УДК 005:623
DOI: https://doi.org/1034169/2414-0651.2024.4(44).79-91

В. І. СЛЮСАР, *доктор технічних наук, професор*
https://orcid.org/0000-0002-2912-3149

ЛОКАЛЬНІ ВЕЛИКІ МОВНІ МОДЕЛІ ДЛЯ ОБРОБКИ КОНФІДЕНЦІЙНОЇ ІНФОРМАЦІЇ

У статті розглядаються концептуальні підходи до використання локальних великих мовних моделей (LLM) для обробки конфіденційної інформації. Використання хмарних сервісів для таких завдань може викликати занепокоєння щодо безпеки та збереження приватності даних. Тому зростає інтерес до локальних LLM, які дозволяють зберігати контроль над конфіденційною інформацією, забезпечуючи при цьому високий рівень продуктивності. Описано технічні вимоги та варіанти реалізації локальних LLM, зокрема україномовної моделі Mistral-7B, яка використовується в захищеному середовищі. В статті також розглянуто мультиагентні системи та їх роль у підвищенні гнучкості та ефективності обробки даних. Описані тестування моделей за різних рівнів квантування і наведено технічні вимоги для їхньої реалізації на локальних платформах. Основна увага приділяється забезпеченню безпеки і конфіденційності інформації під час використання локальних LLM та потенційним можливостям мультиагентних систем у майбутньому.

Ключові слова: *локальні великі мовні моделі, обробка конфіденційної інформації, мультиагентні системи, штучний інтелект, квантування, безпека даних.*

Нарощування процесів цифровізації та генерації великих даних спонукає державні організації та комерційні компанії до необхідності ефективно обробляти величезні обсяги конфіденційної інформації. Великі мовні моделі (LLM), такі як o1 від OpenAI [1], змінюють традиційні підходи до аналізу та інтерпретації текстових даних, пропонуючи безпрецедентні можливості для автоматизації та інтелектуальної обробки інформації. Зокрема, використання LLM у сфері наукового супроводження розробок озброєння та військової техніки (ОВТ) відкриває нові можливості для оптимізації процесів дослідження та проєктування. Застосування LLM в зазначеному контексті сприятиме автоматизації аналізу технічної документації, прискоренню формування технічних завдань та методик випробувань ОВТ, підвищенню точності прогнозування результатів випробувань. Це дозволяє зменшити часові та фінансові витрати на розробку, підвищуючи ефективність використання наявних ресурсів, у тому числі експертних. Проте використання хмарних сервісів для вказаних моделей цілком доречно викликає занепокоєння щодо безпеки та збереження приватності даних. Як результат, зростає інтерес до впровадження локальних LLM [2–4], які дозволяють зберігати контроль над конфіденційною інформацією, забезпечуючи при цьому високий рівень продуктивності.

Метою статті є розгляд концептуальних підходів до використання локальних мовних моделей для обробки конфіденційних даних, а також результатів апробації їх впровадження в захищеному середовищі.

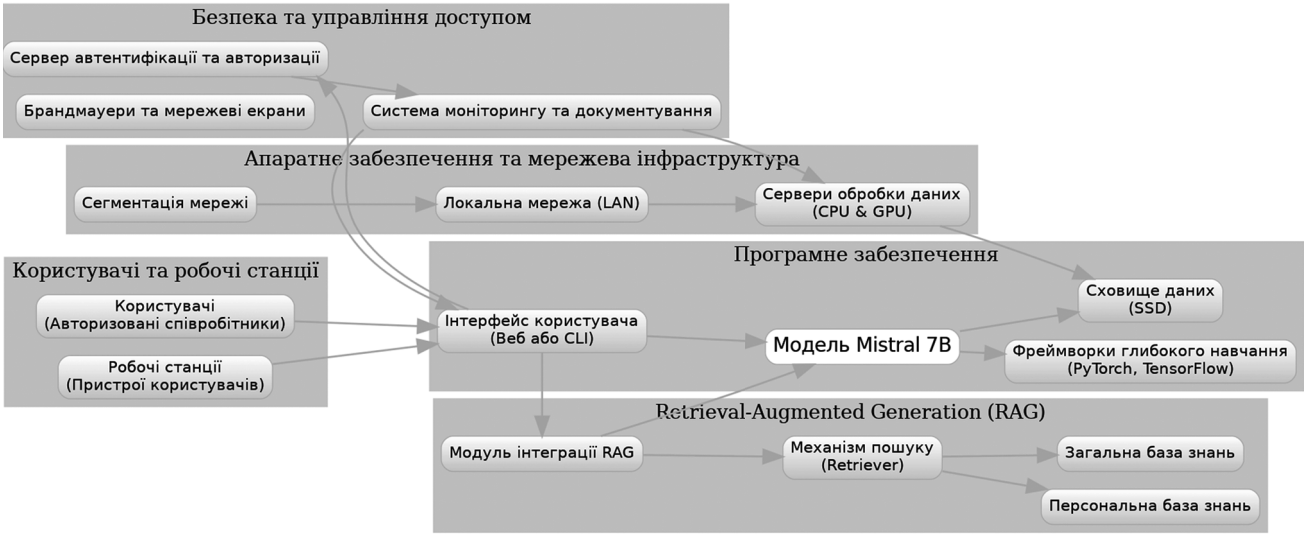
Для розуміння того, як функціонує інформаційна система з локальною LLM, варто розглядати її як складний комплекс взаємопов'язаних компонентів (рис. 1). Розглянемо його осовні складові більш детально.

Центральним, системоутворюючим елементом структури на рис. 1 слід вважати високопродуктивні сервери обробки даних, оснащені потужними процесорами та графічними картами. Вони відповідають за виконання обчислювальних завдань, пов'язаних із роботою LLM, забезпечуючи швидку і ефективну обробку інформації. В якості прикладу локальної LLM на рис. 1 використано Mistral 7B [5]. На платформі Hugging Face доступні різні її версії, а також інші LLM, викладені у відкритий доступ. Вони можуть бути попередньо завантажені з мережі, що має доступ до Інтернет, для відпрацювання загальної методики дослідження спроможностей різних LLM та набуття практичного досвіду їх використання при виконанні типових завдань.

Вказані на рис. 1 серверні потужності підключені до систем зберігання даних, де розміщуються мовні моделі, навчальні дані та результати обробки. Швидкісні накопичувачі гарантують миттєвий доступ до необхідних файлів, а системи резервного копіювання захищають від втрати даних.

Користувачі взаємодіють із системою LLM через зручний інтерфейс – це може бути веб-додаток, спеціалізоване програмне забезпечення, голосова взаємодія або застосування символів доповненої реальності. Через такий інтерфейс надсилаються запити до моделі та отримують результати її роботи.

В якості бюджетного рішення, що дозволяє оперативно розгорнути локальну систему LLM, доцільно задіяти різні фреймворки, які дозволяють оранізувати чат користувача з мовною моделлю. Серед них слід вказати, наприклад, LM Studio [6, 7]. На рис. 2 наведено інтерфейсне вікно цієї програми поточної версії 0.3.2 при тестуванні україномовної LLM Mistral-7B-Instruct-Ukrainian.gguf [8] (версія від 9 вересня 2024 р.) за різних рівнів квантування. Відповідне тестування проводилося з метою формування мінімальних вимог до апаратної реалізації серверного обладнання. Спочатку в якості сервера був задіяний ноутбук з інтегрованою відеокартою. При цьому процес інференсу здійснювався виключно на потужностях процесора (CPU). Через обмежені розміри оперативної пам’яті було використано мовну модель [8] з 3-бітним рівнем квантування вагових коефіцієнтів (Q3_K_S), що мала розмір 3.16 ГБ. В якості тестового завдання фігурував переклад з англійської на українську мову спеціального тексту у вигляді фрагменту статті [9] розміром близько



Р и с . 1. Типовий варіант розгортання локальної LLM на прикладі україномовної версії Mistral 7B

815 токенів. ***Тестовий фрагмент, який потрібно було перекласти на українську мову***: "*According to NATO standards, the identification of lessons from combat actions (LI) and their transformation into lessons learned (LL) are crucial elements in enhancing the effectiveness of military operations. This article examines the primary component of this process – the technical exploitation of enemy weaponry and military equipment samples, regulated by NATO Allied Publication AIntP-10. Technical exploitation involves applying scientific methods to analyze collected exploitable materials (CEM) and identify their useful properties. A special type of CEM is battlefield evidence (BE), which can support law enforcement agencies in legal proceedings.*

The process of technical exploitation includes the collection of samples, their initial examination, detailed technical analysis, functional testing, data collection and analysis, and the preparation of detailed reports. Technical exploitation encompasses three levels: tactical, operational, and strategic, each with its tasks and specifics. At the tactical level, immediate collection and initial analysis of samples are conducted; at the operational level, more detailed analysis and testing are performed; at the strategic level, deep research and the development of recommendations for improving one's weapon systems are carried out.

The importance of the technical exploitation process lies in identifying enemy technologies and enhancing one's weapon systems, which contributes to increased combat efficiency and safety.

The article also discusses methodological recommendations for the structure of strategic-level reports, including sample identification, analysis of technological solutions, assessment of the potential use of examination results, and the development of proposals for government agencies. Special attention is given to studying antenna elements, reengineering onboard computing equipment, using artificial intelligence technologies for sample identification, and applying large language models for report preparation. According to the authors, improving the report cataloging system and using neuro-consultants will facilitate effective expert interaction and the development of new approaches to troop protection."

Результат перекладу: "*Згідно зі стандартами НАТО, ідентифікація уроків з бойових дій (ЛІ) та їх перетворення на уроки, які вивчаються (LL) є критичними елементами для збільшення ефективності військових операцій. Ця стаття оглядає головну складову цього процесу – технічне експлуатування зразків ворожої зброї та військової техніки, регульованого НАТО Allicened Publication AIntP-10. Технічне експлуатування передбачає застосування наукових методів для аналізу зібраних експлуатаційних матеріалів (СЕМ) і визначення їх корисних властивостей. Спеціальний вид СЕМ – це доказ з бою (ВЕ), який можна підтримувати правоохоронними органами у судових справах.*

Процес технічного експлуатування включає збір проб, їх попередню експертизу, детальний технічний аналіз, колекцію даних та аналіз, підготовку детальних звітів. Технічне експлуатування має три рівні: тактичний, операційний і стратегічний, кожен із своїми завданнями і особливостями. На тактичному рівні проводяться швидка збірка та попередня обробка проб; на операційному рівні – більш детальний аналіз та тестування; на стратегічному рівні – глибоке дослідження та розвиток рекомендацій щодо покращення власної зброї.

Важливість процесу технічного експлуатування полягає в тому, щоб ідентифікувати ворожі технології та поліпшити свою зброю, що призводить до збільшення бойової ефективності та безпеки.

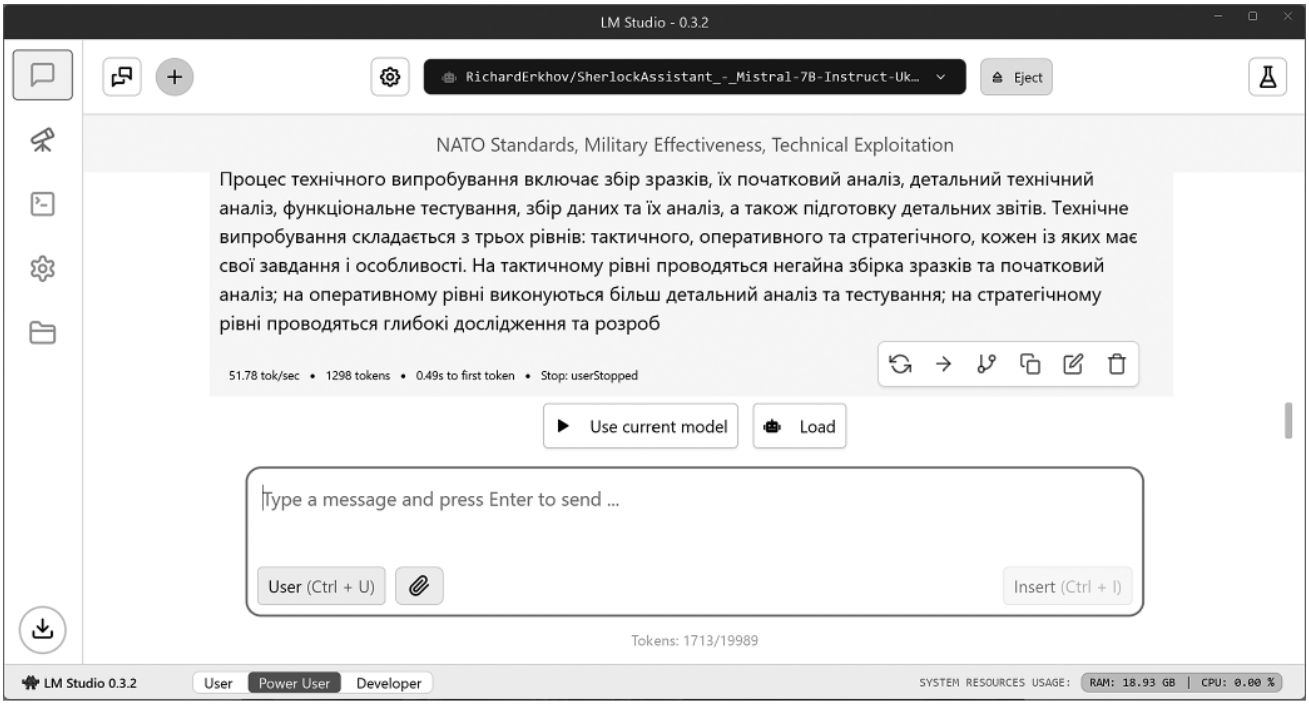
Ця стаття також розглядає рекомендації щодо структури стратегічного рівня звітів, включаючи ідентифікацію зразків, аналіз технічних рішень, оцінку можливого використання результатів досліджень та розроблення пропозицій для державних органів. У ньому особлива увага приділяється вивченню елементів антен, перерозподіленню на бортову обчислювальну апаратуру, використанню технологій штучного інтелекту для ідентифікації зразків та застосуванню великих мовних моделей для підготовки звітів. Автори вважають, що поліпшення системи каталогу звітів і використання нейроконсультантів сприятимуть ефективній взаємодії експертів та розробці нових підходів до захисту військ".

Від моменту відправлення запиту на виконання завдання до появи першого токену у перекладеному тексті пройшло 78,73 с, а подальша швидкість перекладу обмежилася рівнем 3,21 токени/с. Отриманий результат є досить посереднім за швидкістю і свідчить про потребу застосування більш потужних апаратних ресурсів, навіть для такої, необтяжливої за розміром LLM.

Суттєвого зростання швидкості перекладу вдалося досягти після переходу до більш потужного серверного обладнання з графічною картою NVIDIA RTX 3080. В ній реалізовано 8704 CUDA-ядер, що відповідають за навчання нейронних мереж, пов'язані з паралельними обчисленнями. При цьому RTX 3080 має 12 ГБ відеопам'яті, 272 тензорних ядра і 68 RT-ядер, які відповідають за завдання машинного навчання та рендеринг з трасуванням променів відповідно. Ці характеристики дозволили майже в 150 раз скоротити час до появи першого токену при перекладі тестового завдання і більше ніж в 15 раз підвищити швидкість самого перекладу (табл. 1).

Під час тестування моделей Mistral-7B-Instruct-Ukrainian на GPU з 12 ГБ пам'яті було проведено вимірювання їхньої продуктивності з різними рівнями квантування. Відповідні результати наведені в табл. 1 для різних версій зазначеної вище LLM. Наприклад, модель Mistral-7B-Instruct-Ukrainian.Q2_K.gguf має розмір 2,53 ГБ і використовує квантування Q2_k. Вона демонструє швидкість 59.31 токенів на секунду, затримка до першого вихідного токену становить 0.32 секунди, а вихідна кількість токенів – 787. В свою чергу, LLM Mistral-7B-Instruct-Ukrainian.Q3_K.gguf, з розміром 3,52 ГБ і квантуванням Q3_K, працює зі швидкістю 50.09 токенів на секунду, при скороченій 0.49 секунди затримці появи першого токену, при цьому вихідна кількість токенів зросла до 802. Найбільша за розміром модель у тестуванні, Mistral-7B-Instruct-Ukrainian.Q8_0.gguf, мала розмір 7,17 ГБ з квантуванням Q8_0. Вона продемонструвала швидкість інференсу 40.26 токенів на секунду, затримку до візуалізації першого вихідного токену – 0.37 секунди і загальний обсяг вихідних токенів – 858.

Загалом, із збільшенням точності квантування моделей спостерігалося поступове зниження швидкості генерації тексту, збільшення затримки до появи першого токену, а також зміни у кількості згенерованих токенів. Моделі з нижчим рівнем квантування працювали швидше, але з меншою точністю, тоді як моделі з вищим рівнем квантування мали більші затримки, проте забезпечували більш деталізовані результати. Така поведінка



Р и с . 2. Тест україномовної Mistral 7B в LM Studio

Т а б л и ц я 1. Показники тестування на GPU (12 GB)

LLM	Розмір, ГБ	Квантування	Швидкість, токен/с	Затримка до першого вихідного токену, с	Вихідна кількість токенів
Mistral-7B-Instruct-Ukrainian.Q2_K.gguf	2,53	Q2_k	59.31	0.32	787
Mistral-7B-Instruct-Ukrainian.Q3_K.gguf	3,52	Q3_K	50.09	0,49	802
Mistral-7B-Instruct-Ukrainian.Q4_K.gguf	4,07	Q4_K	55.77	0,51	830
Mistral-7B-Instruct-Ukrainian.Q5_0.gguf	4,65	Q5_0	52.15	0.49	2934
Mistral-7B-Instruct-Ukrainian.Q6_K.gguf	5,53	Q6_K	46.15	0.50s	865
Mistral-7B-Instruct-Ukrainian.Q8_0.gguf	7,17	Q8_0	40.26	0.37s	858

є типовою для LLM, оскільки квантування безпосередньо впливає на баланс між швидкістю обробки даних та точністю відповіді.

Зростання кількості токенів на виході може бути пов'язане з тим, як різні рівні квантування впливають на поведінку моделі під час генерації тексту. Моделі з нижчим рівнем квантування (наприклад, Q2 чи Q3) працюють менш точно, тому вони можуть частіше завершувати відповідь раніше, генеруючи менше токенів. З іншого боку, моделі з вищим рівнем квантування (Q5 чи Q8) здатні краще «розуміти» контекст і продовжувати генерацію тексту до логічного завершення, що призводить до більшої кількості токенів на виході. Крім того, високоточні моделі можуть генерувати довші та більш деталізовані відповіді, оскільки вони точніше відтворюють структуру тексту, враховуючи більшу кількість контексту з попередніх частин тексту.

Наявний в табл. 1 статистичний розкид затримки до появи першого токена може бути пов'язаний з тим, що обсяг пам'яті на GPU є достатнім для обробки LLM усіх досліджених версій, і тому збільшення розмірів моделі не призводить до помітного зростання затримки. Іншими словами, GPU не перевантажується навіть при роботі з більшими моделями, що дозволяє їй зберігати стабільну продуктивність. Це може пояснювати мінімальні відхилення у затримці, які є статистичними і не залежать значною мірою від розміру моделі чи квантування, а зумовлені внутрішніми процесами обробки та використання обчислювальних ресурсів, наприклад, асинхронністю переривань на рівні операційної системи.

Щоб оцінити якість функціонування різних версій LLM в описаному експерименті, було використано методику оцінки точності перекладу за допомогою GPT-4o [10]. Доцільність такого рішення може бути обґрунтованою з кількох причин. Перш за все слід вказати, що GPT-4o добре підходить для таких завдань через свою високу лінгвістичну компетенцію і здатність аналізувати переклад в контексті. Ця мультимодальна LLM володіє глибокими знаннями багатьох мов, включаючи англійську й українську, що дозволяє їй точно визначати помилки в синтаксисі, семантиці або стилі перекладу. Вона здатна не лише порівнювати слова, але й аналізувати контекст і правильність передачі значень. Оцінювання перекладу у контексті є особливо важливим для спеціалізованих текстів, де потрібна точність термінології та відповідність структурі оригіналу. Саме через це GPT-4o здатна помічати, чи коректно передані складні концепції та чи не втрачений сенс. Важливо також, відзначити універсальність даної LLM, адже вона може бути використана для оцінки різних типів текстів − від загальнолітературних до технічних і наукових, що робить її ідеальною для оцінки ефективності різноманітних LLM, які перекладають спеціалізовані тексти. Крім того, GPT-4o можна розглядати як референсну модель, що надає оцінку, близьку до тієї, яку може сформувати експерт-лінгвіст. Це дозволяє більш об'єктивно оцінювати якість перекладу LLM, зокрема, коли важко провести незалежну оцінку з залученням великої кількості людей. Нарешті, застосування GPT-4o для оцінки також дозволяє швидко перевірити велику кількість перекладів, що було б складно зробити з людською експертизою. Це робить методику не лише точною, але й ефективною, про що свідчать результати оцінювання в описаний спосіб, представлені в табл. 2.

Як видно, LLM з квантизацією 6 біт досить непогано перекладає спеціальні тексти. В той же час, використовувати моделі з рівнем квантування нижче 4 біт слід вважати недоцільним.

Окрім завдань перекладу, спроможності дослідженої LLM Mistral 7B дозволяють здійснювати сумаризацію україномовних текстів, з можливістю завантаження їх через меню LM Studio у вигляді файлів txt, doc та pdf. При цьому слід враховувати, що максимальний розмір вхідного запиту для даного типу LLM не перевищуватиме 32 тисячі токенів (32K), а максимальна величина відповіді становить лише 2048 токенів. Таке обмеження при перекладах текстів у даному випадку потребує їх попередньої фрагментації.

LM Studio дозволяє реалізувати роботу з LLM з розгортанням локального серверу та доступом до LLM через механізм API. Однак суттєвим недоліком при цьому є відсутність можливості створити на основі інтерфейсу LM Studio клієнтський доступ до такого серверу. Для цього на даний момент необхідно застосовувати сторонні веб-інтерфейси, хоча розумно було б встановлювати LM Studio на обладнанні користувача і взаємодіяти через відповідний інтерфейс чату та локальну мережу з серверним варіантом LM Studio, де розгорнута потрібна LLM. Можливо, в подальшому цей недолік буде усунуто.

Слід зазначити, що розглянута україномовна модель Mistral-7B-Instruct-Ukrainian.gguf була кращою станом на 20 вересня 2024 р. серед відомих альтернатив такого розміру. Навіть за появи більш просунутих за спроможностями моделей її доцільно використовувати в якості проксі-моделі для балансування локальних датасетів, задіяних для реалізації механізму RAG [11] та донавчання LLM шляхом файн-тюнингу (fine-tuning) [12]. Балансування локальних датасетів є критичним завдан-

Т а б л и ц я 2. Точність перекладу тестового тексту з англійської на українську мову за оцінкою GPT-4o

LLM	Розмір, ГБ	Квантування	Точність перекладу за оцінкою GPT-4o
Mistral-7B-Instruct-Ukrainian.Q2_K.gguf	2,53	Q2_k	6 з 10
Mistral-7B-Instruct-Ukrainian.Q3_K.gguf	3,52	Q3_K	7 з 10
Mistral-7B-Instruct-Ukrainian.Q4_K.gguf	4,07	Q4_K	8 з 10
Mistral-7B-Instruct-Ukrainian.Q5_0.gguf	4,65	Q5_0	8 з 10
Mistral-7B-Instruct-Ukrainian.Q6_K.gguf	5,53	Q6_K	8.5 з 10
Mistral-7B-Instruct-Ukrainian.Q8_0.gguf	7,17	Q8_0	8 з 10

ням у машинному навчанні, особливо при класифікації з нерівномірно представленими класами. Нерівномірність у розподілі класів може призвести до того, що модель переважно навчиться на більш представлених класах, ігноруючи менш репрезентативні, що знижує загальну ефективність моделі. Як вказано в [13], використання проксі-моделі є ефективним інструментом для вирішення цієї проблеми. Згідно з [13], проксі-модель − це спрощена або попередня версія основної LLM, яка використовується для аналізу та розуміння структури даних перед повноцінним навчанням. У контексті балансування датасетів, проксі-модель може бути використана для виявлення слабких місць, тобто визначення, які класи або підмножини даних модель класифікує неправильно або з низькою впевненістю. На основі аналізу функціонування проксі-моделі можна цілеспрямовано додавати або синтезувати дані для тих класів чи областей, де модель показує найгірші результати, що називається спрямованим балансуванням. Це дозволяє економити ресурси, оскільки замість безрозбірливого збільшення даних для всіх класів, зусилля сфокусовані на найбільш проблемних зонах. Переваги використання проксі-моделі включають економію у використанні обчислювальних ресурсів та часу, уникненні перенавчання і глибшому розумінні даних, оскільки проксі-модель може виявити приховані патерни або аномалії, які можуть бути втрачені при традиційних методах балансування. У контексті текстових датасетів, таких як ті, що отримані з Wikipedia або бази препринтів arXiv, балансування даних спрямоване на усунення нерівномірного представлення тем, жанрів або стилів. Наприклад, при побудові LLM для завдань класифікації текстів з arXiv за темами, може виникнути ситуація, коли деякі теми представлені значно більше, ніж інші. Проксі-модель, натренована на цьому датасеті, може показати, що модель має труднощі з правильною класифікацією статей з менш популярних тематик. На основі такого аналізу можна збільшити кількість текстів відповідного спрямування шляхом збору додаткових даних або використання методів аугментації тексту, таких як перефразування чи синонімізація.

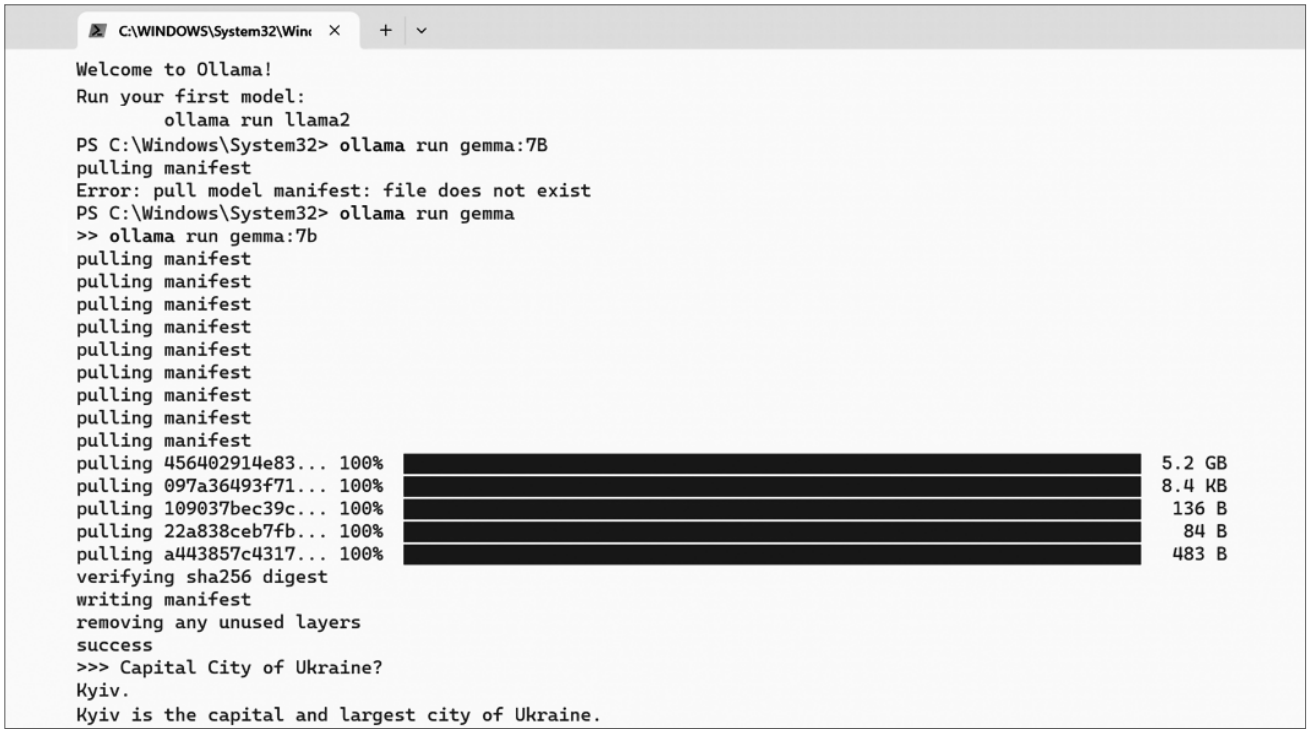
Крім того, в текстових даних можуть бути присутніми дисбаланси у стилях написання, довжині текстів або складності мовлення. Проксі-модель дозволяє виявити, що LLM погано працює з текстами певного стилю або складності. Це створить підґрунтя для цілеспрямованого додавання або генерації текстів з необхідними характеристиками для покращення загальної продуктивності моделі. У випадку з даними з Wikipedia, також може мати місце дисбаланс у мовах або їх регіональних варіаціях. Проксі-модель дозволить виявити, що модель гірше обробляє статті, написані з використанням специфічної термінології. Це сигналізуватиме про необхідність балансування текстових датасетів і сприятиме ефективному вирішенню проблеми нерівномірного представлення їх різних категорій. На основі ітераційного донавчання проксі-моделі за методикою [13] можливо оптимізувати процентне співвідношення складу локальних джерел конфіденційних даних, зокрема, текстових, з метою досягнення максимальної точності результатів функціонування проксі-моделі. Такий збалансований датасет далі можна використовувати для роботи з більш потужними LLM.

Окрім LM Studio, для попереднього тестування мовних моделей можуть бути використані й інші відомі фреймворки, зокрема Ollama [14], Jan [15], Anything LLM [16] тощо. Ollama є інструментом для локального розгортання та управління LLM, який надає інтерфейс командного рядка для завантаження, запуску та налаштування моделей на локальному комп'ютері, наприклад, у вигляді ollama run llama3 [14]. Приклад використання Ollama для запуску мовної моделі Gemma 7B наведено на рис. 3. Фреймворк підтримує різноманітні моделі та спрощує процес їх інтеграції, забезпечуючи ефективне використання ресурсів системи, у тому числі, аналогічно LM Studio, в серверному режимі з API-доступом. Однак, як видно, з рис. 3, робота з командним рядком є не такою зручною, як з інтефейсом LM Studio. До того ж, за допомогою LM Studio можливо задіяти значно більшу кількість мовних моделей з числа доступних на Hugging Face, у тому числі для одночасного їх запуску.

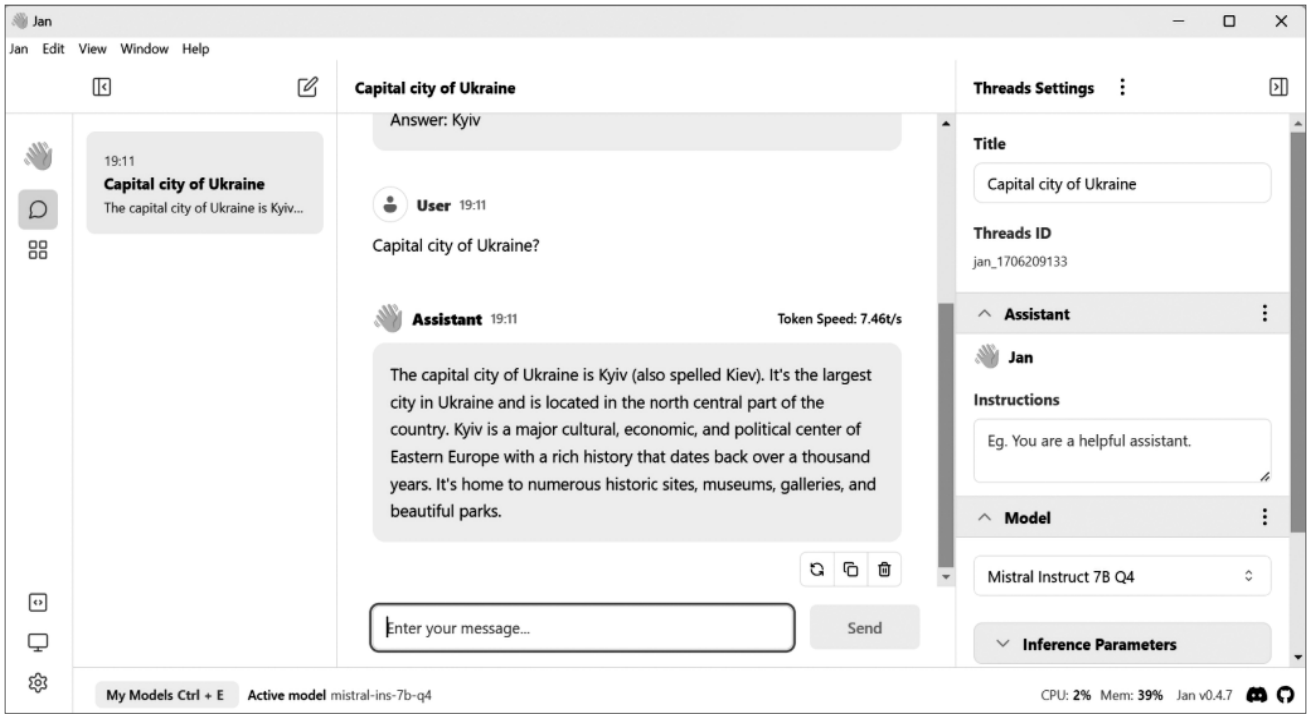
В якості компромісного варіанту можливо розгядати застосування фрейморку Jan [15] (рис. 4), який теж може працювати повністю офлайн, подібно до LM Studio. Такий режим дозволяє запускати LLM на локальних пристроях без необхідності підключення до Інтернету, що забезпечує високу конфіденційність та безпеку даних. Jan.AI підтримує різні апаратні платформи, включаючи настільні ПК, системи на базі NVIDIA GPU, Apple M-series та інші, що робить його адаптивним до широкого кола користувачів. За великим рахунком, Jan.AI є певною мірою клоном LM Studio і може слугувати альтернативою для автономного використання LLM, надаючи функціонал, схожий на LM Studio у тих випадках, коли при роботі LM Studio виникають проблеми з безкінечною генерацією відповідей на запит або похибки при запуску несумісних з фреймворком моделей.

Роль інтегратора згаданих фреймворків при локлаьному сервісі може виконувати Anything LLM [16]. Однією з головних його функцій є здатність використовувати локально встановлені фреймворки LM Studio та Ollama для спільної обробки даних. Це дозволяє користувачам комбінувати можливості різних моделей, оптимізуючи продуктивність і забезпечуючи більш точні та контекстуально релевантні відповіді. Anything LLM підтримує функції завантаження документів для реалізації технології Retrieval Augmented Generation (RAG) [11] (рис. 1), щоб LLM-система могла залучати додаткову інформацію для поліпшення контексту відповідей.

Технологія RAG працює, поєднуючи мовну модель з базою даних, зокрема векторного типу, які зберігають векторні подання документів чи даних. Під час запиту модель використовує ці дані для пошуку релевантної інформації та збагачення відповіді. Наприклад, Anything LLM дозволяє завантажувати різні типи документів (PDF, текстові файли, аудіо та інші формати), які потім індексуються і зберігаються у векторній базі даних. Це забезпечує швидкий доступ до специфічної інформації під час генерації відповідей, підвищуючи точність завдяки виокристанню ресурсу попередньо завантажених даних. Для цього використовується техніка Embedding



Р и с . 3. Приклад роботи з LLM Gemma 7B у фреймворку Ollama



Р и с . 4. Інтерфейсне вікно фреймворку Jan

та векторного пошуку, що дозволяє визначити релевантні сегменти інформації на основі подібності запитів.

У структурі, представленій на рис. 1, інтеграція RAG відбувається через додатковий модуль пошуку, який тісно взаємодіє з основною мовною моделлю. Механізм RAG працює таким чином, що під час обробки запиту користувача система не лише аналізує відповідний текст за допомогою LLM, але й на основі змісту запиту здійснює пошук релевантних фрагментів документів або записів у базі даних спеціалізованого сховища. Це дозволяє моделі отримувати актуальну та специфічну інформацію, яка може бути відсутньою в її внутрішніх параметрах або навчальних даних. Отримана інформація передається, наприклад, через внутрішній API-доступ до LLM, яка використовує її для генерації більш точних і контекстно обґрунтованих відповідей.

Цей підхід вимагає наявності ефективно організованих сховищ даних та індексаційних механізмів для швидкого доступу до інформації, самостійна розробка яких є надто складним завданням. Сховища можуть

містити накази, розпорядження, архіви донесень, технічну документацію, внутрішні звіти або будь-які інші релевантні дані. Важливо, що всі ці дані зберігаються в межах локальної ізольованої мережі, що забезпечує високий рівень безпеки та конфіденційності.

Механізм RAG підпорядковується тим самим політикам, що й основна система. Доступ до сховищ даних контролюється, а всі дії користувачів і системи фіксуються в журналах для подальшого аудиту. Це гарантує, що конфіденційна інформація не буде розкрита або використана не за призначенням.

Інтеграція RAG також вимагає додаткових обчислювальних ресурсів і підтримки з боку ІТ-фахівців. Вони відповідають за оновлення баз даних, оптимізацію пошукових алгоритмів і забезпечення безперебійної роботи системи. Регулярне оновлення інформації у сховищах гарантує актуальність та точність відповідей, що особливо корисно в управлінні критичними процесами, коли потрібна актуальна інформація, яка може швидко змінюватися з часом.

Використання механізму RAG, інтегрованого в Anything LLM, підвищує ефективність LLM-системи без необхідності її перенавчання. Нова ж технологія RAG 2.0 передбачає можливість ембеддінгу не тільки текстових, а й мультимедійних даних, що особливо актуально для обробки відеоданих, зображень, транскрибування аудіозаписів тощо. Однак її реалізація в Anything LLM лишається наразі питанням часу.

Для реалізації RAG в LM Studio, можна скористатися його вбудованими можливостями для створення текстових векторних подань (text embeddings), які є ключовими для RAG. Починаючи з версії 0.2.19, LM Studio включає в себе сервер для векторизації текстів, який дозволяє генерувати вектори для подальшого пошуку і відбору релевантної інформації з локальних документів. Цей процес можна реалізувати повністю локально, без необхідності підключення до Інтернету через POST-запит на локальний сервер з додаванням до мережевого адресу ключа / v1/embeddings. Згенеровані вектори можуть бути збережені в локальній базі даних (наприклад, ChromaDB). Для більш складної реалізації RAG можна інтегрувати LM Studio з зазначеним фреймворком AnythingLLM.

Таким чином, системи з інтегрованим механізмом RAG не лише зберігають усі переваги локальної LLM, але й значно розширюють її спроможності. Вони стають більш адаптивними, здатними швидко реагувати на нові запити та надавати інформацію, яка відповідає оперативно отриманим даним. Це відкриває нові перспективи для використання LLM в різних місіях, підвищуючи цінність та ефективність інтегрованих рішень.

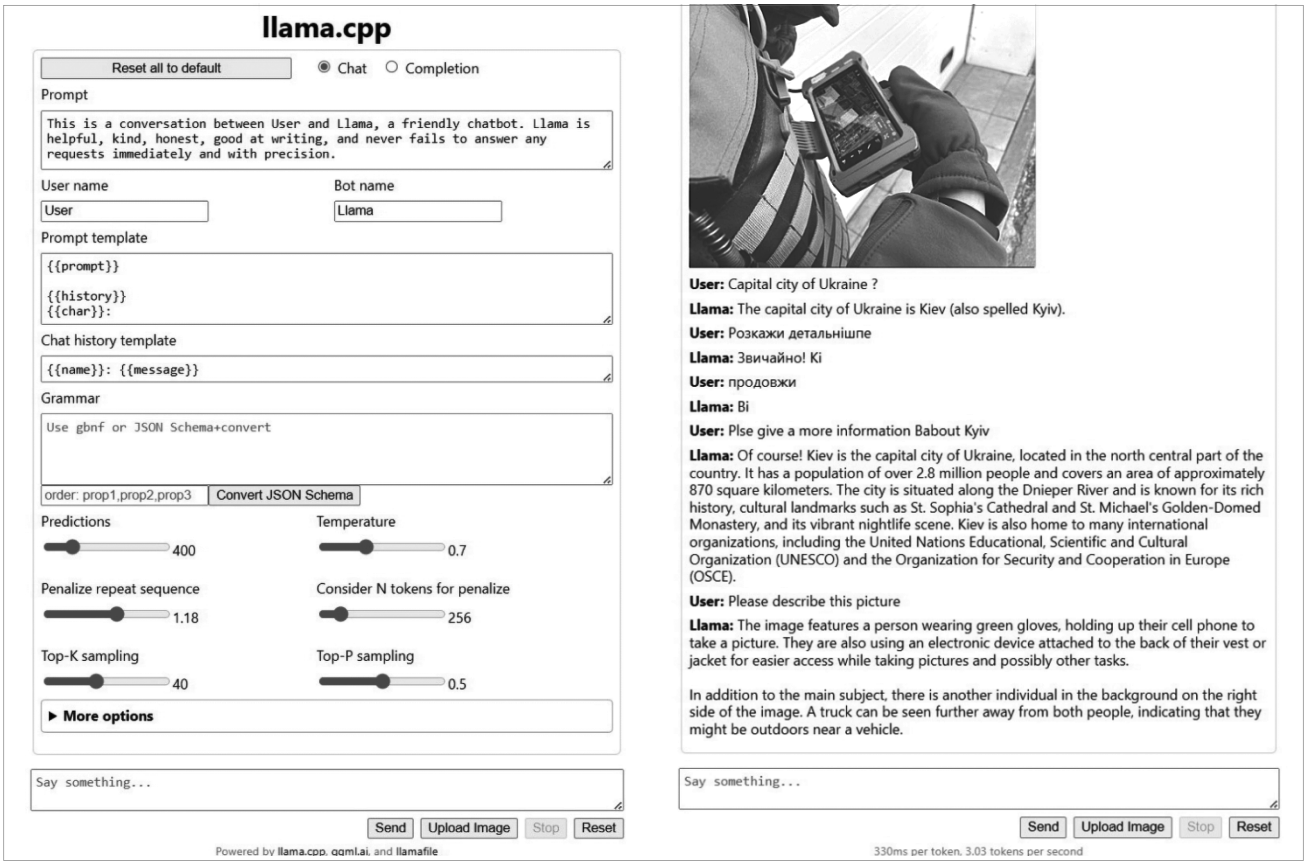
Альтернативою до розглянутих фреймворків є використання інструменту Llamafile від Mozilla [17] для створення виконуваних файлів моделей LLaMA, який дозволяє упаковувати LLM, наприклад, в один файл формату `.exe`. LLaMA (Large Language Model Meta AI) − це серія мовних моделей, розроблених дослідницькою командою Meta AI, які були оптимізовані для ефективного використання обчислювальних ресурсів. Використання Llamafile дозволяє обійти потребу в сторонніх фреймворках, надаючи можливість безпосередньо працювати з моделлю через стандартні бібліотеки глибокого навчання, такі як PyTorch або TensorFlow. Це сприяє більшому контролю над процесом розгортання та налаштування моделі, що є критичним при роботі з конфіденційними даними в ізольованому середовищі. Варіант запуску мультимодальної моделі LLaVa у вигляді Llamafile, перейменованого в exe-файл, наведено на рис. 5. Як видно, у форматі Llamafile модель LLaVa має вбудований графічний інтерфейс для реалізації чату й забезпечує генерацію текстових описів зображень, наданих користувачем. На даний момент на Github доступні Llamafile для 29 мовних моделей в різних конфігураціях, що дозволяє обрати оптимальний баланс між продуктивністю та використанням ресурсів відповідно до потреб конкретного проєкту. Цей перелік може бути розширений самостійно.

Повертаючись до розгляду структурної схеми на рис. 1, слід зазначити, що безпека доступу до локально розгорнутої LLM забезпечується системами автентифікації та авторизації, які перевіряють особистість користувачів і контролюють їхні права доступу. Це гарантує, що тільки авторизовані особи можуть користуватися можливостями системи.

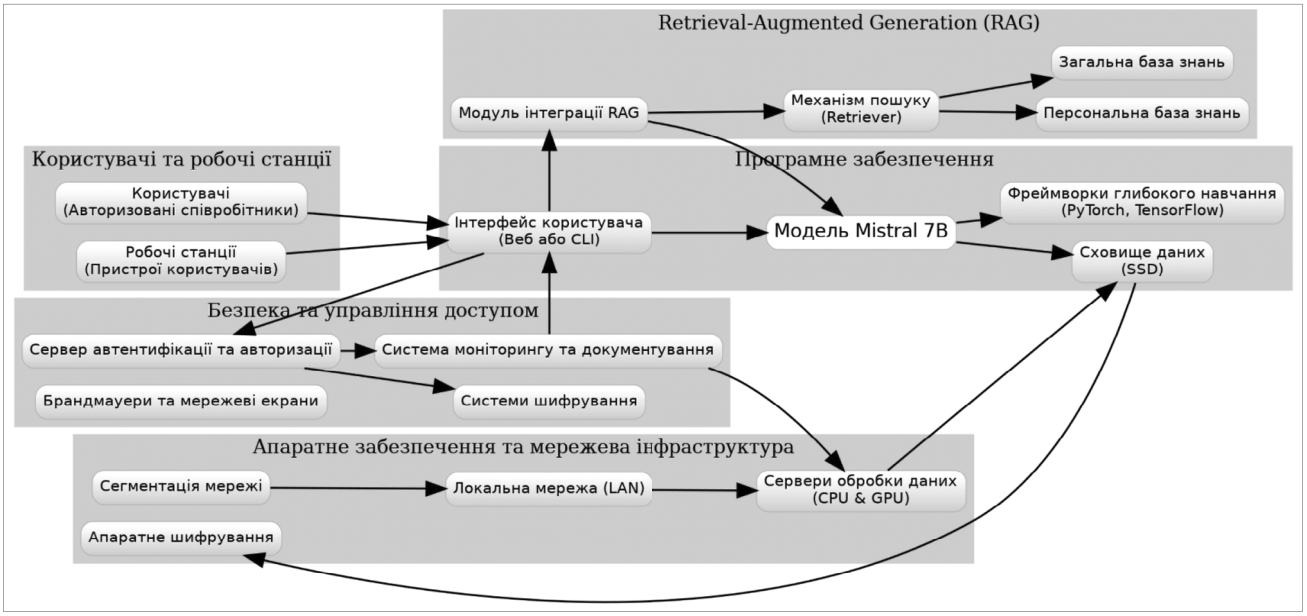
Хоча уся зазначена інфраструктура і функціонує в межах локальної ізольованої мережі, для додаткового захисту запобігання несанкціонованому доступу ззовні використовуються брандмауери та системи виявлення вторгнень, які контролюють мережевий трафік і відстежують підозрілу активність. З цією ж метою мережа розділена на сегменти, що дозволяє ізолювати різні компоненти системи і контролювати доступ між ними. Крім того, має здійснюватися шифрування даних на дисках (at rest) та під час їх передачі (in transit) за допомогою SSL/TLS (рис. 6).

Користувачами системи мають бути виключно авторизовані спеціалісти – аналітики-дослідники, експерти та інші особи, які працюють на робочих станціях, підключених до локальної мережі. Їхні дії фіксуються системами моніторингу та документування, що допомагає відстежувати роботу системи та реагувати на можливі інциденти з подальшим проведенням аудиту при необхідності. Політики безпеки визначають правила доступу до даних і ресурсів, регламентують процедури оновлення та управління системою.

Критичним аспектом роботи системи є управління даними, які мають вводитися і зберігатися з дотриманням усіх вимог безпеки. Системи резервного копіювання дозволяють швидко відновити інформацію та файли LLM у разі збоїв або інших непередбачуваних ситуацій. Команда ІТ-фахівців відповідає за підтримку та обслуговування всієї інфраструктури. Вони здійснюють оновлення програмного забезпечення та моделей, стежать за продуктивністю функціонування системи і розширюють її можливості при необхідності. Коли користувачам надсилається запит через інтерфейс, той проходить через усі компоненти системи безпеки та потрапляє до моделі LLM. Результат повертається користувачу тим самим шляхом, забезпечуючи інформаційний конвейєр з мінімальною затримкою. Таким чином, система з локальною LLM є комплексним поєднанням апаратного та



Р и с . 5. Інтерфейс та результат запуску LLaVa 1.5 у вигляді Llamafile

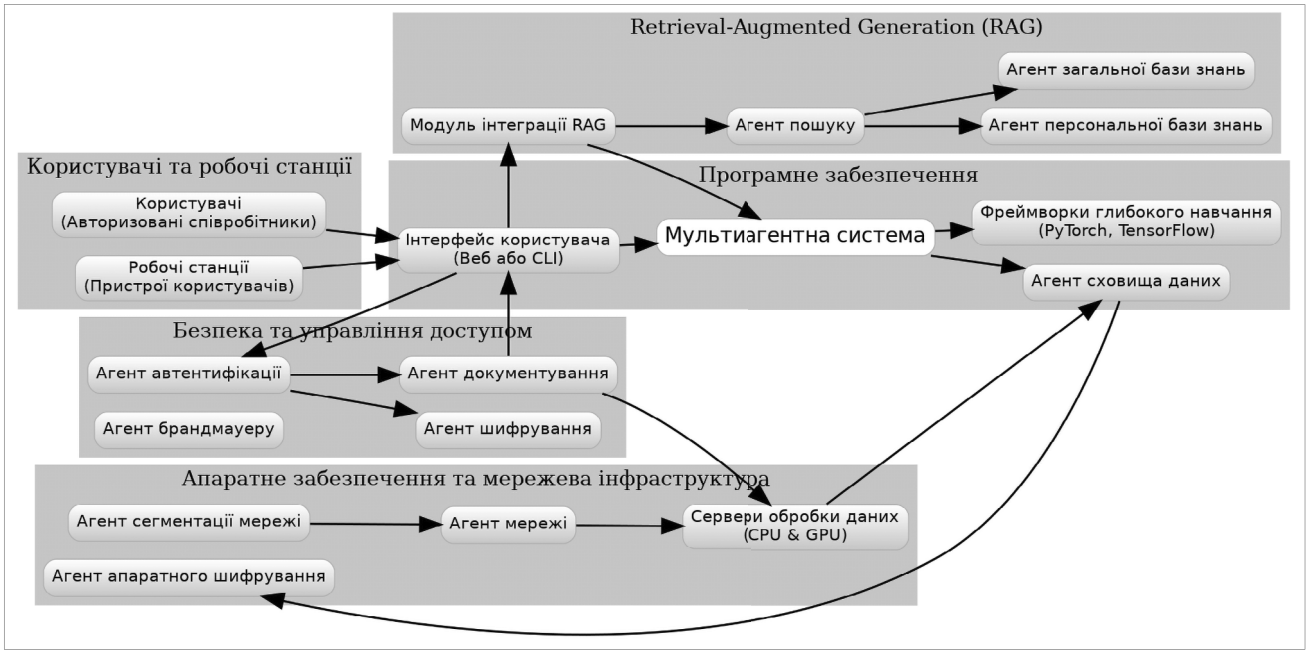


Р и с . 6. Варіант схеми на рис. 1 з доданим шифруванням даних

програмного забезпечення, мережевої інфраструктури та заходів безпеки. Її ефективне функціонування залежить від злагодженої роботи всіх компонентів і чіткого управління взаємодією між ними. Це дозволяє максимально використовувати потужність LLM, забезпечуючи при цьому високий рівень продуктивності та захисту даних.

Однак, попри значні можливості, які надає взаємодія з локальною LLM, наступним кроком у розвитку відповідної концепції може стати перехід до мультиагентної системи у вигляді так званої великої моделі дій (Large Action Model, LAM). Такий підхід передбачає розподіл функціональності між кількома автономними агентами [18] (рис. 7), кожен з яких реалізується на основі LLM і спеціалізується на виконанні певних завдань шляхом взаємодії з іншими для досягнення спільної мети.

Доцільність такого переходу обумовлена потребою в підвищенні гнучкості, масштабованості та ефективності системи. У розподіленій архітектурі мультиагентних



Р и с . 7. Мультиагентна система

систем (рис. 7) кожен агент може бути налаштований на виконання специфічних функцій: один відповідає за обробку природної мови, інший – за пошук і вилучення даних, третій – за аналітику та прийняття рішень. Це дозволяє оптимізувати кожен компонент під конкретні завдання. Наприклад, у межах АСУ командного пункту або штабу тактичного підрозділу рівня бригади, розглянута мультиагентна система може реалізувати широкий спектр функцій, спрямованих на оптимізацію управлінських та бойових процесів.

У разі збою одного з агентів система загалом продовжить своє функціонування, оскільки інші агенти можуть компенсувати або обійти непрацюючий компонент. Це особливо важливо в критичних застосуваннях, де безперервність роботи має вирішальне значення. Крім того, така система легше адаптується до змінних вимог і умов. Додавання нових агентів або оновлення існуючих не вимагає повного перероблення всієї системи. Це спрощує процес масштабування та впровадження нових функцій, що є ключовим у швидкоплинному технологічному середовищі. Мультиагентна система також підвищує ефективність обробки даних завдяки можливості паралельного виконання завдань. Агенти можуть працювати незалежно один від одного, одночасно обробляючи різні запити або частини складного завдання. Це скорочує час реакції системи та підвищує її продуктивність.

Передбачається, що взаємодія між агентами відбувається через чітко визначені протоколи та інтерфейси, що забезпечує узгодженість і цілісність системи. Це полегшує розробку, тестування та підтримку окремих компонентів, а також сприяє повторному використанню коду та модулів. У контексті безпеки мультиагентна архітектура надає додаткові переваги. Розподіл функцій дозволяє більш точно контролювати доступ до різних частин системи, встановлювати спеціалізовані механізми захисту для кожного агента та мінімізувати потенційні вразливості.

Важливою є і можливість інтеграції різних технологій штучного інтелекту. Мультиагентна система може об'єднувати моделі машинного навчання, експертні системи, алгоритми оптимізації та інші інструменти, створюючи більш потужну та універсальну платформу.

Таким чином, перехід від монолітної великої мовної моделі до мультиагентної системи відкриває нові напрями в розвитку інтелектуальних систем. Це дозволяє створювати більш адаптивні, стійкі та ефективні рішення, які краще відповідають сучасним вимогам та викликам. В умовах, коли інформаційні потоки стають все складнішими, а потреба в оперативній обробці даних зростає, мультиагентні системи можуть стати ключовим елементом успішних технологічних інновацій.

Слід зазначити, що на платформі Hugging Face вже доступна велика кількість агентних LLM. Їх пошук і тестування доцільно здійснювати за допомогою фреймморку LM Studio. Зокрема, корисною може бути модель Agent-Flow-Phone_Demo_3GB_RAM для генерації діаграми Mermaid за допомогою переліку послідовності кроків, наданих у промпті [19]. Агент обробляє вхідний текст і автоматично створює опис блок-схеми у форматі Mermaid, яку для візуалізації достатньо завантажити до редактора відповідного фреймворку (https://mermaid.live/edit). Модель суворо дотримується визначеного шаблону і не потребує додаткового налаштування запиту. При 2-бітному рівні квантування модель агента має розмір 1,36 ГБ. В якості прикладу вказаному агенту було надано тестову інструкцію для демонстрації його спроможностей:

«Create an ideogram from the sequence of learning stages in a virtual environment. Main stages:

1. Initiation of training:
   – the virtual environment is prepared for learning with all the necessary equipment and materials.
2. Command execution.
3. Feedback.

4. Collaborative interaction.
5. Summing up».

Результатом роботи агента став код:

```
graph TD
A[Start Training] --> B{Initiate VR}
B --> D[Prepare Virtual Environment]
D --> F[Enter Command Execution]
F --> G[Receive Commands]
G --> L[Handle Feedback]
L --> M[Interact Collaboratively]
M --> N[Summing Up]
N --> O[End Training]
```

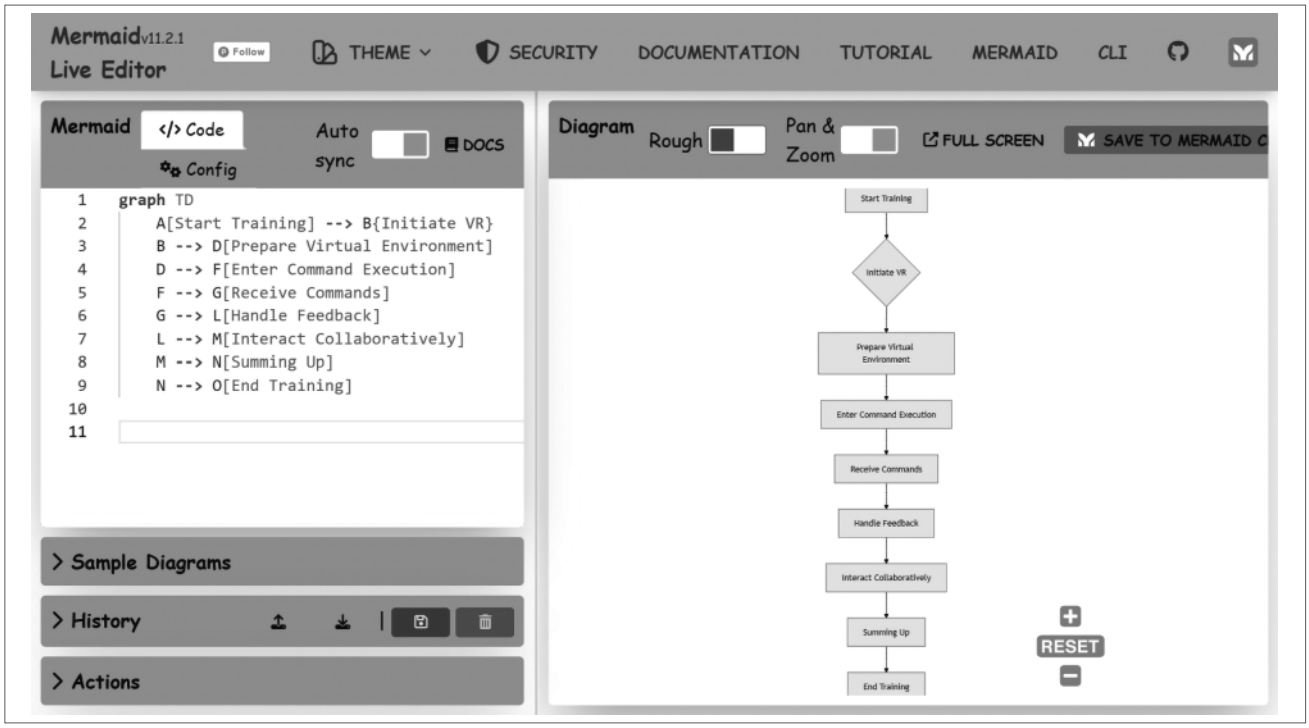
На рис. 8 наведено інтерфейс редактора фреймворку Mermaid, в якому було відтворено відповідну діаграму.

Сферами застосування описаного агенту, як зазначено в файлі Readme.md, є візуалізація покрокових процесів, перетворення письмових інструкцій на чіткі та легкі для сприйняття діаграми. Такий агент може стати в нагоді для документування робочих потоків, створення навчальних посібників тощо.

На завершення слід вказати, що для впровадження локальних LLM, призначених для обробки конфіденційної інформації, необхідно забезпечити ряд технічних вимог. Апаратні ресурси повинні бути достатньо потужними і мати високопродуктивні багатоядерні процесори для підтримки паралельних обчислень та графічні процесори (GPU), такі як NVIDIA з підтримкою CUDA, щоб прискорити навчання й інференс моделі. Разом з тим, апаратні вимоги відрізняється залежно від того, чи буде LLM використовуватися виключно для інференсу, чи вона має донавчатися на локальних даних у режимі файн-тюнингу за технологією LoRa (Low-Rank Adaptation) або іншою.

На рівні інференсу запуск мовних моделей на згаданому вище графічному процесорі RTX 3080 з 12 ГБ відеопам'яті можливий з повним їхнім завантаженням у відеокарту, якщо розмір LLM не перевищує 10 ГБ. Це можуть бути, наприклад, усі моделі з кількістю параметрів 7B, 8B та 9B при 8-бітних рівнях квантування або більші моделі (13B, 2x7B, 2x8B, 20B, 22B, 32x1.1B та ін.) з нижчими рівнями квантування. При розмірі LLM понад 10 ГБ їх функціонування у відеопам'яті можливе на основі шардингу (від англ. sharding), тобто процесу розбиття моделі на фрагменти для її завантаження. Така технологія дозволяє розділити велику модель на менші частини (фрагменти або шард), щоб полегшити її зберігання, обробку та завантаження. При цьому кожен фрагмент працює по черзі з іншими, приймаючи результати від попереднього фрагменту і передаючи свої вихідні дані наступному. Звісно, що це знижує швидкість інференсу і тому є запасним варіантом з урахуванням намагання розмістити максимально більшу за розміром LLM у відеопам'яті графічної карти.

Для пошуку компромісного рішення щодо вартості та продуктивності апаратних рішень слід розглядати усю лінійку наявних на світовому ринку відеокарт з обсягом відеопам'яті від 24 ГБ і більше. Серед відеокарт з одночипним графічним процесором найбільший обсяг пам'яті 94 ГБ має Nvidia H100. Ця відеокарта призначена для центрів обробки даних і високопродуктивних обчислень, включаючи навчання та інференс великих за розміром LLM. Також варто згадати AMD Instinct MI250X, в якій загальний обсяг пам'яті становить 128 ГБ HBM2e. Проте слід зазначити, що ця відеокарта налічує два графічних чіпи (GPU) на одній платі, кожен з яких має 64 ГБ пам'яті. Тому з точки зору одночипового рішення NVIDIA H100 94 ГБ лишається досить затребуваним рекордсменом. Більш бюджетними варіантами є перехід до застосування графічної карти Nvidia A100 з 80 ГБ або 40 ГБ відеопам'яті, а також RTX 4090 або



Р и с . 8. Синтез діаграми в Mermaid Chart за кодом, згенерованим LLM-агентом

RTX 6000 Ada з відеопамят'ю 48 ГБ. Такі ресурси дозволяють доволі ефективно оперувати з квантованими версіями 70B LLM. Для використання ж гіпермоделей типу LLaMa 3.1 з кількістю параметрів 405B, Mixtral–8x22B, MiquMaid-v2-2x70B, DBRX-16x12B та їм подібних доцільно, крім шардингу, передбачити використання одразу кількох GPU-карт розглянутих типів (до 4 або 8). Наприклад, для налаштування середньорівневого кластеру LLM підходить робоча станція типу Lambda Labs GPU Workstation, яка пропонує конфігурації з 4 GPU. У високому сегменті повноцінне інтегроване рішення з 8 GPU, оптимізоване для досліджень у сфері AI, надає NVIDIA DGX Station.

Серед процесорів для таких станцій пропонується обирати рішення типу Intel Xeon W, що забезпечують велику кількість ядер і PCIe ліній для підтримки кількох GPU. Зокрема, процесор з 32 ядрами чудово підходить для балансування навантаження між підготовкою даних і обробкою на GPU. В інтересах ефективного використання таких процесорів оперативна пам'ять (RAM) має бути мінімум 256 ГБ DDR4/DDR5 з можливістю збільшення до 1 ТБ для обробки великих наборів даних і моделей. З метою швидкого доступу до датасетів тренування необхідним є накопичувач NVMe SSD обсягом 1–2 ТБ, з додатковим накопичувачем великої ємності (4 ТБ і більше), що забезпечить зберігання великих наборів даних і контрольних точок моделі. Високошвидкісні NVMe SSD допоможуть уникнути вузьких місць при завантаженні та збереженні великих обсягів даних.

Оскільки вказані GPU і CPU споживають багато енергії, важливими для їх стабільної роботи під час інтенсивних обчислень є ефективна система охолодження і потужне джерело живлення (1 кВт і більше). Інфраструктура розгортання LLM повинна підтримувати масштабування та ефективне управління ресурсами, для чого можуть використовуватися системи оркестрації на зразок Kubernetes.

Дотримання цих технічних вимог забезпечить надійну, безпечну та ефективну роботу локальної LLM, яка здатна обробляти конфіденційну інформацію відповідно до всіх встановлених стандартів та норм. Разом з тим, слід визнати, що в межах статті неможливо охопити всі нюанси та аспекти функціонування локальних систем LLM. Накопичення кращих практик та засвоєння уроків з отриманого досвіду дозволить в подальшому неодноразово повертатися до обговорення цієї теми як перспективного напряму досліджень. Зокрема, після оволодіння методологією розгортання робочих місць для роботи з LLM безпосередньо на робочих станціях наступним кроком стане реалізація оптимальних технологій мережевого доступу до мультиагентних мультимодальних LLM, розосереджених за архітектурою Mixture of Experts (MoE) [20−23] на кількох серверних платформах. Формування вимог до подібних масштабованих рішень має супроводжуватися удосконаленням україномовних мультимодальних датасетів спеціального призначення та методик тестування локальних LLM з метою відбору кращих із них за спроможностями та якістю інференсу.

СПИСОК ПОСИЛАНЬ

1. Marino, R. (2024). Fast Analysis of the OpenAI O1-Preview Model in Solving Random K-SAT Problem: Does the LLM Solve the Problem Itself or Call an External SAT Solver. arXiv preprint arXiv:2409. P. 11232. [Online]. Available at: https://arxiv.org/abs/2409.11232.
2. Slyusar, V.I., Kondratenko, Y.P., Shevchenko, A.I. & Yeroshenko, T.V. (2024). Some Aspects of Artificial Intelligence Development Strategy for Mobile Technologies. J. of Mobile Multimedia. Vol. 20_3. Pp. 525–554. https://orcid.org/10.13052/jmm1550-4646.2031.
3. Slyusar Vadym. (2023). Large language models (LLM) in the military area. Artificial Intelligence and Intellegent Systems: XXIII Intern. Scientific and Technical Conf. (AIIS'2023), 10−11 October. https://orcid.org/10.13140/RG.2.2.30196.94086.
4. Slyusar Vadym. Reducing the Cognitive Burden of a Soldier with the Help of Personal AI and LLM Assistant. The LCGDSS Human System Integration (HSI) symposium, 12 January 2024. https://orcid.org/10.13140/RG.2.2.10264.57605/1.
5. A. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. Singh Chaplot, D. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. Renard Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, W. E. Sayed. Mistral 7B. 2023. 9 p. Available at: https://arxiv.org/pdf/2310.06825.pdf.
6. LM Studio. 2024. [Online]. Available: https://lmstudio.ai/.
7. Vakulenko, M. & Slyusar, V. (2024) Automatic smart subword segmentation for the reverse Ukrainian physical dictionary task. Proc. of the Modern Data Science Technologies Workshop (MoDaST-2024). Lviv. Ukraine. May 31 − June 1. Pp. 59–73.
8. Erkhov, R. (2024). SherlockAssistant - Mistral-7B-Instruct-Ukrainian-gguf. Hugging Face [Online]. Available at: https://huggingface.co/RichardErkhov/SherlockAssistant_-_Mistral-7B-Instruct-Ukrainian-gguf.
9. Слюсар В.І., Сотник В.В., Чепков І.Б. Методологія досліджень систем озброєння: теоретичні та практичні аспекти. Озброєння та військова техніка. 2024. № 3(43). С. 3–8. https://doi.org/1034169/2414-0651.2024.3(43).3-8.
10. OpenAI. GPT-4 technical report. arXiv, 2024. [Online]. Available at: https://arxiv.org/abs/2303.08774.
11. P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, et al. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. Advances in Neural Information Processing Systems. Vol. 33. Pp. 9459–9474. [Online]. Available at: https://arxiv.org/abs/2005.11401.
12. Parthasarathy, V.B., Zafar, A., Khan, A. & Shahid, A. (2024). The Ultimate Guide to Fine-Tuning LLMs from Basics to Breakthroughs: An Exhaustive Review of Technologies, Research, Best Practices, Applied Research Challenges and Opportunities. arXiv preprint arXiv:2408. P. 13296. [Online]. Available at: https://arxiv.org/abs/2408.13296.
13. S. Fan, M. Pagliardini & M. Jaggi (2023). DoGE: Domain Reweighting with Generalization Estimation. arXiv preprint arXiv:2310.15393, [Online]. Available at: https://arxiv.org/abs/2310.15393.

14. Ollama. 2024. [Online]. Available at: https://ollama.com/.
15. Rethink the Computer. 2024. [Online]. Available at: https://jan.ai/.
16. AnythingLLM. The all-in-one AI application. 2024. [Online]. Available at: https://anythingllm.com/.
17. Lamafile. 2024. [Online]. Available at: https://github.com/Mozilla-Ocho/llamafile/.
18. Tomasz Wieroński. (2023). Sztuczna inteligencja w strategicznych grach planszowych: czy algorytm może zastąpić człowieka? Krakow: AGH. 80 s.
19. TroyDoesAI/Agent-Flow-Phone_Demo_3GB_RAM. 2024. [Online]. Available at: https://huggingface.co/TroyDoesAI/Agent-Flow-Phone_Demo_3GB_RAM.
20. Zhi-Hua Zhou, (2012). Ensemble Methods: Foundations and Algorithms (Chapman & Hall/CRC Machine Learning & Pattern Recognition), CRC Press, Taylor & Francis Group, Boca Raton, FL, USA.
21. Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. (2017). Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. arXiv preprint arXiv:1701.06538, [Online]. Available at: https://arxiv.org/pdf/1701.06538.
22. Mixtral of experts. A high quality Sparse Mixture-of-Experts. https://mistral.ai/news/mixtral-of-experts/.
23. Zixiang Chen, Yihe Deng, Yue Wu, Quanquan Gu, Yuanzhi Li. Towards Understanding Mixture of Experts in Deep Learning. 04 August 2022. Available at: https://arxiv.org/abs/2208.02813.

REFERENCES

1. Marino, R. (2024). Fast Analysis of the OpenAI O1-Preview Model in Solving Random K-SAT Problem: Does the LLM Solve the Problem Itself or Call an External SAT Solver. arXiv preprint arXiv:2409. P. 11232. [Online]. Available at: https://arxiv.org/abs/2409.11232.
2. Slyusar, V.I., Kondratenko, Y.P., Shevchenko, A.I. & Yeroshenko, T.V. (2024). Some Aspects of Artificial Intelligence Development Strategy for Mobile Technologies. J. of Mobile Multimedia. Vol. 20_3. Pp. 525–554. https://orcid.org/10.13052/jmm1550-4646.2031.
3. Slyusar Vadym. (2023). Large language models (LLM) in the military area. Artificial Intelligence and Intellegent Systems: XXIII Intern. Scientific and Technical Conf. (AIIS'2023), 10−11 October. https://orcid.org/10.13140/RG.2.2.30196.94086.
4. Slyusar Vadym. Reducing the Cognitive Burden of a Soldier with the Help of Personal AI and LLM Assistant. The LCGDSS Human System Integration (HSI) symposium, 12 January 2024. https://orcid.org/10.13140/RG.2.2.10264.57605/1.
5. A. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. Singh Chaplot, D. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. Renard Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, W. E. Sayed. Mistral 7B. 2023. 9 p. Available at: https://arxiv.org/pdf/2310.06825.pdf.
6. LM Studio. 2024. [Online]. Available: https://lmstudio.ai/.
7. Vakulenko, M. & Slyusar, V. (2024) Automatic smart subword segmentation for the reverse Ukrainian physical dictionary task. Proc. of the Modern Data Science Technologies Workshop (MoDaST-2024). Lviv. Ukraine. May 31 − June 1. Pp. 59–73.
8. Erkhov, R. (2024). SherlockAssistant - Mistral-7B-Instruct-Ukrainian-gguf. Hugging Face [Online]. Available at: https://huggingface.co/RichardErkhov/SherlockAssistant_-_Mistral-7B-Instruct-Ukrainian-gguf.
9. Slyusar, V.I., Sotnyk, V.V. & Chepkov, I.B. (2024). "Metodologiia doslidzhen system ozbroennia: teoretychni ta praktychni aspekty" [Research methodology of weapons systems: theoretical and practical aspects]. Weapons and military equipment. № 3(43). Pp. 3–8. https://doi.org/1034169/2414-0651.2024.3(43).3-8.
10. OpenAI. GPT-4 technical report. arXiv, 2024. [Online]. Available at: https://arxiv.org/abs/2303.08774.
11. P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, et al. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. Advances in Neural Information Processing Systems. Vol. 33. Pp. 9459–9474. [Online]. Available at: https://arxiv.org/abs/2005.11401.
12. Parthasarathy, V.B., Zafar, A., Khan, A. & Shahid, A. (2024). The Ultimate Guide to Fine-Tuning LLMs from Basics to Breakthroughs: An Exhaustive Review of Technologies, Research, Best Practices, Applied Research Challenges and Opportunities. arXiv preprint arXiv:2408. P. 13296. [Online]. Available at: https://arxiv.org/abs/2408.13296.
13. S. Fan, M. Pagliardini & M. Jaggi (2023). DoGE: Domain Reweighting with Generalization Estimation. arXiv preprint arXiv:2310.15393, [Online]. Available at: https://arxiv.org/abs/2310.15393.
14. Ollama. 2024. [Online]. Available at: https://ollama.com/.
15. Rethink the Computer. 2024. [Online]. Available at: https://jan.ai/.
16. AnythingLLM. The all-in-one AI application. 2024. [Online]. Available at: https://anythingllm.com/.
17. Lamafile. 2024. [Online]. Available at: https://github.com/Mozilla-Ocho/llamafile/.
18. Tomasz Wieroński. (2023). Sztuczna inteligencja w strategicznych grach planszowych: czy algorytm może zastąpić człowieka? Krakow: AGH. 80 s.
19. TroyDoesAI/Agent-Flow-Phone_Demo_3GB_RAM. 2024. [Online]. Available at: https://huggingface.co/TroyDoesAI/Agent-Flow-Phone_Demo_3GB_RAM.
20. Zhi-Hua Zhou, (2012). Ensemble Methods: Foundations and Algorithms (Chapman & Hall/CRC Machine Learning & Pattern Recognition), CRC Press, Taylor & Francis Group, Boca Raton, FL, USA.
21. Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. (2017). Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. arXiv preprint arXiv:1701.06538, [Online]. Available at: https://arxiv.org/pdf/1701.06538.
22. Mixtral of experts. A high quality Sparse Mixture-of-Experts. https://mistral.ai/news/mixtral-of-experts/.
23. Zixiang Chen, Yihe Deng, Yue Wu, Quanquan Gu, Yuanzhi Li. Towards Understanding Mixture of Experts in Deep Learning. 04 August 2022. Available at: https://arxiv.org/abs/2208.02813.

Slyusar V.I.

LOCAL LARGE LANGUAGE MODELS FOR CONFIDENTIAL INFORMATION PROCESSING

The growing demand for digitalization and the generation of vast amounts of data drives government organizations and commercial companies toward the need for effective processing of large volumes of confidential information. Large language models, such as o1 from OpenAI, are transforming traditional approaches to data analysis and interpretation, offering unprecedented capabilities for automating and intellectually processing information. However, the use of cloud-based services for these models raises concerns about security and privacy. As a result, there is a rising interest in implementing local LLMs that allow control over confidential information while maintaining high levels of productivity. The article delves into conceptual approaches to the application of local language models for processing confidential data and examines the results of their deployment in secure environments. At the same time, various quantization levels of language models are evaluated, including a case study of the Ukrainian-language model Mistral-7B, assessing their performance in different configurations.

In addition to LLMs, the article also introduces multi-agent systems, exploring their role in enhancing the flexibility and efficiency of data processing. These systems, which consist of multiple agents working collaboratively to solve complex problems, can be configured to perform specific tasks. For example, one agent may handle natural language processing, while another focuses on data retrieval or decision-making analytics. This modular approach enhances system adaptability, scalability and efficiency, making multi-agent systems particularly valuable in dynamic environments. Additionally, this article also examines the technical and hardware requirements necessary for implementing these solutions on local infrastructures.

Keywords: *local large language models, confidential information processing, multi-agent systems, artificial intelligence, quantization, data security.*

Відомості про автора:

Слюсар Вадим Іванович
доктор технічних наук, професор
https://orcid.org/0000-0002-2912-3149

Information about the author:

Vadym Slyusar
Doctor of Technical Sciences, Professor,
Kyiv, Ukraine
https://orcid.org/0000-0002-2912-3149

Стаття надійшла до редколегії 24.09.2024.