Báo cáo thực nghiệm về Fed-iMAML

Nguyễn Bảo Long, Lê Hoài Bắc

01/04/2023

Abstract

Hệ thống Fed-MAML hoạt động trên dữ liệu non-IID, dù cho kết quả cao và khả năng hội tụ nhanh nhưng tiêu tốn rất nhiều phần cứng. Báo cáo đề xuất kết hợp `iMAML` (thuật toán meta-learning) và hệ thống FL và chứng minh bằng thực nghiệm rằng: (1) Hệ thống Fed-iMAML cho độ chính xác và khả năng hội tụ nhanh tương đương hệ thống Fed-MAML trên tập dữ liệu CIFAR-10, tuy nhiên, trên tập dữ liệu MNIST, thuật toán đề xuất bị giảm hiệu suất so với Fed-MAML; (2) Hệ thống Fed-iMAML tiêu thụ ít GPU hơn và tiêu thụ CPU tương đương khi so sánh với hệ thống Fed-MAML.

1 Thí nghiệm

1.1 Mục tiêu

- Thực hiện đánh giá 2 thuật toán `Fed-iMAML` (thuật toán đề xuất) và `Fed-MAML` về 2 phương diện: (1) - Độ chính xác; (2) - Chi phí phần cứng.
- Kỳ vọng: Thuật toán đề xuất cho độ chính xác tương đương với `Fed-MAML` nhưng tiêu tốn phần cứng ít hơn. Các tiêu chí đánh giá phần cứng là thời gian chạy 1 global epoch trên CPU và dung lượng VRAM tiêu tốn trên GPU.

1.2 Dữ liệu

- Thực hiện thí nghiệm trên bộ dữ liệu MNIST và CIFAR-10.
- MNIST:
  - Bao gồm 70.000 ảnh thuộc 10 phân lớp.
  - 75% được sử dụng làm dữ liệu huấn luyện. Dữ liệu này được phân bố cho 50 clients theo kịch bản dữ liệu non-IID. Mỗi client chỉ chứa các mẫu thuộc 2/10 label bất kỳ.
  - 25% được sử dụng làm dữ liệu kiểm tra. Dữ liệu này được phân bố cho 30 clients theo kịch bản dữ liệu non-IID. Phân phối của các clients này không giống với các clients đã tham gia huấn luyện.

- CIFAR-10: Bao gồm 60.000 ảnh thuộc 10 phân lớp. Được chia tương tự như tập dữ liệu MNIST.
- Tại mỗi client, dữ liệu đều được chia thành 2 tập: support (chứa 20% dữ liệu) và query (chứa 80% dữ liệu).

1.3 Thang đánh giá

- acc_{macro}: Được tính bằng cách lấy trung bình độ chính xác trên các clients. Đơn vị: %.
- Thời gian chạy 1 global epoch của CPU. Đơn vị: second/iteration.
- Lượng VRAM mà GPU tiêu tốn trong quá trình chạy. Đơn vị: %.

1.4 Tiến hành thí nghiệm

- Tiến thành thí nghiệm với các tham số sau:
  - Số global epochs: 200
  - Số clients tham gia trong một lần huấn luyện toàn cục: 5
  - Batch-size: 32 samples/batch
- Sau mỗi 20 bước huấn luyện toàn cục, thực hiện kiểm tra trên toàn bộ 30 test clients.
- Các tham số sau được điều chỉnh để thu được mô hình tốt nhất:
  - Siêu tham số toàn cục: β
  - Siêu tham số cục bộ: α
- Các tham số sau được điều chỉnh để đánh giá quá trình sử dụng phần cứng của các thuật toán:
  - Số bước huấn luyện cục bộ: $local\_epoch \in \{1, 5, 10, 15\}$
  - Số bước tối ưu của `iMAML`: $cg\_step \in \{1, 5, 10\}$.

2 Kết quả

2.1 Độ chính xác

2.2 Lượng VRAM trên GPU

- Từ hình 1 và hình 2, có thể thấy mức tiêu thụ VRAM của `Fed-MAML` tăng tuyến tính theo số bước tối ưu cục bộ của thuật toán. Tuy nhiên, thuật toán `Fed-iMAML` lại cho mức tiêu thụ VRAM là hằng số khi tăng cả hai thông số là số bước huấn luyện cục bộ và số bước tối ưu của `iMAML`.

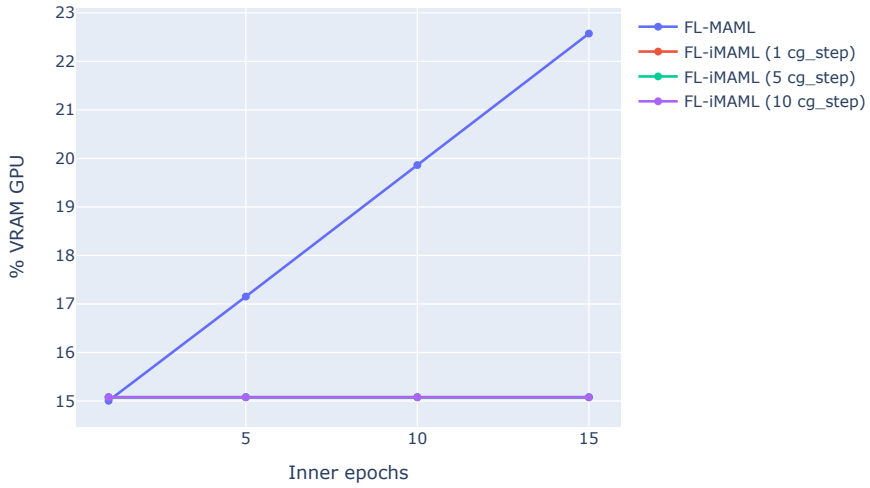

Figure 1: Lượng VRAM tiêu thụ của các thuật toán `Fed-MAML` và `Fed-iMAML` (cg_step lần lượt bằng 1, 5, 10) trên tập dữ liệu MNIST. Các đường biểu diễn `Fed-iMAML` nằm chồng khít lên nhau và không tăng dù cho số bước tối ưu là tăng dần.

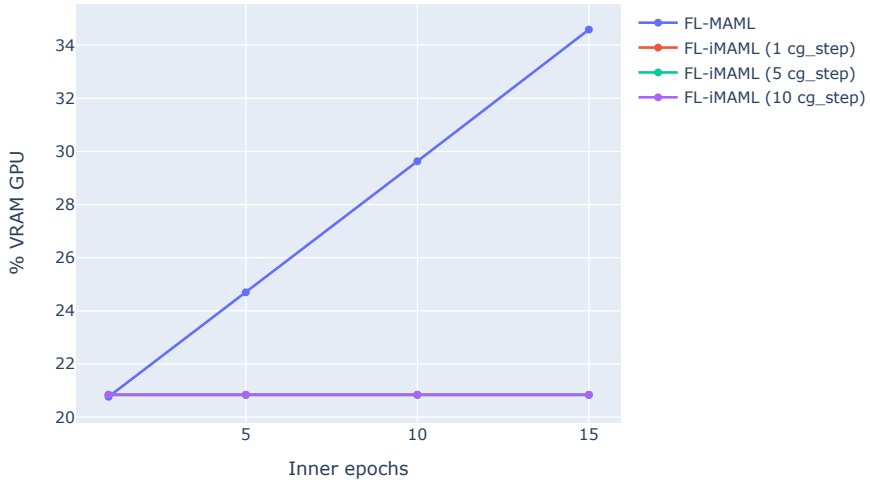

Figure 2: Lượng VRAM tiêu thụ của các thuật toán `Fed-MAML` và `Fed-iMAML` (cg_step lần lượt bằng 1, 5, 10) trên tập dữ liệu CIFAR-10. Các đường biểu diễn `Fed-iMAML` nằm chồng khít lên nhau và không tăng dù cho số bước tối ưu là tăng dần.

2.3 Thời gian chạy trên CPU

- Từ hình **??** và hình **??**, có thể thấy, thời gian tính toán của CPU cho 1 global epoch giữa thuật toán `Fed-MAML` và `Fed-iMAML` là như nhau.

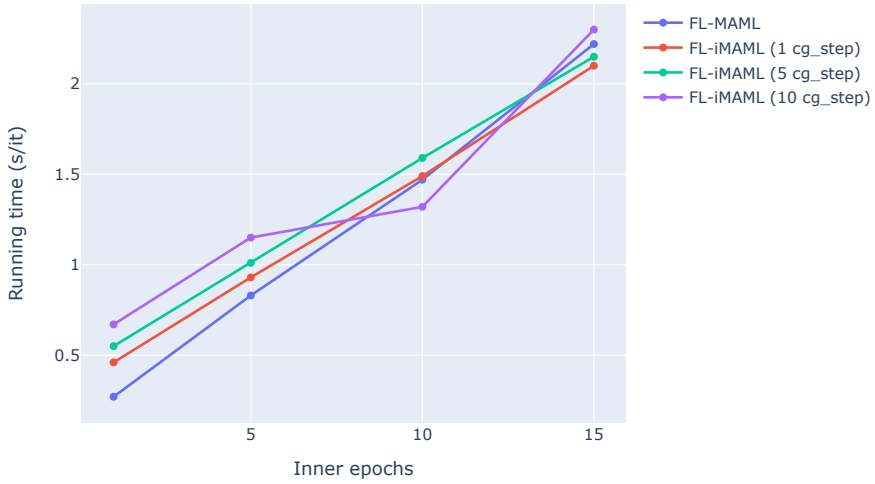

Figure 3: Thời gian tính toán của CPU cho 1 global epoch cho 2 thuật toán `Fed-MAML` và `Fed-iMAML` trên tập MNIST.

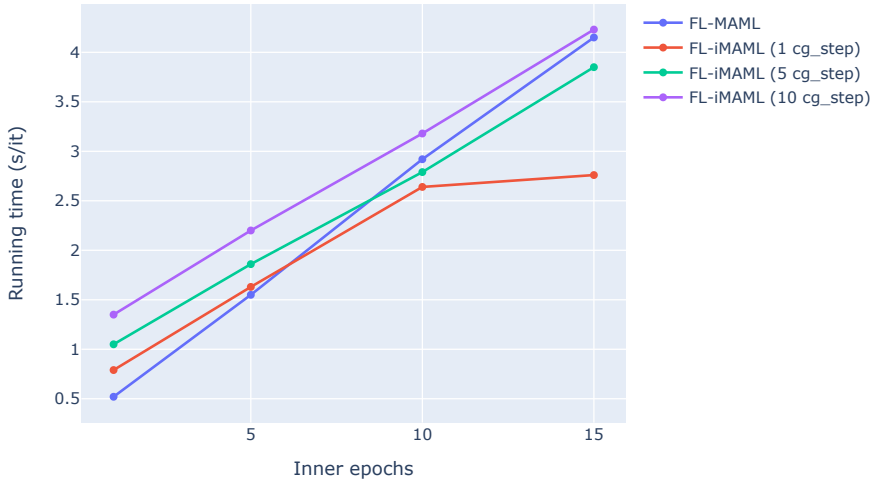

Figure 4: Thời gian tính toán của CPU cho 1 global epoch cho 2 thuật toán `Fed-MAML` và `Fed-iMAML` trên tập CIFAR-10.