**BÁO CÁO KỸ THUẬT VÀ CHIẾN LƯỢC: TỐI ƯU HÓA HUẤN LUYỆN AI TRONG MÔI TRƯỜNG TÀI NGUYÊN GIỚI HẠN**

Việc huấn luyện các mô hình Trí tuệ Nhân tạo (AI) hiện đại, đặc biệt là các mô hình Học Sâu (Deep Learning), thường đòi hỏi tài nguyên tính toán và bộ nhớ GPU (VRAM) khổng lồ, vượt xa khả năng của các máy tính cá nhân có cấu hình yếu. Báo cáo này trình bày các chiến lược tối ưu hóa kỹ thuật cốt lõi (algorithmic optimization) và các phương pháp huấn luyện trực tuyến hiệu quả về chi phí (Cloud Training), cùng với quy trình quản lý và truy xuất dữ liệu mô hình một cách an toàn và có hệ thống.

**PHẦN I: TỐI ƯU HÓA KỸ THUẬT NỘI TẠI (ALGORITHMIC OPTIMIZATION)**

Trước khi chuyển sang nền tảng đám mây, việc can thiệp vào các tham số và thuật toán huấn luyện là bước đầu tiên và quan trọng nhất để giảm thiểu nhu cầu tài nguyên, đặc biệt là Bộ nhớ GPU (VRAM).

**1.1. Phân tích Các Giới hạn Tài nguyên Cốt lõi**

Giới hạn lớn nhất đối với người dùng có máy yếu khi huấn luyện AI là Bộ nhớ GPU (VRAM).1 VRAM xác định kích thước lô (Batch Size) tối đa mà mô hình có thể xử lý đồng thời. Khi Batch Size bị giới hạn ở mức quá nhỏ do thiếu VRAM, quá trình tính toán Gradient sẽ trở nên nhiễu loạn (noisy), có khả năng làm chậm quá trình hội tụ hoặc làm giảm chất lượng cuối cùng của mô hình. Do đó, các kỹ thuật tối ưu hóa tập trung vào việc mô phỏng hiệu ứng của Batch Size lớn mà không yêu cầu VRAM tương ứng.

Bên cạnh đó, nếu máy yếu bao gồm cả ổ cứng và I/O chậm, việc tải dữ liệu huấn luyện cục bộ sẽ trở thành nút thắt cổ chai lớn.2 Điều này củng cố chiến lược bắt buộc là phải chuyển tất cả dữ liệu training lên nền tảng Cloud Storage trước khi bắt đầu huấn luyện.

**1.2. Kỹ thuật Tích lũy Gradient (Gradient Accumulation - GA)**

Tích lũy Gradient là một cơ chế thiết yếu cho người dùng tài nguyên giới hạn. Kỹ thuật này cho phép mô hình mô phỏng việc sử dụng một Batch Size lớn ($B\_{simulated}$) bằng cách chia nó thành nhiều Mini-Batch nhỏ hơn ($B\_{actual}$) được xử lý tuần tự.3

Cơ chế hoạt động của GA là thực hiện truyền ngược (backpropagation) qua $N$ Mini-Batch ($N$ là tham số gradient\_accumulation\_steps) mà không cập nhật trọng số.3 Gradient của từng bước này được tích lũy lại. Sau khi tích lũy đủ $N$ Mini-Batch, một lần cập nhật trọng số duy nhất sẽ được thực hiện bằng cách sử dụng tổng Gradient đã tích lũy.

Lợi ích chính của GA là giảm đáng kể Memory Footprint (dấu chân bộ nhớ).1 Vì mô hình chỉ cần lưu trữ các kích hoạt (intermediate activations) và gradient cho một Mini-Batch nhỏ ($B\_{actual}$) tại một thời điểm, nhu cầu VRAM tổng thể được giảm thiểu. Mặc dù việc tăng số bước tích lũy ($N$) giúp giảm bộ nhớ, cần lưu ý rằng nó cũng sẽ tăng tổng số lần lặp (iterations) cần thiết để xử lý cùng một lượng dữ liệu, điều này có thể kéo dài tổng thời gian huấn luyện.1 Kỹ thuật này là biện pháp can thiệp đầu tiên cần áp dụng để vượt qua giới hạn VRAM.

**1.3. Huấn luyện Độ chính xác Hỗn hợp (Mixed Precision Training - MP)**

Huấn luyện Độ chính xác Hỗn hợp là một kỹ thuật tối ưu hóa hiệu suất và bộ nhớ bằng cách kết hợp sử dụng kiểu dữ liệu dấu phẩy động 16-bit (float16 hoặc bfloat16) với 32-bit (float32).4

Kỹ thuật này mang lại hai lợi ích chính:

1. **Tiết kiệm Bộ nhớ:** Kiểu dữ liệu 16-bit chỉ cần một nửa dung lượng so với 32-bit, do đó giảm 50% nhu cầu bộ nhớ để lưu trữ trọng số và kích hoạt.4
2. **Tăng Tốc độ:** Các bộ tăng tốc hiện đại, đặc biệt là GPU NVIDIA và TPU, có phần cứng chuyên dụng để thực hiện các phép toán 16-bit nhanh hơn đáng kể so với FP32 (có thể tăng hiệu suất lên tới $3\times$ trên các GPU mới).4

Tuy nhiên, MP giới thiệu một thách thức về độ ổn định số học (Numeric Stability). Kiểu dữ liệu 16-bit có phạm vi giá trị nhỏ hơn, có thể dẫn đến hiện tượng underflow (gradient trở nên quá nhỏ và bị làm tròn thành 0).4 Để khắc phục, cần giữ các biến quan trọng và một số phép tính (như tổng tích lũy gradient) ở định dạng FP32.4

Để đảm bảo mô hình hội tụ thành công, việc sử dụng **Kỹ thuật Gradient Scaling** là bắt buộc khi dùng FP16. Gradient Scaling giúp "mở rộng" các giá trị gradient nhỏ, ngăn chúng bị làm tròn thành 0.4 Các framework như PyTorch cung cấp tính năng Automatic Mixed Precision (AMP) với autocast() và GradScaler() để tự động hóa quy trình này một cách hiệu quả.5 Việc kết hợp Tích lũy Gradient và Mixed Precision tạo ra sự hiệp lực mạnh mẽ: MP giúp Batch Size vật lý nhỏ chạy nhanh hơn và chiếm ít VRAM hơn, trong khi GA cho phép sử dụng Batch Size logic lớn để đảm bảo chất lượng hội tụ, là công thức tối ưu cho việc sử dụng các tài nguyên Cloud giá rẻ.

**1.4. Tổng quan về Tối ưu hóa Mô hình Sau Huấn luyện (Post-Training)**

Sau khi mô hình được huấn luyện hoàn tất, cần xem xét các kỹ thuật tối ưu hóa sau huấn luyện. **Lượng tử hóa Mô hình (Model Quantization)** là quá trình làm cho các mô hình nhỏ hơn và nhanh hơn đáng kể.6 Mặc dù mục tiêu chính của nó là giảm độ trễ suy luận (inference latency), nó cho phép triển khai mô hình trên các phần cứng bị hạn chế tài nguyên (như điện thoại di động, hệ thống nhúng, hoặc trong trường hợp này, máy tính cục bộ yếu của người dùng).6 Bằng cách giảm kích thước mô hình, việc tải mô hình về máy cục bộ ở Phần IV cũng sẽ được thực hiện nhanh chóng hơn.

Các kỹ thuật tối ưu hóa được tổng hợp trong bảng sau:

Table 1: So sánh Kỹ thuật Tối ưu hóa Huấn luyện cho Tài nguyên Giới hạn

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Kỹ thuật** | **Mục tiêu Chính** | **Lợi ích về Bộ nhớ (VRAM)** | **Tác động đến Tốc độ** | **Ghi chú Kỹ thuật Quan trọng** |
| Tích lũy Gradient [1, 3] | Mô phỏng Batch Size lớn | Rất Cao (giảm Footprint) | Giảm nhẹ/Không đáng kể (do tăng Iterations) | Cần thay đổi logic cập nhật trọng số. |
| Mixed Precision (FP16/BF16) [4, 5] | Tăng tốc tính toán & Giảm kích thước dữ liệu | Cao (giảm 50% trọng số) | Tăng Cao (trên GPU/TPU hiện đại) | BẮT BUỘC sử dụng Gradient Scaling cho sự ổn định số học. |
| Lượng tử hóa Mô hình 6 | Giảm kích thước mô hình (Inference) | Rất Cao (giảm 75% hoặc hơn) | Tăng Cao (tốc độ suy luận) | Chủ yếu áp dụng sau khi huấn luyện để triển khai. |

**PHẦN II: CHIẾN LƯỢC HUẤN LUYỆN TRỰC TUYẾN (CLOUD TRAINING) CHO NGÂN SÁCH THẤP**

Với giới hạn về phần cứng cục bộ, việc chuyển sang các dịch vụ Đám mây (Cloud Services) là chiến lược hiệu quả và tiết kiệm nhất. Các dịch vụ học máy trên nền tảng đám mây (ML Cloud Services) cung cấp công cụ và hạ tầng cần thiết mà không yêu cầu đầu tư phần cứng phức tạp.7

**2.1. Đánh giá Nền tảng Cloud cho Cá nhân và Thử nghiệm**

Đối với người dùng cá nhân hoặc các dự án thử nghiệm với ngân sách thấp, việc lựa chọn nền tảng Cloud phải dựa trên sự cân bằng giữa chi phí, tính dễ sử dụng và khả năng truy cập tài nguyên mạnh mẽ.

**Google Colab**

Google Colab là dịch vụ Jupyter Notebook miễn phí, được host, cung cấp quyền truy cập không cần thiết lập (no-setup) vào các tài nguyên tính toán như GPU và TPU.8 Đây là lựa chọn tiên quyết cho giai đoạn thử nghiệm ban đầu do đường cong học tập dễ dàng và chi phí đầu vào thấp nhất.9

Colab cung cấp nhiều tùy chọn tính phí linh hoạt:

* **Colab Free:** Miễn phí, nhưng tài nguyên và thời gian chạy không được đảm bảo.
* **Colab Pro ($9.99/tháng):** Cung cấp các GPU nhanh hơn và 100 đơn vị điện toán (Compute Units) được đảm bảo hàng tháng.10
* **Pay As You Go:** Cho phép người dùng mua các gói đơn vị điện toán ($9.99/100 đơn vị) để chỉ thanh toán cho những gì được sử dụng, rất phù hợp nếu khối lượng công việc không liên tục.11

**Nền tảng ML Cloud Cấp Doanh nghiệp**

Các nền tảng như AWS SageMaker, Google Vertex AI, và Azure Machine Learning cung cấp môi trường tự động hóa cho toàn bộ vòng đời ML (MLOps).7

* **AWS SageMaker:** Được coi là bộ công cụ toàn diện (Swiss Army Knife of ML), hỗ trợ từ chuẩn bị dữ liệu, huấn luyện, đến triển khai.8
* **Google Vertex AI:** Tích hợp sâu vào hệ sinh thái Google Cloud, cung cấp các công cụ nâng cao như Pipelines và tích hợp TensorBoard.7

Mặc dù mạnh mẽ, các nền tảng này phù hợp hơn cho các quy trình triển khai sản phẩm (Production).9 Đối với việc thử nghiệm ban đầu và ngân sách thấp, chiến lược kết hợp giữa Colab (cho tính toán giá rẻ) và các công cụ MLOps độc lập (để quản lý thí nghiệm) thường mang lại giá trị cao nhất.

Table 2: Đánh giá Chiến lược Các Nền tảng Huấn luyện AI trên Đám mây

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nền tảng** | **Đặc điểm Cốt lõi** | **Phù hợp cho User Persona (Máy yếu/Chi phí thấp)** | **Mô hình Chi phí** | **Hạn chế Chính** |
| Google Colab Pro/Pay As You Go 10 | Môi trường Notebook được host sẵn, GPU/TPU sẵn có. | **TỐI ƯU (Thử nghiệm & Ngân sách thấp).** | Đăng ký hoặc Trả theo Đơn vị Điện toán (Compute Units). | Tài nguyên không được đảm bảo tuyệt đối (trừ Colab Enterprise). |
| Google Vertex AI 7 | Nền tảng MLOps toàn diện, tích hợp GCP. | Tốt (Nếu cần triển khai Production). | Pay-As-You-Go theo Instance/Dịch vụ. | Đường cong học tập dốc, chi phí có thể cao. |
| AWS SageMaker 8 | Bộ công cụ ML End-to-End, lý tưởng cho Doanh nghiệp. | Tốt (Nếu đã ở trong hệ sinh thái AWS). | Pay-As-You-Go theo Instance/Dịch vụ. | Phức tạp hơn cho người dùng cá nhân/dự án nhỏ. |

**2.2. Xây dựng Quy trình Dữ liệu Cloud-Native**

Để đảm bảo hiệu suất huấn luyện, dữ liệu phải được chuẩn bị kỹ lưỡng và chuyển lên Cloud Storage. Chất lượng và độ chính xác của dữ liệu tiền xử lý (preprocessing) ảnh hưởng trực tiếp đến hiệu suất của mô hình.13

**Tải Dữ liệu lên Cloud Storage:** Dữ liệu huấn luyện phải được lưu trữ tập trung trên các dịch vụ lưu trữ đối tượng của Cloud (ví dụ: Google Cloud Storage - GCS hoặc Amazon S3).2 Các công cụ dòng lệnh (CLI) như gsutil hoặc **gcloud storage cp** phải được sử dụng để tải dữ liệu lên bucket GCS.14 Các lệnh này được tối ưu hóa cho việc truyền tải dung lượng lớn. Ví dụ, gcloud storage cp --recursive gs:// BUCKET\_NAME / FOLDER\_NAME. có thể được sử dụng để sao chép dữ liệu đệ quy.15 Sau khi dữ liệu nằm trên Cloud Storage, job huấn luyện trên Cloud sẽ truy cập trực tiếp bằng đường dẫn gs://....2

**Chiến lược Tối ưu hóa Hiệu suất:** Đối với các mô hình phức tạp, việc sử dụng các kỹ thuật như **Huấn luyện Phân tán (Distributed Training)** là cần thiết.16 Huấn luyện phân tán, thường thông qua Tính song song dữ liệu (Data Parallelism), chia khối lượng công việc tính toán cho nhiều bộ xử lý (GPU).16 Đối với người dùng máy yếu, việc chuyển lên Cloud và sử dụng Distributed Training giúp tối ưu hóa hiệu suất và giảm thời gian chờ đợi, qua đó tối ưu hóa chi phí sử dụng tài nguyên đắt tiền.17

**PHẦN III: QUẢN LÝ THÍ NGHIỆM TRÊN ĐÁM MÂY (MLOPS & MONITORING)**

Huấn luyện trên Cloud tính phí dựa trên thời gian sử dụng tài nguyên. Do đó, việc giám sát chặt chẽ và quản lý thí nghiệm từ xa là cần thiết để tránh lãng phí chi phí.

**3.1. Theo dõi Thí nghiệm với MLflow và TensorBoard**

Các công cụ quản lý vòng đời học máy (MLOps) cho phép theo dõi mọi khía cạnh của quá trình huấn luyện:

* **MLflow Tracking:** Công cụ này được sử dụng để ghi lại (log) các siêu tham số (hyperparameters), mã nguồn, metrics (loss, accuracy), và các Artifacts cho từng lần chạy thí nghiệm.18 Việc này giúp người dùng so sánh ảnh hưởng của các cấu hình khác nhau (ví dụ: learning rate và batch size) để xác định thiết lập tối ưu.18
* **TensorBoard:** Là bộ công cụ trực quan hóa cốt lõi của TensorFlow, cung cấp khả năng theo dõi metrics theo thời gian, xem biểu đồ mô hình và histogram của trọng số.19

**3.2. Thiết lập Giám sát Từ xa**

Để giám sát job huấn luyện đang chạy trên Cloud, cần đảm bảo rằng các log được truy cập từ máy cục bộ.

**Tích hợp TensorBoard với Cloud Storage:** Kịch bản huấn luyện (training script) cần được cấu hình để ghi log TensorBoard trực tiếp vào thư mục trên GCS. Ví dụ, trong Vertex AI, dịch vụ huấn luyện sẽ tự động cung cấp biến môi trường (AIP\_TENSORBOARD\_LOG\_DIR) trỏ đến vị trí lưu trữ trên Cloud Storage.12 Điều này cho phép người dùng theo dõi training trong thời gian gần thực (near real-time) từ bảng điều khiển Cloud Console hoặc TensorBoard cục bộ.

**Sử dụng SSH Port Forwarding:** Nếu người dùng sử dụng máy ảo Cloud (VM/Compute Engine), họ có thể thiết lập SSH Port Forwarding để "chuyển tiếp" cổng 6006 (cổng mặc định của TensorBoard) từ Cloud VM về máy cục bộ (localhost:6006).20 Điều này được thực hiện bằng lệnh: gcloud compute ssh -- -NfL 6006:localhost:6006. Sau đó, chỉ cần truy cập localhost:6006 trên trình duyệt cục bộ.20

**3.3. Tầm quan trọng của MLOps**

MLOps là tập hợp các phương pháp nhằm chuẩn hóa và tự động hóa quy trình triển khai hệ thống ML.21 Đối với người dùng cá nhân, MLOps, đặc biệt là Experiment Tracking, đóng vai trò quan trọng trong việc đảm bảo tính tái lập (repeatability) của các thí nghiệm.

Đồng thời, việc giám sát thời gian thực thông qua TensorBoard và MLflow là một công cụ kiểm soát chi phí hiệu quả. Do Cloud tính phí theo thời gian sử dụng, nếu người dùng nhận thấy các metrics (ví dụ: loss không giảm hoặc bắt đầu phân kỳ) thông qua dashboard giám sát 18, họ có thể dừng job huấn luyện ngay lập tức, tránh lãng phí đơn vị điện toán (Compute Units) hoặc chi phí giờ chạy máy.

**PHẦN IV: QUY TRÌNH QUẢN LÝ VÀ TẢI DỮ LIỆU MODEL VỀ MÁY CỤC BỘ**

Sau khi hoàn tất huấn luyện và đạt được hiệu suất mong muốn, bước cuối cùng là truy xuất các thành phẩm (Artifacts) về máy cục bộ.

**4.1. Khái niệm Model Artifacts và Quản lý Phiên bản (Versioning)**

Thành phẩm mô hình (Model Artifacts) không chỉ là các tệp trọng số (model weights), mà còn bao gồm các tệp cấu hình, tokenizer, và siêu dữ liệu (metadata) của lần chạy huấn luyện đó.22

Để đảm bảo tính tái lập và dễ dàng quản lý, việc sử dụng các công cụ MLOps để quản lý phiên bản (Versioning) là cần thiết. Các nền tảng như Weights & Biases (W&B) cho phép người dùng đăng ký Artifacts với các phiên bản cụ thể. Điều này giải quyết vấn đề truy xuất mô hình một cách tái lập, đảm bảo rằng mô hình được tải về đi kèm với tất cả các thông số huấn luyện chính xác.22

**4.2. Phương pháp Tải Xuống Trực tiếp từ Cloud Storage (CLI)**

Đối với các mô hình lớn, phương pháp tải xuống trực tiếp thông qua giao diện dòng lệnh (CLI) là hiệu quả nhất vì nó được tối ưu hóa cho băng thông và khả năng truyền tải khối lượng lớn.

**Sử dụng Google Cloud CLI:** Lệnh **gcloud storage cp** là công cụ được khuyến nghị để tải các đối tượng và thư mục từ GCS về local.15 Việc tải toàn bộ mô hình và các checkpoint thường yêu cầu tùy chọn tải đệ quy (recursive download):

gcloud storage cp --recursive gs://<BUCKET\_NAME>/<FOLDER\_NAME>.

Ngoài ra, công cụ truyền thống gsutil với tùy chọn -m cũng hỗ trợ truyền tải song song, giúp tăng tốc độ tải xuống.2

**4.3. Tải Xuống Thông qua API MLOps**

Nếu quy trình huấn luyện sử dụng các nền tảng MLOps, việc tải mô hình thông qua API sẽ đơn giản và có tổ chức hơn. Ví dụ, thư viện W&B cho phép tải xuống một Artifact được phiên bản hóa cụ thể bằng lệnh wandb.use\_artifact().22 Phương pháp này đảm bảo tính toàn vẹn của mô hình và các siêu dữ liệu liên quan.

Table 3: Hướng dẫn Kỹ thuật Tải Xuống Model Artifacts

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Phương pháp** | **Tài nguyên Lưu trữ (Ví dụ)** | **Công cụ/Lệnh Kỹ thuật Cốt lõi** | **Ưu điểm** | **Khi nào nên sử dụng** |
| Tải trực tiếp (CLI) | Google Cloud Storage (GCS) | gcloud storage cp --recursive gs://<BUCKET>/<PATH>. 15 | Nhanh chóng, hiệu quả cho thư mục lớn; Tối ưu hóa băng thông. | Tải toàn bộ thư mục checkpoint hoặc mô hình chưa được versioning. |
| Tải qua API MLOps | W&B Artifacts/MLflow Model Registry | wandb.use\_artifact() 22 | Đảm bảo tính phiên bản (versioning) và dễ dàng tích hợp vào code. | Khi quy trình MLOps đã được thiết lập. |
| Tải bằng Airflow Operator | GCS | GCSToLocalFilesystemOperator [23] | Tự động hóa quy trình tải xuống, tích hợp vào Data Pipeline. | Khi xây dựng hệ thống tự động hóa cấp doanh nghiệp. |

Sau khi mô hình được tải về máy yếu, cần lưu ý đến chiến lược suy luận (inference). Việc tải mô hình vào bộ nhớ đệm (cache) một lần và tải từ cache cho các yêu cầu dự đoán tiếp theo, được gọi là **Lazy Loading**, là chiến lược tối ưu để giảm độ trễ và chi phí truy xuất lặp lại.24 Tuy nhiên, cần lưu ý rằng đối với các ứng dụng yêu cầu phản hồi thời gian thực nghiêm ngặt, độ trễ ban đầu khi tải mô hình vào bộ nhớ có thể là một vấn đề.25

**PHẦN V: KẾT LUẬN VÀ KHUYẾN NGHỊ CHIẾN LƯỢC**

Để huấn luyện AI hiệu quả với máy tính cá nhân yếu và ngân sách thấp, chiến lược toàn diện phải kết hợp tối ưu hóa thuật toán và tận dụng nền tảng đám mây một cách khôn ngoan.

**1. Tối ưu hóa Kỹ thuật:**

* Bắt buộc áp dụng Tích lũy Gradient 1 và Mixed Precision.4 Sự kết hợp này giảm thiểu nhu cầu VRAM và tăng tốc độ tính toán trên GPU Cloud, giúp tiết kiệm chi phí thuê giờ máy. Đặc biệt, việc triển khai Mixed Precision phải đi kèm với Gradient Scaling để đảm bảo độ ổn định số học.5

**2. Chiến lược Huấn luyện Cloud Tối ưu Chi phí:**

* Ưu tiên Google Colab Pro/Pay As You Go.10 Đây là lựa chọn tốt nhất cho thử nghiệm do chi phí thấp và quyền truy cập GPU/TPU dễ dàng.
* **Chuyển dữ liệu lên Cloud Storage trước.** Bắt buộc sử dụng các công cụ CLI mạnh mẽ như gcloud storage cp để di chuyển dữ liệu huấn luyện lên GCS, đảm bảo hiệu suất I/O cao nhất khi training.15

**3. Quản lý và Kiểm soát Chi phí:**

* **Triển khai MLOps từ sớm.** Sử dụng các công cụ như MLflow hoặc W&B để theo dõi thí nghiệm và Vertex AI TensorBoard để giám sát hiệu suất mô hình trong thời gian thực.12 Việc này cho phép người dùng dừng job huấn luyện ngay lập tức nếu thấy metrics không cải thiện, là cơ chế kiểm soát chi phí trực tiếp và quan trọng nhất.

**4. Truy xuất và Triển khai Mô hình:**

* **Tải về bằng CLI hoặc API MLOps.** Sử dụng gcloud storage cp --recursive để tải toàn bộ thư mục mô hình một cách hiệu quả.15 Nếu đã thiết lập MLOps, sử dụng API như wandb.use\_artifact() để đảm bảo tải về đúng phiên bản kèm theo siêu dữ liệu.22
* **Lượng tử hóa Mô hình trên Cloud** trước khi tải về sẽ làm giảm kích thước tệp, giảm thời gian tải xuống do giới hạn băng thông cục bộ.6 Khi triển khai cục bộ, áp dụng chiến lược Lazy Loading để giảm độ trễ suy luận.24

**Nguồn trích dẫn**

1. Gradient accumulation | Continuum Labs, truy cập vào tháng 10 30, 2025, <https://training.continuumlabs.ai/training/the-fine-tuning-process/hyperparameters/gradient-accumulation>
2. Hướng dẫn training và dự đoán trên Cloud ML Engine, truy cập vào tháng 10 30, 2025, <https://blog.cloud-ace.vn/huong-dan-training-va-du-doan-tren-cloud-ml-engine/>
3. What is Gradient Accumulation in Deep Learning? | by Raz Rotenberg - Medium, truy cập vào tháng 10 30, 2025, <https://medium.com/data-science/what-is-gradient-accumulation-in-deep-learning-ec034122cfa>
4. Mixed precision | TensorFlow Core, truy cập vào tháng 10 30, 2025, <https://www.tensorflow.org/guide/mixed_precision>
5. Automatic Mixed Precision for Deep Learning | NVIDIA Developer, truy cập vào tháng 10 30, 2025, <https://developer.nvidia.com/automatic-mixed-precision>
6. Lượng Tử Hóa Mô Hình: Tối Ưu Hóa Học Sâu | Ultralytics, truy cập vào tháng 10 30, 2025, <https://www.ultralytics.com/vi/glossary/model-quantization>
7. Machine Learning Cloud Services là gì - Viện FMIT, truy cập vào tháng 10 30, 2025, <https://fmit.vn/tu-dien-quan-ly/machine-learning-cloud-services-la-gi>
8. Amazon SageMaker vs. Azure Notebooks vs. Google Colab Comparison - SourceForge, truy cập vào tháng 10 30, 2025, <https://sourceforge.net/software/compare/Amazon-SageMaker-vs-Azure-Notebooks-vs-Google-Colab/>
9. Best Cloud Platforms for Deep Learning: AWS vs Google vs Azure | by Sam Austin | Medium, truy cập vào tháng 10 30, 2025, <https://medium.com/@digitalconsumer777/best-cloud-platforms-for-deep-learning-aws-vs-google-vs-azure-24ce77e48482>
10. Chọn gói Colab phù hợp với bạn - Google, truy cập vào tháng 10 30, 2025, <https://colab.research.google.com/signup?hl=vi>
11. Colab Paid Services Pricing - Google, truy cập vào tháng 10 30, 2025, <https://colab.research.google.com/signup>
12. Use Vertex AI TensorBoard with Vertex AI Pipelines - Google Cloud, truy cập vào tháng 10 30, 2025, <https://cloud.google.com/vertex-ai/docs/experiments/tensorboard-with-pipelines>
13. Train AI là gì? Giải pháp và quy trình training đào tạo AI bài bản - Lạc Việt, truy cập vào tháng 10 30, 2025, <https://lacviet.vn/train-ai/>
14. Copy data to or from instances | Filestore - Google Cloud, truy cập vào tháng 10 30, 2025, <https://cloud.google.com/filestore/docs/copying-data>
15. Download objects - Storage - Google Cloud Documentation, truy cập vào tháng 10 30, 2025, <https://docs.cloud.google.com/storage/docs/downloading-objects>
16. Huấn luyện phân tán (Distributed Training) - Ultralytics, truy cập vào tháng 10 30, 2025, <https://www.ultralytics.com/vi/glossary/distributed-training>
17. Điện toán phân tán là gì? - AWS, truy cập vào tháng 10 30, 2025, <https://aws.amazon.com/vi/what-is/distributed-computing/>
18. MLflow - FlowHunt, truy cập vào tháng 10 30, 2025, <https://www.flowhunt.io/vi/tu-dien/mlflow/>
19. TensorBoard - TensorFlow, truy cập vào tháng 10 30, 2025, <https://www.tensorflow.org/tensorboard>
20. Using TensorBoard with a Google Cloud Platform Instance - Monte Fischer, truy cập vào tháng 10 30, 2025, <https://www.montefischer.com/2020/02/20/tensorboard-with-gcp.html>
21. MLOps là gì? Những Kỹ năng cần thiết cho vị trí MLOps | 200Lab Blog, truy cập vào tháng 10 30, 2025, <https://200lab.io/blog/mlops-la-gi>
22. Download and use artifacts - Weights & Biases Documentation - Wandb, truy cập vào tháng 10 30, 2025, <https://docs.wandb.ai/models/artifacts/download-and-use-an-artifact>
23. Production | Machine Learning / Optimizing ML Inference Pipelines with Lazy Loading, truy cập vào tháng 10 30, 2025, <https://www.mdinesh.com/lazy-loading.html>
24. Why Lazy Loading Can Hinder Your AI Development. | by Debasis Acharya - Medium, truy cập vào tháng 10 30, 2025, <https://medium.com/@debasisacharya1977/why-lazy-loading-can-hinder-your-ai-development-6b89cb87ad83>