DOI: https://doi.org/10.36910/6775-2524-0560-2025-61-17
УДК 004.85:004.75
Левицька Тетяна Олександрівна[1], к.т.н., доцент
https://orcid.org/0000-0003-3359-1313
Копійка Олександр Сергійович[1], студент
https://orcid.org/0009-0004-4117-4899
Грищак Дмитро Дмитрович[2], к. фіз-мат.н.,т, докторант
https://orcid.org/0000-0001-8956-8468
[1] ДВНЗ «Приазовський державний технічний університет», м. Дніпро, Україна
[2] Національний технічний університет «Дніпровська політехніка», м. Дніпро, Україна

ПОРІВНЯЛЬНА ОЦІНКА СТРАТЕГІЙ НАВЧАННЯ МОДЕЛЕЙ ТРАНСФОРМЕРІВ У ПУБЛІЧНИХ ХМАРНИХ ПЛАТФОРМАХ

Левицька Т. О., Копійка О. С., Грищак Д. Д. Порівняльна оцінка стратегій навчання моделей трансформерів у публічних хмарних платформах. У статті представлено всебічну порівняльну оцінку сучасних стратегій навчання моделей на основі архітектури Трансформер, зокрема Великих Мовних Моделей, на провідних публічних хмарних платформах: Amazon Web Services, Google Cloud Platform та Microsoft Azure. Дослідження систематизує та аналізує ключові технічні, економічні, екологічні та пов'язані з масштабованістю виклики, що постають перед розробниками та дослідниками. Розглянуто основні архітектури LLM, такі як BERT, GPT, LLaMA та Falcon, і детально проаналізовано методології навчання: повне попереднє навчання, тонке налаштування та параметро-ефективні методи, включно з LoRA та QLoRA. У роботі наведено порівняльний аналіз специфічних для хмарних провайдерів обчислювальних ресурсів (GPU A100/H100, TPU, AWS Trainium), інструментів MLOps, мережевих рішень та моделей ціноутворення. На основі синтезу емпіричних даних та бенчмарків сформульовано науково обґрунтовані рекомендації щодо вибору оптимальних стратегій навчання та хмарних конфігурацій залежно від сценаріїв використання та ресурсних обмежень. Робота спрямована на надання практичних знань для фахівців та визначення напрямків для майбутніх досліджень у сфері оптимізації навчання LLM.
Ключові слова: LLM, Трансформер, хмарні обчислення, навчання моделей, тонке налаштування, PEFT, LoRA, TPU

Levytska T., Kopiika O., Hryshchak D. Comparative evaluation of training strategies for transformer models on public cloud platforms. This paper presents a comprehensive comparative evaluation of modern training strategies for Transformer-based models, particularly Large Language Models, on leading public cloud platforms: Amazon Web Services, Google Cloud Platform, and Microsoft Azure. The study systematizes and analyzes the key technical, economic, environmental, and scalability challenges facing developers and researchers. It examines major LLM architectures such as BERT, GPT, LLaMA, and Falcon, and provides a detailed analysis of training methodologies: full pretraining, fine-tuning, and Parameter-Efficient Fine-Tuning methods, including LoRA and QLoRA. The paper features a comparative analysis of cloud-provider-specific computational resources (GPU A100/H100, TPU, AWS Trainium), MLOps tools, networking solutions, and pricing models. Based on a synthesis of empirical data and benchmarks, scientifically-grounded recommendations are formulated for selecting optimal training strategies and cloud configurations depending on use-case scenarios and resource constraints. The work aims to provide actionable insights for practitioners and identify directions for future research in optimizing LLM training.
Keywords: LLM, Transformer, cloud computing, model training, fine-tuning, PEFT, LoRA, TPU

Постановка проблеми. Архітектура Трансформер та похідні від неї Великі Мовні Моделі (LLM), такі як серії GPT, LLaMA та Falcon, стали де-факто стандартом у галузі обробки природної мови та генеративного штучного інтелекту. Їхня здатність вирішувати широкий спектр завдань, від перекладу до написання коду, спричинила технологічну революцію в багатьох галузях [1].

Однак цей прогрес супроводжується «гонкою озброєнь» за розміром моделей, що створює фундаментальні виклики. Технічно, тренування моделей, що налічують мільярди параметрів, вимагає величезних обсягів пам'яті (VRAM) та обчислювальної потужності, доступних лише на найдорожчих прискорювачах, як-от NVIDIA H100 [12]. Це, у свою чергу, породжує колосальні економічні бар'єри: вартість повного попереднього навчання моделей класу GPT-3 оцінюється в мільйони доларів [3], а вартість підготовки якісних навчальних даних може бути на порядок вищою. Одночасно виникають проблеми масштабованості: ефективне розподілене навчання на сотнях і тисячах GPU є складною інженерною задачею, що вимагає глибокої оптимізації інфраструктури та програмного забезпечення [10]. Нарешті, не можна ігнорувати значний екологічний вплив, пов'язаний з енергоспоживанням та вуглецевим слідом таких обчислень [7].

У відповідь на ці виклики академічна та індустріальна спільноти розробили параметро-ефективні методи тонкого налаштування (Parameter-Efficient Fine-Tuning, PEFT), зокрема LoRA (Low-Rank Adaptation) та його квантизовану версію QLoRA [6, 8]. Ці підходи дозволяють

адаптувати попередньо навчені LLM до нових завдань зі значно меншими обчислювальними витратами порівняно з повним тонким налаштуванням

Таким чином, сьогодні перед дослідниками та компаніями стоїть складний вибір між різними стратегіями навчання (повне навчання, тонке налаштування, PEFT) та їхньою реалізацією на публічних хмарних платформах (AWS, GCP, Azure), які пропонують різноманітні конфігурації апаратного забезпечення (GPU, TPU) та моделей ціноутворення. Проте, попри доступність цих технологій, бракує систематизованого порівняльного аналізу, який би дозволив об'єктивно оцінити компроміси між вартістю, продуктивністю, часом навчання та складністю імплементації для кожної комбінації "стратегія-платформа". Ця прогалина у знаннях ускладнює прийняття обґрунтованих архітектурних та фінансових рішень.

Огляд та принципи роботи стратегій навчання. Вибір стратегії навчання або адаптації попередньо навченої моделі є визначальним фактором, що впливає на вартість, час та кінцеву якість проєкту. Існує спектр підходів, що варіюються від повномасштабного втручання в усі параметри моделі до мінімалістичних, хірургічно точних змін. У таблиці 1 наведено порівняння основних стратегій.

Повне тонке налаштування (Full Fine-Tuning, FFT) - це класичний підхід, що став стандартом у парадигмі "pre-train & fine-tune" [5]. Процес полягає у завантаженні ваг попередньо навченої моделі та продовженні її тренування на новому, специфічному для завдання наборі даних, оновлюючи абсолютно всі параметри моделі. Цей метод дозволяє досягти потенційно найвищої точності, оскільки модель має повну свободу для адаптації своїх внутрішніх представлень. Однак це досягається ціною надзвичайно високих вимог до ресурсів VRAM, оскільки в пам'яті потрібно зберігати повні ваги, градієнти та стани оптимізатора. Крім того, результатом кожного налаштування є нова повнорозмірна копія моделі, що створює значні проблеми зі зберіганням

Таблиця 1 – Порівняльна характеристика стратегій навчання

Критерій порівняння	Повне тонке налаштування (FFT)	LoRA	QLoRA
Кількість параметрів для тренування	100% (всі параметри моделі)	~0.01% - 1% від загальної кількості	~0.01% - 1% (аналогічно LoRA)
Вимоги до VRAM (пам'яті GPU)	Дуже високі (сотні ГБ)	Середні / Низькі (десятки ГБ)	Дуже низькі (<24 ГБ для великих моделей)
Розмір артефакту на виході	Повнорозмірна копія моделі (ГБ)	Лише ваги адаптера (МБ)	Лише ваги адаптера (МБ)
Вплив на швидкість інференсу	Базова (без додаткових затримок)	Відсутній (після злиття ваг)	Незначний (можлива затримка на деквантизацію)
Потенційна точність	Максимально можлива (еталон)	Висока, дуже близька до еталону	Висока, з мінімальними втратами порівняно з LoRA
Основний сценарій використання	Критичні завдання з найвищими вимогами до якості; навчання з нуля.	Більшість завдань з адаптації; оптимальний баланс якості та вартості.	Навчання на обмежених ресурсах; швидкі експерименти; демократизація доступу

LoRA (Low-Rank Adaptation) - це найвпливовіший параметро-ефективний метод (PEFT), розроблений для розв'язання проблем FFT [8]. Його ідея полягає в тому, щоб "заморозити" мільярди ваг оригінальної моделі й додати до деяких її шарів невеликі, низькорангові "адаптивні" матриці. Під час налаштування тренуються лише ці нові матриці, що становить мізерну частку від загальної кількості параметрів. Це різко скорочує обчислювальні вимоги, дозволяючи проводити налаштування на значно менш потужному обладнанні. Важливо, що для кожного нового завдання зберігаються лише невеликі файли адаптерів (кілька мегабайтів), а ваги адаптерів можна математично злити з основними перед розгортанням, уникаючи затримок при інференсі.

QLoRA (Quantized Low-Rank Adaptation) є подальшою, ще більш агресивною оптимізацією LoRA, що робить процес налаштування максимально доступним [6]. Цей метод поєднує LoRA з квантуванням: базова "заморожена" модель стискається шляхом переведення її ваг зі стандартного 16-бітного у значно ефективніший 4-бітний формат (NF4). Тренування адаптерів LoRA відбувається поверх цієї квантованої моделі. Завдяки цьому QLoRA досягає екстремальної ефективності використання пам'яті, що дозволяє налаштовувати величезні моделі на одному споживчому GPU. При цьому, завдяки застосуванню спеціальних технік, якість фінальної моделі майже не поступається результатам 16-бітного налаштування LoRA.

Огляд хмарних платформ та їхніх AI/ML сервісів. Навчання та розгортання великих мовних моделей у промислових масштабах практично неможливе без інфраструктури провідних постачальників хмарних послуг. Amazon Web Services (AWS), Google Cloud Platform (GCP) та Microsoft Azure пропонують потужні екосистеми для машинного навчання, кожна з яких має свої унікальні переваги, архітектурні підходи та апаратні пропозиції, що робить вибір між ними нетривіальним завданням.

Amazon Web Services як лідер ринку пропонує найбільш зрілу та всеосяжну екосистему. Її центральним сервісом є Amazon SageMaker - повністю керована платформа, що охоплює весь життєвий цикл МЛ-проєктів. Для обчислень AWS надає широкий вибір інстансів на базі GPU NVIDIA, включно з A100 (інстанси P4) та H100 (інстанси P5). Ключовою стратегією AWS є розвиток власних прискорювачів - AWS Trainium для навчання та AWS Inferentia для інференсу, що спрямовано на оптимізацію співвідношення вартості та продуктивності. Платформа також спрощує розподілене навчання за допомогою власних бібліотек та пропонує хаб готових моделей через сервіс SageMaker JumpStart.

Google Cloud Platform історично позиціонується як платформа, народжена з інновацій Google у галузі ШІ. Її сервіс Vertex AI тісно інтегрований з іншими продуктами для обробки даних, як-от BigQuery. Поряд зі стандартними GPU від NVIDIA, головною конкурентною перевагою GCP є власні прискорювачі - Tensor Processing Units (TPU). Це спеціалізовані чипи, розроблені для глибокого навчання, які об'єднуються у великі кластери (Pods) за допомогою надшвидкісної власної мережі. Така архітектура робить TPU ідеальним рішенням для масивного, паралельного навчання найбільших мовних моделей, особливо при використанні фреймворку JAX.

Microsoft Azure та її платформа Azure Machine Learning (Azure ML) вирізняються сильною орієнтацією на корпоративний сектор та унікальним стратегічним партнерством з OpenAI. Це надає користувачам Azure ексклюзивний доступ до найсучасніших моделей, як-от GPT-4, через керований та безпечний сервіс Azure OpenAI. Для обчислень Azure використовує кластери на базі GPU NVIDIA, з'єднані високопродуктивною мережею InfiniBand, що є критично важливим для ефективного розподіленого навчання. Платформа також пропонує найкращу інтеграцію з бібліотекою DeepSpeed від Microsoft Research, що є ще однією її важливою перевагою.

Аналіз економічної ефективності та енергоспоживання. Економічні бар'єри є однією з головних перешкод для широкого застосування та дослідження великих мовних моделей. Оцінка фінансової ефективності є комплексним завданням, оскільки загальна вартість навчання залежить не лише від погодинної ціни на обчислювальні ресурси, а й від швидкості досягнення необхідної якості моделі та архітектури ціноутворення хмарних провайдерів. Загальні витрати на навчання формуються не лише з прямих обчислювальних витрат, які є найбільшою статтею, але й з вартості зберігання терабайтів даних та моделей, а також з потенційно значної вартості мережевого трафіку. Хмарні платформи пропонують гнучкі моделі ціноутворення, такі як спотові інстанси, що можуть значно скоротити витрати для відмовостійких завдань, проте вимагають ретельного архітектурного планування [14].

Вибір апаратного прискорювача безпосередньо впливає на загальний бюджет. Незалежні дослідження, зокрема від Databricks (колишня MosaicML), показали, що новітні покоління GPU, як-от NVIDIA H100, попри вищу погодинну вартість, часто виявляються більш економічно ефективними для тренування великих моделей порівняно з A100. Це досягається завдяки значно вищій пропускній здатності та ефективності, що скорочує загальний час навчання настільки, що нівелює різницю в ціні [4]. З іншого боку, екосистема Google TPU пропонує конкурентну економіку для завдань масивного масштабу, де їхня спеціалізована архітектура та високошвидкісна мережа забезпечують майже лінійне масштабування продуктивності, що може призвести до нижчої загальної вартості при навчанні моделей з нуля [9]. Власні чипи, як-от AWS Trainium, також

розробляються з метою покращення співвідношення ціни та продуктивності порівняно з універсальними GPU.

Однак найбільш фундаментальний вплив на економіку проєкту має вибір стратегії навчання. Повне тонке налаштування (FFT) за замовчуванням є найдорожчим, оскільки вимагає оренди найпотужніших конфігурацій на тривалий час. На противагу цьому, параметро-ефективні методи кардинально змінюють ситуацію. LoRA дозволяє досягти значної економії, уможливлюючи використання менш потужних інстансів [8]. Метод QLoRA йде ще далі, знижуючи вимоги до пам'яті до рівня, що дозволяє проводити налаштування на відносно дешевих споживчих або низькорівневих хмарних GPU, що робить кінцеву вартість адаптації моделі на порядки нижчою порівняно з FFT [6].

Наведена нижче таблиця 2 узагальнює порівняльний аналіз економічної ефективності різних підходів, базуючись на синтезі даних з технічних звітів та документації провайдерів. Важливо зазначити, що замість абсолютних цін використовуються відносні якісні оцінки. Це зроблено свідомо, оскільки точні ціни є вкрай волатильними. Проте на момент червня 2025 року погодинна вартість On-Demand інстансу з одним GPU NVIDIA T4, що відповідає оцінці "Дуже низька" становила приблизно $0.5-$1, тоді як вартість інстансу з кластером 8xH100 (оцінка "Найвища") могла перевищувати $70-$100.

Таблиця 2 - Порівняльний аналіз економічної ефективності стратегій навчання

Стратегія	Апаратний ресурс	Відносна вартість	Ключовий фактор та обґрунтування
Повне тонке налаштування	Кластер H100/A100	Найвища	Потреба в оренді найдорожчих інстансів на тривалий час для оновлення мільярдів параметрів.
Повне тонке налаштування	Google TPU Pod	Висока	Оптимізовано для навчання з нуля; висока ефективність при масивному масштабуванні може знизити загальну вартість.
LoRA	Один або кілька A100/H100	Середня	Значно менший час навчання та можливість використання менш потужних інстансів порівняно з FFT.
LoRA	Спотовий інстанс GPU	Низька	Оптимальний баланс ціни та якості для більшості завдань адаптації; потребує відмовостійкості.
QLoRA	Недорогий GPU (T4/L4)	Дуже низька	Мінімальні вимоги до VRAM дозволяють використовувати найдешевші доступні прискорювачі.
QLoRA	Спотовий інстанс T4/L4	Найнижча	Ультимативна економія для експериментів та завдань, що не потребують найвищої точності.

Аналіз продуктивності та масштабованості. Продуктивність систем для навчання LLM вимірюється двома основними показниками: пропускною здатністю (throughput), тобто кількістю даних, наприклад, токенів, які система може обробити за одиницю часу, та часом до отримання результату (time-to-train) - загальним часом, необхідним для досягнення цільової якості моделі. При переході від навчання на одному вузлі до розподілених систем, що налічують сотні чи тисячі прискорювачів, ці показники починають критично залежати від ефективності комунікації між вузлами та швидкості доступу до даних. Саме тому аналіз мережевих технологій та рішень для зберігання даних є ключовим для розуміння реальної продуктивності хмарних платформ.

Ефективність розподіленого навчання прямо залежить від швидкості, з якою вузли можуть обмінюватися даними, такими як градієнти або ваги моделі. У випадку повільної мережі, потужні прискорювачі простоюватимуть в очікуванні даних, що створює "комунікаційний вузол" (communication bottleneck) і нівелює переваги від додавання нових ресурсів. Кожен хмарний провайдер пропонує власне рішення цієї проблеми. Microsoft Azure у своїх кластерах віртуальних

машин серії ND використовує технологію InfiniBand, що є стандартом у галузі високопродуктивних обчислень (HPC). Вона забезпечує наднизьку затримку та високу пропускну здатність, що є критичним для синхронізації градієнтів у таких фреймворках, як DeepSpeed. Amazon Web Services пропонує власну технологію Elastic Fabric Adapter (EFA), яка дозволяє обходити ядро операційної системи, надаючи додаткам прямий доступ до мережевого обладнання. Це значно знижує затримку і є стандартною практикою для побудови великих тренувальних кластерів на базі інстансів P4/P5. Однак найбільш інтегрований підхід демонструє Google Cloud Platform зі своїми TPU. Архітектура TPU Pods включає в себе власну, спеціалізовану оптичну мережу з топологією 2D-тороуса, яка забезпечує колосальну пропускну здатність для зв'язку між чипами. Саме ця тісна інтеграція обчислювальних ресурсів та мережі є причиною, чому TPU демонструють майже лінійне масштабування продуктивності для завдань навчання LLM [9].

Не менш важливим є вплив системи зберігання даних. Процес навчання висуває дві основні вимоги до сховища: швидке зчитування навчального датасету та швидке збереження контрольних точок (checkpoints). Якщо швидкість подачі даних з диска є нижчою, ніж швидкість їх обробки прискорювачами, виникає "вузол вводу-виводу" (I/O bottleneck). Стандартні об'єктні сховища, такі як AWS S3, Google Cloud Storage та Azure Blob Storage, є ідеальними для довготривалого та економічного зберігання терабайтів даних, але їхня пропускна здатність може бути недостатньою для прямої подачі даних у великий тренувальний кластер. Для вирішення цієї проблеми провайдери пропонують високопродуктивні паралельні файлові системи, наприклад, Amazon FSx for Lustre або Azure NetApp Files. Загальноприйнятою архітектурною практикою є зберігання основного датасету в об'єктному сховищі, звідки він копіюється або стріміться на тимчасову паралельну файлову систему, яка вже підключена до всіх вузлів тренувального кластера. Такий підхід забезпечує як економічність, так і максимальну швидкість доступу до даних під час навчання [2].

Отже, продуктивність та масштабованість у хмарі є результатом синергії трьох компонентів: обчислювальних ресурсів, мережі та системи зберігання. Теоретична пікова продуктивність GPU або TPU не має сенсу без відповідної інфраструктурної підтримки. Підхід GCP з TPU пропонує найбільш цілісну та інтегровану систему, що демонструє виняткові результати при масивному навчанні. AWS та Azure надають більшу гнучкість у побудові гетерогенних кластерів на базі GPU, використовуючи передові мережеві стандарти, однак досягнення максимальної продуктивності на цих платформах вимагає від користувача більш ретельного проєктування архітектури системи зберігання даних.

Аналіз операційної складності та інтеграції MLOps. Окрім економічних та технічних показників продуктивності, вирішальний вплив на успіх та швидкість реалізації проєкту має операційна складність - практичні аспекти роботи з платформою, що включають легкість налаштування, гнучкість конфігурації та зрілість інструментів для керування життєвим циклом моделей (MLOps). Цей розділ аналізує компроміси, які пропонують хмарні платформи між високорівневою абстракцією та низькорівневим контролем.

Використання високорівневих керованих сервісів (PaaS - Platform-as-a-Service), таких як SageMaker, Vertex AI та Azure ML. Вони абстрагують значну частину інфраструктурної складності, надаючи готові контейнери, API для запуску тренувальних завдань та інтегровані середовища розробки. Це дозволяє командам швидше розпочинати роботу, фокусуючись на моделі, а не на адмініструванні серверів. Другий підхід - це побудова власного середовища на базі базових обчислювальних ресурсів (IaaS - Infrastructure-as-a-Service), як-от AWS EC2, Google Compute Engine або Azure VMs. Цей шлях надає максимальну гнучкість та контроль, але вимагає від команди значно глибшої експертизи в галузі DevOps та адміністрування розподілених систем.

AWS SageMaker традиційно вважається платформою, що пропонує найвищий рівень абстракції та найнижчий поріг входження. Її керовані бібліотеки для розподіленого навчання значно спрощують запуск складних тренувальних завдань, приховуючи від користувача деталі імплементації фреймворків, як-от PyTorch FSDP. Це є перевагою для команд, що прагнуть отримати результат з мінімальними інфраструктурними зусиллями, проте іноді може ускладнювати глибоке налагодження (debugging) або використання нестандартних конфігурацій.

Google Cloud Platform та її Vertex AI пропонують збалансований підхід. Платформа надає потужні керовані сервіси, але водночас зберігає філософію відкритості та гнучкості, що проявляється у глибокій інтеграції з відкритими стандартами, зокрема Kubernetes (через Google Kubernetes Engine) та Kubeflow. Це робить її привабливою для команд, що вже мають досвід роботи

з цими інструментами. Водночас, екосистема TPU, попри свою потужність, має вищий поріг входження і вимагає від розробників адаптації до специфіки апаратної архітектури та фреймворку JAX.

Microsoft Azure ML демонструє свої найсильніші сторони в корпоративному сегменті. Платформа забезпечує безшовну інтеграцію з усталеними практиками DevOps та інструментами, такими як Azure DevOps та GitHub Actions. Це дозволяє організаціям вбудовувати процеси навчання та розгортання моделей у вже існуючі CI/CD-пайплайни для розробки програмного забезпечення. Така глибока інтеграція є значною перевагою для великих компаній, що прагнуть уніфікувати свої робочі процеси [11].

Зрілість MLOps-інструментів є ще одним критичним фактором. Усі три платформи пропонують повний набір сервісів для управління життєвим циклом моделей, включно з реєстрами моделей, інструментами для відстеження експериментів та системами для оркестрації пайплайнів. Ключові відмінності полягають у філософії: AWS пропонує цілісну, дещо закриту екосистему "все в одному". GCP робить ставку на відкриті стандарти, як-от Kubeflow, що забезпечує кращу портативність. Azure, як зазначалося, фокусується на інтеграції з існуючими інструментами розробки. Вибір між цими підходами часто залежить від культури та технологічного стеку конкретної організації [13].

Таким чином, відповідь на питання про операційну складність не є однозначною. Не існує єдиної "найпростішої" чи "найгнучкішої" платформи. Вибір є компромісом, що залежить від наявних навичок команди та організаційного контексту. AWS SageMaker може бути оптимальним для швидкого старту та команд, що хочуть абстрагуватися від інфраструктури. GCP Vertex AI приваблює своєю гнучкістю та потужністю для тих, хто готовий інвестувати час у її освоєння. Azure ML є логічним вибором для великих підприємств, де ключовою вимогою є інтеграція МЛ-процесів у загальну інженерну культуру.

Висновки. У даній дослідницькій роботі було проведено комплексний порівняльний аналіз стратегій навчання великих мовних моделей на базі архітектури Трансформер на трьох провідних публічних хмарних платформах: Amazon Web Services (AWS), Google Cloud Platform (GCP) та Microsoft Azure. Метою дослідження було систематизувати знання про сучасні підходи та інфраструктурні рішення, а також надати науково обґрунтовані відповіді на ключові питання щодо економічної ефективності, продуктивності та операційної складності, що постають перед дослідниками та інженерами.

Проведений аналіз дозволяє сформулювати вичерпні відповіді на поставлені дослідницькі питання. По-перше, дослідження показало, що вартість навчання LLM є нелінійною функцією, де найдорожчий погодинний ресурс не завжди призводить до найвищих загальних витрат. Для масштабних завдань навчання з нуля новітні прискорювачі, як-от NVIDIA H100 або Google TPU, завдяки своїй продуктивності можуть скоротити час навчання настільки, що стають економічно вигіднішими за дешевші альтернативи. Однак фундаментальний зсув в економіці прикладних завдань стався завдяки параметро-ефективним методам: QLoRA, зокрема, дозволяє знизити вартість адаптації моделей на порядки, уможливлюючи використання найдешевших спотових інстансів і роблячи технологію доступною для проєктів з мінімальним бюджетом. По-друге, було встановлено, що теоретична пікова продуктивність обчислювальних ресурсів реалізується лише за наявності відповідної мережевої та дискової інфраструктури. Архітектура Google TPU, що поєднує обчислювальні чипи та власну високошвидкісну мережу в єдине ціле, демонструє найкращу масштабованість для масивних завдань. AWS та Azure, використовуючи передові HPC-технології, такі як EFA та InfiniBand, пропонують гнучкі та потужні кластери на базі GPU, проте досягнення максимальної пропускної здатності на цих платформах вимагає від користувача ретельного проєктування архітектури системи зберігання даних. По-третє, аналіз показав, що вибір платформи значною мірою залежить від наявних навичок команди та організаційного контексту. AWS SageMaker пропонує найвищий рівень абстракції та найнижчий поріг входження для швидкого старту. GCP Vertex AI надає потужну та гнучку платформу, орієнтовану на відкриті стандарти, що є перевагою для команд з досвідом у Kubernetes. Microsoft Azure ML, у свою чергу, є оптимальним вибором для корпоративного сектору завдяки безшовній інтеграції з існуючими DevOps-процесами та унікальному доступу до моделей OpenAI.

Ключовий висновок роботи полягає в тому, що не існує єдиної найкращої стратегії чи платформи. Оптимальне рішення завжди є результатом розв'язання багатовимірної задачі

оптимізації в межах компромісів між вартістю, продуктивністю та операційними можливостями. Вибір конкретної моделі, методу її налаштування та хмарної інфраструктури має базуватися на ретельному аналізі специфічних вимог проєкту, його бюджету та компетенцій команди.

Водночас варто визнати певні обмеження даного дослідження. По-перше, сфера LLM та хмарних технологій розвивається надзвичайно динамічно, і будь-які конкретні порівняння апаратного забезпечення чи цін є лише знімком у часі. По-друге, аналіз базувався на синтезі публічно доступних даних, технічної документації та існуючих бенчмарків, а не на проведенні власних повномасштабних порівняльних тренувань, що було б неможливо через надмірні фінансові витрати. Деякі аспекти, як-от операційна складність, мають частково якісний характер.

Результати роботи відкривають кілька напрямків для майбутніх досліджень. Нагальною є потреба у створенні стандартизованих, відкритих та відтворюваних бенчмарків для крос-хмарного порівняння продуктивності та вартості навчання й адаптації LLM. Необхідні подальші незалежні дослідження економічної ефективності та продуктивності спеціалізованих прискорювачів, таких як TPU та Trainium. Нарешті, майбутні порівняльні аналізи повинні включати не лише вартість і швидкість, а й енергоефективність як першочерговий показник, що дозволить оцінювати технології також з точки зору їхнього екологічного впливу.

Список бібліографічного опису

1. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. In Advances in Neural Information Processing Systems (Vol. 30, pp. 5998–6008). https://doi.org/10.48550/arXiv.1706.03762
2. Amazon Web Services. (2022, March 15). Amazon FSx for Lustre: Best practices guide. Retrieved from https://docs.aws.amazon.com/fsx/latest/LustreGuide/
3. Brown, T. B. et al. (2020). Language Models are Few-Shot Learners. arXiv:2005.14165. https://doi.org/10.48550/arXiv.2005.14165
4. Databricks. (2023, June 30). Benchmarking large language models on NVIDIA H100 GPUs with CoreWeave (Part 1). Databricks Engineering Blog. Retrieved from https://www.databricks.com/blog/coreweave-nvidia-h100-part-1
5. Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv Preprint, arXiv:1810.04805. https://doi.org/10.48550/arXiv.1810.04805
6. Dettmers, T., Pagnoni, A., Holtzman, A., & Zettlemoyer, L. (2023). QLoRA: Efficient finetuning of quantized large language models. arXiv Preprint, arXiv:2305.14314. https://doi.org/10.48550/arXiv.2305.14314
7. Henderson, P., Hu, J., Romoff, J., Brunskill, E., Jurafsky, D., & Pineau, J. (2020). Towards the systematic reporting of the energy and carbon footprints of machine learning. Journal of Machine Learning Research, 21(248), 1–43. https://doi.org/10.48550/arXiv.2002.05651
8. Hu, E. J., Shen, Y., Wallis, P., Allen, D., & Wang, L. (2021). LoRA: Low-rank adaptation of large language models. arXiv Preprint, arXiv:2106.09685. https://doi.org/10.48550/arXiv.2106.09685
9. Jouppi, N. P., Kurian, G., Li, S., Ma, P., Nagarajan, R., Nai, L., … Patterson, D. (2023). TPU v4: An optically reconfigurable supercomputer for machine learning with hardware support for embeddings. In Proceedings of the 50th Annual International Symposium on Computer Architecture (Article 82, pp. 1–14). https://doi.org/10.1145/3579371.3589350
10. Le Scao, T., Fan, A., Akiki, C., et al. (2022). BLOOM: A 176B-parameter open-access multilingual language model. arXiv Preprint, arXiv:2211.05100. https://doi.org/10.48550/arXiv.2211.05100
11. Microsoft. (2022, December 14). Use Azure DevOps for CI/CD with Azure Machine Learning. Microsoft Learn. Retrieved from https://learn.microsoft.com/azure/machine-learning/how-to-devops-machine-learning
12. MosaicML. (2023, July 11). Mosaic LLMs: GPT-3 quality for < $500k. MosaicML Blog. Retrieved from https://www.mosaicml.com/blog/gpt-3-quality-for-500k
13. Paleyes, A., Urma, R.-G., & Lawrence, N. D. (2022). Challenges in deploying machine learning: A survey of case studies. ACM Computing Surveys, 55(6), Article 114. https://doi.org/10.1145/3533378
14. Peven, B. (2020, April 16). Cost optimize your Jenkins CI/CD pipelines using EC2 Spot Instances. AWS Compute Blog. Retrieved from https://aws.amazon.com/blogs/compute/cost-optimize-your-jenkins-ci-cd-pipelines-using-ec2-spot-instances/

References

1. Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A. N., Kaiser L., Polosukhin I. Attention is all you need. *Advances in Neural Information Processing Systems.* 2017. Vol. 30. P. 5998–6008. URL: https://doi.org/10.48550/arXiv.1706.03762.
2. Amazon Web Services. Amazon FSx for Lustre: Best practices guide. URL: https://docs.aws.amazon.com/fsx/latest/LustreGuide/.
3. Brown T. B. et al. Language Models are Few-Shot Learners. *arXiv.* 2020. arXiv:2005.14165. URL: https://doi.org/10.48550/arXiv.2005.14165.
4. Databricks. Benchmarking large language models on NVIDIA H100 GPUs with CoreWeave (Part 1). *Databricks Engineering Blog.* URL: https://www.databricks.com/blog/coreweave-nvidia-h100-part-1.
5. Devlin J., Chang M.-W., Lee K., Toutanova K. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv.* arXiv:1810.04805. URL: https://doi.org/10.48550/arXiv.1810.04805.

6. Dettmers T., Pagnoni A., Holtzman A., Zettlemoyer L. QLoRA: Efficient finetuning of quantized large language models. *arXiv.* arXiv:2305.14314. URL: https://doi.org/10.48550/arXiv.2305.14314.
7. Henderson P., Hu J., Romoff J., Brunskill E., Jurafsky D., Pineau J. Towards the systematic reporting of the energy and carbon footprints of machine learning. *Journal of Machine Learning Research.* Vol. 21, no. 248. P. 1–43. URL: https://doi.org/10.48550/arXiv.2002.05651.
8. Hu E. J., Shen Y., Wallis P., Allen D., Wang L. LoRA: Low-rank adaptation of large language models. *arXiv.* arXiv:2106.09685. URL: https://doi.org/10.48550/arXiv.2106.09685.
9. Jouppi N. P., Kurian G., Li S., Ma P., Nagarajan R., Nai L., … Patterson D. TPU v4: An optically reconfigurable supercomputer for machine learning with hardware support for embeddings. *Proceedings of the 50th Annual International Symposium on Computer Architecture.* Article 82. P. 1–14. URL: https://doi.org/10.1145/3579371.3589350.
10. Le Scao T., Fan A., Akiki C., et al. BLOOM: A 176B-parameter open-access multilingual language model. *arXiv.* arXiv:2211.05100. URL: https://doi.org/10.48550/arXiv.2211.05100.
11. Microsoft. Use Azure DevOps for CI/CD with Azure Machine Learning. *Microsoft Learn.* URL: https://learn.microsoft.com/azure/machine-learning/how-to-devops-machine-learning.
12. MosaicML. Mosaic LLMs: GPT-3 quality for < $500k. *MosaicML Blog.* URL: https://www.mosaicml.com/blog/gpt-3-quality-for-500k.
13. Paleyes A., Urma R.-G., Lawrence N. D. Challenges in deploying machine learning: A survey of case studies. *ACM Computing Surveys.* Vol. 55, no. 6. Article 114. URL: https://doi.org/10.1145/3533378.
14. Peven B. Cost optimize your Jenkins CI/CD pipelines using EC2 Spot Instances. *AWS Compute Blog.* URL: https://aws.amazon.com/blogs/compute/cost-optimize-your-jenkins-ci-cd-pipelines-using-ec2-spot-instances/.