai/blog/best-ai-coding-assistant-tools/