УДК: 004.93:316.774:004.89

О.В. Бісікало
В.В. Дадиверін

ОГЛЯД ТА АНАЛІЗ АРХІТЕКТУРИ RECURSIVE LANGUAGE MODELS У КОНТЕКСТІ РОЗПІЗНАВАННЯ ФЕЙКОВОЇ ІНФОРМАЦІЇ

Вінницький Національний Технічний Університет

Анотація

У роботі проведено дослідження архітектурного підходу Recursive Language Models як альтернативи традиційним інструментам обробки довгих текстових фрагментів у системах перевірки інформації на достовірність. Показано, що на відміну від інших підходів, RLM має можливість зберігати текст за межами контексту головної моделі та проводити його обробку через згенеровану логіку з рекурсивними викликами. Зроблено акцент на здатність RLM підвищувати точність виявлення неправдивої інформації за рахунок збереження контекстних зв'язків та прозорості логіки аналізу. Окремим блоком проаналізовано переваги (масштабованість, модульність, ефективність) та недоліки (затримки, витрати, маршрутизаційні помилки) архітектури. Робота показує перспективність RLM-підходу для побудови агентних систем обробки тексту в умовах інформаційної перенасиченості.

Ключові слова: *Recursive Language Models, фактчекінг, RAG, рекурсія, вікно контексту, дезинформація, обробка тексту, велика мовна модель.*

Abstract

The paper studies the architectural approach of Recursive Language Models as an alternative to traditional tools for processing long text fragments in information verification systems. It is shown that, unlike other approaches, RLM has the ability to store text outside the context of the main model and process it through generated logic with recursive calls. Emphasis is placed on the ability of RLM to increase the accuracy of detecting false information by preserving contextual connections and transparency of the analysis logic. A separate block analyzes the advantages (scalability, modularity, efficiency) and disadvantages (delays, costs, routing errors) of the architecture. The paper shows the prospects of the RLM approach for building agent-based text processing systems in conditions of information overload.

Keywords: *Recursive Language Models, fact-checking, RAG, recursion, context window, disinformation, text processing, large language model.*

Вступ

Проблематика обмеженості "вікна контексту" сучасних великих мовних моделей (далі — LLM) спонукає до розвитку нових підходів обробки великих доробків текстової інформації. При досягненні межі довжини контексту зазвичай спостерігається втрата зв'язків між фрагментами тексту, що напряму впливає на кількість помилок при обробці та якість кінцевого результату. Такий процес називають "контекстним гниттям" [5]. Збільшення розміру контекстного вікна потребує додаткових апаратних можливостей або архітектурних змін, що позначається на рівні обчислювальної складності, а отже, і на вартості використання подібних моделей, але, найголовніше, немає ніякої гарантії на якісне покращення аналізу на довгих послідовностях.

У статті Recursive Language Models for Large-Scale Context Reasoning було запропоновано концепцію LLM з використанням рекурсії, де текст не передається напряму на вхід моделі, а залишається в зовнішньому середовищі обчислення [1]. Декларується, що на відміну від RAG-систем,

рекурсивні LLM дозволяють здійснювати ітеративний доступ до всього тексту під час обробки в поєднанні з алгоритмічним аналізом звичних мовних моделей. В цій роботі буде розглянуто переваги та недоліки цього підходу.

Результати дослідження

Задачі розпізнавання фейкової інформації виділяють стабільність зберігання логічних зв'язків між різними джерелами інформації як один з ключових аспектів точності інструменту. Далі ми розглянемо загальновідомі підходи до автоматизованого фактчекінгу.

Аналіз різних підходів:

1. Традиційний спосіб або Feed-all: Тут спостерігається головна проблема, що зазначена у вступі, так зване "контекстне гниття" [5]. Через те, що на вхід LLM подається увесь текст, в якійсь момент обробки досягається вичерпання контексного вікна і втрачається зв'язок з початковими фрагментами тексту, і це призводить до неповного порівняння окремих фрагментів або, ще гірше, призводить до "вигадування" мовною моделлю.
2. Retrieval-Augmented Generation (RAG): У даного підходу є можливість шукати "правильні" фрагменти у зовнішніх великих корпусах через індексацію [2], але якість результату залежить від довершеності алгоритму пошуку та частоти самої індексації. В ситуаціях, у яких дезінформація формується через поєднання розрізнених фактів, часто пропускаються критично важливі фрагменти, які є прямим доказом фейковості. Розширені підходи, такі як GraphRag [3], значно просунулися у структурному аналізі, але все одно не усунули проблему фрагментарного доступу до тексту.
3. Summarize and chunking: Цей підхід розділяє текстову інформацію на частини або стискає блоки за контекстом у підсумкове заключення. У статті "Lost in the Middle: How Language Models Use Long Contexts" підхід "LOST cross-chunk context" має проблеми у вигляді втрат зв'язку між шматками інформації та неправильного формулювання заключення з втратами контексних нюансів, що може критично впливати на можливість знаходити маніпулятивні вставки [5].
4. Recursive Language Models (RLM): На відміну від вище зазначених методів, цей підхід виносить повну версію тексту за межі контексту моделі і контактує з цією інформацією програмними засобами. Під час роботи модель генерує код, який займається обробкою обраних фрагментів і рекурсивними викликами самого себе або інших моделей для виконання потрібних підзадач.

Переваги RLM:

1. Модульність та складність застосування.

RLM варто застосовувати як модуль складнішої архітектури або у вигляді надбудови над готовою мовною моделлю. Складність інтеграції маленька, бо це сформована бібліотека, яка легко застосовується у потрібному місці у вигляді звичайного виклику, наприклад, "rlm.completion(prompt, model)". Автори вказують, що одним з оптимальних застосувань може бути такий підхід: обирається головна LLM (наприклад, GPT-5), яка займається стратегічними прийняттями рішень, а доступ до інформації та управління виконання обробки делегується меншій або дешевшій моделі (наприклад LLaMa), що покращує завадостійкість системи і робить її більш придатною до налаштувань.

2. Контекстна гнучкість.

RLM має можливість працювати з довільними зовнішніми даними. Для перевірки фактів з бази знань завантажує відповідні дані у змінну "context" і алгоритмічно проводить перевірку взятого фрагмента. Менеджментом порядку перевірки фрагментів займається агент (головний LLM), і завдяки цьому не втрачається повний контекст, бо обсяг обробленої інформації оброблюється поступово, без втрати зв'язків.

3. Менеджмент пам'яттю.

Контекст RLM постійно зберігається назовні від головного агента, тобто модель буде бачити тільки "корисну" інформацію, яка буде поступово нарощуватися в процесі поступового аналізу довгого тексту, розділеного на релевантні сегменти. Цим принципом усувається інформаційний шум за великого контексту та накопичується результат кожного кроку (виклику). Останнє можна відносити до "довготривалої" пам'яті, що може слугувати проміжним доказом або базою для повторного аналізу з поглибленням контексту.

4. Продуктивність системи.

У статті про RLM наводяться дані, які вказують на високу ефективність підходу у завданнях з довгим контекстом. Зокрема, наводиться приклад аналізу контекстів на мільйони токенів, де досягається істотне покращення у якості показників точності в порівнянні з базовою моделлю GPT-5 [1]. Також наводиться метрика, що вказує на зниження вартості запитів на 80–90 % для одного виклику головної моделі на той самий фрагмент даних.

5. Пристосованість до аналізу.

Кожний крок виконання, рекурсивний виклик або логічний фільтр фіксується у журналі кроків у REPL та у журналі запитів до підмоделей. Таким чином, можна проаналізувати розвиток "думки" агента та зрозуміти, яким чином була обрана послідовність аналізу фрагментів тексту, бо це є важливим для додаткового налаштування системи та захисту отриманих результатів обробки — так званої адвокації висновків роботи.

Недоліки / обмеження RLM:

1. Час роботи.

RLM-логіка складається з послідовних кроків: генерація коду, його виконання та рекурсивні виклики, тому час роботи буде довшим за традиційний LLM-запит. У звичних реалізаціях всі задачі виконуються послідовно, а це може стати критичною проблемою для задач з обмеженими часовими рамками.

2. Ресурси для роботи.

Рекурсивні виклики є досить витратними, тому при неправильній генерації коду чи великій кількості запитів об'єм ресурсів, що потрібні для виконання підзадач, буде рости експоненційно. В комерційних проєктах це може призвести до високих витрат на звичних задачах пошуку.

3. Помилки маршрутизації.

Головний агент контролює порядок аналізу фрагментів тексту, якщо допускається помилка релевантності або коректності підзапиту, то обов'язково втрачається деякий об'єм даних, що може бути критичним при перевірці на достовірність. Виявлення таких помилок є досить складною задачею, тому варто вводити у подібні системи інструменти контролю, такі як моніторинг стану середовища та обмежувач стеку рекурсії.

4. Індексація корпусів.

Практика ідентифікаторів фейків підказує, що RLM потребує наявності зовнішніх векторних індексів (наприклад, Wikipedia dump) для того, щоб агент міг дослідити фрагменти на релевантність і не витрачати час на послідовне сканування. Саме тому для уникнення зазначеного вище недоліку потрібно створювати систему, де RLM працює у парі з Lazy Loading у Google ADK [4] або RAG-агентом. Також варто враховувати якість зовнішнього корпусу знань та його частоту індексації.

Попри зазначені вище недоліки подібного підходу, у задачах перевірки на достовірність великого масиву тексту переваги переважають через можливість динамічно зберігати всю потрібну інформацію, що позитивно впливає на надійність кінцевого результату. Також варто зазначити, що більшість негативних аспектів розглянутого підходу відноситься до множини розв'язуваних задач, тому не можна їх назвати критичними вадами. До прикладу, одним з перспективних напрямів індексації текстової інформації та зменшення обсягу ресурсів, що потрібні для виконання підзадач, які виникають, може бути запропонований в [6] підхід до образного аналізу та синтезу природно-мовних конструкцій. Зокрема йде мова про образний рівень індексації та пошуку, на якому лексичні конструкти розглядаються лише як один (вербальний) тип з багатьох типів образних ознак. Розвиток архітектури RLM на основі застосування результатів моделювання ментальних процесів природного інтелекту безумовно потребує подальших досліджень.

Висновки

RLM слід розглядати як архітектурний патерн організації взаємодії головного LLM з довгим текстом. Використання такого патерну дозволяє розширити масштаб аналізу текстової інформації до мільйонів токенів, зберігаючи ключові контексти та більшість логічних зв'язків. Це є дуже важливим аспектом для систем автоматизованої перевірки тексту, бо завдяки рекурсивним запитам відбувається

перехресна перевірка та більш глибоке дослідження дрібних нюансів. Результати замірів роботи над задачами з великим обсягом інформації вказують на те, що створення складної системи з RLM-агентом якісно переважає систему без подібного інструменту, подекуди різниця може досягати 100%.

Беручи до уваги всю архітектурну складність (управління контекстом та REPL), можна сміливо рекомендувати впровадження RLM для задач, що потребують якісного пошуку маніпуляцій, перевірки джерел або навіть правого аналізу. Такі архітектурні рішення в перспективі мають суттєво підвищити довіру до LLM-систем для фактчекінгу через свою прозорість у прийнятті рішень під час аналізу.

У підсумку можна стверджувати, що RLM – це не просто відповідь на технічне обмеження контексту в LLM, а цілісна архітектурна концепція, що відкриває нові можливості для систем достовірної автоматичної перевірки фактів, особливо в умовах інформаційного перенасичення та зростаючих викликів дезінформації.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Zhang A., et al. Recursive Language Models for Large-Scale Context Reasoning [Електронний ресурс]. – 2025. – URL: https://arxiv.org/html/2512.24601v1.
2. Lewis P., et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks [Електронний ресурс]. – 2020. – URL: https://arxiv.org/abs/2005.11401.
3. Peng H., et al. GraphRAG: Leveraging Knowledge Graphs for Retrieval-Augmented Generation [Електронний ресурс]. – 2023. – URL: https://arxiv.org/abs/2308.03284.
4. Google DeepMind. Autoformalization with RLM Agents (ADK Toolkit) [Електронний ресурс]. – 2025. – URL: https://github.com/deepmind/autoformalization-adk.
5. Press O., et al. Lost in the Middle: How Language Models Use Long Contexts [Електронний ресурс]. – 2023. – https://arxiv.org/abs/2307.03172.
6. Бісікало О. В. Формальні методи образного аналізу та синтезу природно-мовних конструкцій : монографія. — Вінниця : ВНТУ, 2013. – 240 с.

Бісікало Олег Володимирович – д-р техн. наук, професор, завідувач кафедри АІІТ, Вінницький національний технічний університет, м. Вінниця, e-mail: obisikalo@vntu.edu.ua

Дадиверін Віталій Валерійович – аспірант групи 126-24а, факультет автоматизації та інтелектуальних інформаційних технологій, Вінницький національний технічний університет, Вінниця, e-mail: vetaldadyverin@gmail.com

Bisikalo Oleg V. – Dr.Sc. (Eng.), Professor of the Faculty for Computer Systems and Automatic, Vinnytsia National Technical University, Vinnytsia, e-mail: obisikalo@vntu.edu.ua

Dadyverin Vitalii V. – Department of Automation and intelligent information technologies, Vinnytsia National Technical University, Vinnytsia, e-mail: vetaldadyverin@gmail.com