https://doi.org/10.31891/2219-9365-2025-84-49
УДК 681.3; 004.8:004.85

СТАРЖИНСЬКИЙ Валерій
Вінницький національний технічний університет
https://orcid.org/0009-0009-3827-0122
e-mail: 3372292@gmail.com
БІСІКАЛО Олег
Вінницький національний технічний університет
https://orcid.org/0000-0002-7607-1943
e-mail: obisikalo@vntu.edu.ua

ВИКОРИСТАННЯ ЛОКАЛЬНИХ LLM-МОДЕЛЕЙ ДЛЯ СТАНДАРТИЗАЦІЇ ТА БАГАТОМОВНОГО ПЕРЕКЛАДУ ТЕХНІЧНИХ НАЗВ ВИРОБІВ

У роботі розглянуто підхід до автоматизованої стандартизації технічних назв метизів із використанням локальних великих мовних моделей (LLM), що дозволяє суттєво підвищити точність обробки спеціалізованих даних. Докладно описано архітектуру взаємодії Python-скриптів із локальним сервером Lm studio, включаючи організацію API-запитів, структуру вхідних і вихідних даних у форматі JSON, а також механізми збереження результатів у базі даних sqlite (items.db). У роботі наведено приклади формування запитів (prompt), що містять не точні найменування виробів, очікуваний формат стандарту та розмір виробу, а також приклади перевірки правильності форматів після генерації результатів. Показано, як реалізується обробка серії записів із бази даних, контроль якості отриманих результатів та відображення інформації про помилки, що дозволяє оперативно коригувати як промпти, так і навчальні дані моделі.

Обґрунтовано доцільність переходу на локальні рішення для обробки технічних даних замість використання хмарних сервісів. Такий підхід дозволяє значно знизити фінансові витрати на інфраструктуру, підвищити безпеку та конфіденційність даних, а також забезпечити стабільний контроль над робочими процесами обробки інформації. Показано, що локальна інференція великих мовних моделей може ефективно вирішувати завдання стандартизації у прикладних системах, забезпечуючи високу точність розпізнавання шаблонів, коректне дотримання міжнародних стандартів (DIN, ISO, ДСТУ) та можливість інтеграції у існуючі робочі процеси підприємств без залучення сторонніх сервісів. Крім того, робота демонструє практичну реалізацію багатомовного перекладу технічних назв, що підвищує універсальність рішення та полегшує взаємодію з міжнародними базами даних і документацією.

Ключові слова: мовна модель, локальна LLM, стандартизація назв, метизи, стандартизація.

STARZHYNSKYI Valerii, BISIKALO Oleh
Vinnytsia National Technical University

USING LOCAL LLM MODELS FOR STANDARDIZATION AND MULTILINGUAL TRANSLATION OF TECHNICAL PRODUCT NAMES

The paper considers an approach to automated standardization of technical names of hardware using local large language models (LLM), which allows to significantly increase the accuracy of processing specialized data. The architecture of interaction of python scripts with the local Lm studio server is described in detail, including the organization of api requests, the structure of input and output data in JSON format, as well as the mechanisms for saving results in the sqlite database (items.db). The paper provides examples of generating requests (prompts) containing raw product names, the expected standard format and product size, as well as examples of checking the correctness of formats after generating results. It shows how the processing of a series of records from the database is implemented, the quality control of the results obtained and the display of information about errors are implemented, which allows to promptly correct both prompts and training data of the model.

The feasibility of switching to local solutions for processing technical data instead of using cloud services is substantiated. This approach allows to significantly reduce financial costs for infrastructure, to increase data security and confidentiality, and to ensure stable control over information processing workflows. It is shown that local inference of large language models can effectively solve standardization problems in application systems, ensuring high accuracy of pattern recognition, correct compliance with international standards (DIN, ISO, DSTU) and the possibility of integration into existing enterprise workflows without involving third-party services. In addition, the work demonstrates the practical implementation of multilingual translation of technical names, which increases the versatility of the solution and facilitates interaction with international databases and documentation.

Keywords: language model, local LLM, name standardization, hardware, standardization.

Стаття надійшла до редакції / Received 22.09.2025
Прийнята до друку / Accepted 20.11.2025

ПОСТАНОВКА ПРОБЛЕМИ У ЗАГАЛЬНОМУ ВИГЛЯДІ ТА ЇЇ ЗВ'ЯЗОК ІЗ ВАЖЛИВИМИ НАУКОВИМИ ЧИ ПРАКТИЧНИМИ ЗАВДАННЯМИ

У сучасних умовах розвитку промислових інформаційних систем (у тому числі ERP, CAD/PDM та інших), виникає гостра потреба у уніфікації технічних назв виробів. Зокрема, завдання стандартизації метизів – таких як болти, гайки, гвинти тощо – доповнюється необхідністю їхнього багатомовного перекладу для забезпечення цілісності даних з метою інтеграції в міжнародні ланцюги постачання та автоматизації поповнення каталогів. При цьому традиційні NMT (нейронні системи перекладу), хоч і показують добрі результати, обмежені специфікою технічної лексики та, зазвичай, «глухі» до форматних стандартів, як-от DIN, ISO чи ДСТУ [1].

Натомість, останні досягнення у сфері великих мовних моделей (LLM) відкривають нові можливості: вони здатні одночасно виконувати стандартизацію формату та переклад, використовуючи один і той самий універсальний нейронний інструмент. Наприклад, у контексті термінологічної узгодженості, було запропоновано підхід LLM-BT, що застосовує механізм back-translation для автоматичної верифікації та стабілізації термінів [2]. Додатково, для локальних систем, що не залежать від хмарних API, стає актуальним використання локальних LLM-серверів, що забезпечують безпеку, економію ресурсів та автономність у рамках підприємств.

У зв'язку з цим, у нашій роботі розглядається архітектура автономної системи на основі Python-скриптів, локального LM Studio-сервера та SQLITE бази даних, що забезпечує повний цикл: стандартизація формату → багатомовний переклад → збереження → валідація. Обґрунтовується та досліджується підхід, що поєднує prompt-інженеринг (формування підказок моделі), логіку валідації форматів (DIN/ISO → розмір → переклад) та метрики якості результатів, що дозволяють кількісно оцінити коректність вихідних назв.

АНАЛІЗ ДОСЛІДЖЕНЬ ТА ПУБЛІКАЦІЙ

У публікації “Використання LLM для вилучення та нормалізації значень атрибутів продукту” [3] досліджується здатність моделей GPT-3.5 та GPT-4 витягувати значення атрибутів товарів із торгових пропозицій і нормалізувати їх для використання в застосунках електронної комерції, зокрема у фасетній фільтрації. Для оцінки застосовано набір даних WDC PAVE, що містить вручну перевірені еталонні значення для обох завдань: простого витягування та витягування з подальшою нормалізацією. Автори проводять експерименти з різними варіантами побудови підказок, використовуючи приклади значень і демонстрації, відібрані з навчального набору.

Робота “Використання LLM для вилучення та нормалізації значень атрибутів продукту”[4] ілюструє, що зворотний переклад на основі LLM є не лише життєздатною методологією стандартизації термінології, але й узагальненою та інтерпретованою семантичною основою для міждоменної мовної інфраструктури в епоху генеративного штучного інтелекту. LLM-BT (метод перевірки та стабілізації термінології: модель перекладає назву → зворотний переклад → порівнюється консистентність термінів) перетворює зворотний переклад зі статичного інструменту оцінки на інтерпретований, динамічний та семантично насичений механізм для багато-мовного моделювання термінів. Замість вбудовування тексту в непрозорі вектори, він створює зрозумілі для людини, оборотні та структурно простежувані результати, що робить його переконливим розширенням теорії вбудовування.

Дослідження “NormTab: покращення символічного мислення в LLM за допомогою нормалізації табличних даних” [5] представляє NormTab, фреймворк, спрямований на покращення продуктивності LLM для табличних даних шляхом нормалізації веб-таблиць. Використовуючи текстове розуміння LLM для очищення та нормалізації даних, NormTab покращує логічне мислення в таблицях. Експерименти на складних наборах даних демонструють його ефективність. Робота присвячена вдосконаленню методів для LLM для обробки табличних даних, підкреслюючи важливість вирішення проблем веб-таблиць для покращення продуктивності.

ФОРМУЛЮВАННЯ ЦІЛЕЙ СТАТТІ

У сучасних ERP-, e-commerce системах та базах даних критичною є єдність та коректність технічних назв виробів, оскільки вони безпосередньо впливають на якість пошуку, фільтрації та інтеграції даних із різних джерел. Проблема ускладнюється багатомовністю середовища та наявністю різних стандартів позначень, що призводить до неоднозначностей, дублювання та ускладнює подальшу обробку інформації. Традиційні алгоритми нормалізації та перекладу часто виявляються недостатньо гнучкими, особливо при роботі з технічними термінами, які можуть мати різні скорочення або національні варіації. Локальне розгортання великих мовних моделей (LLM) відкриває нові можливості для автоматизації цих процесів без передачі конфіденційних даних на зовнішні сервери. Це особливо актуально для підприємств, що працюють із закритими технічними базами даних або з обмеженнями на використання хмарних сервісів.

Потребують дослідження аспекти практичного тестування та донавчання великої мовної моделі “mistral-7b-instruct” для задачі стандартизації та багатомовного перекладу технічних назв виробів. У межах дослідження на першому етапі доцільно провести запуск локального серверного середовища Lm Studio, що забезпечує інтерфейс взаємодії з моделлю через API-запити. Наступним кроком має бути донавчання моделі на спеціально підготовленому корпусі даних, що містить приклади правильного форматування та перекладу технічних найменувань. Для оцінки ефективності підходу варто організувати обмін вхідними даними (JSON-запити із назвами та шаблонами стандартизації) та вихідними результатами (згенеровані переклади у визначеній структурі). Такий експеримент дозволить дослідити, наскільки ефективно локальна LLM може інтегруватися в робочі процеси обробки технічної інформації та які саме помилки виникають при генерації.

Метою роботи є розробка інтегрованої методики обробки технічних назв – метизів, яка базується на локальних великих мовних моделях, забезпечує форматну стандартизацію (відповідність структурам

стандартів) та якісний багатомовний переклад, а також включає механізм валідації результатів із можливістю вимірювання точності та відсотка помилок.

ПРИНЦИПИ ТА РЕЗУЛЬТАТИ ДОНАВЧАННЯ LLM НА ДОМЕННИХ ДАНИХ

Навчання великих мовних моделей (LLM) є ключовим етапом для забезпечення їхньої здатності якісно вирішувати специфічні завдання в обмежених професійних областях, таких як специфічна технічна термінологія й стандарти. Загальні моделі, натреновані на відкритих великих корпусах, демонструють хорошу універсальність, але часто неадекватно обробляють вузькоспеціалізовані терміни та форматні вимоги. Для вирішення цієї проблеми широко застосовують три основні підходи: Retrieval-Augmented Generation (RAG) – для забезпечення доступу до спеціалізованих знань у режимі реального часу; Parameter-Efficient Fine-Tuning (PEFT), наприклад, LoRA – що дозволяє адаптувати модель без повного донавчання; та класичне fine-tuning — прицільне донавчання на доменних даних (James Tang, Medium-публікація, 2025) [6-7].

Повне перенавчання потребує масштабних обчислювальних ресурсів і великих обсягів даних, що робить його недоцільним для задач із відносно вузьким корпусом. Підхід RAG дозволяє збагачувати відповіді зовнішніми базами знань, однак він більше орієнтований на динамічний пошук і не гарантує дотримання чіткого шаблону форматування. У свою чергу, методи PEFT, зокрема supervised fine-tuning та LoRA, забезпечують можливість навчання моделі на невеликих доменних наборах даних, зберігаючи при цьому узагальнювальні властивості базової LLM. У проведених в даному дослідженні експериментах було застосовано метод PEFT (з використанням LoRA) для адаптації моделі “mistral-7b-instruct-v0.2” на корпусі технічних назв метизів, які включали шаблони із стандартами (DIN, ISO, ГОСТ), розмірами та перекладом. Було зібрано понад сто prompt-completion прикладів, де prompt містить «сире» (зазвичай не зовсім точне) найменування, а completion – стандартизовану назву з перекладом. Це дало змогу прискорити донавчання, що займало лише кілька годин на типових GPU, при цьому забезпечено помітне зниження кількості помилок у форматуванні назв (у порівнянні з базовою моделлю), без значущої втрати загальних мовних здібностей.

Принцип донавчання (тренування) моделі “mistral-7b-instruct-v0.2” відбувався через метод стандартизації назви. Основне завдання полягало у точному форматуванні назви згідно вказаного шаблону. Як і планувалося у постановці задачі дослідження, обмін вхідними та вихідними даними з моделлю було реалізовано у форматі JSON-запитів. Зокрема модель отримує в якості вхідних даних JSON з об’єктів, при цьому кожен об’єкт містить ключі: itm_id(унікальний код запису), onms_value(шаблон формату назви), itm_name (назва метиза). У таблиці 1 наведено завдання та відповіді на завдання до та після навчання. Завдання та відповіді подано у форматі об’єктів JSON.

Таблиця 1

Вхідні та вихідні дані обміну з моделлю

Завдання	Відповідь до навчання	Відповідь після навчання
[{ "itm_id": 401001, "onms_value": "[Назва виробу] [Стандарт] [Тип][Діаметр]-[Довжина] [Матеріал/Покриття]", "itm_name": "M12x50 гвинт DIN 933 оцинкований" }]	[{ "itm_id": 401001, "onms_value": "[Назва виробу] [Стандарт] [Тип][Діаметр-Довжина] [Матеріал/Покриття]", "itm_name": "гвинт M12x50 DIN 933 оцинкований" }]	[{ "itm_id": 401001, "onms_value": "[Назва виробу] [Стандарт] [Тип][Діаметр-Довжина] [Матеріал/Покриття]", "itm_name": "гвинт DIN 933 M12-50 оцинкований" }]
[{ "itm_id": "94678", "itm_name": "Болт M14x60 ГОСТ 7798-70 фосфатований", "onms_value": "[Назва виробу] [Стандарт] [Тип різьби][Діаметр]x[Довжина] [Матеріал/Покриття] [Шліц]" }, { "itm_id": "94679", "itm_name": "Болт M6x16 DIN 933 A2 з хрестовим шліцом", "onms_value": "[Назва виробу] [Стандарт] [Тип різьби][Діаметр]x[Довжина] [Матеріал/Покриття] [Шліц]"}, { "itm_id": "94680", "itm_name": "Болт M8x30 з фланцем оцинкований ISO 4162", "onms_value": "[Назва Виробу] [Стандарт] [Тип різьби][Діаметр]X[Довжина] [Матеріал/Покриття] [Шліц]" }]	[{ "itm_id": "94678", "itm_name": "Болт M14x60 ГОСТ 7798-70 фосфатований", "onms_value": "[Назва виробу] [Стандарт] [Тип різьби][Діаметр]x[Довжина] [Матеріал/Покриття] [Шліц]"}, { "itm_id": "94679", "itm_name": "Болт M6x16 DIN 933 A2 з хрестовим шліцом", "onms_value": "[Назва виробу] [Стандарт] [Тип різьби][Діаметр]x[Довжина] [Матеріал/Покриття] [Шліц]"}, { "itm_id": "94680", "itm_name": "Болт M8x30 з фланцем оцинкований ISO 4162", "onms_value": "[Назва Виробу] [Стандарт] [Тип різьби][Діаметр]X[Довжина] [Матеріал/Покриття] [Шліц]" }]	[{ "itm_id": "94678", "itm_name": "болт ГОСТ 7798-70 M14x60 фосфатований", "onms_value": "[Назва виробу] [Стандарт] [Тип різьби][Діаметр]x[Довжина] [Матеріал/Покриття] [Шліц]" }, { "itm_id": "94679", "itm_name": "Болт DIN 933 M6x16 A2 хрестовий", "onms_value": "[Назва виробу] [Стандарт] [Тип різьби][Діаметр]x[Довжина] [Матеріал/Покриття] [Шліц]"}, { "itm_id": "94680", "itm_name": "болт ISO 4162 M8x30 оцинкований з фланцем", "onms_value": "[Назва Виробу] [Стандарт] [Тип різьби][діаметр]x[довжина] [матеріал/покриття] [шліц]" }]

Після проведення навчання та тренування моделі було виконано аналіз отриманих результатів, Зразки вхідних та вихідних даних та їх відповідність очікуваним форматам наведено у Таблиці 2.

Таблиця 2

Аналіз отриманих результатів

itm_id	Очікуване форматування	Відповідь до навчання	Відповідь після навчання	Відхилення до навчання	Відхилення після навчання
401001	гвинт DIN 933 M12x50 оцинкований	гвинт M12x50 DIN 933 оцинкований	Гвинт DIN 933 M12-50 оцинкований	Порушено порядок: діаметр раніше за стандарт	OK (змінено порядок відповідно до шаблону)
94678	Болт ГОСТ 7798-70 M14x60 фосфатований	болт M14x60 ГОСТ 7798-70 фосфатований	Болт ГОСТ 7798-70 M14x60 фосфатований	Порядок неправильний (стандарт після розміру)	OK (порядок виправлено)
94649	Болт DIN 933 M6x16 A2 хрестовий	болт M6x16 DIN 933 A2 з хрестовим шліцом	Болт DIN 933 M6x16 A2 хрестовий	Стандарт після розміру, зайве «з», шліц наприкінці	OK (структура шаблону збережена)
94680	Болт ISO 4162 M8x30 оцинкований з фланцем	болт M8x30 з фланцем оцинкований ISO 4162	Болт ISO 4162 M8x30 оцинкований з фланцем	Стандарт наприкінці, розрив послідовності шаблону	OK (усі елементи в правильному порядку)

Реалізація поставленої задачі

За допомогою мови програмування Python було сформовано базу даних з однієї таблиці. Спеціальний запит вибирає записи з бази, формує для кожного prompt (завдання) і відправляє POST-запит на локальний сервер LM Studio з моделлю. Сервер запускає модель, яка обробляє дані і повертає JSON із результатом. Python розбирає JSON, отримує стандартизовані назви та переклади, і оновлює ці значення у таблиці бази.

Для візуалізації архітектури рішення було створено Component / Deployment Diagram (схему підключення), яка відображає підключення локального середовища LM Studio, Python-клієнта та локального сервера моделі. Представлена на рис. 1. Component / Deployment diagram демонструє взаємозв'язок компонентів, потоки даних між ними та місця розгортання програмного забезпечення на локальному комп'ютері.

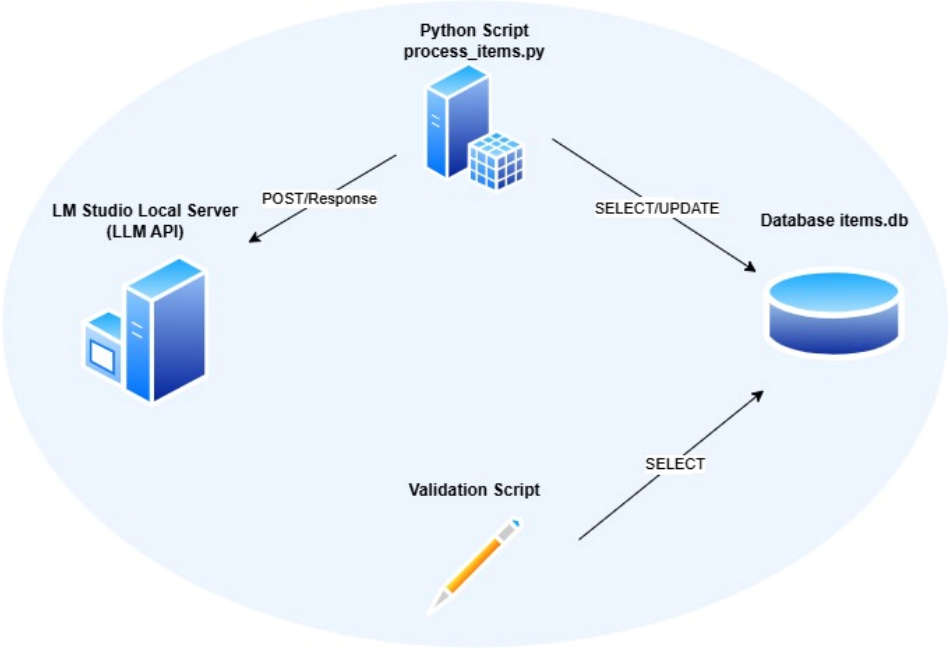

Рис. 1. Component / Deployment Diagram (Схема підключення)

Дана діаграма розкриває архітектуру системи та її ключові компоненти, зокрема:

– Python Script (process_items.py) – головний скрипт, який запускає обробку записів, формує запити для моделі та зберігає результати у базі даних.

– Database (items.db) – локальна база даних, що зберігає записи метизів з полями raw_name, ukr_name, eng_name. Скрипт Python звертається до неї через SQL-запити (SELECT, UPDATE).

– LLM Local Server (LM Studio) – локальний сервер, який виконує модель mistral-7b-instruct-v0.2. Приймає JSON-запити через HTTP POST і повертає JSON-відповідь з полями ukr_name та eng_name.

– Validation Script (validate_items.py) – додатковий скрипт, що перевіряє правильність форматування та складає статистику некоректних записів.

З метою упорядкування обміну даними між користувачем та моделлю було створено Sequence Diagram (Обмін даними з моделлю), що відображає послідовність запитів та відповідей у форматі JSON. Діаграма Sequence Diagram (рис. 2) демонструє, як Python-клієнт надсилає вхідні дані на локальний сервер моделі, як обробляється запит і як формується відповідь із стандартизованими назвами та перекладом.

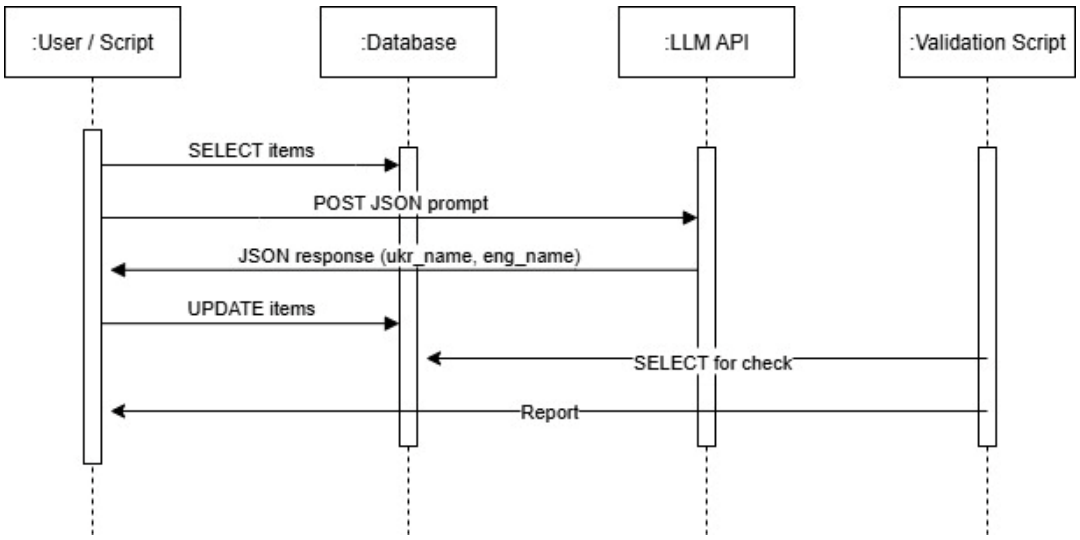

Рис. 2. Sequence Diagram (Обмін даними з моделлю)

Для візуалізації послідовності дій та робочого процесу обробки даних було створено Activity Diagram (Workflow). Представлена на рис. 3 Activity Diagram (Workflow) ілюструє ключові етапи обробки технічних назв: надсилання вхідних даних до моделі, отримання стандартизованих результатів та їх запису у базу даних.

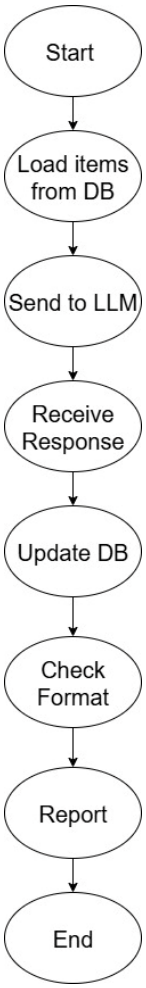

Рис. 3. Activity Diagram (Workflow)

Значна роль у проведеному дослідженні полягала у розроблені необхідних для донавчання моделі promt-ів. Основна задача promt – максимально роз'яснити моделі mistral-7b-instruct-v0.2 інструкцію по стандартизації назв метизів. Приклад promt №1 наведено на рис.4.

```
prompt = (
    f"Ти – модель для стандартизації назв метизів. Задача стандартизувати назву метиза і зробити переклад її на 2 мови. "
    f"Вхід(приклад): 'Болт оцинкований M10x120 DIN 933'. "
    f"Вихід(приклад): ukr_name: 'Болт DIN 933 M10x120 оцинкований'; eng_name: 'Bolt DIN 933 M10x120 galvanized'. "
    f"Тепер стандартизуй: '{raw_name}' у форматі '<назва виробу> <стандарт> <розмір> <додаткові параметри при наявності>'. "
    f"Відповідь у JSON без додаткових полів, одразу ukr_name, eng_name."
)
```

Рис. 4. Приклад promt №1

Реалізовано перевірку точності за допомогою розбиття оновленої назви українською мовою на секції, що в результаті є форматом назви та дозволяє порівняти оновлену назву з шаблоном. Формат назви має наступний вид: <назва> <стандарт> <номер стандарту> <розмір> [опції]. Код перевірки точності за допомогою скрипта реалізовано на мові Python та зображено на рис. 5.

```
pattern = re.compile(
    r"^(болт|гайка|шайба|цвях|шуруп|саморіз|гвинт|винт|шпилька)\s+"  # назва
    r"(DIN|ГОСТ|ISO)\s*\d+"                                          # стандарт
    r"(?:\s+[A-ZA-Я]?\d+(?:x\d+)*)*"                                 # розмір/параметри (гнучкі)
    r"(.*)?$",                                                       # решта
    re.IGNORECASE
)
```

Рис. 5. Умова перевірки точності стандартизованої назви

На основі prompt №1 було проведено тестування моделі з використанням 100 записів, які були занесені до бази даних items.db. Вибірку з 10 записів для ілюстрації наведено у Таблиці 3.

Таблиця 3

Виконання prompt №1

Назва метиза	Оновлений запис ukr_name	Оновлений запис eng_name	Час обробки запису
Болт M6x20 DIN 933	Болт DIN 933 M6x20 оцинкований	DIN 933 M6x20 Bolt, Oiled	9.70 секунд
Гайка M10 DIN 934	Гайка M10 DIN 934	M10 DIN 934 Nut	7.83 секунд
Болт M12x69 DIN 444	Болт M12x69 DIN 444	M12x69 bolt DIN 444	9.94 секунд
Гайка 8 DIN 934	Гайка DIN 934 8	nut DIN 934 8	10.21 секунд
Шайба M6 DIN 125	Шайба M6 DIN 125	M6 washer DIN 125	8.57 секунд
Цвях 50 ГОСТ 4028	Цвях 50 ГОСТ 4028	Nail 50 GOST 4028	11.32 секунд
Шпилька M20x100 DIN 976	Шпилька M20x100 DIN 976	M20x100 stud DIN 976	12.08 секунд
Винт M5x16 ISO 7380	Винт M5x16 ISO 7380	M5x16 screw ISO 7380	7.56 секунд
Саморіз 4.2x16 ГОСТ 1145	Саморіз ГОСТ 1145 4.2x16	screw GOST 1145 4.2x16	13.44 секунд
Шайба M8 DIN 125	Шайба M8 DIN 125	M8 washer DIN 125	8.51 секунд

Загальний час обробки 100 записів склав 998.46 секунд (~17 хвилин 4 секунди). Результат перевірки коректності результатів через promt №1 при загальній кількісті 100 записів становить 67 (67%) некоректних записів.

У таблиці 4 наведено порівняння результатів з шаблоном виконаних моделлю з використанням promt №1. Шаблон: <назва> <стандарт> <номер стандарту> <розмір> [опції].

Таблиця 4

Порівняння результатів з використанням prompt №1

№	Запис ukr_name / eng_name	Примітки
1	**2**	**3**
1	Болт DIN 933 M6x20 оцинкований DIN 933 M6x20 Bolt, Oiled	Укр: відповідає шаблону Eng: не відповідає шаблону (правильно: Bolt DIN 933 M6x20 Oiled)
2	Гайка M10 DIN 934 M10 DIN 934 Nut	Укр: не відповідає шаблону (правильно: Гайка DIN 934 M10) Eng: не відповідає шаблону (правильно: Nut DIN 934 M10)
3	Болт M12x69 DIN 444 M12x69 bolt DIN 444	Укр: не відповідає шаблону (правильно: Болт DIN 444 M12x69) Eng: не відповідає шаблону (правильно: bolt DIN 444 M12x69)
4	Гайка DIN 934 8 nut DIN 934 8	Укр: відповідає шаблону Eng: відповідає шаблону
5	Шайба M6 DIN 125 M6 washer DIN 125	Укр: не відповідає шаблону (правильно: Шайба DIN 125 M6) Eng: не відповідає шаблону (правильно: washer DIN 125 M6)
6	Цвях 50 ГОСТ 4028 Nail 50 GOST 4028	Укр: не відповідає шаблону (правильно: Цвях ГОСТ 4028 50) Eng: не відповідає шаблону (правильно: Nail GOST 4028 50)

1	2	3
7	Шпилька M20x100 DIN 976 M20x100 stud DIN 976	Укр: не відповідає шаблону (правильно: Шпилька DIN 976 M20x100) Eng: не відповідає шаблону (правильно: stud DIN 976 M20x100)
8	Винт M5x16 ISO 7380 M5x16 screw ISO 7380	Укр: не відповідає шаблону (правильно: Винт ISO 7380 M5x16) Eng: не відповідає шаблону (правильно: screw ISO 7380 M5x16)
9	Саморіз ГОСТ 1145 4.2x16 screw GOST 1145 4.2x16	Укр: відповідає шаблону Eng: відповідає шаблону
10	Шайба M8 DIN 125 M8 washer DIN 125	Укр: не відповідає шаблону (правильно: Шайба DIN 125 M8) Eng: не відповідає шаблону (правильно: washer DIN 125)

Prompt №1 було доопрацьовано для підвищення точності стандартизації та перекладу, після чого проведено повторне тестування моделі, результатом якого став prompt №2. Приклад promt №2 наведено на рис.6.

```
prompt = (
    f"Ти — модель для стандартизації назв метизів. Задача стандартизувати назву метиза і зробити переклад її на 2 мови. "
    f"Вхід(приклад): 'Болт оцинкований M10x120 DIN 933'. "
    f"Вихід(приклад): ukr_name: 'Болт DIN 933 M10x120 оцинкований'; eng_name: 'Bolt DIN 933 M10x120 galvanized'. "
    f"В конкретному прикладі Болт - Назва виробу(може бути Гайка, Шайба тощо), DIN 933 - Стандарт(може бути ГОСТ та ISO), M10x120 - розмір виробу, оцинкований - інші додатковий параметр"
    f"Тепер стандартизуй: '{raw_name}' у форматі '<назва виробу> <стандарт> <розмір> <додаткові параметри при наявності>'. "
    f"Забезпеч, щоб переклади були точними: "
    f"- ukr_name українською, "
    f"- eng_name англійською. "
    f"Відповідь у JSON без додаткових полів, одразу ukr_name, eng_name."
)
```

Рис. 6. Приклад promt №2

На основі prompt №2 було проведено тестування моделі з використанням 100 записів, які були занесені до бази даних items.db. Вибірку з 10 записів для ілюстрації наведено у Таблиці 5.

Таблиця 5

Виконання prompt №2

Назва метиза	Оновлений запис ukr_name	Оновлений запис eng_name	Час обробки запису
Винт M10x40 DIN 912	Винт DIN 912 M10x40	Bolt DIN 912 M10x40	9.94 секунд
Болт M12x85 DIN 603	Болт DIN 603 M12x85	Bolt DIN 603 M12x85	10.14 секунд
Гайка M12 DIN 934	Гайка DIN 934 M12	Screw DIN 934 M12	8.99 секунд
Шайба 12 DIN 125	Шайба DIN 12 M12	Washer DIN 12 M12	8.26 секунд
Гвинт M6x25 ISO 7380	Гвинт ISO 7380 M6x25	Bolt ISO 7380 M6x25	8.77 секунд
Шпилька M8x80 DIN 976	Шпилька DIN 976 M8x80	Screw DIN 976 M8x80	9.30 секунд
Винт M8x35 DIN 912	Винт DIN 912 M8x35	Bolt DIN 912 M8x35	9.01 секунд
Болт M10x70 DIN 931	Болт DIN 931 M10x70	Bolt DIN 931 M10x70	12.21 секунд
Гайка M10 DIN 934	Гайка DIN 934 M10	Screw DIN 934 M10	10.45 секунд
Шайба 10 DIN 125	Шайба DIN 125 10	Washer DIN 125 M10	10.04 секунд

Загальний час обробки 100 записів: 987.59 секунд (~16 хвилин 45 секунди). Результат перевірки коректності результатів через promt №2 при загальній кількісті 100 записів становить 0 (0,0%) некоректних записів. Приклад виконання запиту зображено на рис.7.

У таблиці 6 наведено порівняння результатів з шаблоном виконаних моделлю з використанням promt №2. Шаблон: <назва> <стандарт> <номер стандарту> <розмір> [опції].

Таблиця 6

Порівняння результатів з використанням prompt №2

№	Запис ukr_name / eng_name	Примітки
1	Винт DIN 912 M10x40 Bolt DIN 912 M10x40	Укр: відповідає шаблону Eng: відповідає шаблону)
2	Болт DIN 603 M12x85 Bolt DIN 603 M12x85	Укр: відповідає шаблону Eng: відповідає шаблону
3	Гайка DIN 934 M12 Screw DIN 934 M12	Укр: відповідає шаблону Eng: відповідає шаблону
4	Гвинт ISO 7380 M6x25 Bolt ISO 7380 M6x25	Укр: відповідає шаблону Eng: відповідає шаблону
5	Шпилька DIN 976 M8x80 Screw DIN 976 M8x80	Укр: відповідає шаблону Eng: відповідає шаблону
6	Винт DIN 912 M8x35 Bolt DIN 912 M8x35	Укр: відповідає шаблону Eng: відповідає шаблону
7	Болт DIN 931 M10x70 Bolt DIN 931 M10x70	Укр: відповідає шаблону Eng: відповідає шаблону
8	Гайка DIN 934 M10 Screw DIN 934 M10	Укр: відповідає шаблону Eng: відповідає шаблону
9	Шайба DIN 125 10 Washer DIN 125 M10	Укр: відповідає шаблону Eng: відповідає шаблону (додано до назви індекс M)
10	Шайба DIN 12 M12 Washer DIN 12 M12	Укр: відповідає шаблону Eng: відповідає шаблону

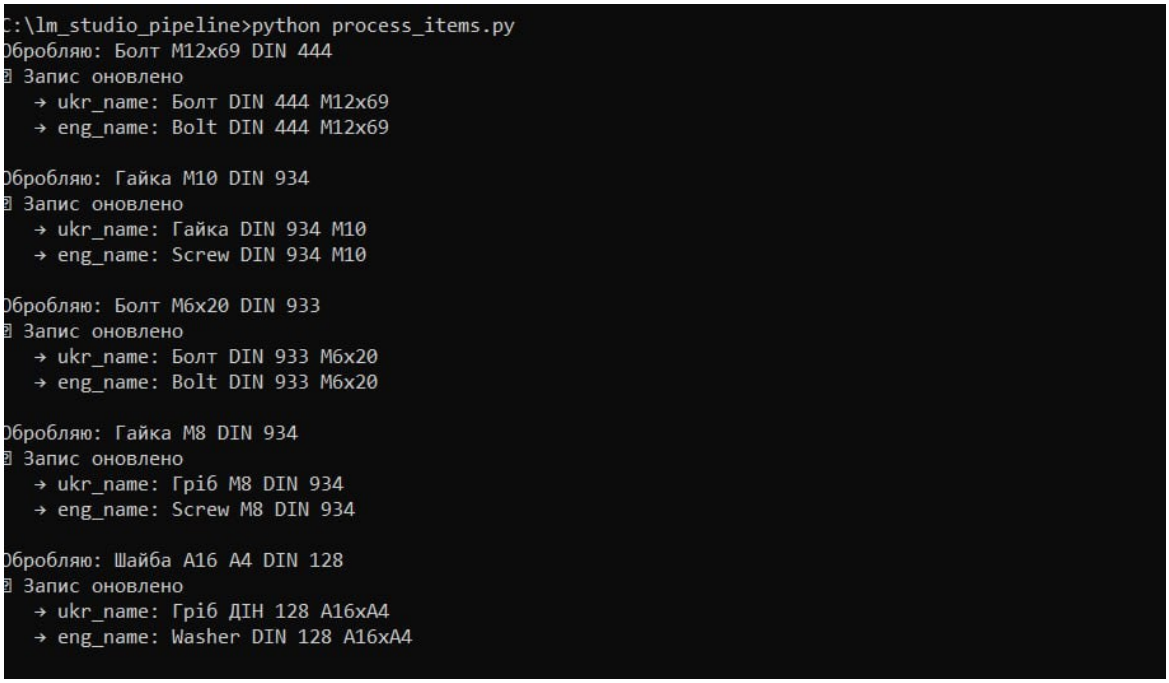

Рис. 7. Виконання проміжного запиту з стандартизації назви метизів

ВИСНОВКИ З ДАНОГО ДОСЛІДЖЕННЯ І ПЕРСПЕКТИВИ ПОДАЛЬШИХ РОЗВІДОК У ДАНОМУ НАПРЯМІ

У результаті проведеного дослідження обґрунтовано ефективність використання локальних великих мовних моделей (LLM) для автоматизованої стандартизації та багатомовного перекладу технічних назв метизів. Після донавчання моделі mistral-7b-instruct на спеціально підготовленому доменному корпусі та розробки коректного промпту (prompt №2) точність обробки назв досягла 100%, що підтверджує доцільність запропонованого підходу для локального використання у прикладних системах.

У порівнянні з цим, проведене тестування з використанням початкового недостатньо коректного промпту (prompt №1) показало, що 67% отриманих відповідей були неправильними, що підкреслює критичну роль правильного формування запитів і підготовки навчальних даних для LLM. Отримані результати демонструють, що локальна інференція моделі в поєднанні з донавчанням і точним промптом забезпечує повну відповідність міжнародним стандартам форматування назв та дозволяє одержувати коректні багатомовні переклади.

Отже, запропонована методика інтеграції локальних LLM у процеси обробки технічних даних є ефективною, дозволяє значно підвищити точність автоматизованої стандартизації і може бути використана як основа для впровадження подібних рішень у промислові та аналітичні системи.

Література

1. Machine translation in the era of large language models:a survey of historical and emerging problems / d. Ataman et al. *Information*. 2025. Vol. 16, no. 9. P. 723. URL: https://doi.org/10.3390/info16090723 (дата звернення: 05.09.2025)

2. Llm-bt-terms: back-translation as a framework for terminology standardization and dynamic semantic embedding. *Arxiv.org*. URL: https://arxiv.org/abs/2506.08174 (дата звернення: 05.09.2025)

3. Brinkmann A., Baumann N., Bizer C. Using LLMs for the Extraction and Normalization of Product Attribute Values. *arXiv.org e-Print archive*. URL: https://arxiv.org/html/2403.02130 (дата звернення: 08.09.2025)

4. Weigang L., Brom P. C. LLM-BT: Back-Translation as a Framework for Terminology Standardization and Dynamic Semantic Embedding. *Arxiv.org e-print archive*. URL: https://arxiv.org/html/2506.08174v1 (дата звернення: 08.09.2025)

5. Nahid M. M. H., Rafiei D. NormTab: improving symbolic reasoning in llms through tabular data normalization. *Findings of the association for computational linguistics*. 2024. Emnlp 2024. P. 3569–3585. URL: https://aclanthology.org/2024.findings-emnlp.203.pdf (дата звернення: 08.09.2025)

6. Shah K. How to Build Domain Specific LLMs?. *Best Artificial Intelligence Blogs*. URL: https://www.projectpro.io/article/domain-specific-llms/1111 (дата звернення: 08.09.2025)

7. Tang J. Fine-tuning llms for specific domains: why, how, and what to consider. *Medium*. Url: https://medium.com/@jamestang/fine-tuning-llms-for-specific-domains-why-how-and-what-to-consider-8b2fd5781615 (дата звернення: 08.09.2025)

References

1. Machine translation in the era of large language models:a survey of historical and emerging problems / d. Ataman et al. *Information*. 2025. Vol. 16, no. 9. P. 723. URL: https://doi.org/10.3390/info16090723 (date of access: 05.09.2025)
2. Llm-bt-terms: back-translation as a framework for terminology standardization and dynamic semantic embedding. *Arxiv.org*. URL: https://arxiv.org/abs/2506.08174 (date of access: 05.09.2025)
3. Brinkmann A., Baumann N., Bizer C. Using LLMs for the Extraction and Normalization of Product Attribute Values. *arXiv.org e-Print archive*. URL: https://arxiv.org/html/2403.02130 (date of access: 08.09.2025)
4. Weigang L., Brom P. C. LLM-BT: Back-Translation as a Framework for Terminology Standardization and Dynamic Semantic Embedding. *Arxiv.org e-print archive*. URL: https://arxiv.org/html/2506.08174v1 (date of access: 08.09.2025)
5. Nahid M. M. H., Rafiei D. NormTab: improving symbolic reasoning in llms through tabular data normalization. *Findings of the association for computational linguistics*. 2024. Emnlp 2024. P. 3569–3585. URL: https://aclanthology.org/2024.findings-emnlp.203.pdf (date of access: 08.09.2025)
6. Shah K. How to Build Domain Specific LLMs?. *Best Artificial Intelligence Blogs*. URL: https://www.projectpro.io/article/domain-specific-llms/1111 (date of access: 08.09.2025)
7. Tang J. Fine-tuning llms for specific domains: why, how, and what to consider. *Medium*. Url: https://medium.com/@jamestang/fine-tuning-llms-for-specific-domains-why-how-and-what-to-consider-8b2fd5781615 (date of access: 08.09.2025)