TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

Nguyễn Bảo Long - Cao Tất Cường

Ứng dụng của Meta Learning và Personalization Layer trong hệ thống Federated Learning

KHÓA LUẬN TỐT NGHIỆP CỬ NHÂN
CHƯƠNG TRÌNH CHÍNH QUY

Tp. Hồ Chí Minh, tháng 03/2022

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

Nguyễn Bảo Long - 18120201
Cao Tất Cường - 18120296

Ứng dụng của Meta Learning và Personalized Layer trong hệ thống Federated Learning

KHÓA LUẬN TỐT NGHIỆP CỬ NHÂN
CHƯƠNG TRÌNH CHÍNH QUY

GIÁO VIÊN HƯỚNG DẪN
GS.TS. Lê Hoài Bắc

Tp. Hồ Chí Minh, tháng 03/2022

Tóm tắt

Đề cương chi tiết

Thông tin chung

- Tên đề tài: Ứng dụng của Meta Learning và Personalization Layer trong hệ thống Federated Learning
- Giảng viên hướng dẫn: GS. TS. Lê Hoài Bắc
- Nhóm sinh viên thực hiện:
  - Nguyễn Bảo Long - MSSV: 18120201
  - Cao Tất Cường - MSSV: 18120296
- Thời gian thực hiện: Từ 09/2021 đến 03/2022
- Loại đề tài: Nghiên cứu

Nội dung thực hiện

Giới thiệu đề tài

Trong bối cảnh bùng nổ thông tin cũng như việc đề cao tính riêng tư dữ liệu người dùng như hiện nay, các mô hình huấn luyện tập trung truyền thống dần bộc lộ nhiều điểm yếu khiến chúng không còn phù hợp. Ba điểm yếu làm cho cách tiếp cận cũ này trở nên tốn kém, không năng suất và ảnh hưởng đến quyền riêng tư của người dùng có thể kể đến:

- Việc truyền dữ liệu từ máy người dùng về máy chủ để tiến hành huấn luyện tiềm ẩn nguy cơ lộ dữ liệu quan trọng của người dùng.
- Chi phí truyền dữ liệu từ người dùng về máy chủ để huấn luyện ngày càng lớn do lượng dữ liệu sinh ra tại thiết bị cuối ngày càng tăng cao.

- Cần một máy chủ thật mạnh mẽ để huấn luyện mô hình với lượng dữ liệu lớn như vậy.

Khái niệm federated learning (FL) được Google lần đầu giới thiệu trong nghiên cứu [14] với ý tưởng chính là huấn luyện mô hình máy học trên các tập dữ liệu riêng biệt được phân bố trên các thiết bị biên (được gọi là huấn luyện phân tán). Với ý tưởng này, việc triển khi mô hình máy học đến người dùng không còn gặp phải vấn đề về chi phí truyền tin, giúp bảo vệ quyền riêng tư dữ liệu và không đòi hỏi một máy chủ quá mạnh để huấn luyện mô hình. Tuy nhiên, dữ liệu của người dùng trong hệ thống thường không đồng nhất và có tính cá nhân hóa rất cao (tuân theo phân phối Non-IID). Điều này khiến cho hiệu suất của hệ thống FL suy giảm nghiêm trọng [22].

Một cách ngắn gọn, hiệu suất của mô hình bị giảm là do mô hình không khái quát tốt trên tập dữ liệu của người dùng. Các thuật toán meta learning (ML) được biết đến với khả năng thích ứng nhanh trên tập dữ liệu mới [9]. Điều này giải quyết chính xác vấn đề dữ liệu FL đang gặp phải. Song song với đó, để tăng thêm tính các nhân hóa mô hình cho từng người dùng, các nghiên cứu [2, 12] đề xuất sử dụng kỹ thuật personalization layer (PL), giúp tăng đáng kể hiệu suất trên hệ thống FL. Tuy nhiên, các phương pháp tối ưu kể trên vẫn tồn tại nhiều khuyết điểm và có khả năng bù trừ cho nhau. Do đó, chúng tôi tiến hành nghiên cứu về ML, PL. Từ đó, kết hợp chúng vào hệ thống FL để đạt được hiệu suất tốt hơn.

Mục tiêu đề tài

Mục tiêu chính của đề tài này bao gồm: (1) - nghiên cứu, khảo sát các thuật toán theo hướng FL, ML, PL và (2) - kết hợp cài đặt các thuật toán trên, giúp nâng cao hiệu suất của hệ thống FL khi đối mặt với dữ liệu dạng Non-IID.

Việc nghiên cứu, khảo sát các thuật toán nhằm đưa ra đánh giá về ưu, nhược điểm của từng thuật toán. Từ đó, biết cách kết hợp chúng để đạt hiệu suất tốt.

Việc kết hợp cài đặt nhằm mục đích chứng minh thực nghiệm tính hiệu quả của thuật toán đề xuất khi làm việc với dữ liệu Non-IID.

Phạm vi đề tài

Nghiên cứu [19] chỉ ra ba hướng nghiên cứu chính khi đề cập đến một hệ thống FL: (1) - Cải thiện hiệu suất của hệ thống FL, (2) - Cải thiện khả năng bảo mật của hệ thống FL, (3) - Cải thiện vấn đề về quyền riêng tư của người dùng trong hệ thống FL.

Về việc phân loại hệ thống FL, dựa trên dữ liệu đầu vào, hệ thống FL được chia thành ba loại [19]: (1) - Horizontal FL, (2) - Vertical FL, (3) - Federated transfer learning.

Về việc phân loại các kịch bản Non-IID, nghiên cứu [22] chỉ ra bốn kịch bản chính: (1) - Phân phối thuộc tính khác nhau giữa các máy khách, (2) - Phân phối nhãn khác nhau giữa các máy khách, (3) - Phân phối thời gian khác nhau giữa các máy khách, (4) - Các kịch bản khác.

Chúng tôi giới hạn phạm vi và xây dựng phương án giải quyết của đề tài dựa trên ba giả định sau:

- Loại hệ thống: Môi trường thí nghiệm (bao gồm các yếu tố như số lượng người dùng, dữ liệu, cấu hình, khả năng lưu trữ của thiết bị cuối,...) tuân theo đặc điểm của hệ thống Horizontal FL.
- Bảo mật & quyền riêng tư: Hệ thống đã đảm bảo tính bảo mật cũng như duy trì tốt quyền riêng tư của người dùng.
- Kịch bản Non-IID: Phân phối nhãn dữ liệu trên các máy khách là khác nhau.

Cách tiếp cận

Phương pháp chính.

Hệ thống FL huấn luyện trực tiếp các mô hình máy học trên các tập dữ liệu của từng người dùng, sau đó tiến hành tổng hợp tham số của mô hình thu được từ các máy khách này để thu được một mô hình toàn cục. Do đó, kiến trúc máy chủ - máy khách nghiễm nhiên trở được nghĩ đến và trở nên phổ biến. Nghiên cứu [19] nêu ra các đặc điểm chính của máy chủ và máy khách trong một hệ thống FL:

- Máy chủ: Điều phối các hoạt động huấn luyện mô hình và duy trì một bộ tham số toàn cục bằng cách tổng hợp các tham số mô hình do máy khách gửi về.

- Máy khách: Đóng vai trò là nơi huấn luyện các mô hình học theo sự chỉ đạo của máy chủ. Chúng nhận tham số toàn cục từ máy chủ, huấn luyện mô hình dựa trên bộ tham số này và gửi tham số của mô hình mới về máy chủ để tổng hợp.

Phương pháp đề xuất của chúng tôi nhằm tích hợp các thuật toán ML, PL vào hệ thống nêu trên. Trong đó, các thuật toán ML được sử dụng để tạo ra một khởi tạo tốt, giúp máy khách hội tụ mô hình nhanh chóng (chỉ sau một hoặc một vài bước huấn luyện); các thuật toán PL được thêm vào như một phương pháp giúp tăng tính cá nhân hóa của từng mô hình máy học phân bố trên máy khách.

Dữ liệu thực nghiệm. Chúng tôi sử dụng tập dữ liệu CIFAR-10 và tập dữ liệu MNIST trong tất cả các thí nghiệm. Cả hai tập dữ liệu đều sử dụng 25% số điểm dữ liệu để kiểm thử và 75% số điểm dữ liệu để huấn luyện.

Phương pháp đối sánh. Chúng tôi sử dụng thuật toán `FedAvg` làm mô hình baseline. Để so sánh công bằng, chúng tôi cho phép mô hình này fine-tune một hoặc một vài bước trên một phần tập dữ liệu kiểm thử (thuật toán `FedAvgMeta`) trước khi bước vào kiểm thử thực sự. Các kết quả của thuật toán đề xuất sẽ được đem so sánh với kết quả của `FedAvg` và `FedAvgMeta`.

Kết quả đề tài

Sau khi tiến hành nghiên cứu, chúng tôi kỳ vọng đạt được những kết quả sau:

- Nắm được ý tưởng huấn luyện mô hình của các thuật toán theo hướng FL, ML, PL. Cài đặt mô hình baseline.

- Cài đặt được hệ thống FL có tích hợp ML. So sánh độ chính xác thu được với mô hình baseline.

- Cài đặt hệ thống FL có tích hợp ML và PL. So sánh độ chính xác thu được với mô hình baseline.

Kế hoạch thực hiện

Kế hoạch thực hiện khóa luận bao gồm 4 giai đoạn, được trình bày trong bảng sau:

Bảng 1: Bảng phân chia công việc

Giai đoạn	Công việc	Người thực hiện
1 (01/09/2021 - 15/09/2021)	Tìm hiểu kiến thức nền tảng về ML	Nguyễn Bảo Long
	Tìm hiểu kiến thức nền tảng về FL	Cao Tất Cường
	Tìm hiểu kiến thức nền tảng về PL	Cao Tất Cường
	Trao đổi 2 mảng kiến thức	Cả hai
2 (16/09/2021 - 31/01/2022)	Cài đặt các thuật toán FedAvg, FedMeta(Meta-SGD)	Nguyễn Bảo Long
	Cài đặt các thuật toán FedAvg(Meta), FedAvg(MAML)	Cao Tất Cường
	Cài đặt hệ thống FL có tích hợp ML và PL	Nguyễn Bảo Long
	Phân tích và đánh giá kết quả	Cả hai
3 (01/02/2022 - 25/02/2022)	Viết luận văn	Nguyễn Bảo Long
	Làm slide thuyết trình	Cao Tất Cường
	Tập thuyết trình	Cả hai

Lời cảm ơn

Xin phép gửi lời cảm ơn chân thành nhất đến GS. TS. Lê Hoài Bắc, người đã cung cấp, chia sẻ tài liệu và kiến thức, cũng như đã trực tiếp tham gia giảng dạy, hướng dẫn chúng tôi hoàn thiện đồ án này.

Ngoài ra, chúng tôi đặc biệt cảm ơn TS. Nguyễn Tiến Huy vì những lời khuyên rất hữu ích mỗi khi chúng tôi gặp khó khăn.

Đồ án này sẽ không thể hoàn thiện được nếu không có sự hỗ trợ kiến thức đến từ chị Bùi Thị Cẩm Nhung, cựu sinh viên chương trình Cử nhân tài năng, khoa Công Nghệ Thông Tin, khóa 2017. Chúng tôi rất cảm ơn trước những đóng góp của chị.

Mục lục

Danh sách hình

Danh sách bảng

Chương 1

Giới thiệu

1.1 Đặt vấn đề & Động lực

Hiện nay, các thiết bị biên như điện thoại, máy tính bảng, thậm chí máy giặt, máy hút bụt thông minh có thể sinh ra lượng lớn dữ liệu trong quá trình hoạt động. Lượng dữ liệu này, nếu tận dụng được, có thể mang lại sự cải thiện rất lớn về độ chính xác cho các mô hình máy học hiện tại. Ví dụ, dữ liệu thu thập được từ bàn phím điện thoại có thể phục vụ tối ưu cho các mô hình ngôn ngữ; ảnh chụp được lưu trữ trong bộ nhớ điện thoại hoàn toàn có thể được sử dụng làm dữ liệu để huấn luyện cho mô hình nhận dạng ảnh; hay lịch sử duyệt web của người dùng có thể được dùng cho bài toán đề xuất sản phẩm. Những lý do trên trở thành một động lực to lớn, thúc đẩy việc tìm ra một phương pháp giúp tận dụng nguồn dữ liệu dồi dào này.

Việc ngày càng nhiều dữ liệu được sinh ra tại các thiết bị biên khiến cho phương pháp huấn luyện mô hình theo cách tiếp cận truyền thống (được gọi là huấn luyện tập trung) bộc lộ nhiều khuyết điểm. Ba điểm yếu khiến cho các tiếp cận này không còn mạnh mẽ có thể kể đến: sự vi phạm về quyền riêng tư dữ liệu, chi phí truyền tin và chi phí phần cứng máy chủ.

Sự vi phạm về quyền riêng tư dữ liệu. Phương pháp truyền thống đòi hỏi phải gửi dữ liệu người dùng về một máy chủ để tiến hành huấn luyện mô hình. Các thông tin nhạy cảm của người dùng hoàn toàn có thể bị nghe lén bởi kẻ tấn công hoặc bị khai thác khi máy chủ bị nhiễm mã độc. Điều này ảnh hưởng nghiêm trọng đến quyền riêng tư dữ liệu của người dùng - một vấn đề mà hiện nay đang nhận được rất nhiều sự quan tâm từ cả người dùng lẫn chính phủ.

Chi phí truyền tin. Dữ liệu sinh ra tại thiết bị biên đang ngày một tăng lên do văn hóa sử dụng và sự phát triển của công nghệ. Một người

dùng điện thoại thông minh giờ đây có thể thực hiện giao dịch tài chính, nghe nhạc, lướt web, xem phim ngay trên thiết bị của mình. Một máy hút bụi thông minh được trang bị các cảm biến nên hoàn toàn có thể sử dụng dữ liệu cảm biến này như một "time series". Chi phí truyền tin từ các thiết bị biên đến máy chủ để huấn luyện trở nên tốn kém và có thể gây mất thông tin, ảnh hưởng đến hiệu suất học của mô hình.

Chi phí phần cứng máy chủ. Sau khi dữ liệu được gửi về máy chủ, cần một cấu hình máy mạnh mẽ cùng khả năng lưu trữ lớn để có thể xử lý hết lượng dữ liệu khổng lồ trên trong thời gian giới hạn.

Việc các phương pháp tiếp cận máy học theo hướng truyền thống đang dần bộc lộ các nhược điểm về chi phí vận hành và bảo trì ngày càng cao, cũng như các mối nguy hiểm tiềm tàng có thể xảy ra đối với dữ liệu của người dùng, một lần nữa thúc đẩy việc nghiên cứu về một phương pháp huấn luyện giúp làm giảm chi phí phần cứng (sử dụng cho đường truyền và máy chủ), đồng thời đảm bảo tính riêng tư dữ liệu cho người dùng. Khái niệm federated learning (FL) và thuật toán `FedAvg` được đưa ra vào năm 2016 bởi Google trong nghiên cứu [14] nhằm mục đích huấn luyện mô hình máy học trên các tập dữ liệu riêng biệt được phân bố trên các thiết bị biên (được gọi là huấn luyện phân tán). Do đó, một hệ thống FL không cần một máy chủ quá mạnh để vận hành (thậm chí có thể sử dụng một máy khách để vận hành [19] , không đòi hỏi chi phí truyền tin quá lớn và đảm bảo được quyền riêng tư dữ liệu của người dùng vì không diễn ra bất cứ quá trình thu thập dữ liệu từ người dùng nào (điều mà mô hình huấn luyện tập trung bắt buộc phải làm). Dễ thấy rằng, phần lớn quá trình tính toán được chuyển đến các thiết bị biên. Tuy nhiên, khả năng lưu trữ và tính toán tại các thiết bị này ngày càng được cải thiện, khiến cho việc việc huấn luyện phân tán dần trở nên khả thi và đạt hiệu quả cao hơn.

Mặt khác, nghiên cứu [22] chỉ ra rằng, hệ thống FL hoạt động trên nền thuật toán `FedAvg` bị giảm hiệu suất nghiêm trọng khi xử lý dữ liệu đầu vào tuân theo phân phối Non-IID. Trong khi đó, dữ liệu phân bố trên máy khách là không đồng nhất và có tính cá nhân hóa tất cao. Nói cách khác, các tập dữ liệu này tuân theo phân phối Non-IID. Do đó, việc nghiên cứu và cải tiến hệ thống FL để thu được kết quả cao trên dữ liệu Non-IID là rất cần thiết. Đây chính là vấn đề mà khóa luận hướng tới giải quyết.

1.2 Phạm vi đề tài

Nghiên cứu [19] chỉ ra ba hướng nghiên cứu chính khi đề cập đến một hệ thống FL: (1) - Cải thiện hiệu suất của hệ thống FL, (2) - Cải thiện khả năng bảo mật của hệ thống FL, (3) - Cải thiện vấn đề về quyền riêng tư của người dùng trong hệ thống FL.

Về việc phân loại hệ thống FL, dựa trên dữ liệu đầu vào, hệ thống FL được chia thành ba loại [19]: (1) - Horizontal FL, (2) - Vertical FL, (3) - Federated transfer learning.

Về việc phân loại các kịch bản Non-IID, nghiên cứu [22] chỉ ra bốn kịch bản chính: (1) - Phân phối thuộc tính khác nhau giữa các máy khách, (2) - Phân phối nhãn khác nhau giữa các máy khách, (3) - Phân phối thời gian khác nhau giữa các máy khách, (4) - Các kịch bản khác.

Chúng tôi giới hạn phạm vi và xây dựng phương án giải quyết của đề tài dựa trên ba giả định sau:

- Loại hệ thống: Môi trường thí nghiệm (bao gồm các yếu tố như số lượng người dùng, dữ liệu, cấu hình, khả năng lưu trữ của thiết bị cuối,...) tuân theo đặc điểm của hệ thống Horizontal FL.
- Bảo mật & quyền riêng tư: Hệ thống đã đảm bảo tính bảo mật cũng như duy trì tốt quyền riêng tư của người dùng.
- Kịch bản Non-IID: Phân phối nhãn dữ liệu trên các máy khách là khác nhau.

1.3 Đóng góp chính

Chúng tôi chia các đóng góp chính của mình thành hai loại: đóng góp về mặt lý thuyết và đóng góp về mặt thực nghiệm.

1.3.1 Đóng góp lý thuyết

- Nghiên cứu hệ thống FL và thách thức về phân phối dữ liệu mà hệ thống Horizontal FL gặp phải.

- Khảo sát các phương pháp tối ưu hóa hệ thống Horizontal FL trên dữ liệu dạng Non-IID. Trong đó, tập trung nghiên cứu các phương pháp theo hướng Personalized Federated Averaging [6, 4] và Personalization Layer [12, 2].
- Phương pháp đề xuất của chúng tôi đã cho thấy khả năng đạt độ chính xác cao hơn trong quá trình kiểm thử với hai đối tượng người dùng (người dùng cục bộ và người dùng mới) so với các phương pháp trước đó (chỉ sử dụng `FedAvg`, chỉ sử dụng Personalized Federated Averaging, hoặc chỉ sử dụng Personalization Layer).

1.3.2 Đóng góp thực nghiệm

- Tổ chức bộ dữ liệu MNIST và CIFAR-10 theo hai hướng IID và Non-IID để tiến hành thí nghiệm.
- Cài đặt thuật toán `FedAvg`, `FedAvgMeta`, các thuật toán kết hợp giữa `FedAvg` và ML (thuật toán `FedMeta(MAML)`, `FedMeta(Meta-SGD)`).
- Cài đặt thuật toán kết hợp giữa `FedAvg` và PL (thuật toán `FedPer`, `LG-FedAvg`).
- Kết hợp các thuật toán ML và PL vào hệ thống FL.
- Fine-tune các siêu tham số như số lượng máy khách tham gia huấn luyện, số bước huấn luyện cục bộ, các siêu tham số học để mô hình đạt độ chính xác tốt nhất.

1.4 Bố cục

Trong luận văn này, chương 2 trình bày về tổng quan lý thuyết được sử dụng trong khóa luận, các lý thuyết này làm nền tảng cho nghiên cứu và đề xuất thuật toán; chương 3 đề xuất thuật toán giúp giải quyết vấn đề vừa nêu ở chương 1; chương 4 trình bày về cài đặt thực nghiệm để kiểm chứng tính hiệu quả của thuật toán; chương 5 đi vào phân tích kết quả đạt được; chương 6 nêu kết luận, những điều chưa làm được và hướng phát triển tương lai của khóa luận.

Chương 2

Tổng quan lý thuyết

2.1 Hệ thống Federated Learning

2.1.1 Định nghĩa

Định nghĩa về FL [18]: Giả sử có n máy khách, máy khách thứ i ký hiệu là c_i ($i \in [1, n]$), chứa tập dữ liệu D_i. FL là một quá trình học mà ở đó, các chủ sở hữu dữ liệu (ở đây có thể hiểu là các thiết bị biên) cùng hợp tác huấn luyện một mô hình $\mathcal{M}$ và đạt được độ chính xác f nhưng không có bất kỳ chủ sở hữu dữ liệu c_i nào chia sẻ tập dữ liệu $\mathcal{D}_i$ của chúng.

Gọi $\bar{\mathcal{M}}$ là mô hình máy học được huấn luyện trên tập dữ liệu $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup ... \cup \mathcal{D}_n$ và cho độ chính xác $\bar{f}$. f và $\bar{f}$ chỉ được phép chênh lệch nhau một khoảng nhỏ. Gọi δ là một giá trị thực không âm, nếu $| f - \bar{f} | < \delta$ ta nói mô hình $\mathcal{M}$ có δ - *accuracy loss*.

Định nghĩa về tính hợp lệ [10]: Ký hiệu $\mathcal{M}_i$ là mô hình được huấn luyện trên tập dữ liệu $\mathcal{D}_i$ và cho độ chính xác f_i. Mô hình $\mathcal{M}$ được gọi là hợp lệ nếu tồn tại $i \in [1, n]$ sao cho $f > f_i$.

2.1.2 Một hệ thống Federated Learning điển hình

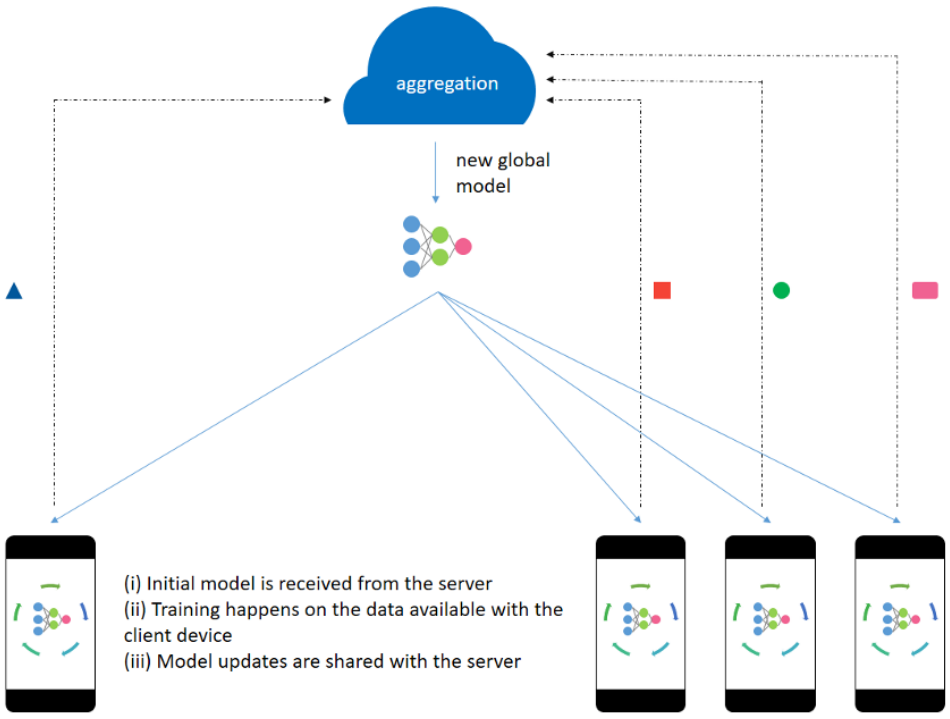
Thành phần và các tương tác trong hệ thống. Một hệ thống FL (Hình 2.1) thường bao gồm hai thành phần chính: máy chủ (đóng vai trò là đối tượng duy trì mô hình toàn cục) và máy khách (đóng vai trò là đối tượng nắm giữ dữ liệu huấn luyện). Hai thành phần này tương tác với nhau theo ba bước sau [13]:

- *Khởi tạo.* Máy chủ khởi tạo trọng số w_G^0 cho mô hình toàn cục và các siêu tham số cho quá trình huấn luyện. Thông tin này sau đó được gửi đến một tập hợp con các máy khách được chọn để tiến hành huấn luyện.

- *Huấn luyện và cập nhật mô hình cục bộ.* Tại bước huấn luyện thứ t, máy khách c_i nhận trọng số w_G^t từ máy chủ và tiến hành huấn luyện cục bộ trên tập dữ liệu $\mathcal{D}_i$. Tham số θ_i^{t+1} thu được sau quá trình huấn luyện (có thể là trọng số w_i^{t+1} hoặc đạo hàm hàm lỗi g_i^{t+1}) được máy khách gửi về máy chủ để tổng hợp.

- *Tổng hợp và cập nhật mô hình toàn cục.* Máy chủ nhận tham số θ_i^{t+1} gửi về từ các máy khách được chọn trước đó, tiến hành tổng hợp w_G^{t+1} - trọng số mới của mô hình toàn cục và gửi trọng số này đến một tập hợp con các máy khách khác để bắt đầu bước huấn luyện toàn cục mới.

Máy chủ sẽ lặp lại bước 2 và bước 3 cho đến khi độ lỗi hội tụ hoặc độ chính xác đạt đến một ngưỡng nhất định. Khi quá trình huấn luyện kết thúc, tham số của mô hình toàn cục sẽ được phân phối đến toàn bộ máy khách trong hệ thống.



Hình 2.1: Hai thành phần chính và quá trình tương tác giữa chúng trong hệ thống FL [3]

Mục tiêu của hệ thống FL. Chúng tôi khảo sát hai mục tiêu của hệ thống FL: (1) - Mục tiêu cục bộ; (2) - Mục tiêu toàn cục.

Các máy khách trong hệ thống hướng đến việc thực hiện mục tiêu cục bộ. Ban đầu, máy khách c_i nhận một trọng số toàn cục w_G từ máy chủ. Máy khách này sau đó sẽ cố gắng tìm kiếm một trọng số w_i^{t+1} giúp cực tiểu hóa hàm lỗi cục bộ. Nói cách khác, w_i^{t+1} phải thỏa mãn thỏa mãn:

$$w_i^{t+1} = \arg\min_{w_i} f_{local}(w_i^t) \tag{2.1}$$

Trong đó, $f_{local}(w_i)$ là hàm lỗi trên tập dữ liệu của c_i. Với α là siêu tham số học cục bộ, $w_{i(j)}$ là trọng số tại bước huấn luyện j của c_i, lời giải của phương trình 2.1 theo phương pháp SGD sau e bước huấn luyện có thể được viết:

$$\begin{cases} w_{i(0)}^t = w_G^t \\ w_{i(j)}^t = w_{i(j-1)}^t - \alpha \nabla f_{local}(w_{i(j-1)}^t) \\ w_i^{t+1} = w_{i(e)}^t \end{cases} \tag{2.2}$$

Hay:

$$w_i^{t+1} \leftarrow w_i^t - \alpha \nabla f_{local}(w_i^t) \tag{2.3}$$

Mặt khác, mục tiêu toàn cục, cũng là mục tiêu chính của hệ thống FL, được máy chủ thực hiện bằng cách tìm kiếm một trọng số w_G^* giúp tối thiểu hóa hàm lỗi của cả hệ thống [19]:

$$\begin{aligned} w_G^* &= \arg\min_{w_G} f_{global}(w_G) \\ &= \arg\min_{w_G} \frac{1}{n} \sum_{i=1}^{n} f_{local}(w_i) \end{aligned} \tag{2.4}$$

Trong đó, $f_{global}(w_G)$ là hàm lỗi toàn cục của hệ thống. Để giải phương trình 2.4, máy chủ thực hiện tổng hợp tham số gửi về từ máy khách bằng một trong hai cách: lấy trung bình trọng số [14, 1, 20] hoặc lấy trung bình đạo hàm [4, 15].

Đặt $n_i = |\mathcal{D}_i|$ là số điểm dữ liệu của tập $\mathcal{D}_i$, $N = \sum_{i=1}^{n} n_i$ là tổng số điểm dữ liệu có trong cả hệ thống. Phương pháp lấy trung bình trọng số

tính toán trọng số toàn cục tại bước huấn luyện thứ t từ các trọng số của máy khách như sau [14]:

$$w_G^{t+1} = \sum_{i=1}^{n} \frac{n_i}{N} w_i^{t+1} \tag{2.5}$$

Trái lại, phương pháp lấy trung bình đạo hàm đòi hỏi máy khách gửi về đạo hàm hàm lỗi sau khi kết thúc quá trình huấn luyện cục bộ. Với β là siêu tham số học toàn cục, quá trình tổng hợp tại bước huấn luyện t được biểu diễn theo công thức:

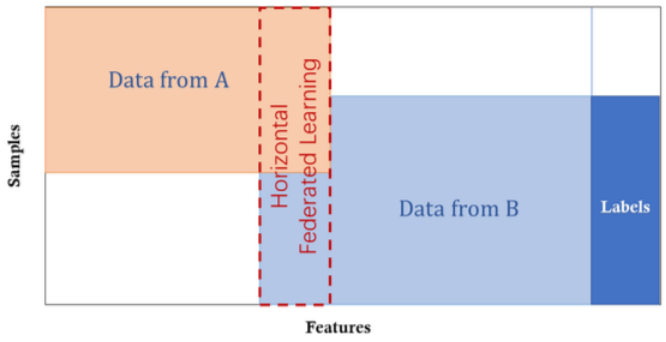
$$\begin{aligned} w_G^{t+1} &= w_G^t - \beta \sum_{i=1}^{n} \frac{n_i}{N} \nabla f_{local}(w_i^{t+1}) \\ &= w_G^t - \beta g^{t+1} \end{aligned} \tag{2.6}$$

Sau khi khảo sát cả hai phương pháp tổng hợp tham số của máy chủ, nghiên cứu [19] chỉ ra rằng, việc lấy trung bình trọng số giúp hệ thống có khả năng chịu được việc mất cập nhật, nhưng không đảm bảo việc hội tụ. Trái lại, việc lấy trung bình đạo hàm giúp hệ thống đảm bảo sự hội tụ nhưng tiêu tốn nhiều chi phí truyền tin hơn. Trong nghiên cứu này, chúng tôi tổng hợp trọng số toàn cục bằng phương pháp lấy trung bình trọng số để phù hợp hơn với giới hạn về chi phí giao tiếp và lưu trữ.

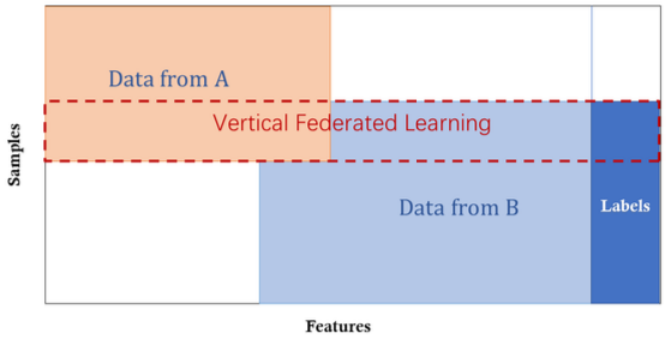
Phân loại hệ thống Federated Learning. Nghiên cứu [19] đề xuất các phân loại các hệ thống FL dựa trên phân bố dữ liệu đầu vào của chúng. Theo đó, ba phân bố dữ liệu: (1) - Phân bố dữ liệu theo chiều ngang (Horizontal data partitioning), (2) - Phân bố dữ liệu theo chiều dọc (Vertical data partitioning), (3) - Phân bố dữ liệu hỗn hợp (Hybrid data partitioning) sẽ ứng với ba loại hệ thống FL (Hình 2.2): (1) - Hệ thống FL theo chiều ngang (Horizontal FL), (2) - Hệ thống FL theo chiều dọc (Vertical FL), (3) - Hệ thống học chuyển giao tri thức (Federated Transfer Learning).

Hệ thống Horizontal FL. Phân bố dữ liệu theo chiều ngang là kiểu phân bố dữ liệu mà ở đó các bên tham gia vào hệ thống cùng sở hữu các đặc tính dữ liệu giống nhau nhưng giá trị định danh của mẫu dữ liệu của các bên là khác nhau. Ví dụ, khi các bên tham gia hệ thống là các trường đại học,

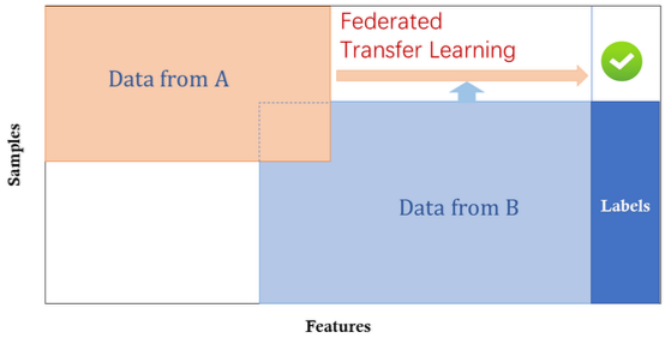
họ sẽ muốn quản lý các thông tin giống nhau về sinh viên như họ và tên, mã số sinh viên,... nhưng không có một sinh viên nào tham gia hai trường đại học cùng một lúc. Kiến trúc Horizontal FL rất phù hợp để huấn luyện mô hình học tuân theo phân phối này [19].



(a) Horizontal Federated Learning



(b) Vertical Federated Learning



(c) Federated Transfer Learning

Hình 2.2: Ba loại hệ thống FL với phân bố dữ liệu tương ứng [18]

Dựa vào kiến trúc giao tiếp, có thể chia Horizontal FL ra làm hai loại: Kiến trúc client-server và kiến trúc peer-to-peer (P2P). Kiến trúc client-server, hay còn gọi là kiến trúc FL tập trung, về cơ bản sẽ thực hiện các bước huấn luyện giống như đã trình bày trong phần **Thành phần và các tương tác trong hệ thống**. Trong khi đó, kiến trúc P2P, hay còn gọi là kiến trúc FL phân tán không có một máy chủ cố định. Tại mỗi bước huấn luyện toàn cục, một máy khách trong hệ thống được chọn làm máy chủ. Quá trình huấn luyện sau đó được thực hiện giống như kiến trúc client-server.

Một hệ thống Horizontal FL thường có số lượng máy khách rất lớn, khả năng lưu trữ và tính toán tại các máy khách không cần quá cao (ví dụ như điện thoại thông minh, máy tính bảng) và tần suất một máy khách tham gia huấn luyện là rất thấp.

Hệ thống Vertical FL. Đây là kiến trúc phù hợp với phân bố dữ liệu theo chiều dọc. Trong phân bố dữ liệu dạng này, các bên tham gia hệ thống sở hữu các đặc tính dữ liệu khác nhau nhưng giá trị định danh của mẫu dữ liệu của các bên là giống nhau. Ví dụ, khi các bên tham gia hệ thống là ngân hàng và trường đại học. Thuộc tính mà ngân hàng và trường đại học lưu trữ là rất khác nhau nhưng lại chứa thông tin của cùng một người dùng.

Hệ thống Federated Transfer Learning. Khi phân bố dữ liệu của các bên tham gia hệ thống không sở hữu chung các đặc tính dữ liệu hay giá trị định danh của từng mẫu, người ta gọi đây là phân bố dữ liệu hỗn hợp. Ví dụ, khi các bên tham gia hệ thống là một ngân hàng ở Hoa Kỳ và một trường đại học ở Việt Nam. Do cản trở địa lý và nhu cầu quản lý thông tin khác nhau, chủ sở hữu dữ liệu này sẽ không có chung thuộc tính hay giá trị định danh nào. Trong trường hợp này, FTL có thể được sử dụng để chuyển giao tri thức giữa các bên tham gia.

Dựa vào các đặc điểm phân loại nêu trên, chúng tôi xếp nghiên cứu của mình vào nhóm hệ thống Horizontal FL tập trung.

2.2 Khảo sát dữ liệu Non-IID

Dữ liệu tại các máy khách thường được sinh ra dựa trên nhu cầu của người dùng cuối. Do đó, loại dữ liệu này thường có tính cá nhân hóa cao và không đồng nhất. Nói cách khác, không có bất kỳ phân phối dữ liệu cục bộ nào có thể đại diện cho phân phối trên toàn bộ dữ liệu, phân phối dữ liệu trên hai máy khách khác nhau là khác nhau [22]. Đây chính là ý tưởng mà thuật ngữ *dữ liệu Non-IID* muốn truyền đạt.

Mặt khác, nghiên cứu [21] chỉ ra rằng hệ thống FL có thể bị giảm hiệu quả nghiêm trọng khi đối mặt với dữ liệu Non-IID. Để hiểu rõ vấn đề mình đang đối mặt, dựa trên nghiên cứu [22], chúng tôi tiến hành khảo sát các kịch bản về dữ liệu Non-IID.

Gọi (x, y) là cặp thuộc tính và nhãn dữ liệu. Theo kịch bản dữ liệu Non-IID, phân phối dữ liệu của hai máy khách c_i, c_j bất kỳ là khác nhau: $P_i(x, y) \neq P_j(x, y)$. Nghiên cứu [22] nêu ra bốn kịch bản dữ liệu Non-IID: (1) - Phân phối thuộc tính khác nhau giữa các máy khách, (2) - Phân phối nhãn khác nhau giữa các máy khách, (3) - Phân phối thời gian khác nhau giữa các máy khách, (4) - Các kịch bản khác.

2.2.1 Phân phối thuộc tính khác nhau giữa các máy khách

Với kịch bản này, phân phối thuộc tính $P(x)$ trên các máy khách là đôi một khác nhau. Không gian thuộc tính của các máy khách có thể khác nhau hoàn toàn, trùng lặp một vài thuộc tính hoặc trùng lặp hoàn toàn.

Các hệ thống Vertical FL thường rơi vào trường hợp đầu tiên. Ví dụ, trong trường hợp dữ liệu dạng bảng, máy khách A có thể quản lý các thuộc tính A_1, A_2, A_3 trong khi máy khách B quản lý các thuộc tính B_1, B_2 của cùng một người dùng.

Đối với trường hợp thứ hai, hai máy khách có thể cùng quản lý một số thuộc tính dữ liệu. Ví dụ, đối với dữ liệu của một hệ thống camera dám sát, hai camera bất kỳ có thể cùng lưu hình một người với các góc chụp khác nhau.

Trường hợp cuối chính là đặc điểm chính của hệ thống Horizontal FL. Tại đây, không gian thuộc tính của các máy khách là hoàn toàn giống nhau.

Ví dụ, tập dữ liệu MNIST chứa các mẫu chữ viết tay của nhiều người. Do đó, cùng là một chữ số, nhưng độ đậm nhạt, cách viết người dùng A sẽ khác với người dùng B.

2.2.2 Phân phối nhãn khác nhau giữa các máy khách

Đây là trường hợp dữ liệu Non-IID phổ biến nhất, gây hại nghiêm trọng cho hệ thống FL [22], cũng chính là trường hợp mà chúng tôi hướng tới giải quyết. Tại đây, phân phối nhãn của hai máy khách c_i, c_j bất kỳ là khác nhau: $P_i(y) \neq P_j(y)$ và xác suất thuộc tính x có nhãn dữ liệu y: $P(x|y)$ của các máy khách là như nhau. Một kịch bản thường thấy của trường hợp này được trình bày trong nghiên cứu [14]: Mỗi máy khách sẽ chỉ chứa các điểm dữ liệu thuộc về k nhãn. Trong đó, k là một siêu tham số biểu diễn độ mất cân bằng về nhãn. k càng nhỏ, hệ thống mất cân bằng nhãn càng mạnh. Trong khóa luận này, chúng tôi thiết đặt môi trường dữ liệu theo cách tương tự như vậy. `#todo: chèn thêm ảnh`

Ngoài ra, còn một trường hợp phổ biến khác liên quan đến việc dữ liệu Non-IID trên nhãn. Đối với trường hợp này, xác suất thuộc tính x được gán nhãn y: $P(y|x)$ là khác nhau giữa các máy khách. Ví dụ, với một bức ảnh trên mạng xã hội, người dùng A có thể gán nhãn "yêu thích", trong khi người dùng B gán nhãn "không yêu thích".

2.2.3 Phân phối thời gian khác nhau giữa các máy khách

Một ví dụ dễ hiệu cho trường hợp này là việc hai người dùng c_i, c_j thu thập dữ liệu trong hai khoảng thời gian khác nhau. Dẫn đến việc phân phối $P_i(x, y|t) \neq P_j(x, y|t)$, với t là một thời điểm nhất định. Một trường hợp khác của kịch bản này là phân phối $P(x, y|t)$ của một máy khách bị thay đổi liên tục theo thời gian. Ví dụ, một hệ thống camera giám sát có thể ghi nhận hình ảnh của rất nhiều người vào các ngày làm việc trong tuần nhưng lại có rất ít hình ảnh vào những ngày nghỉ.

2.2.4 Các kịch bản khác

Các kịch bản còn lại thường rơi vào hai trường hợp: (1) - Phân phối thuộc tính và nhãn là khác nhau giữa các máy khách, (2) - Số lượng dữ

liệu huấn luyện là khác nhau giữa các máy khách.

2.3 Tối ưu hệ thống Federated Learning trên dữ liệu Non-IID

2.3.1 Tối ưu dựa trên dữ liệu

Việc mô hình toàn cục làm việc với dữ liệu có phân phối không đồng nhất dẫn đến việc các lớp được học một cách không đồng đều (có một số lớp được học ít hơn/nhiều hơn các lớp khác), khiến cho mô hình bị giảm hiệu suất [22]. Hướng tối ưu dựa trên dữ liệu trực tiếp giải quyết vấn đề này bằng hai cách: (1) - Chia sẻ dữ liệu, (2) - Tăng cường dữ liệu.

Chia sẻ dữ liệu. Chia sẻ dữ liệu [21] được thực hiện bằng cách xây dựng một tập dữ liệu chứa dữ liệu của tất cả các nhãn theo phân phối đều. Dữ liệu trong tập này được thu thập trực tiếp từ các máy khách và gửi về máy chủ để kết hợp huấn luyện mô hình toàn cục.

Tăng cường dữ liệu. Cùng với ý tưởng cho phép mô hình toàn cục được học trên các tập dữ liệu có phân phối đều các nhãn trong hệ thống, tăng cường dữ liệu [16] nhằm mục đích gia tăng sự đa dạng của dữ liệu huấn luyện. Phương pháp này đòi hỏi các máy khách phải gửi phân phối dữ liệu của mình về máy chủ. Máy chủ theo đó yêu cầu các máy khách tạo ra ảnh mới [5] với số lượng và nhãn lớp biết trước, hoặc tự mình tạo ra ảnh mới bằng cách sử dụng GAN [22] để có thể huấn luyện trên một tập dữ liệu chứa tất cả các nhãn với phân phối trung bình.

Các phương pháp nêu trên đều giúp hệ thống FL "chống chịu" tốt trước dữ liệu Non-IID. Tuy nhiên, đòi hỏi máy khách gửi thông tin cá nhân về máy chủ là vi phạm mục tiêu ban đầu của hệ thống FL - bảo vệ quyền riêng tư dữ liệu của người dùng.

2.3.2 Tối ưu dựa trên thuật toán

Tinh chỉnh cục bộ

Tinh chỉnh cục bộ, hay local fine-tuning, là kỹ thuật mạnh mẽ trong việc cá nhân hóa mô hình học cho các tập dữ liệu riêng biệt. Kỹ thuật này

hướng đến việc tinh chỉnh mô hình học tại các máy khách sau khi nhận được mô hình từ máy chủ [17].

Một hướng tiếp cận phổ biến được đề ra là sử dụng ML trong việc tạo ra một mô hình toàn cục tốt, có thể thích ứng với tập dữ liệu mới trên máy khách một cách nhanh chóng. Các thuật toán theo hướng này [4, 6] sử dụng các kỹ thuật ML có khả năng tạo ra một khởi tạo tốt như Model-Agnostic Meta-Learning (MAML) [7] hay Meta-SGD [11] để huấn luyện mô hình toàn cục. Mô hình toàn cục này, trong quá trình chạy thực tế trên một máy khách mới, hoàn toàn có thể đạt hội tụ chỉ sau một hoặc một vài bước huấn luyện.

Lớp cá nhân hóa

Các thuật toán theo hướng này cho phép duy trì một phần của mạng học sâu trên máy khách. Cụ thể, thuật toán chia mạng học sâu thành hai thành phần: phần chung và phần riêng. Phần chung được hợp tác huấn luyện bởi các máy khách và được tổng hợp bởi máy chủ. Phần riêng tồn tại riêng biệt trên từng máy khách, được máy khách trực tiếp duy trì và huấn luyện.

Một thuật toán điển hình theo hướng tiếp cận này là FedPer [2]. FedPer quy định phần chung của mạng học sâu là các lớp rút trích đặc trưng, phần riêng của mạng là các lớp còn lại. Các thí nghiệm đã cho thấy thuật toán này đạt điệu quả cao hơn nhiều so với FedAvg khi làm việc trên dữ liệu Non-IID.

Thuật toán LG-FedAvg [12] cũng được xếp vào nhóm thuật toán sử dụng lớp cá nhân hóa. Tuy nhiên, ngược lại với FedPer, LG-FedAvg chỉ định phần riêng là các lớp rút trích đặc trưng trong mạng học sâu. Thực nghiệm cho thấy, LG-FedAvg đạt hiệu quả tốt hơn FedAvg trên cả các máy khách sẵn có trong hệ thống lẫn các máy khách chỉ vừa mới tham gia hệ thống.

Bằng các lớp cá nhân hóa, các thuật toán nêu trên đã giải quyết được sự khác nhau về dữ liệu giữa các máy khách, từ đó tránh được phần nào sự giảm hiệu suất trên dữ liệu Non-IID. Nhưng các lớp thuộc phần chung của mạng học sâu vẫn có thể bị thiên kiến (bias) do dữ liệu huấn luyện

không tuân theo phân phối đều. **Vậy có thể làm gì để các lớp này có thể thích ứng nhanh với tập dữ liệu của máy khách chỉ sau một vài bước huấn luyện?**

2.3.3 Tối ưu dựa trên hệ thống

Gom cụm người dùng

Các tiếp cận FL truyền thống giả định rằng hệ thống này chỉ bao gồm một máy chủ. Điều này làm cho việc học các đặc tính của tất cả các máy khách trong môi trường dữ liệu Non-IID là khó khả thi. Để giải quyết vấn đề này, một hệ thống huấn luyện với nhiều máy chủ được đề xuất. Câu hỏi đặt ra là: Làm sao để biết một máy khách nên huấn luyện cùng với máy chủ nào?

Trong ngữ cảnh đa máy chủ, thuật toán IFCA [8] trả lời câu hỏi trên bằng cách gửi trọng số của tất cả các máy chủ cho từng máy khách. Các máy khách theo đó tìm ra được trọng số cho độ lỗi nhỏ nhất trên tập dữ liệu cục bộ và gửi thông tin sau khi huấn luyện cục bộ của mình về máy chủ đó để cập nhật mô hình toàn cục. Việc gửi toàn bộ trọng số của các máy chủ đến một máy khách khiến cho thuật toán này tăng chi phí giao tiếp lên gấp k lần, với k là số lượng máy chủ của hệ thống.

Một cách khác để trả lời câu hỏi trên là đánh giá sự tương đồng của trọng số do máy khách gửi về [22]. Một thang độ đo sự tương đồng như độ đo cosine được máy chủ sử dụng trên các trọng số của máy khách. Từ đó biết được nên tổng hợp trọng số của các máy khách nào với nhau.

Với việc lượng dữ liệu và thiết bị biên ngày càng tăng lên, việc duy trì nhiều máy chủ là thực sự cần thiết khi đối mặt với nhu cầu nâng cấp hệ thống học. Tuy nhiên, chi phí giao tiếp và phương pháp gom cụm vẫn là những vấn đề rất lớn cần giải quyết.

Chương 3

Phương pháp đề xuất

Trong chương này, chúng tôi khảo sát hai và phân tích về ML và PL - hai hướng tiếp cận giúp cải thiện hiệu suất của hệ thống FL trên dữ liệu Non-IID. Từ đó đề xuất thuật toán FedMeta-Per, là sự kết hợp của hai kỹ thuật ML và PL vào hệ thống FL.

3.1 Tiếp cận hệ thống theo hướng Meta Learning

Meta Learning được áp dụng vào hệ thống FL như một phương pháp tối ưu thuộc nhóm "Tinh chỉnh cục bộ (local fine-tuning)" [22]. Các thuật toán ML được sử dụng trong hệ thống FL nhằm mục đích tạo ra một mô hình toàn cục tốt, giúp hội tụ nhanh trên tập dữ liệu phân bố trên các máy khách.

3.1.1 Diễn giải Meta Learning

ML là một phương pháp học mới, cho phép mô hình học có thể gia tăng kinh nghiệm qua việc thực hiện nhiều nhiệm vụ khác nhau trong cùng một phân phối nhiệm vụ. Dẫn đến việc các mô hình ML có khả năng thích ứng nhanh trên nhiệm vụ mới chỉ sau một vài bước huấn luyện và dữ liệu huấn luyện giới hạn. Đây là một phát kiến quan trọng, đóng vai trò trong việc đưa cách học tập của máy trở nên tiệm cận với cách học tập của con người chèn chú thích 8 trong survey ML.

Đối với phương pháp huấn luyện mô hình học truyền thống, chúng ta huấn luyện mô hình dự đoán $\hat{y} = f_\theta(x)$ trên tập dữ liệu $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^m$ của nhiệm vụ T gồm các cặp thuộc tính và nhãn tương ứng. Ký hiệu $\mathcal{L}$ là hàm lỗi, ω là giả định ban đầu của hệ thống học, mục tiêu của việc học là tối thiểu hóa hàm lỗi trên tập dữ liệu $\mathcal{D}$ bằng cách tìm một bộ trọng số w^* thỏa mãn:

$$w^* = \arg\min_w \mathcal{L}(\mathcal{D}; w, \omega) \tag{3.1}$$

Hướng tiếp cận của ML nằm ở chỗ cố gắng học một giả định ban đầu ω thật tốt. Điều này đạt được thông qua việc học một phân phối các nhiệm vụ $p(T)$ [9]. Sau khi học được một giả định ban đầu tốt, có thể áp dụng giả định này cho các nhiệm vụ mới trong cùng phân phối nhiệm vụ: $T \sim p(T)$.

Về mặt công thức, ký hiệu $\mathcal{L}(\mathcal{D}, \omega)$ là hàm số biểu diễn sự hiệu quả việc sử dụng ω trong huấn luyện nhiệm vụ T có tập dữ liệu $\mathcal{D}$, chúng ta có thể biểu diễn hàm mục tiêu của ML như sau:

$$\min_\omega \mathop{\mathbb{E}}_{T\sim p(T)} \mathcal{L}(\mathcal{D}, \omega) \tag{3.2}$$

Trong thực tế, người ta thực hiện mục tiêu trên bằng cách huấn luyện mô hình học trên tập dữ liệu $\mathcal{D}_{train} = \{(\mathcal{D}^{support}_{train(i)}, \mathcal{D}^{query}_{train(i)})\}_{i=1}^{|\mathcal{D}_{train}|}$ và kiểm thử trên tập dữ liệu $\mathcal{D}_{test} = \{(\mathcal{D}^{support}_{test(i)}, \mathcal{D}^{query}_{test(i)})\}_{i=1}^{|\mathcal{D}_{test}|}$. Mục tiêu của việc huấn luyện là tìm ra một giá trị ω^*, sao cho khi sử dụng giá trị này trong huấn luyện một nhiệm vụ $T \sim p(T)$ thì đạt được hiệu quả cao:

$$\omega^* = \arg\max_\omega \log p(\omega|\mathcal{D}_{source}) \tag{3.3}$$

Trong quá trình kiểm thử, tham số ω^* được sử dụng trong việc huấn luyện mô hình giải quyết nhiệm vụ T_{new}: $w^* = \arg\max_w \log p(w|\omega^*, \mathcal{D}^{support}_{test(new)})$. Để đánh giá hiệu quả của việc sử dụng ω trong huấn luyện nhiệm vụ T_{new}, người ta dựa vào kết quả của w^* trên tập $\mathcal{D}^{query}_{test(new)}$.

Để giải phương 3.3, chúng tôi nhìn nhận ML dưới góc độ một bài toán tối ưu hai cấp độ [9]. Dưới góc nhìn này, phương trìnhh 3.3 được giải bằng cách đạt được mục tiêu tại hai cấp độ: (1) - Cấp độ thấp, (2) - Cấp độ cao. Đối với cấp độ thấp, mục tiêu là giải quyết nhiệm vụ T_i dựa vào ω:

$$w_i^*(\omega) = \arg\min_w \mathcal{L}^{task}(w, \omega, \mathcal{D}^{support}_{train(i)}) \tag{3.4}$$

Giải phương trình 3.4 bằng kỹ thuật SGD, ta được lời giải sau:

$$\begin{cases} w_{i(0)} = \omega \\ w_{i(j)} = w_{i(j-1)} - \alpha \nabla \mathcal{L}^{task}(w_{i(j-1)}, \omega, \mathcal{D}^{support}_{train(i)}) \end{cases} \tag{3.5}$$

Hay:

$$w_i \leftarrow w_i - \alpha \nabla \mathcal{L}^{task}(w_i) \tag{3.6}$$

Đối với cấp độ cao, mục tiêu là tìm ra tham số ω^* tối ưu, giúp việc học một nhiệm vụ mới $T_{new} \sim p(T)$ được thực hiện nhanh chóng và đạt hiệu suất cao:

$$\omega^* = \arg\min_{\omega} \sum_{i=1}^{|\mathcal{D}_{souece}|} \mathcal{L}^{meta}(w_i^*(\omega), \omega, \mathcal{D}^{query}_{train(i)}) \tag{3.7}$$

Áp dụng kỹ thuật SGD, lời giải cần tìm của bài toán ML được biểu diễn như sau:

$$\begin{cases} \omega_0 = \Omega, \Omega \text{ là giá trị khởi tạo ngẫu nhiên} \\ \omega_j = \omega_{j-1} - \beta \nabla \sum_{i=1}^{|\mathcal{D}_{souece}|} \mathcal{L}^{meta}\left(w_i^*(\omega), \omega, \mathcal{D}^{query}_{train(i)}\right) \end{cases} \tag{3.8}$$

Hay:

$$\begin{aligned} \omega &\leftarrow \omega - \beta \nabla \sum_{i=1}^{|\mathcal{D}_{souece}|} \mathcal{L}^{meta}\left(w_i^*(\omega)\right) \\ &\leftarrow \omega - \beta \nabla \sum_{i=1}^{|\mathcal{D}_{souece}|} \mathcal{L}^{meta}\left(w_i - \alpha \nabla \mathcal{L}^{task}(w_i)\right) \\ &\leftarrow \omega - \beta \sum_{i=1}^{|\mathcal{D}_{souece}|} \left(I - \alpha \nabla^2 \mathcal{L}^{task}(w_i)\right) \times \nabla \mathcal{L}^{meta}\left(w_i - \alpha \nabla \mathcal{L}^{task}(w_i)\right) \end{aligned} \tag{3.9}$$

3.1.2 Hệ thống Federated Learning và Meta Learning

Đối chiếu hai phương trình 3.4 và 3.7 với hai mục tiêu của hệ thống FL được trình bày trong phần 2.1.2, có thể thấy rõ điểm tương đồng giữa hệ thống FL và hệ thống ML. Theo đó, các mục tiêu cấp thấp trong ML

tương đồng với các mục tiêu cục bộ trong hệ thống FL, mục tiêu cấp cao trong hệ thống ML tương đồng với mục tiêu toàn cục của hệ thống FL. Từ đây, có thể dễ dàng kết hợp ML và FL bằng cách thay thế hệ hàm mục tiêu của hệ thống FL bằng hệ hàm mục tiêu của ML.

Thật vậy, nghiên cứu [6] đã tích hợp thuật toán MAML vào hệ thống FL và biểu diễn lại hàm mục tiêu toàn cục của hệ thống từ phương trình 2.4 như sau:

$$\min_{w_G} f_{global}(w_G) = \min_{w_G} \frac{1}{n} \sum_{i=1}^{n} f_{local}\left(w_i - \alpha \nabla f_{local}(w_i, \mathcal{D}_{train(i)}^{support}), \mathcal{D}_{train(i)}^{query}\right) \tag{3.10}$$

Trong hàm số 3.10, ta có thể nhận thấy sự xuất hiện của hai tập dữ liệu $\mathcal{D}_{train(i)}^{support}$ và $\mathcal{D}_{train(i)}^{query}$. Điều này có nghĩa là hệ thống FL huấn luyện theo hướng ML cần phải chia lại tập dữ liệu tại các máy khách theo kiểu ML. Một máy khách c_i tham gia huấn luyện bao gồm hai tập dữ liệu $\mathcal{D}_{train(i)}^{support}$ và $\mathcal{D}_{train(i)}^{query}$. Trong quá trình kiểm thử, dữ liệu của máy khách c_i cũng được chia thành hai tập $\mathcal{D}_{test(i)}^{support}$ và $\mathcal{D}_{test(i)}^{query}$. Trong đó, c_i cần thực hiện tinh chỉnh (fine-tune) mô hình trên tập $\mathcal{D}_{test(i)}^{support}$ và đánh giá mô hình trên tập $\mathcal{D}_{test(i)}^{query}$.

Từ phương trình tổng hợp mô hình toàn cục bằng phương pháp lấy trung bình trọng số 2.5, bằng phương pháp SGD như phương trình 3.9, phương trình tổng hợp mô hình toàn cục trong hệ thống FL tích hợp ML tại bước huấn luyện toàn cục thứ t có dạng:

$$\begin{aligned} w_G^{t+1} &= \sum_{i=1}^{n} \frac{n_i}{N} w_i^{t+1} \\ &= \sum_{i=1}^{n} \frac{n_i}{N} \left[w_i^t - \beta \left(I - \alpha \nabla^2 f_{local}(w_i^t)\right) \times \nabla f_{local}\left(w_i^t - \alpha \nabla f_{local}(w_i^t)\right)\right] \end{aligned} \tag{3.11}$$

Ngoài thuật toán MAML, dựa theo nghiên cứu [4], chúng tôi tích hợp thêm vào hệ thống của mình thuật toán Meta-SGD [11]. Tương tự như MAML, Meta-SGD được cấu trúc theo hướng tối ưu hai cấp độ. Tuy nhiên, có một thay đổi nhỏ giúp thuật toán này đạt độ chính xác cao hơn MAML: thuật toán coi siêu tham số học α là một tham số có thể học được và tiến hành tối ưu α trong mục tiêu cấp cao.

Tóm lại, cả hai nghiên cứu [4, 6] đã chứng minh được việc tích hợp ML vào hệ thống FL giúp đạt hiệu quả cao hơn `FedAvg` về độ chính xác trên cả hai phương diện lý thuyết và thực nghiệm. Chúng tôi dựa theo nghiên cứu [4] để cài đặt hai thuật toán `FedMetaMAML` và `FedMetaSGD` với mục đích tìm ra các điểm hạn chế, tứ đó tối ưu chúng để hệ thống đạt độ chính xác cao hơn.

3.1.3 Thuật toán $FedMeta(MAML)$ và $FedMeta(Meta-SGD)$

Algorithm 1 FedMeta(MAML) và FedMeta(Meta-SGD) [4]

// Tại máy chủ
AlgorithmUpdate:
Khởi tạo w cho MAML hoặc (w, α) cho Meta-SGD.
for $t = 0, 1, 2, ...$ **do**
    Chọn một tập C_t gồm m máy khách
    Phân phối w (đối với MAML) hoặc (w, α) (đối với Meta-SGD) tới các máy khách $c \in C_t$.
    for $c \in C_t$ **do**
        Tính toán $g_c \leftarrow$ ModelTrainingMAML(w) cho MAML
        Tính toán $g_c \leftarrow$ ModelTrainingMetaSGD(w, α) cho Meta-SGD

    // n_c, N_m là số điểm dữ liệu trên máy khách c và trên m máy khách
    Cập nhật $w_G \leftarrow w_G - \beta \sum_{c \in C_t} \frac{n_c}{N_m} g_c$ cho MAML
    Cập nhật $(w, \alpha) \leftarrow (w, \alpha) - \beta \sum_{c \in C_t} \frac{n_c}{N_m} g_c$ cho Meta-SGD

// Tại máy khách c
ModelTrainingMAML(w):
Chọn tập support $\mathcal{D}^{support}_{train(c)}$ và tập query $\mathcal{D}^{query}_{train(c)}$
$w' \leftarrow w - \alpha \nabla f_{local}(w, \mathcal{D}^{support}_{train(c)})$
$g_c \leftarrow \nabla_w f_{local}(w', \mathcal{D}^{query}_{train(c)})$
Gửi g_c về máy chủ

ModelTrainingMetaSGD(θ, α):
Chọn tập support $\mathcal{D}^{support}_{train(c)}$ và tập query $\mathcal{D}^{query}_{train(c)}$
$w' \leftarrow w - \alpha \circ \nabla f_{local}(w, \mathcal{D}^{support}_{train(c)})$
$g_c \leftarrow \nabla_{(w,\alpha)} f_{local}(w', \mathcal{D}^{query}_{train(c)})$
Gửi g_c về máy chủ

Nghiên cứu [4] đề xuất kết hợp MAML và Meta-SGD vào hệ thống FL của họ (thuật toán 1). Tuy nhiên, tác giả sử dụng kỹ thuật cập nhật bằng cách lấy trung bình đạo hàm của hàm lỗi. Trước hết, máy khách sau khi nhận được trọng số toàn cục sẽ tiến hành tinh chỉnh (fine-tune) trọng số này trên tập dữ liệu support (phương trình 3.12 và 3.13). Như vậy, mô hình sau khi tinh chỉnh sẽ nắm bắt được các đặc trưng riêng của bộ dữ liệu. Từ đó thích ứng tốt với dữ liệu query.

$$\text{MAML: } w' \leftarrow w - \alpha \nabla f_{local}(w, \mathcal{D}^{support}_{train(c)}) \tag{3.12}$$

$$\text{Meta-SGD: } w' \leftarrow w - \alpha \circ \nabla f_{local}(w, \mathcal{D}^{support}_{train(c)}) \tag{3.13}$$

Trọng số sau khi tinh chỉnh được dùng để dự đoán phân lớp trên tập dữ liệu query và tính toán hàm lỗi dựa trên kết quả dự đoán:

$$\text{MAML: } g_c \leftarrow \nabla_w f_{local}(w', \mathcal{D}^{query}_{train(c)}) \tag{3.14}$$

$$\text{MAML: } g_c \leftarrow \nabla_{(w,\alpha)} f_{local}(w', \mathcal{D}^{query}_{train(c)}) \tag{3.15}$$

Kết quả đạo hàm trên được gửi trực tiếp về máy chủ để tổng hợp bộ trọng số toàn cục mới:

$$\text{MAML: } w_G \leftarrow w_G - \beta \sum_{c \in C_t} \frac{n_c}{N_m} g_c \tag{3.16}$$

$$\text{Meta-SGD: } (w, \alpha) \leftarrow (w, \alpha) - \beta \sum_{c \in C_t} \frac{n_c}{N_m} g_c \tag{3.17}$$

Trong nghiên cứu của mình, chúng tôi sử dụng kỹ thuật lấy trung bình trọng số trong việc tổng hợp trọng số toàn cục vì chi phí giao tiếp của việc truyền thông tin đạo hàm từ máy khách về máy chủ vượt quá giới hạn phần cứng được cung cấp. Đặt trong ngữ cảnh của ML, nếu bỏ qua việc mất gói tin trong quá trình giao tiếp giữa máy chủ và máy khách, việc sử dụng trọng số để truyền tin không ảnh hưởng đến kết quả của bài toán và được chứng minh đối với MAML như sau:

Tại bước huấn luyện toàn cục thứ t với sự tham gia của m máy khách, phương trình 3.11 trở thành:

$$
\begin{aligned}
w_G^{t+1} &= \sum_{i=1}^{m} \frac{n_i}{N_m} \left[w_i^t - \beta \left(I - \alpha \nabla^2 f_{local}(w_i^t) \right) \times \nabla f_{local} \left(w_i^t - \alpha \nabla f_{local}(w_i^t) \right) \right] \\
&= w_G^t - \beta \sum_{i=1}^{m} \frac{n_i}{N_m} \left(I - \alpha \nabla^2 f_{local}(w_i^t) \right) \times \nabla f_{local} \left(w_i^t - \alpha \nabla f_{local}(w_i^t) \right) \\
&= w_G^t - \beta \sum_{i=1}^{m} \frac{n_i}{N_m} g_c^{t+1}
\end{aligned}
\tag{3.18}
$$

Từ phương trình 3.18 và 3.16, ta suy ra điều cần chứng minh. Đối với Meta-SGD và các bước thự hiện như trên, ta thu được kết quả chứng minh tương tự.

3.2 Tiếp cận hệ thống theo hướng Personalization Layer

Chúng tôi tiến hành khảo sát kết quả của các kỹ thuật tối ưu hệ thống FL trên dữ liệu Non-IID và nhận thấy các thuật toán theo hướng PL đạt kết quả rất cao #todo: chèn bảng kq lấy từ paperwithcode. Các thuật toán theo hướng này chia mạng học sâu ra làm hai phần [22]: phần chung và phần riêng. Theo đó, phần chung được hợp tác huấn luyện bởi tất cả các máy khách trong hệ thống còn phần riêng được từng máy khách huấn luyện riêng biệt trên tập dữ liệu cục bộ. Chính các lớp học sâu trong phần riêng đã làm cho cách tiếp cận này trở nên mạnh mẽ trên dữ liệu có tính cá nhân hóa cao như dữ liệu Non-IID.

Tuy nhiên, như câu hỏi đã được nêu ra trong phần 2.3.2, chúng tôi nhận thấy, hiệu suất của hệ thống FL cài đặt theo hướng PL vẫn có thể được cải thiện vì các lớp phần chung chưa thực sự mạnh mẽ và còn phụ thuộc nhiều vào dữ liệu. Cụ thể, các lớp phần chung này có thể bị bias do phân bố dữ liệu không đồng đều trong kịch bản Non-IID. Do đó, cần cải thiện cách huấn luyện của các lớp học sâu trong phần chung để chúng bớt phụ thuộc vào dữ liệu hơn. Nói cách khác, các lớp này cần có khả năng làm việc khách quan, "đối xử" công bằng hơn với các tập dữ liệu riêng biệt trong hệ thống FL.

Như đã thảo luận trước đó, các thuật toán ML có khả năng thích ứng nhanh trên tập dữ liệu mới thông qua việc học nhiều nhiệm vụ khác nhau. Hơn nữa, dựa vào phân tích tính tương đồng về hệ hàm mục tiêu của FL và ML ở trên, ta có thể coi các nhiệm vụ khác nhau của ML chính là các máy khách trong hệ thống FL. Do đó, chúng tôi sử dụng phương pháp huấn luyện mô hình của ML vào huấn luyện các lớp phần chung nhằm thu được các lớp phần chung có khả năng ít bị bias trên dữ liệu huấn luyện và thích ứng tốt trên dữ liệu mới.

Hai thuật toán điển hình của hướng tối ưu này là `FedPer` [2] và `LG-FedAvg` [12]. Trong phần này, chúng tôi tiến hành tìm hiểu, phân tích hai thuật toán trên và đưa ra các nhận xét về ưu, nhược điểm của từng thuật toán. Từ đó kết hợp chúng với các thuật toán `FedMeta` ở trên.

3.2.1 Thuật toán *FedPer*

`#todo: chèn ảnh`

Nghiên cứu [2] đề xuất kiến trúc hệ thống FL theo hướng PL bằng cách chia mạng học sâu ra làm hai phần. Phần chung được hợp tác huấn luyện bởi các máy khách trong hệ thống và được tổng hợp bởi máy chủ. Phần riêng được các máy khách độc lập huấn luyện trên dữ liệu của mình.

Các lớp học sâu trong phần chung là các lớp rút trích đặc trưng của mạng. Vì được hợp tác huấn luyện, các lớp phần chung này được tiếp xúc với đầy đủ các phân lớp dữ liệu. Do đó, sẽ có được khả năng rút trích được đặc trưng dữ liệu của tất cả các nhãn dữ liệu trong hệ thống. Điều này theo chúng tôi là rất quan trọng và là điểm khác biệt chính khi so sánh giữa `FedPer` và `LG-FedAvg`.

Phần riêng của mạng bao gồm các lớp tuyến tính ở mức cao. Phần này sẽ sử dụng các đặc trưng dữ liệu được rút trích ở trên để tính toán và quyết định một mẫu dữ liệu đầu vào sẽ thuộc phân lớp nào. Vì được duy trì riêng tại mỗi máy khách, các lớp phần riêng này được tối ưu riêng cho dữ liệu trên máy khách đó. Đây chính là điểm đáng giá của thuật toán `FedPer` khi nó giúp nắm bắt điểm riêng biệt trong phân phối dữ liệu của từng máy khách mà thuật toán `FedAvg` không thể nào làm được.

Ký hiệu w_{c_i}, w_B lần lượt là trọng số của các lớp phần riêng của máy

khách c_i và phần chung của hệ thống. Hàm phân lớp cần huấn luyện tại máy khách c_i là $\hat{y}_i = g(x, w_B, w_{c_i})$. Trong đó, trọng số w_B có nhiệm vụ rút trích các đặc trưng của dữ liệu đầu vào x còn w_{c_i} chịu trách nhiệm phân lớp dữ liệu x cũng như lưu trữ tính cá nhân hóa của máy khách c_i. Theo đó, hàm mục tiêu của hệ thống có thể được biểu diễn:

$$\min_{w_B, w_{c_1}, ..., w_{c_n}} f_{global}(w_B, w_{c_1}, ..., w_{c_n}) = \min_{w_B, w_{c_1}, ..., w_{c_n}} \frac{1}{n} \sum_{i=1}^{n} f_{local}(w_B, w_{c_i}) \quad (3.19)$$

Các hoạt động chính của máy chủ và máy khách được trình bày trong thuật toán 3 và 2. Đầu tiên, máy chủ khởi tạo trọng số phần chung w_B^0 và gửi trọng số này đến các máy khách trong một bước huấn luyện. Máy khách c_i nhận w_B^0 từ máy chủ đồng thời khởi tạo trọng số phần riêng $w_{c_i}^0$. Bộ trọng số $(w_B^0, w_{c_i}^0)$ sau đó được máy khách c_i sử dụng để dự đoán và huấn luyện cục bộ. Tại bước huấn luyện toàn cục thứ t, ta có:

$$H = h(x, w_{B_i}^t) \quad (3.20)$$

$$\hat{y} = g(H, w_{c_i}^t) \quad (3.21)$$

$$w_{B_i}^{t+1} \leftarrow w_B^t - \alpha \nabla_{w_B^t} f_{local}(x, w_B^t, w_{c_i}^t) \quad (3.22)$$

$$w_{c_i}^{t+1} \leftarrow w_{c_i}^t - \alpha \nabla_{w_{c_i}^t} f_{local}(x, w_B^t, w_{c_i}^t) \quad (3.23)$$

Kết thúc quá trình huấn luyện cục bộ, $w_{c_i}^{t+1}$ được lưu trữ lại còn $w_{B_i}^{t+1}$ được gửi về máy chủ để tổng hợp w_B^{t+1}:

$$w_B^{t+1} = \sum_{i=1}^{n} \frac{n_i}{N} w_{B_i}^{t+1} \quad (3.24)$$

Algorithm 2 FEDPER-CLIENT(c_i) [2]

Require: Siêu tham số học α

1: Khởi tạo $w_{c_i}^0$
2: Nhận w_B^t từ máy chủ
3: Tính toán $(w_{B_i}^t, w_{c_i}^t) \leftarrow (w_{B_i}^t, w_{c_i}^t) - \alpha \nabla_{(w_{B_i}^t, w_{c_i}^t)} f_{local}(w_{B_i}^t, w_{c_i}^t)$
4: Gửi $w_{B_c}^t$ và số điểm dữ liệu n_i về máy chủ

Algorithm 3 FEDPER-SERVER [2]

Khởi tạo w_B^0
Gửi w_B^0 cho các máy khách
for $t = 0, 1, 2, ...$ **do**
    Chọn một tập C_t gồm m máy khách
    for $c_i \in C_t$ **do**
        Gửi w_B^t cho c_i
        Tính toán $(w_{B_i}^{t+1}, n_i) \leftarrow$ FEDPER-CLIENT(c_i)
    Cập nhật $w_B^{t+1} \leftarrow \sum_{i=1}^{n} \frac{n_i}{\sum n_i} w_{B_i}^{t+1}$

Sau khi hoàn thành giai đoạn huấn luyện toàn cục, hệ thống thu được một trọng số phần chung và n trọng số phần riêng. Quá trình kiểm thử trên tập dữ liệu của máy khách mới được tiến hành bằng thông qua bộ tham số (w_B, w_P). Trong đó, w_P là trung bình cộng của các trọng số $(w_{c_1}, ..., w_{c_n})$ được tính bằng phương trình 3.25.

$$w_P = \sum_{i=1}^{n} \frac{n_i}{N} w_{c_i} \tag{3.25}$$

Một lần nữa, trọng số của hệ thống có thể bị bias trên dữ liệu huấn luyện. Bộ trọng số $(w_{c_1}, ..., w_{c_n})$ có thể mang đến sự cá nhân hóa rất tốt trên các máy khách đã tồn tại từ lâu trong hệ thống. Trọng số w_B được hợp tác huấn luyện nên có thể đã "quen thuộc" hơn đối với dữ liệu thuộc các nhãn khác nhau nhưng vẫn khó tránh khỏi hiện tượng bias trên dữ liệu huấn luyện. Trên các máy khách vừa tham gia hệ thống, hai trọng số này sẽ gặp tình trạng tương tự như thuật toán `FedAvg`: Bị giảm hiệu suất nghiêm trọng trên dữ liệu Non-IID.

3.2.2 Thuật toán $LG - FedAvg$

Ngoại trừ việc phần chung của mạng học sâu là các lớp tuyến tính còn phần riêng của mạng là các lớp rút trích đặc trưng, ý tưởng huấn luyện của thuật toán `LG-FedAvg` (thuật toán 4) gần như giống hoàn toàn với thuật toán `FedPer`. Các lớp phần riêng của thuật toán này được kỳ vọng sẽ học những đặc trưng dữ liệu của từng máy khách một cách riêng biệt. Tuy nhiên, khi đối mặt với các phân phối dữ liệu lạ trên các máy khách vừa

tham gia vào hệ thống, các lớp phần riêng tỏ ra khó khăn trong việc nắm bắt các đặc trưng mới vì chúng đã được cá nhân hóa rất cao cho các đặc trưng cục bộ trước đó. Dẫn đến việc độ chính xác của hệ thống giảm từ 31% đến 34% khi hoạt động trên các máy khách mới khi so sánh với độ chính xác trên các máy khách cũ [12].

Algorithm 4 LG-FEDAVG [12]

Require: Khởi tạo w_B tại máy chủ và các $(w_{c_1}, ..., w_{c_n})$ tại các máy khách

Server executes:
for $t = 1, 2, ...$ **do**
    Chọn một tập C_t gồm m máy khách
    for $c_i \in C_t$ **do**
        Tính toán $w_{B_i}^{t+1} \leftarrow ClientUpdate(c_i, w_B^t)$
    // n_i, N_m là số điểm dữ liệu trên máy khách c_i và trên m máy khách
    Cập nhật $w_B^{t+1} \leftarrow \sum_{i=1}^{m} \frac{n_i}{N_m} w_{B_i}^{t+1}$

ClientUpdate (c_i, w_B):
Tính toán $(w_{B_i}, w_{c_i}) \leftarrow (w_{B_i}, w_{c_i}) - \alpha \nabla_{(w_{B_i}, w_{c_i})} f_{local}(w_{B_i}, w_{c_i})$
Gửi w_B về máy chủ

Điểm làm chúng tôi chú ý đến nghiên cứu này chính là kịch bản mà nó xây dựng trong quá trình kiểm thử rất phù hợp với các tình huống thực tế của một hệ thống client-server. Kịch bản kiểm thử của nghiên cứu [12] chia người dùng ra làm hai loại: (1) - người dùng cũ, (2) - người dùng mới.

Đối với người dùng cũ, nghiên cứu cho rằng loại người dùng này đã tồn tại đủ lâu trong hệ thống để có thể xây dựng được một lớp cá nhân hóa cho chính nó. Khi làm việc với loại dữ liệu trên máy khách của họ, hệ thống biết chính xác nên sử dụng bộ trọng số nào là phù hợp nhất.

Đối với người dùng mới, nghiên cứu giả định họ là những người vừa tham gia vào hệ thống. Do đó, hệ thống không thể biết được nên sử dụng trọng số nào để làm việc với phân phối dữ liệu của họ. Chính vì vậy, nghiên cứu đề xuất thực hiện ensemble test trên loại người dùng này.

Dựa trên việc phân chia dữ liệu theo hướng ML, chúng tôi không đồng ý với cách tiếp cận ensemble test trên người dùng mới vì hai lý do. Thứ nhất, khả năng thích ứng trên tập dữ liệu mới của các lớp phần riêng của hệ thống bị triệt tiêu, thay vào đó là tính cá nhân hóa cho từng tập dữ liệu

cục bộ mà hệ thống đã làm việc trước đó. Đứng trước một tập dữ liệu mới, các lớp phần riêng này hầu như không đạt được hiệu suất cao, dẫn đến kết quả của ensemble test không được như kỳ vọng. Thứ hai, cách tổ chức dữ liệu của hệ thống FL theo hướng ML yêu cầu chia tập dữ liệu cục bộ ra thành hai tập con (tập support và query). Mô hình học sẽ được thích ứng với dữ liệu kiểm tra thông qua một vài bước tinh chỉnh (fine-tune) trên tập support. Vì vậy, không những có thể chọn ra lớp phần riêng phù hợp nhất trong quá trình tinh chỉnh, lớp phần riêng được chọn còn có khả năng thích ứng nhanh trên tập dữ liệu mới do đã được huấn luyện bằng các thuật toán ML.

3.3 Kết hợp Meta Learning và Personalization Layer vào hệ thống Federated Learning

3.3.1 Huấn luyện cục bộ

3.3.2 Tổng hợp toàn cục

3.3.3 Giai đoạn kiểm thử

Chương 4

Cài đặt thực nghiệm

4.1 Mô tả dữ liệu

4.2 Phương pháp đánh giá

4.3 Mô tả thực nghiệm

4.3.1 Kiến trúc mô hình

4.3.2 Huấn luyện tập trung

4.3.3 Huấn luyện phân tán

Chương 5

Kết quả & Thảo luận

Chương 6

Kết luận

Tài liệu tham khảo

References

[1] Yoshinori Aono et al. "Privacy-preserving deep learning via additively homomorphic encryption". In: *IEEE Transactions on Information Forensics and Security* 13.5 (2017), pp. 1333–1345.

[2] Manoj Ghuhan Arivazhagan et al. "Federated learning with personalization layers". In: *arXiv preprint arXiv:1912.00818* (2019).

[3] Kapil Chandorikar. *Introduction to Federated Learning and Privacy Preservation*. 2020. URL: `https://towardsdatascience.com/introduction-to-federated-learning-and-privacy-preservation-75644686b559`.

[4] Fei Chen et al. "Federated meta-learning with fast convergence and efficient communication". In: *arXiv preprint arXiv:1802.07876* (2018).

[5] Moming Duan et al. "Astraea: Self-balancing federated learning for improving classification accuracy of mobile deep learning applications". In: *2019 IEEE 37th international conference on computer design (ICCD)*. IEEE. 2019, pp. 246–254.

[6] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. "Personalized federated learning: A meta-learning approach". In: *arXiv preprint arXiv:2002.07948* (2020).

[7] Chelsea Finn, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks". In: *International Conference on Machine Learning*. PMLR. 2017, pp. 1126–1135.

[8] Avishek Ghosh et al. "An efficient framework for clustered federated learning". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 19586–19597.

[9] Timothy Hospedales et al. "Meta-learning in neural networks: A survey". In: *arXiv preprint arXiv:2004.05439* (2020).

[10] Qinbin Li et al. “A survey on federated learning systems: vision, hype and reality for data privacy and protection”. In: *IEEE Transactions on Knowledge and Data Engineering* (2021).

[11] Zhenguo Li et al. “Meta-sgd: Learning to learn quickly for few-shot learning”. In: *arXiv preprint arXiv:1707.09835* (2017).

[12] Paul Pu Liang et al. “Think locally, act globally: Federated learning with local and global representations”. In: *arXiv preprint arXiv:2001.01523* (2020).

[13] Wei Yang Bryan Lim et al. “Federated learning in mobile edge networks: A comprehensive survey”. In: *IEEE Communications Surveys & Tutorials* 22.3 (2020), pp. 2031–2063.

[14] Brendan McMahan et al. “Communication-efficient learning of deep networks from decentralized data”. In: *Artificial intelligence and statistics*. PMLR. 2017, pp. 1273–1282.

[15] H Brendan McMahan et al. “Learning differentially private recurrent language models”. In: *arXiv preprint arXiv:1710.06963* (2017).

[16] Martin A Tanner and Wing Hung Wong. “The calculation of posterior distributions by data augmentation”. In: *Journal of the American statistical Association* 82.398 (1987), pp. 528–540.

[17] Kangkang Wang et al. “Federated evaluation of on-device personalization”. In: *arXiv preprint arXiv:1910.10252* (2019).

[18] Qiang Yang et al. “Federated machine learning: Concept and applications”. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 10.2 (2019), pp. 1–19.

[19] Xuefei Yin, Yanming Zhu, and Jiankun Hu. “A Comprehensive Survey of Privacy-preserving Federated Learning: A Taxonomy, Review, and Future Directions”. In: *ACM Computing Surveys (CSUR)* 54.6 (2021), pp. 1–36.

[20] Tehrim Yoon et al. “Fedmix: Approximation of mixup under mean augmented federated learning”. In: *arXiv preprint arXiv:2107.00233* (2021).

[21] Yue Zhao et al. “Federated learning with non-iid data”. In: *arXiv preprint arXiv:1806.00582* (2018).

[22] Hangyu Zhu et al. “Federated Learning on Non-IID Data: A Survey”. In: *arXiv preprint arXiv:2106.06843* (2021).