УДК 004.8:004.4:681.518
DOI https://doi.org/10.32782/2307-1222.2025-59-22

АВТОМАТИЗОВАНЕ ФОРМУВАННЯ ТЕХНІЧНОЇ ДОКУМЕНТАЦІЇ В ГАЛУЗІ ІТ ІЗ ВИКОРИСТАННЯМ ВЕЛИКИХ МОВНИХ МОДЕЛЕЙ

Володимир ПАСІЧНИК
доктор технічних наук, професор,
професор кафедри інформаційних систем та мереж
Національного університету «Львівська політехніка»
вул. С. Бандери, 12, м. Львів
ORCID: 0000-0002-5231-6395
vpasichnyk@gmail.com

Максим ЯРОМИЧ
аспірант кафедри прикладної лінгвістики
Національного університету «Львівська політехніка»
вул. С. Бандери, 12, м. Львів
ORCID: 0009-0005-3299-6695
yamax0312@gmail.com

Автоматизоване формування технічної документації є однією із ключових проблем сучасної ІТ-галузі, оскільки якісна й актуальна документація необхідна для ефективної роботи розробників, тестувальників і кінцевих користувачів. У статті розглянуто можливості використання великих мовних моделей (LLM) для генерації технічної документації, оцінено потенціал онтологічного підходу у структуризації процесу документування.

Відзначено, що традиційні методи створення документації супроводжуються низкою труднощів, зокрема великим обсягом ручної праці, складністю підтримки актуальності текстів та забезпечення узгодженості між різними розділами документації. Неналежна інтеграція документації із процесами розроблення програмного забезпечення призводить до застарівання матеріалів і ускладнює їх оновлення. Також наголошено, що в умовах швидких циклів розроблення фахівці часто приділяють не досить уваги документації, що ускладнює її підтримку в актуальному стані.

Особлива увага приділена концептуальній інтеграції великих мовних моделей з онтологічними підходами, що дозволяє не лише генерувати тексти, а й забезпечувати їхню структурованість відповідно до формальних моделей знань. Використання онтологій сприяє підвищенню точності документації, зменшенню помилок, а також створенню єдиного стандарту оформлення текстів. У статті представлено огляд підходів до інтеграції великих мовних моделей з онтологіями, зокрема й методи автоматизованого витягування знань із текстів, застосування онтологічного моделювання для структуризації документації.

У дослідженні також проаналізовано переваги й обмеження автоматизованого підходу. До основних переваг належать швидкість створення документації, забезпечення стандартизації, зменшення навантаження на розробників і зниження кількості людських помилок. Водночас відзначаються такі виклики, як необхідність навчання моделей на специфічних доменних корпусах, контроль точності згенерованого тексту й інтеграція великих мовних моделей у процеси життєвого циклу розробки.

Окремий розділ статті присвячено аналізу практичних кейсів застосування великих мовних моделей для генерації документації. Наведено приклад автоматичного створення документації для відкритого програмного продукту, реалізованого мовою Python, де GPT-4 генерувала технічні описи й інструкції користувача. Отримані результати показали, що використання мовних моделей дозволяє значно покращити якість документації, забезпечити чітку структуру, правильне форматування та відповідність загальноприйнятим стандартам. Водночас зазначено, що для повної відповідності офіційним вимогам може знадобитися додаткове редагування текстів і включення більш детальних описів параметрів і змінних.

Також розглянуто перспективи розвитку автоматизації документації, зокрема використання гібридних підходів, які поєднують мовні моделі, онтології та алгоритми контролю якості тексту. Зазначено можливість подальшої інтеграції великих мовних моделей у середовища розробки (IDE), що дозволить автоматично оновлювати документацію під час змін у кодовій базі.

Підкреслено, що автоматизація створення технічної документації за допомогою великих мовних моделей і онтологій є перспективним напрямом, що дозволяє значно підвищити ефективність роботи розробників, мінімізувати помилки та зменшити витрати часу на підтримку документації. Проте для забезпечення високої точності та відповідності технічним стандартам необхідні подальші дослідження у сфері адаптації мовних моделей до конкретних галузей, розроблення методів контролю точності текстів та інтеграції автоматичних систем документації з наявними інструментами DevOps та керування кодом.

Ключові слова: *автоматизація, технічна документація, великі мовні моделі, онтології генерація тексту, структуризація знань.*

AUTOMATED GENERATION OF TECHNICAL DOCUMENTATION IN THE IT INDUSTRY USING LARGE LANGUAGE MODELS

Volodymyr PASICHNYK
DSc of Technical Science, Professor
Professor at the Department of Information Systems and Networks
Lviv Polytechnic National University
12 Bandery str., Lviv
ORCID: 0000-0002-5231-6395
vpasichnyk@gmail.com

Maksym YAROMYCH
PhD Student at the Department of Applied Linguistics
Lviv Polytechnic National University
Bandery str., 12, Lviv
ORCID: 0009-0005-3299-6695
yamax0312@gmail.com

Automated generation of technical documentation is one of the key challenges in the modern IT industry, as high-quality and up-to-date documentation is essential for the effective work of developers, testers, and end users. This paper explores the potential of using large language models (LLMs) for generating technical documentation and evaluates the role of the ontological approach in structuring the documentation process.

It has been noted that traditional methods of documentation creation face several difficulties, including a significant amount of manual labor, challenges in maintaining content relevance, and ensuring consistency across different sections. The lack of integration between documentation and software development processes leads to outdated materials and complicates their updates. Additionally, it is emphasized that in fast-paced development cycles, developers often pay insufficient attention to documentation, making it difficult to keep it up to date.

Special attention is given to the conceptual integration of LLMs with ontological approaches, which allows not only the generation of texts but also ensures their structured organization according to formal knowledge models. The use of ontologies enhances documentation accuracy, reduces errors, and helps establish a unified documentation standard. This study presents an overview of approaches to integrating LLMs with ontologies, including automated knowledge extraction methods and the application of ontological modeling to documentation structuring.

The research also analyzes the advantages and limitations of automated documentation generation. Key benefits include faster document creation, standardization, reduced workload for developers, and fewer human errors. However, several challenges are identified, such as the need for model training on domain-specific corpora, ensuring text accuracy, and integrating LLMs into the software development lifecycle.

A separate section of the study is dedicated to analyzing practical cases of using LLMs for documentation generation. An example is presented of automatically creating documentation for an open-source software project written in Python, where GPT-4 was used to generate technical descriptions and user instructions. The results indicate that language models significantly improve documentation quality by ensuring a clear structure, proper formatting, and compliance with widely accepted standards. However, it is noted that for full compliance with official requirements, additional text editing and the inclusion of more detailed descriptions of parameters and variables may be necessary.

The paper also discusses the future prospects of documentation automation, particularly through hybrid approaches that combine language models, ontologies,

and quality control algorithms. It highlights the potential integration of LLMs into development environments (IDEs), which would allow automatic updates to documentation as code changes occur.

It is emphasized that automating technical documentation generation using LLMs and ontologies is a promising direction that significantly increases developer efficiency, minimizes errors, and reduces the time required to maintain documentation. However, to ensure high accuracy and compliance with technical standards, further research is required in adapting language models to specific domains, developing methods for text accuracy control, and integrating automatic documentation systems with existing DevOps and code management tools.

Key words: *automation, technical documentation, large language models, ontologies, text generation, knowledge structuring.*

Постановка проблеми. На сучасному етапі розвитку ІТ-галузі критично важливим завданням є створення якісної і актуальної документації. Документація забезпечує розробників, тестувальників і кінцевих користувачів необхідною інформацією для розуміння, використання та підтримки програмних продуктів. Проте процес її створення та підтримки стикається з низкою проблем.

По-перше, обсяг і складність сучасних програмних систем ускладнюють підтримку документації в актуальному стані. Зміни в коді часто не супроводжуються відповідними оновленнями в документації, що призводить до її застарівання та неточностей [3]. Це може спричинити неправильне розуміння функціоналу та помилки під час використання або розширенні системи [2]. По-друге, створення документації є трудомістким процесом, який потребує значних ресурсів. Розробники часто зосереджені на написанні коду та впровадженні нових функцій, тому на документацію може не вистачати часу й уваги. Це особливо актуально в умовах швидкого циклу розробки та частих релізів [1]. По-третє, забезпечення узгодженості між різними частинами документації та її відповідності стандартам є складним завданням. Різні автори можуть використовувати різні стилі та підходи до написання, що призводить до непослідовності й ускладнює сприйняття інформації користувачами [5].

Формування документації складається з кількох ключових етапів, як-от аналіз вимог, визначення структури документа, написання та редагування тексту, а також його узгодження із зацікавленими сторонами. Кожен із цих етапів потребує значних ресурсів, як часових, так і кадрових.

За оцінками фахівців, трудомісткість підготовки технічної документації може становити від 15 до 30–40% загального часу розроблення проєкту, залежно від складності системи та вимог до документації. У великих проєктах цей показник може бути ще вищий, особливо якщо потрібна деталізована специфікація, інструкції користувача та супровідні матеріали. Окрім того, постійне оновлення документації у процесі життєвого циклу продукту збільшує загальну витрату ресурсів.

Автоматизація за допомогою великих мовних моделей дозволяє значно зменшити ці витрати, скоротити час написання документації на 50–70% завдяки швидкому генеруванню текстів, узгодженню термінології, автоматичному оновленню змін у програмному коді. Використання великих мовних моделей (далі – LLM) на основі онтологічного підходу пропонує перспективні рішення для автоматизації процесу створення та підтримки документації. LLM здатні аналізувати великі обсяги коду та супутньої інформації, генерувати зрозумілі й узгоджені описи функціоналу, API та інших компонентів системи [4]. Це може значно зменшити навантаження на розробників і забезпечити актуальність і точність документації.

Проте впровадження таких технологій також має свої виклики. Необхідно забезпечити точність і релевантність згенерованого тексту, а також інтеграцію із процесами розробки й інструментами, що існують. Також важливо враховувати питання безпеки та конфіденційності даних під час використання LLM у корпоративному середовищі [6].

Отже, актуальність дослідження використання великих мовних моделей для генерації документації до програмного забезпечення зумовлена потребою в ефективних і автоматизованих підходах до створення та підтримки якісної документації, що відповідає сучасним вимогам розроблення програмних систем.

Мета роботи. Проаналізувати використання великих мовних моделей і онтологічного підходу для автоматизованого формування документації на ІТ-проєкти, розкрити їхній потенціал у подоланні ключових викликів створення технічної документації, як-от актуальність, структурованість і ефективність, а також оцінити переваги й обмеження запропонованого підходу.

Виклад основного матеріалу. *Аналіз стану досліджень.* Автоматизація створення технічної документації є важливим напрямом сучасних ІТ-досліджень, оскільки якісна документація відіграє ключову роль у розробленні та підтримці програмного забезпечення. Традиційно створення документації виконувалося вручну, що потребувало значних ресурсів і часу, особливо у великих проєктах. Це призводило до проблем з актуальністю документації, людськими помилками та труднощами в узгодженні між командами розробників і технічних письменників. Дослідження в цій сфері зосереджені на пошуку методів автоматизації, що дозволяють значно зменшити трудомісткість процесу [17].

Сучасні підходи до автоматизації передбачають використання великих мовних моделей, які вже продемонстрували свою ефективність у генеруванні текстів, узагальненні інформації, аналізі структурованих даних. Дослідження [22] показує, що інтеграція LLM із системами керування документацією може значно прискорити процес написання та підтримки актуальності технічних матеріалів.

Попри значний прогрес, за автоматизованого підходу до створення документації також є деякі виклики. Зокрема, проблема точності та релевантності згенерованої документації залишається відкритою. Багато досліджень відзначають, що LLM можуть створювати тексти, які містять стилістичні або фактичні помилки, що потребує подальшої перевірки людиною. Окрім того, інтеграція

LLM у процеси розроблення програмного забезпечення потребує спеціалізованих налаштувань для адаптації моделей до специфіки проєктів. Для підвищення точності згенерованих текстів дослідники пропонують використовувати такі підходи, як Retrieval Augmented Generation (далі – RAG) [31], які поєднують можливості LLM із доступом до зовнішніх джерел знань, зменшують кількість «галюцинацій» і підвищують достовірність інформації.

Іншим важливим аспектом є стандартизація автоматично створених документів. Хоча LLM можуть генерувати технічні описи, специфікації, інструкції, питання відповідності індустріальним стандартам залишається нерозв'язаним. Дослідження підкреслюють необхідність розроблення спеціалізованих шаблонів і моделей, які б гарантували узгодженість вихідних матеріалів із загальноприйнятими нормами.

Отже, аналіз наукових робіт демонструє, що використання LLM у сфері технічної документації є перспективним, проте ще потребує вдосконалення. Майбутні дослідження можуть бути спрямовані на покращення механізмів перевірки точності згенерованих текстів, інтеграцію LLM у системи керування життєвим циклом програмного забезпечення та створення адаптивних моделей для роботи з різними доменами технічної інформації.

Основи великих мовних моделей. Великі мовні моделі (LLM) є фундаментом сучасних систем обробки природної мови, забезпечують можливості для розуміння, генерації та перекладу тексту. Вони базуються на архітектурі трансформерів, яка дозволяє ефективно обробляти послідовності даних завдяки механізму самоуваги. Це забезпечує моделям здатність фокусуватися на різних частинах вхідного тексту, що є ключовим для розуміння контексту та семантики [7].

Історія розвитку LLM бере свій початок із ранніх досліджень у сфері обробки природної мови. Перші мовні моделі, як-от ELIZA (1966 р.), виконували найпростіші діалогові завдання, але не мали здатності до узагальнення чи розуміння контексту. Справжній прорив стався у 2017 р. з появою трансформерів, які дозволили моделям обробляти контекст тексту ефективніше, ніж попередні рекурентні нейронні мережі (далі – RNN). Ця архітектура стала основою для сучасних моделей, як-от BERT [8] та GPT [9].

Сучасні LLM, як-от GPT-3 та GPT-4, мають мільярди параметрів і навчаються на масивних текстових корпусах. Це дає змогу їм виконувати широкий спектр завдань: від генерації тексту та машинного перекладу до автоматизації створення технічної документації [5]. Наприклад, модель GPT-4 демонструє здатність генерувати технічні описи функціоналу, специфікацій API та навіть автоматизувати супровід документації для програмного забезпечення.

Однак великі мовні моделі мають і свої виклики. Одним із ключових є «галюцинації» моделей, коли вони генерують текст, який виглядає правдоподібно, але є хибним. Ця проблема висвітлюється в роботах [1; 2]. Окрім того, для навчання й обслуговування LLM потрібні величезні обчислювальні ресурси, що обмежує їх доступність для малих і середніх компаній.

Отже, великі мовні моделі стали потужним інструментом для обробки природної мови й автоматизації процесів створення документації. Водночас їх упровадження потребує подальших досліджень для подолання поточних викликів і максимізації їхнього потенціалу в різних галузях.

Виклики у створенні документації до програмного забезпечення. Створення якісної, актуальної документації є невід'ємною частиною розроблення програмного забезпечення. Однак цей процес супроводжується низкою значних викликів, які впливають на ефективність і якість кінцевого результату.

Однією з головних проблем є складність підтримки документації в актуальному стані. У динамічному середовищі розробки програмного забезпечення зміни в коді відбуваються швидше, ніж оновлюється супутня документація. Це призводить до виникнення невідповідностей між кодом і документацією, що створює труднощі для розробників, які підтримують або використовують цей код. Автоматизація цього процесу може значно спростити підтримку документації в актуальному стані, особливо у великих проєктах.

Іншою важливою проблемою є трудомісткість процесу. Створення документації часто потребує багато часу та залучення окремих фахівців або цілих команд, які можуть бути зайняті іншими критично важливими завданнями. У роботі [29] зазначено, що автоматизовані інструменти для генерації API-документації можуть знизити час, необхідний для створення технічних описів, і зменшити залежність від людських ресурсів.

Окрім цього, серед викликів можна виділити також забезпечення узгодженості стилю та змісту документації. Різні автори документації можуть використовувати різні підходи до формулювання тексту, що призводить до стилістичних і змістових розбіжностей. У дослідженні [30] наголошується, що інтеграція автоматизованого документування в середовищах розробки (IDE) може допомогти підтримувати стилістичну узгодженість документації завдяки інтерактивному процесу оновлення.

Ще одним важливим аспектом є складність стандартизації документації. Відсутність єдиних стандартів і правил часто призводить до того, що документація не задовольняє вимог замовників або кінцевих користувачів. Наприклад, у роботі [10] розглядається застосування великих мовних моделей для створення стандартизованих і семантично узгоджених описів функцій і API. Це дозволяє усунути розбіжності між різними частинами документації та забезпечити її відповідність стандартам.

Отже, сучасний процес створення документації до програмного забезпечення потребує інноваційних підходів, які б дозволили автоматизувати цей процес, забезпечити водночас точність, узгодженість і актуальність документації. Використання великих мовних моделей у поєднанні з автоматизованими інструментами є перспективним рішенням для подолання цих викликів.

Потенціал великих мовних моделей у генерації документації. Великі мовні моделі стали революційним інструментом у сфері автоматизації процесів,

пов’язаних із текстовою інформацією. Їхній потенціал у генерації документації до програмного забезпечення зумовлений здатністю аналізувати великі обсяги тексту, зчитувати структуровану та неструктуровану інформацію, а також створювати зв’язний і зрозумілий текст.

Однією з основних переваг LLM є автоматизація рутинних завдань, як-от створення API-документації, технічних описів, звітів про тестування й інструкцій для користувачів. Наприклад, у роботі [10] розглядається використання LLM для створення стандартизованої документації API. Автори підкреслюють, що моделі можуть автоматично генерувати технічні описи, які відповідають стандартам документації та знижують ризик помилок.

Важливою вимогою під час створення документації є забезпечення її узгодженості й актуальності. LLM можуть автоматично оновлювати текст на основі змін у коді за допомогою даних із систем контролю версій, як-от Git. У дослідженні [28] описано, як моделі забезпечують узгодженість документації завдяки інтеграції із процесами Continuous Integration/Continuous Delivery (далі – CI/CD).

LLM також мають широкі можливості для створення багатомовної документації. Завдяки можливостям перекладу й адаптації контенту для різних аудиторій мовні моделі можуть автоматично генерувати документацію кількома мовами. Це особливо корисно для програмного забезпечення, орієнтованого на міжнародний ринок. Також LLM можуть використовуватися для витягування знань безпосередньо з коду. Наприклад, модель може аналізувати коментарі до коду, специфікації API або навіть сам код для генерації технічних описів функціоналу. Так, у роботі [28] описано, як моделі можуть автоматично інтегруватися з репозиторіями коду для створення документації, яка відповідає поточному стану проєкту. Окрім цього, вони здатні створювати інтерактивну документацію, яка дозволяє користувачам взаємодіяти з текстом через інтерфейси, як-от чат-боти або інтерактивні запити.

Попри значні можливості, використання LLM для генерації документації має низку викликів. По-перше, моделі можуть генерувати «галюцинації» – інформацію, яка виглядає достовірною, але є хибною. По-друге, навчання моделей потребує великих обчислювальних ресурсів, що може стати перешкодою для малих і середніх компаній.

Отже, великі мовні моделі мають значний потенціал у трансформації підходів до генерації документації. Їх упровадження може знизити трудомісткість процесу, підвищити точність і узгодженість текстів, а також зробити документацію доступною для ширшого кола користувачів. Інтеграція LLM з онтологічними підходами відкриває нові перспективи для автоматизації документації. У роботі [27] розглядається можливість використання онтологій для створення семантично багатої документації, яка не лише описує функціонал, але й ураховує контекст і потреби користувачів.

Роль онтологій у структуризації документації. У створенні документації важливу роль відіграють структуризація й організація, у чому може допомогти

використання онтологічного підходу. Онтології забезпечують формальне представлення знань у визначеній предметній сфері, визначають основні поняття та взаємозв'язки між ними. Це дозволяє створити чітку структуру для технічної документації, що сприяє її узгодженості та легкості сприйняття. Наприклад, у роботі [4] обговорюється використання онтологій для забезпечення міждисциплінарної взаємодії на рівні загальної мови категорій, що сприяє уніфікації та стандартизації інформації

Онтології можуть бути використані для автоматизованого витягування технічної інформації з нормативних документів. Так, у дослідженні [11] представлено методологію, яка дозволяє автоматично витягувати технічні концепції з нормативних документів за допомогою семантичних інструментів, що підвищує ефективність створення й оновлення технічної документації. Використання онтологій у структуризації документації також сприяє покращенню пошуку та доступу до інформації. Онтологічні моделі забезпечують багату семантику домену знань, надають точні метадані, які машини можуть краще зрозуміти й обробити. У результаті механізми пошуку інформації стають більш розумними та точними, здатними обробляти розширені запити та проводити семантичний пошук.

Поєднання онтологій із LLM дозволяє автоматизувати процес генерації технічної документації на основі структур, визначених онтологіями, забезпечує узгодженість і точність інформації. У дослідженні [12] розглядається використання глибокого навчання для розуміння та представлення семантики великих структурованих документів, що може бути застосовано для покращення автоматизованої генерації документації.

Розглянемо ситуацію, коли компанія розробляє складний програмний продукт із багатьма модулями. Створення та підтримка актуальної документації для такого продукту є складним завданням. За допомогою онтології, яка описує всі модулі, їхні функції та взаємозв'язки, можна автоматизувати процес генерації документації. LLM, інтегрована із цією онтологією, може генерувати детальні описи кожного модуля, його призначення, інтерфейсів і взаємодій з іншими модулями, забезпечити узгодженість і актуальність інформації.

Отже, онтології відіграють важливу роль у структуризації технічної документації, забезпечують чітке й узгоджене представлення інформації. Їх інтеграція з великими мовними моделями відкриває нові можливості для автоматизації процесу створення та підтримки документації в галузі IT.

Підходи до інтеграції LLM з онтологіями. Інтеграція великих мовних моделей (LLM) з онтологіями відкриває нові горизонти в автоматизованому створенні технічної документації в галузі IT. Це поєднання дозволяє забезпечити більш структурований, узгоджений і змістовний підхід до генерації документації, що відповідає сучасним вимогам до інформаційних систем.

Одним із підходів до інтеграції є перетворення тексту природної мови на онтологічні структури. У дослідженні [13] автори налаштували модель GPT-3 для перекладу речень природної мови в синтаксис OWL (Web Ontology Language).

Це дозволяє автоматично збагачувати онтології новими поняттями та зв'язками, що сприяє створенню більш детальної та точної технічної документації.

Інший підхід передбачає автоматизоване витягування знань із технічних текстів за допомогою LLM. Моделі аналізують великі обсяги документації, ідентифікують ключові поняття та їхні взаємозв'язки, які потім структуруються у вигляді онтологічних моделей. Це сприяє створенню узгодженої та структурованої технічної документації, як, наприклад, у дослідженні [11].

Окрім того, онтології можуть слугувати основою для генерації технічної документації за допомогою LLM. Структурований набір знань, представлений в онтології, використовується моделлю для створення зв'язного та змістовного тексту. Це дозволяє автоматично генерувати документацію, яка відповідає встановленим стандартам і містить актуальну інформацію. У дослідженні [14] розглядається метод автоматичного заповнення онтологій доменними знаннями за допомогою LLM, що сприяє швидкому й автоматизованому збагаченню онтологій необхідною інформацією.

Практичне застосування цих підходів можна побачити, наприклад, у створенні API-документації. За допомогою онтологій, що описують структуру та функціональність API, LLM можуть генерувати детальні описи методів, параметрів і повернених значень, забезпечувати узгодженість документації та спрощувати процес її оновлення в разі зміни API. У дослідженні [15] розглядається підхід до документування програмних технологій, який передбачає створення моделей, що описують сценарії використання технологій, включаючи артефакти, мови програмування, потоки даних та інші аспекти, що сприяє більш ефективному створенню та підтримці технічної документації.

Ще одним прикладом є генерація посібників користувача. LLM, інтегровані з онтологіями програмного забезпечення, можуть автоматично створювати посібники, які пояснюють функціональність системи, її модулі та взаємодію між ними. Це забезпечує користувачів актуальною та точною інформацією, знижує навантаження на технічних письменників. Також LLM можуть бути використані для створення навчальних матеріалів. Зокрема, у дослідженні [16] розглядається можливість використання LLM для автоматизованого створення навчальних курсів для дорослих. Автори розробили прототип курсу за допомогою LLM, упровадили процес з участю людини для забезпечення точності та ясності згенерованого контенту. Результати показали, що такий підхід може прискорити створення навчальних матеріалів без втрати якості, що є перспективним напрямом у сфері освіти.

Отже, інтеграція LLM з онтологіями дозволяє автоматизувати процес створення технічної документації та навчальних матеріалів, забезпечити їхню точність, узгодженість і актуальність. Цей підхід має значний потенціал для покращення процесу створення та підтримки документації в галузі IT і освіти.

Практичні кейси та застосування. Розглянемо кілька прикладів використання великих мовних моделей для автоматизації створення документації. У дослідженні [17] автори проводять огляд сучасних архітектур автоматизації документів

(далі – DA) з акцентом на інтеграцію LLM. Метою DA є зменшення ручної праці під час створення документів шляхом автоматичного збирання й інтеграції даних із різних джерел і складання документів відповідно до визначених шаблонів. Автори зазначають, що, хоча існують огляди комерційних рішень DA, особливо в юридичній сфері, досі не було всеосяжного огляду академічних досліджень щодо архітектур і технологій DA. Це дослідження заповнює цю прогалину, надає чіткіше визначення та характеристику DA, визначає сучасні архітектури та технології DA в академічних дослідженнях, а також пропонує ідеї для нових дослідницьких можливостей у сфері DA з урахуванням останніх досягнень у генеративному штучному інтелекті та LLM (рис. 1).

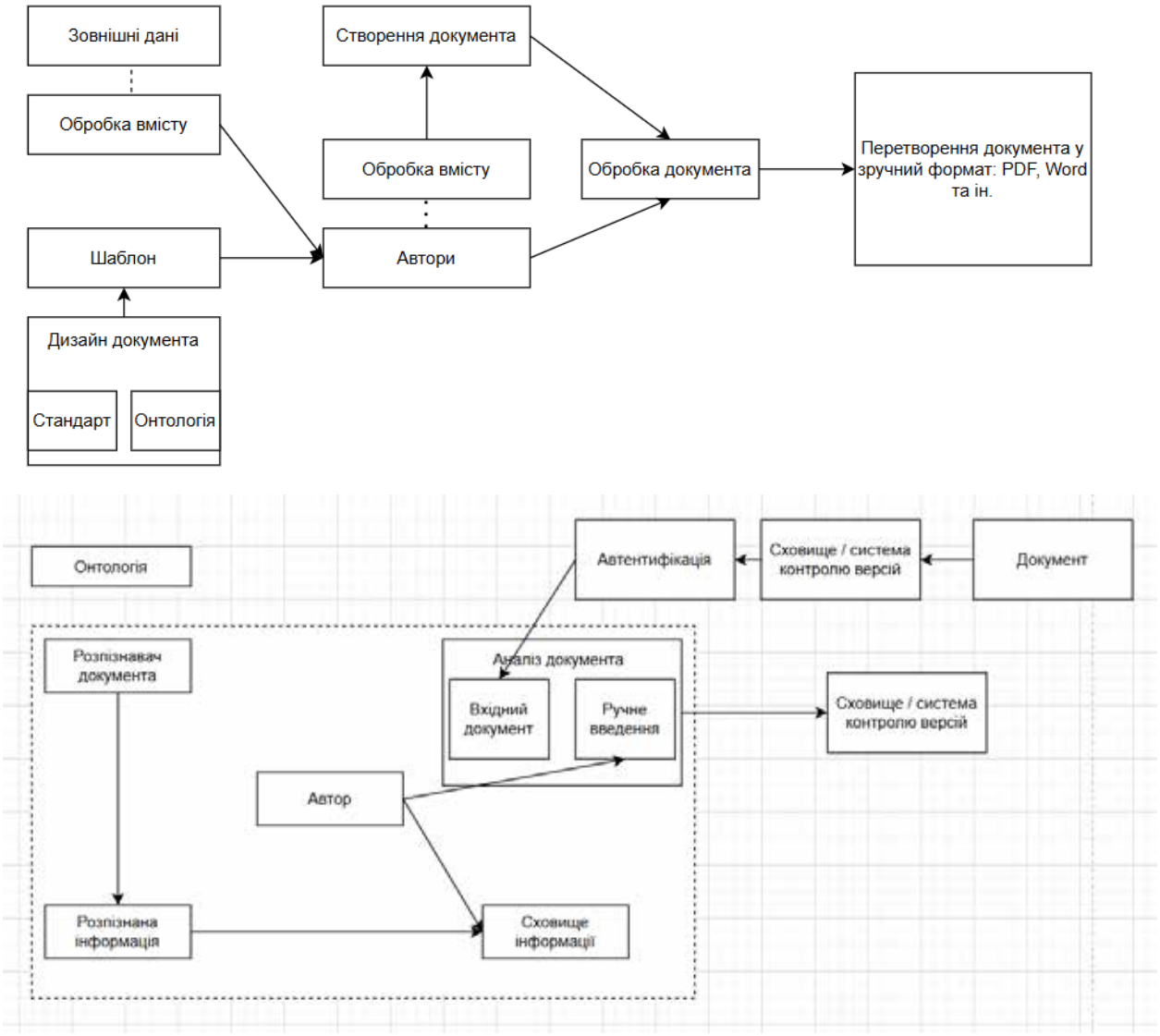


Рис. 1. Архітектура автоматизації документації

Робота [18] представляє комплексну систему, яка використовує LLM для генерації специфікацій OpenAPI з різноманітної API-документації. У цифрову еру широке використання API є очевидним, однак масштабоване використання API ускладнюється через розбіжності у структурі онлайн-документації API. Це

підкреслює потребу в автоматичних інструментах для спрощення використання API. SpeCrawler вирішує цю проблему, переводить документацію у формат специфікації API за допомогою ретельно розробленого конвеєра. Створенням стандартизованого формату для численних API SpeCrawler сприяє оптимізації процесів інтеграції в системах оркестрації API та полегшує включення інструментів у LLM. У дослідженні детально описано методологію SpeCrawler, підкріплену емпіричними доказами та кейсами, продемонстровано її ефективність завдяки можливостям LLM.

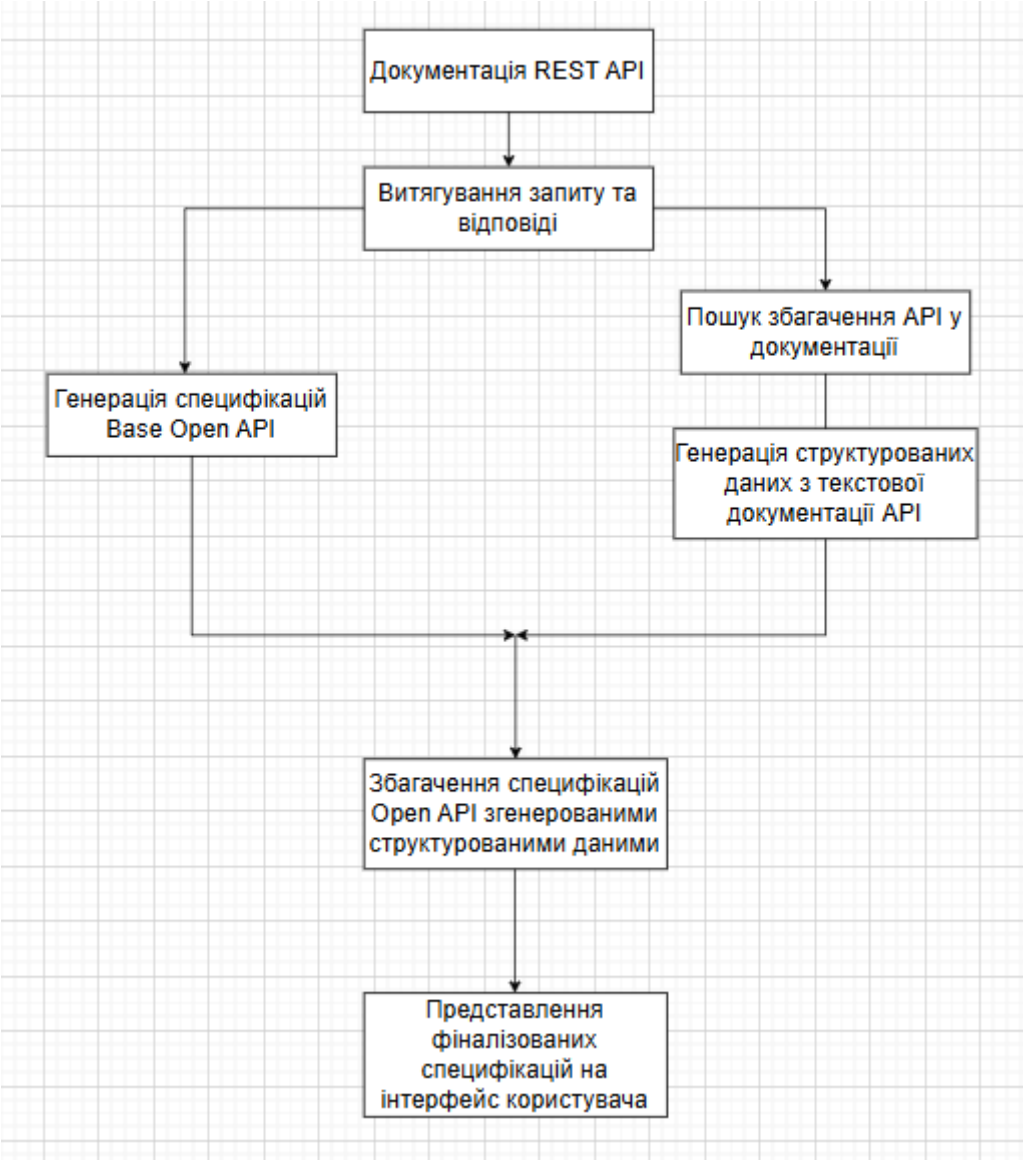
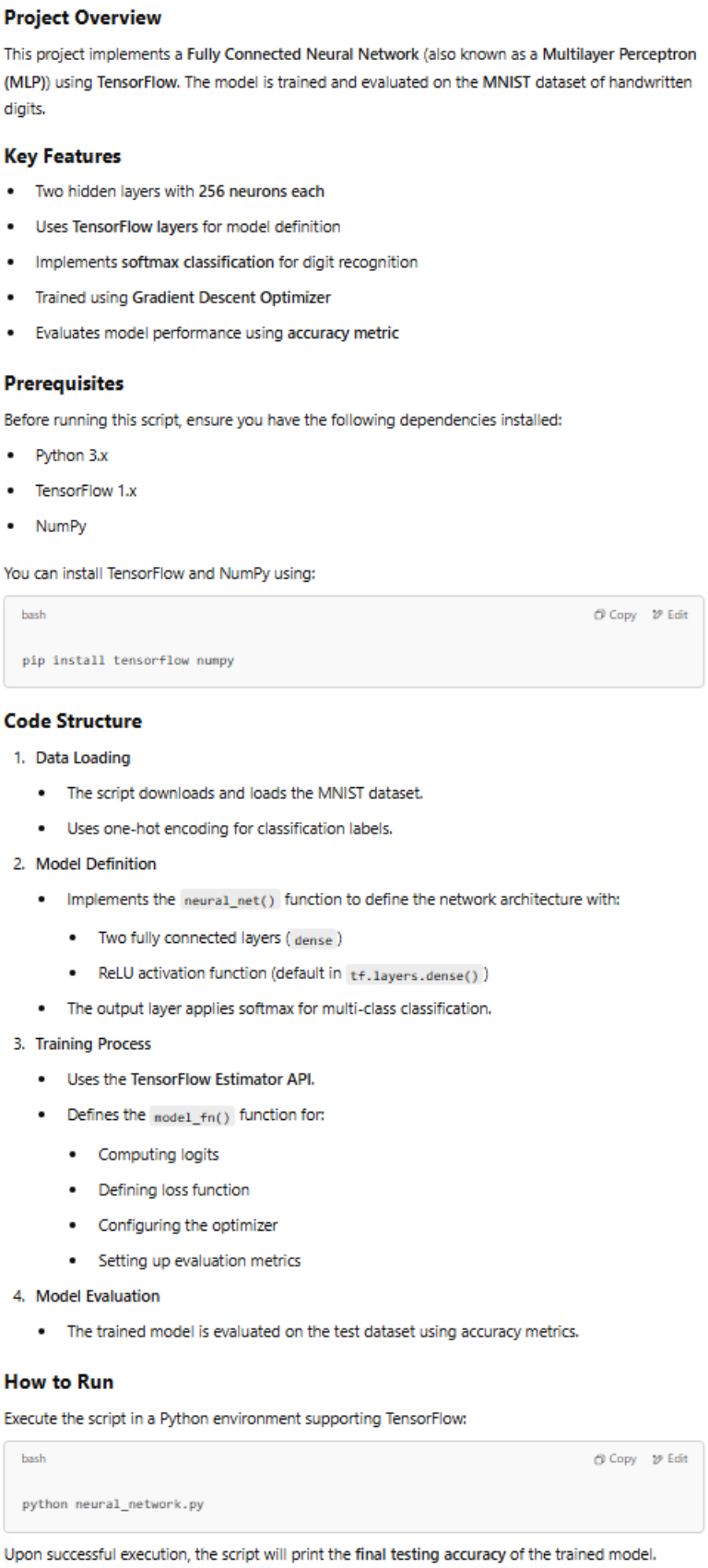


Рис. 2. Процес генерації специфікацій

Спробуємо для наочності згенерувати засобами GPT-4o документацію на невеликий проєкт нейронної мережі з відкритим кодом, реалізований мовою програмування Python, що доступний у відкритому репозиторії на Github (https://github.com/aymericdamien/TensorFlow-Examples/blob/master/examples/3_NeuralNetworks/neural_network.py).

Після запиту згенерувати документацію у двох варіантах – англійською й українською мовами було отримано такі результати (рис. 3, 4):



Project Overview

This project implements a **Fully Connected Neural Network** (also known as a **Multilayer Perceptron (MLP)**) using **TensorFlow**. The model is trained and evaluated on the **MNIST** dataset of handwritten digits.

Key Features

- Two hidden layers with **256 neurons each**
- Uses **TensorFlow layers** for model definition
- Implements **softmax classification** for digit recognition
- Trained using **Gradient Descent Optimizer**
- Evaluates model performance using **accuracy metric**

Prerequisites

Before running this script, ensure you have the following dependencies installed:

- Python 3.x
- TensorFlow 1.x
- NumPy

You can install TensorFlow and NumPy using:

```
bash
pip install tensorflow numpy
```

Code Structure

1. **Data Loading**
   - The script downloads and loads the MNIST dataset.
   - Uses one-hot encoding for classification labels.
2. **Model Definition**
   - Implements the `neural_net()` function to define the network architecture with:
     - Two fully connected layers (`dense`)
     - ReLU activation function (default in `tf.layers.dense()`)
   - The output layer applies softmax for multi-class classification.
3. **Training Process**
   - Uses the **TensorFlow Estimator API**.
   - Defines the `model_fn()` function for:
     - Computing logits
     - Defining loss function
     - Configuring the optimizer
     - Setting up evaluation metrics
4. **Model Evaluation**
   - The trained model is evaluated on the test dataset using accuracy metrics.

How to Run

Execute the script in a Python environment supporting TensorFlow:

```
bash
python neural_network.py
```

Upon successful execution, the script will print the **final testing accuracy** of the trained model.

Рис. 3. Згенерована документація англійською мовою

Огляд проєкту

Цей проєкт реалізує **повнозв'язну нейронну мережу** (Multilayer Perceptron, MLP) за допомогою **TensorFlow**. Модель навчена на наборі даних **MNIST**, який містить зображення рукописних цифр.

Основні особливості

- Два прихованих шари по **256 нейронів**
- Використовуються **TensorFlow layers** для визначення моделі
- Класифікація цифр через **softmax**
- Навчання за допомогою **Gradient Descent Optimizer**
- Оцінка точності моделі через **метрику accuracy**

Необхідні залежності

Перед запуском скрипта потрібно встановити такі бібліотеки:

- Python 3.x
- TensorFlow 1.x
- NumPy

Встановлення TensorFlow та NumPy:

```bash
pip install tensorflow numpy
```

Структура коду

1. **Завантаження даних**
   - Скрипт завантажує та обробляє набір даних MNIST.
   - Використовується one-hot кодування для міток.
2. **Визначення моделі**
   - Функція `neural_net()` визначає архітектуру мережі:
     - Два повнозв'язних шари (`dense`)
     - Функція активації ReLU (за замовчуванням у `tf.layers.dense()`)
   - Вихідний шар застосовує softmax для багатокласової класифікації.
3. **Процес навчання**
   - Використовується **TensorFlow Estimator API**.
   - Функція `model_fn()` визначає:
     - Обчислення логітів
     - Функцію втрат
     - Налаштування оптимізатора
     - Метрики оцінки точності
4. **Оцінка моделі**
   - Тренована модель перевіряється на тестових даних за допомогою метрики точності.

Як запустити

Запустіть скрипт у середовищі, яке підтримує TensorFlow:

```bash
python neural_network.py
```

Після виконання скрипта буде виведено **остаточну точність моделі** на тестових даних.

Рис. 4. Згенерована документація українською мовою

Створена документація відповідає основним стандартам технічного письма, оскільки має чітку структуру, зрозумілий стиль викладу та належне форматування.

Вона містить загальний опис проєкту, перелік ключових особливостей, необхідні залежності, архітектуру коду й інструкцію щодо запуску. Такий підхід спрощує розуміння коду як для початківців, так і для досвідчених розробників. Однак для цілковитої відповідності офіційним стандартам документації, як-от Google Style Guide for Python Documentation або PEP-257, можна було б додати більше деталей у вигляді docstring-коментарів у коді, опис змінних і параметрів функцій, а також приклади їх використання.

Однією з важливих особливостей цієї документації є її двомовність: представлення однакової інформації англійською й українською мовами демонструє можливість автоматизації багатомовної документації. Великі мовні моделі (LLM) можуть значно спростити цей процес завдяки автоматичному перекладу й адаптації текстів під конкретні потреби користувачів. Це є особливо важливим для міжнародних проєктів, які мають аудиторію в різних країнах. Окрім того, такі моделі здатні аналізувати код, виокремлювати ключові аспекти, автоматично структурувати текст і формулювати його у зрозумілій формі. У майбутньому це може суттєво знизити витрати часу на створення документації, особливо у великих командах розробників.

Важливим аспектом є також правильне структурування інформації. Документація містить детальний опис архітектури нейронної мережі, що дозволяє швидко зрозуміти логіку роботи коду. Включення інструкцій зі встановлення залежностей спрощує розгортання проєкту, а чітке форматування та розбиття інформації на логічні блоки покращує читабельність. Це відповідає найкращим практикам у сфері технічного письма.

Переваги та виклики автоматизованого підходу. Автоматизований підхід до створення технічної документації у сфері інформаційних технологій має суттєві переваги, проте водночас породжує низку викликів, які необхідно враховувати під час упровадженні таких рішень. Розглянемо ці аспекти детальніше.

Однією з головних переваг автоматизації є підвищення швидкості й ефективності створення документації. Ручне написання технічних документів може потребувати значної кількості часу, особливо якщо йдеться про великі проєкти із численними модулями, взаємозв'язками та змінами. Використання великих мовних моделей у поєднанні з онтологіями дозволяє генерувати структуровану документацію в режимі реального часу, що суттєво скорочує витрати часу на підготовку технічних текстів.

Окрім того, автоматизація сприяє мінімізації людських помилок. Навіть досвідчені фахівці можуть припускатися помилок, дублювати інформацію або упускати важливі деталі. Автоматизовані системи, що працюють на основі онтологій, забезпечують логічну цілісність тексту, уникають суперечностей або пропусків у документації. У дослідженні [3] було продемонстровано, що використання LLM для витягування ключової інформації та її узгодження з онтологіями дозволяє значно покращити узгодженість створених документів.

Ще однією вагомою перевагою є стандартизація документації. Часто технічні документи створюються різними командами, що може призвести до відмінностей

у стилі, структурі та формулюваннях. Автоматизовані підходи забезпечують єдиний стандарт оформлення текстів, що полегшує сприйняття інформації. Використання LLM у поєднанні із семантичними технологіями дозволяє підтримувати стандартизацію незалежно від зміни авторів чи редакторів. У дослідженні [26] зазначається, що інтеграція мовних моделей з онтологіями підвищує якість і відповідність документації міжнародним стандартам.

Окрім цього, суттєвою перевагою автоматизації є також економія ресурсів. Використання автоматизованих рішень дозволяє скоротити витрати на підготовку документації, оскільки значна частина процесу виконується програмними засобами. Це значно зменшує потребу в залученні до створення документації великої кількості фахівців, що може бути особливо корисним для стартапів або компаній з обмеженим бюджетом. Автоматизовані системи також спрощують процес архівування й управління документацією, що забезпечує зручний доступ до історичних даних і попередніх версій документів [20].

Попри всі переваги, упровадження автоматизованого підходу до створення технічної документації супроводжується деякими викликами, одним із найсуттєвіших із яких є високі початкові витрати. Для налаштування ефективної системи автоматизації необхідні значні інвестиції у програмне забезпечення, обчислювальні ресурси та навчання персоналу. Невеликі компанії або організації, які лише починають використовувати LLM і онтології, можуть зіткнутися із труднощами в адаптації цих технологій.

Ще одним викликом є складність налаштування та підтримки автоматизованих систем. Успішне впровадження потребує інтеграції з іншими корпоративними системами, як-от системи управління життєвим циклом продукту (далі – PLM), платформи DevOps або бази знань. Некоректне налаштування може призвести до неправильного формулювання або втрати важливої інформації. Процес навчання моделей і налаштування онтологій є критичним етапом, що визначає ефективність усієї системи.

Також варто враховувати обмежену гнучкість автоматизованих систем. Вони добре працюють у рамках заданих параметрів, але можуть не враховувати нюанси, які важливі для специфічних проєктів. Наприклад, складні міждисциплінарні системи або інноваційні технології можуть містити концепції, які важко описати за допомогою загальних онтологій. У такому разі автоматизовані підходи потребують постійного налаштування та коригування, що може звести нанівець переваги автоматизації.

Важливим аспектом є також залежність від технологій і постачальників рішень. Багато сучасних платформ автоматизованого створення документації використовують хмарні сервіси або сторонні рішення, що може створити ризики в разі припинення підтримки якогось програмного забезпечення або зміни політики компаній-постачальників. Упровадження відкритих стандартів і онтологічних моделей може зменшити ці ризики, але цілком їх не усуває.

Отже, автоматизований підхід до створення технічної документації в IT-сфері демонструє значні переваги, зокрема підвищення ефективності, стандартизацію, мінімізацію людських помилок і економію ресурсів. Водночас його впровадження передбачає деякі виклики, до яких належать високі початкові витрати, складність налаштування, обмежена гнучкість і залежність від постачальників рішень. Незважаючи на ці труднощі, правильне поєднання великих мовних моделей з онтологіями, ретельне налаштування системи й інтеграція з іншими інструментами дозволяють значно покращити якість і узгодженість технічної документації, що робить автоматизацію перспективним напрямом розвитку.

Перспективи розвитку та подальших досліджень. Автоматизація формування технічної документації в галузі IT із використанням великих мовних моделей має досить широкі перспективи розвитку. Використання таких моделей, як GPT, BERT тощо, відкриває можливості для покращення якості технічних текстів, підвищення їхньої точності та відповідності загальноприйнятим стандартам. Це сприяє зниженню кількості людських помилок і забезпечує узгодженість у стилі та форматуванні документів [22].

Одним із перспективних напрямів розвитку є адаптація мовних моделей до специфічних доменів. Розроблення спеціалізованих моделей, навчання яких базується на текстах, характерних для окремої галузі, дозволяє створювати документацію з урахуванням відповідної термінології та професійних стандартів. Це особливо важливо для вузькоспеціалізованих напрямів IT, де точність формулювань і узгодженість термінології мають критичне значення [32].

Ще одним важливим вектором розвитку є інтеграція мовних моделей безпосередньо в середовища розробки програмного забезпечення. Це дозволить автоматично генерувати супровідну документацію у процесі написання коду, що сприятиме своєчасному оновленню та відповідності змісту програмного продукту його опису [24].

Водночас необхідні подальші дослідження, спрямовані на оптимізацію мовних моделей для технічних текстів. Зокрема, постає питання адаптації моделей до специфіки роботи з кодом, схемами та технічною термінологією, що дозволить значно підвищити точність і релевантність згенерованих текстів [25].

Окрім цього, важливо розглянути етичні аспекти використання мовних моделей у процесі автоматизації документації. Однією із ключових проблем є можливі упередження в текстах, що можуть вплинути на точність і нейтральність згенерованих матеріалів. Подальші дослідження мають бути спрямовані на розроблення методів мінімізації таких ризиків, забезпечення об’єктивності контенту.

Ще одним актуальним питанням є оцінювання якості згенерованої документації. Розроблення надійних метрик і методів перевірки дозволить об’єктивно вимірювати ефективність використання мовних моделей у цьому контексті. Це сприятиме впровадженню механізмів автоматичної перевірки та покращення текстів у реальному часі.

Значну увагу необхідно також приділити питанням безпеки та конфіденційності. Використання мовних моделей у корпоративному середовищі потребує розроблення методів захисту чутливих даних, що дозволить запобігти витокам інформації та несанкціонованому доступу до корпоративної документації.

Загалом подальші дослідження в цій сфері сприятимуть глибшому розумінню можливостей і обмежень мовних моделей у контексті автоматизації технічної документації, що дозволить створювати ефективніші, безпечніші та більш адаптивні рішення для ІТ-галузі.

Висновки. Автоматизація створення технічної документації за допомогою великих мовних моделей є важливим напрямом розвитку сучасної ІТ-галузі. Висока трудомісткість написання та підтримки документації в актуальному стані, часті помилки, зумовлені людським чинником, а також складнощі в забезпеченні узгодженості між різними елементами проєкту створюють значні виклики для розробників. Використання LLM дозволяє значно оптимізувати робочі процеси, забезпечити швидке створення, оновлення й узгодженість документації з вихідним кодом.

Було проаналізовано традиційні проблеми створення документації, серед яких надмірне навантаження на команди розробників, відсутність єдиних стандартів для різних проєктів і необхідність постійного оновлення інформації відповідно до змін у кодовій базі. Описаний підхід із використанням LLM продемонстрував низку переваг, зокрема: можливість автоматичного формування описів функцій, класів і методів, генерацію текстів з урахуванням специфіки конкретного проєкту й адаптацію документації під заданий стиль оформлення.

Однією з основних переваг використання LLM є її здатність опрацьовувати великі обсяги даних і знаходити закономірності в коді, що дає змогу автоматично створювати детальні й узгоджені технічні описи. Окрім того, LLM можуть інтегруватися із сучасними системами контролю версій і DevOps-процесами, що забезпечує динамічне оновлення документації в разі змін у коді. Ще одним важливим аспектом є багатомовність LLM, що дозволяє автоматично перекладати документацію для міжнародних команд, зменшує потребу в ручному редагуванні та підвищує доступність матеріалів для глобальної аудиторії.

Попри значні переваги, використання мовних моделей для автоматизації документації також супроводжується деякими викликами. Однією з основних проблем є необхідність забезпечення високої точності та відповідності текстів реальним технічним вимогам. LLM можуть генерувати зміст, що містить помилки або логічні невідповідності, тому критично важливо розробити ефективні механізми перевірки якості. Для цього можуть використовуватися комбіновані підходи, що включають автоматизовані алгоритми перевірки й експертний аналіз людиною.

Ще однією проблемою є адаптація LLM до специфічних вимог окремих проєктів і галузей. Наприклад, у сфері фінансових технологій або медичної інформатики необхідно враховувати суворі регуляторні вимоги, що ускладнює застосування загальних мовних моделей без додаткового навчання на спеціалізованих

наборах даних. Використання гібридних підходів, що поєднують LLM з онтологічними моделями та правилами доменної експертизи, може стати ефективним рішенням для таких випадків.

Також важливим аспектом є безпека та конфіденційність даних. Оскільки процес автоматизації документації передбачає використання великих обсягів внутрішньої інформації, постають ризики витоку даних або несанкціонованого доступу. Це потребує розроблення додаткових механізмів захисту, як-от шифрування, контроль доступу та використання приватних моделей LLM, які працюють локально, без передавання даних у хмарні сервіси.

Підсумовуючи результати аналізу, можна зробити висновок, що автоматизація створення документації за допомогою LLM є перспективним напрямом, що дозволяє значно підвищити ефективність роботи розробників, покращити якість документації та зменшити витрати часу на її підтримку. Однак для повноцінного впровадження цієї технології необхідні подальші дослідження у сфері підвищення точності текстів, адаптації моделей до специфічних галузей, інтеграції автоматичних систем документації з наявними інструментами розробки. Подальший розвиток цього напряму відкриває можливості для створення інтелектуальних систем, здатних не лише автоматично формувати й оновлювати документацію, а й забезпечувати її відповідність сучасним стандартам якості та безпеки.

ЛІТЕРАТУРА

1. Mukanova A., Milosz M., Dauletkaliyeva A. LLM-powered natural language text processing for ontology enrichment. *Applied Sciences.* 2024. Vol. 14. № 13. P. 5860–5875. DOI: 10.3390/app14135860.

2. Neuhaus F. Ontologies in the era of large language models – a perspective. *Applied Ontologyю.* 2024. Vol. 18. № 4. P. 399–407. DOI: 10.3233/AO-230072.

3. Hu Y., Liu D., Wang Q. Automating Knowledge Discovery from Scientific Literature via LLMs: A Dual-Agent Approach with Progressive Ontology Prompting. 2024. https://doi.org/10.48550/arXiv.2409.00054.

4. Palagin O., Kaverinsky V., Litvin A. OntoChatGPT Information System: Ontology-Driven Structured Prompts for ChatGPT Meta-Learning. *International Journal of Computing.* 2023. Vol. 22. № 2. P. 170–183. DOI: 10.47839/ijc.22.2.3086.

5. Giglou H.B., D'Souza J., Auer S. LLMs4OL: Large Language Models for Ontology Learning. *The Semantic Web – ISWC.* 2023. P. 408–427. DOI: 10.1007/978-3-031-47240-4_22.

6. Zulkipli Z.Z., Maskat R., Teo N.H.I. A systematic literature review of automatic ontology construction. *Indonesian Journal of Electrical Engineering and Computer Science.* 2022. Vol. 28. № 2. P. 878–889. DOI: 10.11591/ijeecs.v28.i2.pp878-889.

7. Vaswani A., Shazeer N., Parmar N. Attention is all you need. *arXiv.1706.03762.* 2017. https://doi.org/10.48550/arXiv.1706.03762.

8. Devlin J., Chang M.W., Lee K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. https://doi.org/10.48550/arXiv.1810.04805.

9. Brown T.B., Mann B., Ryder N. Language Models are Few-Shot Learners. https://doi.org/10.48550/arXiv.2005.14165.

10. Wulf J., Meierhofer J. Utilizing Large Language Models for Automating Technical Customer Support. https://doi.org/10.48550/arXiv.2406.01407.

11. Cutting-Decelle A.F., Digeon A., Young R.I. Extraction Of Technical Information From Normative Documents Using Automated Methods Based On Ontologies: Application To The Iso 15531 Mandate Standard – Methodology And First Results. 2018. https://doi.org/10.48550/arXiv.1806.02242.

12. Rahman M.M., Finin T. Understanding and representing the semantics of large structured documents. 2018. https://doi.org/10.48550/arXiv.1807.09842.

13. Mateiu P., Groza A. Ontology engineering with Large Language Models. 2023. https://doi.org/10.48550/arXiv.2307.16699.

14. Ciatto G., Agiollo A., Magning M. Large language models as oracles for instantiating ontologies with domain-specific knowledge. *Knowledge-Based Systems*. 2025. Vol. 310. https://doi.org/10.48550/arXiv.2404.04108.

15. Härtel J., Härtel L., Lämmel R. Interconnected Linguistic Architecture. *The Art, Science, and Engineering of Programming.* 2017. Vol. 1. № 1. https://doi.org/10.48550/arXiv.1701.08122.

16. Leiker D., Finnigan S., Gyllen A.R. Prototyping the use of Large Language Models (LLMs) for adult learning content creation at scale. 2023. URL: https://doi.org/10.48550/arXiv.2306.01815.

17. Achachlouei M.A., Patil O., Joshi T. Document Automation Architectures: Updated Survey in Light of Large Language Models. 2023. http://dx.doi.org/10.48550/arXiv.2308.09341.

18. Lazar K., Vetzler M., Uziel G. SpeCrawler: Generating OpenAPI Specifications from API Documentation Using Large Language Models. 2024. https://doi.org/10.48550/arXiv.2402.11625.

19. Jayasuriya D., Perera I. Ontology Based Software Design Documentation for Design Reasoning. 2019. DOI: 10.1109/MERCon.2019.8818813.

20. Топчій Н. Електронний документообіг як основа сучасного документування на підприємстві. *Вчені записки Таврійського національного університету імені В.І. Вернадського.* Серія «Технічні науки». 2021. Vol. 32. № 2. С. 246–249.

21. Автоматизація та IT у 2025 р.: ключові тренди, які змінять бізнес. URL: https://inbase.com.ua/trendy-avtomatyzatsiyi-2025/.

22. Миколайчук Р., Миколайчук А. Використання технологій штучного інтелекту для автоматизації процесу обробки документів. *Сучасні інформаційні технології у сфері безпеки та оборони*. 2024. Vol. 50. № 2. С. 111–117. DOI: 10.33099/2311-7249/2024-50-2-111-117.

23. Kim J.L., Woo K. GLEAN: Generative Learning for Eliminating Adversarial Noise. 2024. https://doi.org/10.48550/arXiv.2409.10578.

24. Han B., Wang X., Wang Y. New Interaction Paradigm for Complex EDA Software Leveraging GPT. 2023. https://doi.org/10.48550/arXiv.2307.14740.

25. Sachidananda V., Kessler J.S., Lai Y. Efficient Domain Adaptation of Language Models via Adaptive Tokenization. 2021. https://doi.org/10.48550/arXiv.2109.07460.

26. Bhatia M.P., Kumar A., Beniwal R. Ontology Driven Software Development for Automated Documentation. *Webology*. 2018. Vol. 15. № 2.

27. Makin A. Ontology-Driven Knowledge Management Systems Enhanced by Large Language Models.2024. http://dx.doi.org/10.13140/RG.2.2.32979.18728.

28. Luo Q., Ye Y., Liang S. Repo Agent: An LLM-Powered Open-Source Framework for Repository-level Code Documentation Generation. 2024. https://doi.org/10.48550/arXiv.2402.16667.

29. Yang J., Wittern E., Ying A.T. Automatically Extracting Web API Specifications from HTML Documentation. 2018. https://doi.org/10.48550/arXiv.1801.08928.

30. Wang S., Tang Y., He D. gDoc: Automatic Generation of Structured API Documentation. 2018. https://doi.org/10.48550/arXiv.2303.13041.

31. Toro S., Anagnostopoulos A.V., Bello S.M. Dynamic Retrieval Augmented Generation of Ontologies using Artifcial Intelligence (DRAGON-AI). *Journal of Biomedical Semantics.* 2024. Vol. 15. № 19. DOI: 10.1186/s13326-024-00320-3.

32. Gretter R., Matassoni M., Falavigna D. Seed Words Based Data Selection for Language Model Adaptation. 2021. https://doi.org/10.48550/arXiv.2107.09433.

REFERENCES

1. Mukanova, A., Milosz, M., & Dauletkaliyeva, A. (2024). LLM-powered natural language text processing for ontology enrichment. *Applied Sciences.* Vol. 14. № 13. P. 5860–5875. DOI: 10.3390/app14135860.

2. Neuhaus, F. (2024). Ontologies in the era of large language models – a perspective. *Applied Ontologyю.* Vol. 18. № 4. P. 399–407. DOI: 10.3233/AO-230072.

3. Hu, Y., Liu, D., & Wang, Q. (2024). Automating Knowledge Discovery from Scientific Literature via LLMs: A Dual-Agent Approach with Progressive Ontology Prompting. https://doi.org/10.48550/arXiv.2409.00054.

4. Palagin, O., Kaverinsky, V., & Litvin, A. (2023). OntoChatGPT Information System: Ontology–Driven Structured Prompts for ChatGPT Meta–Learning. *International Journal of Computing.* Vol. 22. № 2. P. 170–183. DOI: 10.47839/ijc.22.2.3086.

5. Giglou, H.B., D'Souza, J., & Auer, S. (2023). LLMs4OL: Large Language Models for Ontology Learning. *The Semantic Web – ISWC.* P. 408–427. DOI: 10.1007/978-3-031-47240-4_22.

6. Zulkipli, Z.Z., Maskat, R., Teo, N.H.I. A systematic literature review of automatic ontology construction. *Indonesian Journal of Electrical Engineering and Computer Science.* 2022. Vol. 28. № 2. P. 878–889. DOI: 10.11591/ijeecs.v28.i2.pp878-889.

7. Vaswani, A., Shazeer, N., & Parmar, N. (2017). Attention Is All You Need. arXiv.1706.03762. https://doi.org/10.48550/arXiv.1706.03762.

8. Devlin, J., Chang, M.W., & Lee, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. https://doi.org/10.48550/arXiv.1810.04805.

9. Brown, T.B., Mann, B., & Ryder, N. Language Models are Few-Shot Learners. https://doi.org/10.48550/arXiv.2005.14165.

10. Wulf, J., & Meierhofer, J. Utilizing Large Language Models for Automating Technical Customer Support. https://doi.org/10.48550/arXiv.2406.01407.

11. Cutting-Decelle, A.F., Digeon, A., & Young, R.I. (2018). Extraction Of Technical Information From Normative Documents Using Automated Methods Based On Ontologies: Application To The Iso 15531 Mandate Standard – Methodology And First Results. https://doi.org/10.48550/arXiv.1806.02242.

12. Rahman, M.M., & Finin, T. (2018). Understanding and representing the semantics of large structured documents. https://doi.org/10.48550/arXiv.1807.09842.

13. Mateiu, P., & Groza, A. (2023). Ontology engineering with Large Language Models. https://doi.org/10.48550/arXiv.2307.16699.

14. Ciatto, G., Agiollo, A., & Magning M. (2025). Large language models as oracles for instantiating ontologies with domain-specific knowledge. *Knowledge-Based Systems*. Vol. 310. https://doi.org/10.48550/arXiv.2404.04108.

15. Härtel, J., Härtel, L., & Lämmel, R. (2017). Interconnected Linguistic Architecture. The Art, Science, and Engineering of Programming. Vol. 1. № 1. https://doi.org/10.48550/arXiv.1701.08122.

16. Leiker, D., Finnigan, S., & Gyllen, A.R. (2023). Prototyping the use of Large Language Models (LLMs) for adult learning content creation at scale. https://doi.org/10.48550/arXiv.2306.01815.

17. Achachlouei, M.A., Patil, O., & Joshi, T. (2023). Document Automation Architectures: Updated Survey in Light of Large Language Models. http://dx.doi.org/10.48550/arXiv.2308.09341.

18. Lazar. K., Vetzler. M., & Uziel. G. (2024). SpeCrawler: Generating OpenAPI Specifications from API Documentation Using Large Language Models. https://doi.org/10.48550/arXiv.2402.11625.

19. Jayasuriya, D., & Perera, I. (2019). Ontology Based Software Design Documentation for Design Reasoning. DOI: 10.1109/MERCon.2019.8818813.

20. Topchii, N. (2021). Elektronnyi dokumentoobih iak osnova suchasnoho dokumentuvannia na pidpryiemstvi [Electronic document management as the basis of modern documentation in an enterprise]. *Scientific Notes of V.I. Vernadsky Taurida National University. Series "Technical Sciences".* Vol. 32, № 2, P. 246–249.

21. Automation and IT in 2025: Key Trends That Will Transform Business. Website. Retrieved from https://inbase.com.ua/trendy-avtomatyzatsiyi-2025/.

22. Mykolaichuk, R., & Mykolaichuk, A. (2024). Vykorystannia tekhnolohii shtuchnoho intelektu dlia avtomatyzatsii protsesu obrobky dokumentiv. Suchasni informatsiini tekhnolihii u sferi bezpeky ta oborony [*Application of artificial intelligence technologies for automating the document processing workflow. Modern Information Technologies in Security and Defense*]. Vol. 50, 2, P. 111–117. DOI: 10.33099/2311-7249/2024-50-2-111-117.

23. Mykolaichuk, R., & Mykolaichuk, A. (2024). Vykorystannia tekhnolohii shtuchnoho intelektu dlia avtomatyzatsii protsesu obrobky dokumentiv [Use of Artificial Intelligence Technologies for Automating the Document Processing].

Suchasni informatsiini tekhnolohii u sferi bezpeky ta oborony [*Modern Information Technologies in the Field of Security and Defense*], 50 (2), 111–117. https://doi.org/10.33099/2311-7249/2024-50-2-111-117 [in Ukrainian].

24. Kim, J.L., & Woo, K. (2024). GLEAN: Generative Learning for Eliminating Adversarial Noise. https://doi.org/10.48550/arXiv.2409.10578.

25. Han, B., Wang, X., & Wang, Y. (2023). New Interaction Paradigm for Complex EDA Software Leveraging GPT. https://doi.org/10.48550/arXiv.2307.14740.

26. Sachidananda, V., Kessler, J.S., & Lai, Y. (2021). Efficient Domain Adaptation of Language Models via Adaptive Tokenization. https://doi.org/10.48550/arXiv.2109.07460.

27. Bhatia, M.P., Kumar, A., & Beniwal, R. (2018). Ontology Driven Software Development for Automated Documentation. *Webology*. Vol. 15. № 2.

28. Makin, A. (2024). Ontology-Driven Knowledge Management Systems Enhanced by Large Language Models. http://dx.doi.org/10.13140/RG.2.2.32979.18728.

29. Luo, Q., Ye, Y., & Liang, S. (2024). Repo Agent: An LLM-Powered Open-Source Framework for Repository-level Code Documentation Generation. https://doi.org/10.48550/arXiv.2402.16667.

30. Yang, J., Wittern, E., & Ying, A.T. (2018). Automatically Extracting Web API Specifications from HTML Documentation. https://doi.org/10.48550/arXiv.1801.08928.

31. Wang, S., Tang, Y., He, D. (2018). gDoc: Automatic Generation of Structured API Documentation. https://doi.org/10.48550/arXiv.2303.13041.

32. Toro, S., Anagnostopoulos, A.V., & Bello, S.M. (2024). Dynamic Retrieval Augmented Generation of Ontologies using Artifcial Intelligence (DRAGON-AI). *Journal of Biomedical Semantics.* Vol. 15. № 19. DOI: 10.1186/s13326-024-00320-3.

33. Gretter, R., Matassoni, M., & Falavigna, D. (2021). Seed Words Based Data Selection for Language Model Adaptation. https://doi.org/10.48550/arXiv.2107.09433.