Meta-learning and Personalization layer in Federated learning

Bao-Long Nguyen[1][0000−0002−6411−8943]
and Tat-Cuong Cao[1][0000−0003−1803−843X]

VNU HCM University of Science, Ho Chi Minh City 749000, Vietnam

Abstract. Khái niệm *Federated Learning* (FL) cùng thuật toán *Federated Averaging* ra đời, được xem như một giải pháp thay thế cho huấn luyện tập trung. Giải pháp này không những đạt hiệu quả gần như tương đương các phương pháp học sâu đã có, giải quyết được vấn đề chi phí phần cứng, mà còn đảm bảo được quyền riêng tư dữ liệu người dùng. Tuy nhiên, đứng trước dữ liệu không đồng nhất và có tính cá nhân hóa cao trên từng người dùng (dữ liệu Non-IID), hệ thống này bị suy giảm hiệu suất nghiêm trọng [20]. Nghiên cứu cải thiện hiệu suất và tăng tính cá nhân hoá của hệ thống FL trên dữ liệu Non-IID bằng thuật toán `FedMeta-Per` - sự kết hợp giữa phương pháp huấn luyện meta-learning [8] và kỹ thuật personalization layer [21] vào hệ thống FL.

Keywords: Federated learning · Non-IID · Meta-learning · Personalization layer.

1 Introduction

Trong kỷ nguyên Internet of Things (IoT), đứng trước một lượng dữ liệu khổng lồ tại các thiết bị biên được sinh ra mỗi giây, quá trình huấn luyện tập trung bộc lộ nhiều nhược điểm. Thứ nhất, dữ liệu tại các thiết bị biên thường hám chứa những thông tin nhạy cảm về người dùng. Khi phải truyền về một máy chủ để huấn luyện, các thông tin này có thể bị lộ, ảnh hưởng nghiêm trọng đến quyền riêng tư dữ liệu. Thứ hai, chi phí truyền tin giữa máy chủ và các máy khách rất tốn kém do phải truyền một lượng lớn dữ liệu về máy chủ. Thứ ba, máy chủ phải có khả năng tính toán và lưu trữ lớn để tiến hành quá trình huấn luyện.

Tính toán biên [11] ra đời nhằm đẩy quá trình tính toán và lưu trữ dữ liệu từ máy chủ đến các máy khách hay thiết bị biên, giúp giảm tải cho máy chủ. Hơn nữa, việc tính toán được chuyển đến gần hoặc ngay tại nơi sản sinh dữ liệu, giúp tiết kiệm một lượng lớn chi phí truyền tin và giảm độ trễ khi tính toán. Không những thế, quá trình trao đổi dữ liệu giữa máy chủ và máy khách không còn diễn ra, giúp tăng đáng kể quyền riêng tư dữ liệu của người dùng.

Trên tinh thần của tính toán biên, *federated learning* (FL) [16] ra đời, nhằm huấn luyện mô hình máy học trên các tập dữ liệu riêng biệt, được phân bố trên các thiết bị biên [19]. Cụ thể, mục tiêu của hệ thống FL là tìm kiếm một tham số toàn cục w_G^* sao cho độ lỗi của hệ thống đạt cực tiểu:

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

Trong đó, n là số máy khách tham gia vào hệ thống, f_{global} là độ lỗi của toàn hệ thống, $f_{local}(w_i)$ là độ lỗi của máy khách i khi sử dụng tham số w_i.

Phương pháp huấn luyện này không những thừa hưởng những lợi ích từ quá trình tính toán biên (chi phí phần cứng và độ trễ giảm, bảo mật dữ liệu tăng) mà còn có khả năng tăng tính cá nhân hoá cho từng người dùng. Tuy nhiên, các máy khách tham gia vào hệ thống thường không đồng nhất về khả năng lưu trữ và tính toán. Dữ liệu phân bố trên các máy khách này cũng không mang một phân phối giống nhau (non-IID). Điều này tạo ra hai vấn đề chính của một hệ thống FL: Thách thức về hệ thống và Thách thức về thống kê [13]. Đối với vấn đề thống kê, nghiên cứu [20] chỉ ra rằng thuật toán truyền thống của FL - Federated Averaging (`FedAvg`) bị giảm hiệu suất nghiêm trọng trên dữ liệu non-IID. Nghiên cứu này nhằm giải quyết một phần vấn đề thống kê. Cục thể, nghiên cứu tập trung vào cải thiện hiệu suất cũng như tính cá nhân hoá trong việc dự đoán trên dữ liệu non-IID.

Trong nghiên cứu này, chúng tôi cải thiện hiệu suất của hệ thống FL trên dữ liệu non-IID bằng meta-learning (ML) [8] và kỹ thuật personalization layer (PL). Các thuật toán ML theo hướng tối ưu hai cấp độ có thể tạo ra một khởi tạo có khả năng thích ứng nhanh trên nhiệm vụ mới. Việc sử dụng các thuật toán ML trong huấn luyện nhằm thu được một tham số toàn cục có khả năng thích ứng nhanh trên tập dữ liệu mới của từng người dùng. Trong khi đó, việc sử dụng các kỹ thuật PL để duy trì một phần mạng học sâu tại các thiết bị biên giúp cải thiện hơn nữa tính cá nhân hoá.

Đóng góp chính. Nghiên cứu đề xuất thuật toán `FedMeta-Per`, một sự kết hợp giữa các thuật toán ML (`MAML` [6], `Meta-SGD` [14]) và các kỹ thuật PL (`FedPer` [1], `LG-FedAvg` [15]), giúp làm tăng độ chính xác lẫn tính cá nhân hoá trên từng người dùng. Qua thực nghiệm, nghiên cứu chỉ ra tính hiệu quả trong việc tăng độ chính xác cũng như tính cá nhân hoá của của thuật toán đề xuất so với thuật toán `FedAvg`, các thuật toán có sự kết hợp của ML vào FL [3] (`FedMeta(MAML)`, `FedMeta(Meta-SGD)`) và thuật toán sử dụng PL (`FedPer`) trên hệ thống FL gồm 50 người dùng, lần lượt chứa dữ liệu của hai tập dữ liệu CIFAR-10 [12] và MNIST [4].

2 Related Work

Từ khi thuật toán `FedAvg` được giới thiệu lần đầu vào năm 2017, tới nay, đã có rất nhiều công trình được đưa ra nhằm cải thiện độ chính xác của hệ thống trên dữ liệu non-IID. Tại đây, nghiên cứu điểm qua một vài công trình tiêu biểu trong việc thực hiện mục tiêu này.

Sử dụng 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 với 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 [7].

Trong ngữ cảnh dữ liệu non-IID, mỗi người dùng, với nhu cầu khác nhau, sinh ra các tập dữ liệu có phân phối rất khác nhau. Nếu coi mỗi người dùng là một nhiệm vụ, việc huấn luyện mô hình toàn cục trên một tập hợp người dùng của FL sẽ tương đương với việc huấn luyện mô hình máy học trên một phân phối nhiệm vụ của ML. Nghiên cứu [3] đề xuất framework `FedMeta` và minh hoạ thực nghiệm bằng hai thuật toán `FedMeta(MAML)`, `FedMeta(Meta-SGD)`. Kết quả cho thấy mô hình toàn cục có khả năng thích ứng nhanh, hội tụ nhanh hơn `FedAvg` trên tập dữ liệu LEAF [2]. Nghiên cứu [5] đề xuất kết hợp `MAML` trên tập dữ liệu CIFAR-10 và nghiên cứu [10] đề xuất kết hợp `Reptile` [17] trên tập dữ liệu EMNIST-62 vào hệ thống FL để tăng tính cá nhân hoá trên từng máy khách.

Mặt khác, một hệ thống FL, theo thời gian sẽ có sự xuất hiện của các máy khách mới. Một điểm trừ cho các hệ thống sử dụng ML trong huấn luyện là chúng đối xử như nhau với những máy khách đã tồn tại lâu trong hệ thống và những máy khách vừa tham gia vào hệ thống. Do đó, dù mô hình toàn cục có khả năng thích ứng nhanh trên dữ liệu mới, hiệu năng của chúng hoàn toàn có thể được cải thiện thêm.

Sử dụng personalization layer. Để nắm bắt được đặc tính dữ liệu, vốn mang tính cá nhân hoá cao trên từng người dùng khác nhau, việc duy trì một phần mạng học sâu tại các máy khách (được gọi là phần riêng) được đề xuất. Theo đó, nghiên cứu [1] đề xuất thuật toán `FedPer` với ý tưởng duy trì các lớp fully-connected của mạng học sâu tại máy khách với mục tiêu tăng tính cá nhân hoá. Thực nghiệm cho thấy, `FedPer` đạt kết quả cao hơn rất nhiều so với `FedAvg` khi thử nghiệm trên các tập dữ liệu CIFAR-10 và CIFAR-100. Bên cạnh đó, nghiên cứu [15] với thuật toán `LG-FedAvg` đề xuất duy trì các lớp rút trích đặc trưng của mạng học sâu tại các máy khách với mục tiêu nắm bắt được các đặc trưng của từng máy khách khác nhau. Kết quả thu được thậm chí còn tốt hơn `FedPer` trong đa số các trường hợp khi thực nghiệm trên hệ thống bao gồm từ 10 đến 100 người dùng lần lượt chứa dữ liệu của các tập dữ liệu CIFAR-10, CIFAR-100 và Omniglot.

Điểm giống nhau giữa hai phương pháp này nằm ở việc huấn luyện các lớp phần chung (phần còn lại của mạng học sâu sau khi đã loại bỏ các lớp phần riêng). Các lớp này được huấn luyện và tổng hợp bằng thuật toán `FedAvg`. Do đó, chúng không có khả năng thích ứng nhanh trên tập dữ liệu mới. Ngoài ra, trong quá trình kiểm thử, thuật toán `FedPer` lấy trung bình cộng các tham số phần riêng, sau đó ghép với tham số của các lớp phần chung để tiến hành kiểm thử. Các tham số phần riêng vốn có tính cá nhân hoá rất cao, giờ đây được tính trung bình cộng, có thể làm cho hiệu quả phân lớp bị giảm đáng kể khi dữ liệu non-IID mạnh. Thuật toán `LG-FedAvg` đề xuất thực hiện ensemble test giữa các lớp phần riêng. Tuy nhiên, chỉ các lớp phần riêng được huấn luyện trên dữ liệu giống với dữ liệu kiểm thử mới thực sự hoạt động tốt. Vì vậy, trong một

số trường hợp, ensemble test cho kết quả không cao do có quá nhiều lớp phần riêng chưa từng làm việc với dữ liệu giống với dữ liệu trong quá trình kiểm thử.

3 Proposed Method

Phương pháp đề xuất của nghiên cứu này là sự kết hợp giữa việc duy trì các lớp phần riêng tại thiết bị biên của thuật toán FedPer và việc sủ dụng meta-learning trong huấn luyện các lớp phần chung của mô hình. Với sự kết hợp này, nghiên cứu nhằm vào hai mục tiêu: (1) - Khả năng tương thích nhanh trên tập dữ liệu mới của các thuật toán ML, (2) - Khả năng cá nhân hoá của kỹ thuật PL.

Tính tương thích nhanh. Như đã đề cập ở trên, các lớp phần chung của các thuật toán sử dụng kỹ thuật PL không thực sự mạnh mẽ vì được huấn luyện theo thuật toán FedAvg, dẫn đến việc chúng chậm chạp trong việc thích nghi với tập dữ liệu mới. Việc huấn luyện các lớp phần chung bằng các thuật toán ML được kỳ vọng sẽ cung cấp cho các lớp phần chung này khả năng thích ứng nhanh trên tập dữ liệu của các máy khách, khắc phục nhược điểm của các thuật toán sử dụng kỹ thuật PL.

Tính cá nhân hoá. Tính cá nhân hoá hệ thống FL kết hợp với ML đến từ việc hệ thống cho phép mô hình toàn cục thực hiện fine-tune trên một phần dữ liệu của máy khách trước khi bước vào kiểm thử trên phần dữ liệu còn lại của máy khách đó. Việc cho phép duy trì một phần mạng học sâu (các lớp phần riêng) trên từng máy khách cũng làm gia tăng đáng kể khả năng cá nhân hoá cho từng máy khách. Tại đây, thuật toán đề xuất kết hợp cả hai phương pháp tăng cường tính cá nhân hoá vừa nêu, kỳ vọng sẽ làm gia tăng hơn nữa tính cá nhân hoá cho hệ thống FL.

Cụ thể, mạng học sâu được chia thành hai phần, phần chung bao gồm các lớp rút trích đặc trưng, phần riêng chứa các lớp fully-connected. Các lớp phần chung được huấn luyện theo hai thuật toán ML (MAML và Meta-SGD). Các lớp phần chung được duy trì tại các máy khách riêng biệt. Với sự kết hợp này, chúng tôi gọi thuật toán đề xuất của mình là FedMeta-Per.

Algorithm 1 FedMeta-Per (Server)

Initialize w_B^0 for MAML or (w_B^0, α_B^0) for Meta-SGD
for $t = 0, 1, 2, ...$ **do**
  Sample a subset C_t of m clients
  for $c_i \in C_t$ **do**
    $w_{B(i)}^{t+1} \leftarrow \text{ModelTrainingMAML}(c_i, w_B^t)$ for MAML
    $(w_{B(i)}^{t+1}, \alpha_{B(i)}^{t+1}) \leftarrow \text{ModelTrainingMetaSGD}(c_i, w_B^t, \alpha_B^t)$ for Meta-SGD

  $n_i = \left|\mathcal{D}_{train(i)}^{query}\right|, N_m = \sum_{i=0}^{m} n_i$
  Aggregate $w_B^{t+1} = \sum_{i=0}^{m} \frac{n_i}{N_m} w_{B(i)}^{t+1}$ for MAML
  Aggregate $(w_B^{t+1}, \alpha_B^{t+1}) = \sum_{i=0}^{m} \frac{n_i}{N_m} (w_i^{t+1}, \alpha_{B(i)}^{t+1})$ for Meta-SGD

Huấn luyện. Tại máy chủ, thuật toán 1 được thi triển để điều phối các hoạt động huấn luyện. Các hoạt động này bao gồm: Khởi tạo tham số phần chung; Phân phối tham số này đến các máy khách; Tổng hợp tham số phần chung toàn cục mới. Quá trình này được tóm tắt trong Bảng 1. Đối với thuật toán `MAML`, tham số phần chung chỉ bao gồm một phần tham số của mô hình toàn cục (w_B^t at round t). Trong khi đó, tham số phần chung của thuật toán `Meta-SGD` chứa thêm một phần giá trị siêu tham số học tại mỗi máy khách (α_B at round t), có kích thước bằng với w_B^t. Giá trị siêu tham số này sẽ được giải thích rõ trong phần huấn luyện tại máy khách. Các giá trị này sẽ được phân phối đến một tập hợp con C_t gồm m máy khách để tiến hành huấn luyện phân tán. Các máy khách sau đó thực hiện huấn luyện trên chính bộ dữ liệu của mình rồi trả về các tham số phần chung ($w_{B(i)}^{t+1}$ for `MAML` and $(w_{B(i)}^{t+1}, \alpha_{B(i)}^{t+1})$ for `Meta-SGD` at round t) cho máy chủ. Tham số phần chung mới được tổng hợp như sau:

$$\text{MAML: } w_B^{t+1} = \sum_{i=0}^{m} \frac{n_i}{N_m} w_{B(i)}^{t+1} \tag{2}$$

$$\text{Meta-SGD: } (w_B^{t+1}, \alpha_B^{t+1}) = \sum_{i=0}^{m} \frac{n_i}{N_m} (w_i^{t+1}, \alpha_{B(i)}^{t+1}) \tag{3}$$

Trong đó, n_i là số mẫu dữ liệu của máy khách $c_i \in C_t$, N_m là tổng số mẫu dữ liệu trên tập query của m máy khách tham gia huấn luyện toàn cục.

Table 1: Server activity summary at round t^{th}

	Algorithm	
	MAML	Meta-SGD
Send	w_B^t	(w_B^t, α_B^t)
Receive	Weights: $w_{B(i)}^{t+1}, i \in [1, m]$	Parameters: $(w_{B(i)}^{t+1}, \alpha_{B(i)}^{t+1}), i \in [1, m]$
Aggregate	w_B^{t+1}	$(w_B^{t+1}, \alpha_B^{t+1})$

Tại máy khách, một trong hai thuật toán 2 và 3 được thi triển, ứng với phương pháp huấn luyện theo kiểu `MAML` và `Meta-SGD`. Để huấn luyện trên các thuật toán ML theo hướng tối ưu hai cấp độ, dữ liệu của máy khách được chia thành tập support ($\mathcal{D}_{train(i)}^{support}$) và tập query ($\mathcal{D}_{train(i)}^{query}$). Các bước huấn luyện tại máy khách bao gồm: Hợp nhất tham số phần chung và phần riêng để tạo thành tham số mô hình hoàn chỉnh; Thực hiện huấn luyện theo các thuật toán ML; Phân giải tham số mô hình mới.

Cụ thể, đối với các máy khách huấn luyện theo thuật toán `MAML`, tại bước huấn luyện toàn cục thứ t, một máy khách c_i ban đầu sẽ nhận được trọng số khởi tạo w_B^t của phần chung do máy chủ gửi đến. Tiếp đến, c_i tiến hành hợp nhất trọng số phần chung w_B^t với trọng số phần riêng $w_{P(i)}^t$ để thu được trọng số của mô hình hoàn chỉnh w_i^t. w_i^t sau đó được huấn luyện như sau:

Algorithm 2 FedMeta-Per (MAML Client)

ModelTrainingMAML(c_i, w_B^t)**:**
Sample support set $\mathcal{D}_{train(i)}^{support}$ and query set $\mathcal{D}_{train(i)}^{query}$
if $t = 0$ **then**
    Initialize $w_{P(i)}^t$
else
    Load $w_{P(i)}^t$ from the external memory
$w_i^t \leftarrow w_B^t \bigoplus w_{P(i)}^t$ ▷ Merge w_B^t and $w_{P(i)}^t$ to form w_i^t
Training:

$$\hat{w}_i^{t+1} \leftarrow w_i^t - \alpha \nabla_{w_i^t} f_{local}\left(w_i^t, \mathcal{D}_{train(i)}^{support}\right)$$

$$w_i^{t+1} \leftarrow w_i^t - \beta \nabla_{w_i^t} f_{local}\left(\hat{w}_i^{t+1}, \mathcal{D}_{train(i)}^{query}\right)$$

$w_{B(i)}^{t+1}, w_{P(i)}^{t+1} \leftarrow w_i^{t+1}$ ▷ Resolve w_i^{t+1} to form $w_{B(i)}^{t+1}$ and $w_{P(i)}^{t+1}$
Send $w_{B(i)}^{t+1}$ to server and store $w_{P(i)}^{t+1}$

$$\text{Train: } \hat{w}_i^{t+1} \leftarrow w_i^t - \alpha \nabla_{w_i^t} f_{local}\left(w_i^t, \mathcal{D}_{train(i)}^{support}\right) \tag{4}$$

$$\text{Meta-train: } w_i^{t+1} \leftarrow w_i^t - \beta \nabla_{w_i^t} f_{local}\left(\hat{w}_i^{t+1}, \mathcal{D}_{train(i)}^{query}\right) \tag{5}$$

Trong đó, $\alpha, \beta \in \mathbb{R}$ là các siêu tham số học.

Đối với các máy khách sử dụng thuật toán Meta-SGD trong huấn luyện, phần chung của mạng học sâu bao gồm hai tham số: trọng số huấn luyện chung w_B^t và siêu tham số huấn luyện chung α_B^t; phần riêng của mạng bao gồm hai tham số $w_{P(i)}^t$ và $\alpha_{P(i)}^t$ (có kích thước bằng tương tự $w_{P(i)}^t$). Quá trình hợp nhất tham số cũng được diễn ra giữa các tham số phần chung và phần riêng để tạo thành bộ tham số hoàn chỉnh w_i^t và α_i^t. Cần lưu ý rằng Meta-SGD cấu hình α là một mảng siêu tham số có kích thước bằng trọng số của mô hình và coi α là một tham số có thể huấn luyện được. Do đó, thuật toán nhân vô hướng α với đạo hàm của hàm lỗi tại phương trình 6 và tiến hành đạo hàm theo α trong phương trình 7.

$$\text{Train: } \hat{w}_i^{t+1} \leftarrow w_i^t - \alpha_i^t \circ \nabla_{w_i^t} f_{local}\left(w_i^t, \mathcal{D}_{train(i)}^{support}\right) \tag{6}$$

$$\text{Meta-train: } (w_i^{t+1}, \alpha_i^{t+1}) \leftarrow (w_i^t, \alpha_i^t) - \beta \nabla_{(w_i^t, \alpha_i^t)} f_{local}\left(\hat{w}_i^{t+1}, \mathcal{D}_{train(i)}^{query}\right) \tag{7}$$

Kiểm thử. Dựa theo quá trình kiểm thử của nghiên cứu [15], chúng tôi chia ra hai loại người dùng: người dùng cục bộ và người dùng mới. Đối với người dùng cục bộ, các lớp học sâu thuộc phần riêng được sử dụng lại để đạt được độ mức cá nhân hóa cao. Đối với người dùng mới, chúng tôi không sử dụng kỹ thuật ensemble test như nghiên cứu [15] đề xuất mà cho từng lớp phần riêng đã

Algorithm 3 FedMeta-Per (Meta-SGD Client)

ModelTrainingMetaSGD(c_i, w_B^t, α_B^t)**:**
Sample support set $\mathcal{D}_{train(i)}^{support}$ and query set $\mathcal{D}_{train(i)}^{query}$
if $t = 0$ **then**
    Initialize $(w_{P(i)}^t, \alpha_{P(i)}^t)$
else
    Load $(w_{P(i)}^t, \alpha_{P(i)}^t)$ from the external memory
$w_i^t \leftarrow w_B^t \bigoplus w_{P(i)}^t$ ▷ Merge w_B^t and $w_{P(i)}^t$ to form w_i^t
$\alpha_i^t \leftarrow \alpha_B^t \bigoplus \alpha_{P(i)}^t$ ▷ Merge α_B^t and $\alpha_{P(i)}^t$ to form α_i^t
Training:

$$\hat{w}_i^{t+1} \leftarrow w_i^t - \alpha_i^t \circ \nabla_{w_i^t} f_{local}\left(w_i^t, \mathcal{D}_{train(i)}^{support}\right)$$

$$(w_i^{t+1}, \alpha_i^{t+1}) \leftarrow (w_i^t, \alpha_i^t) - \beta \nabla_{(w_i^t, \alpha_i^t)} f_{local}\left(\hat{w}_i^{t+1}, \mathcal{D}_{train(i)}^{query}\right)$$

$w_{B(i)}^{t+1}, w_{P(i)}^{t+1} \leftarrow w_i^{t+1}$ ▷ Resolve w_i^{t+1} to form $w_{B(i)}^{t+1}$ và $w_{P(i)}^{t+1}$
$\alpha_{B(i)}^{t+1}, \alpha_{P(i)}^{t+1} \leftarrow \alpha_i^{t+1}$ ▷ Resolve α_i^{t+1} to form $\alpha_{B(i)}^{t+1}$ và $\alpha_{P(i)}^{t+1}$
Send $(w_{B(i)}^{t+1}, \alpha_B^{t+1})$ to server and store $(w_{P(i)}^{t+1}, \alpha_{P(i)}^{t+1})$

được xây dựng trước đó kết hợp với lớp phần chung để hoạt động trên bộ dữ liệu mới. Trong quá trình fine-tune, máy khách sẽ chọn được bộ tham số cho độ lỗi nhỏ nhất. Từ đó sử dụng bộ tham số này để xử lý dữ liệu kiểm thử.

4 Numerical Experiments

Dữ liệu. Nghiên cứu đánh giá thuật toán đề xuất trên 50 máy khách lần lượt chứa hai tập dữ liệu MNIST [4] và CIFAR-10 [12]. Để mô phỏng dữ liệu Non-IID, mỗi máy khách được cấu hình chỉ chứa 2/10 phân lớp dữ liệu, số lượng nhãn giữa các lớp và số lượng dữ liệu giữa các máy khách là không đồng đều. Tại mỗi máy khách, dữ liệu chia cho hai tập support và query có tỷ lệ 2:8. Thống kê dữ liệu được trình bày trong Bảng 2.

Table 2: Statistics on MNIST and CIFAR-10 (Non-IID data)

Dataset	#clients	#samples	#classes	#samples/client				#classes/client
				min	mean	std	max	
MNIST	50	69,909	10	135	1,398	1,424	5,201	2
CIFAR-10	50	52,497	10	506	1,049	250	1,986	2

Thang đánh giá. Nghiên cứu đánh giá mô hình qua các thang đo: Độ chính xác trong tương quan với tất cả các điểm dữ liệu, Độ chính xác trong tương quan với tất cả các máy khách và F1-score (macro).

Huấn luyện. Nghiên cứu thực hiện các thí nghiệm tập trung và phân tán trên cùng một kiến trúc mô hình. Đối với huấn luyện phân tán, nghiên cứu thực nghiệm trên các thuật toán `FedAvg`, `FedPer`, `FedMeta(MAML)`, `FedMeta(Meta-SGD)`. Các kết quả được đem so sánh với thuật toán đề xuất `FedMeta-Per`. Để so sánh công bằng, nghiên cứu cho phép mô hình toàn cục thu được bởi thuật toán `FedAvg` và `FedPer` thực hiện fine-tune một hoặc một vài lần trên tập support của máy khách trong quá trình kiểm thử. Đây chính là ý tưởng của hai thuật toán `FedAvgMeta` và `FedPerMeta`.

Chi tiết về cài đặt thực nghiệm được trình bày trong Phụ lục A.

5 Results & Discussion

Centralized & Decentralized. Kết quả quá trình huấn luyện tập trung và huấn luyện phân tán (`FedAvg` trên IID và non-IID data) được trình bày trong Bảng 3. Dễ dàng nhận thấy, mô hình huấn luyện tập trung đạt kết quả cao hơn trên tất cả các thang đánh giá. Các kết quả thu được trên `FedAvg` (dữ liệu IID) thấp hơn từ 7% đến 8% do mô hình toàn cục nắm bắt các đặc trưng một cách gián tiếp(thông qua quá trình lấy trung bình tham số tại máy chủ). Tuy nhiên, khi dữ liệu đầu vào là non-IID, các giá trị độ đo giảm từ 9% đến 13% trên MNIST và không đạt hội tụ trên CIFAR-10. Các giá trị độ đo trên dữ liệu Non-IID cũng chênh lệch nhau khá nhiều (2% - 8% trên MNIST và 1% - 6% trên CIFAR-10) so với chênh lệch trên dữ liệu IID (dưới 1%). Điều này chứng tỏ phân phối dữ liệu không đều trên các máy khách đã ảnh hưởng xấu đến chất lượng phân lớp.

Table 3: Classification results (%) of centralized and decentralized (FedAvg on IID and Non-IID data) using MNIST and CIFAR-10 datasets. Best results per metrics are boldfaced.

		acc_{micro}	acc_{macro}	$F1_{macro}$
MNIST	Centralized	**97.07**	-	**97.04**
	FedAvg (IID data)	90.36	90.34±2.24	90.12±2.29
	FedAvg (local client)	85.03	82.14±14.76	79.43±16.83
	FedAvg (new client)	83.92	81.69±19.71	77.66±22.54
CIFAR-10	Centralized	**61.91**	-	**61.72**
	FedAvg (IID data)	53.83	53.83±3.14	53±3.21
	FedAvg (local client)	19.02	19.29±25.11	16.85±23.92
	FedAvg (new client)	24.63	24.83±22.57	20.52±20.45

Khả năng hội tụ. Kết quả trên các thang đo của thuật toán `FedMeta-Per` và các thuật toán dùng trong so sánh được trình bày trong Bảng 4. Để giải quyết vấn đề dữ liệu Non-IID, kỹ thuật fine-tune cục bộ vào hệ thống FL dưới dạng đơn giản (thuật toán `FedAvgMeta`), nhằm cải thiện khả năng thích ứng hoặc cá nhân hoá của mô hình trên tập dữ liệu mới. Tuy nhiên, kết quả thu trong bảng 4

cũng như trong Hình 2 không quá khả quan (hội tụ ở mức thấp trên tập MNIST và không hội tụ trên CIFAR-10). Nguyên nhân cho việc `FedAvgMeta` không hội tụ nằm ở chỗ khả năng thích ứng trên tập dữ liệu mới của mô hình toàn cục là rất thấp. Kỹ thuật PL cũng được sử dụng dưới dạng hai thuật toán `FedPer` và `FedPerMeta`. Tuy nhiên, kết quả thu được thậm chí có phần tệ hơn `FedAvg` và `FedAvgMeta`. Điều này là dễ hiểu khi kiến trúc mô hình học mà nghiên cứu [1] sử dụng là mạng pre-trained `MobileNet-v1` [9] - một mạng học sâu phức tạp hơn của nghiên cứu này rất nhiều (xem Phụ lục A.2). Điều này cũng chứng minh rằng, sử dụng kỹ thuật PL là không đủ trong việc xử lý dữ liệu non-IID.

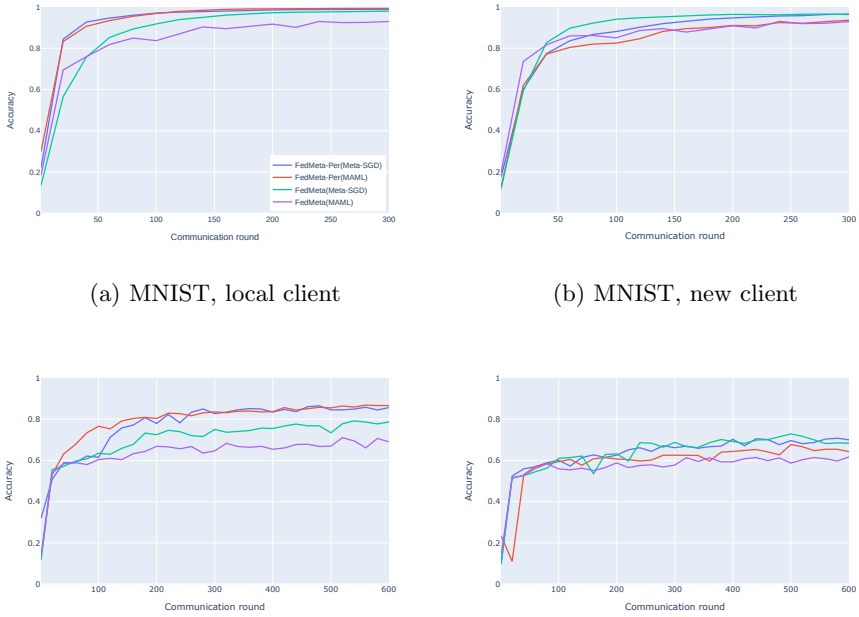
Table 4: Classification results (%) of FedAvg, FedPer and FedMeta-Per using MNIST and CIFAR-10 datasets. Best results per metrics are boldfaced.

		CIFAR-10			MNIST		
		acc_{micro}	acc_{macro}	$F1_{macro}$	acc_{micro}	acc_{macro}	$F1_{macro}$
local client	FedAvg	19.02	19.29±25.11	16.85±23.92	85.03	82.14±14.76	79.43±16.83
	FedPer	13.22	12.99±19.39	10.52±14.91	77.29	75.48±14.84	72.32±15.99
	FedAvgMeta	40.3	38.47±31.52	33.81±30.61	84.84	81.56±16.68	78.31±19.8
	FedPerMeta	18.57	17.48±22.55	14.54±18.67	75.91	74.11±16.2	71.22±16.77
	FedMeta(MAML)	69.02	68.76±14.86	61.14±20	92.99	91.14±5.99	90.16±6.28
	FedMeta(Meta-SGD)	78.63	78.73±11.59	72.87±18.31	98.02	96.35±4.62	95.80±5.51
	FedMeta-Per(MAML)	**86.6**	**86.52±6.31**	**85.33±6.77**	**99.37**	**99.12±1.29**	**98.94±1.6**
	FedMeta-Per(Meta-SGD)	85.61	85.68±7.22	85.08±7.32	98.92	98.15±3.32	98.20±2.94
new client	FedAvg	24.63	24.83±22.57	20.52±20.45	83.92	81.69±19.71	77.66±22.54
	FedPer	14.4	14.52±20.15	10.66±13.79	78.3	76.19±18.79	72.72±19.3
	FedAvgMeta	43.39	43.54±18	35.14±17.22	84.34	82.37±17.42	78.78±19.31
	FedPerMeta	13.33	13.57±19.62	10.05±13.17	77.47	75.56±20.33	72.60±21.37
	FedMeta(MAML)	61.69	61.64±12.49	50.76±19.2	92.96	91.88±5.88	90.02±7.34
	FedMeta(Meta-SGD)	68.36	67.89±15.11	60.24±21.52	96.39	93.53±8.39	89.31±14.56
	FedMeta-Per(MAML)	64.22	63.70±12.29	53.68±19.06	93.6	93.57±5.58	91.83±6.43
	FedMeta-Per(Meta-SGD)	**69.97**	**69.13±14.63**	**62.42±20.94**	**96.62**	**95.88±3.58**	**94.85±4.61**

Khi so sánh khả năng hội tụ của `FedMeta` và `FedMeta-Per`, vì cả hai thuật toán đều có sự tham gia của ML và có sự tương đồng trong quá trình hội tụ của hai thuật toán (Hình 1). Chứng tỏ ML đã có sự tác động nhất định đến hệ thống FL.

Khả năng của ML trong thuật toán đề xuất trong so sánh với các thuật toán `FedAvg`, `FedAvgMeta`, `FedPer` và `FedPerMeta` được trình bày trong Hình 2. Theo đó, ML cung cấp cho hệ thống khả năng thích ứng nhanh trên máy khách mới, dẫn đến việc `FedMeta-Per` hội tụ nhanh hơn hẳn so với các thuật toán cũ, đạt các ngưỡng hội tụ từ khoảng 60% đến 85% trên CIFAR-10 và cải thiện khoảng 20% độ chính xác trên MNIST. Đặc biệt hơn, khả năng hội tụ này đến từ việc thực hiện duy nhất một bước fine-tune trên 20% dữ liệu tại máy khách, một lần nữa chứng minh khả năng thích ứng rất nhanh của hệ thống FL-ML.

Ngoài ra, nhờ vào việc xây dựng các lớp phần chung tốt, quá trình kiểm thử trên người dùng cục bộ cũng đạt hội tụ ở mức cao hơn khi so sánh thuật toán đề xuất với `FedMeta` (Hình 1). Tuy nhiên, đứng trước một phân phối dữ liệu lạ của người dùng mới, hai thuật toán này cho khả năng hội tụ gần như tương đương nhau.



(a) MNIST, local client

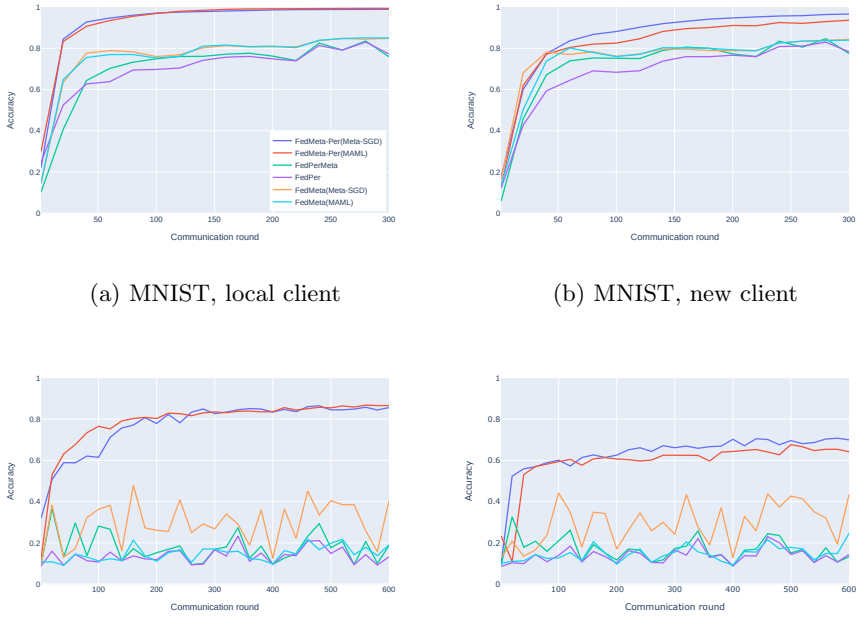
(b) MNIST, new client

(c) CIFAR-10, local client

(d) CIFAR-10, new client

Fig. 1: Độ chính xác trên điểm dữ liệu của FedMeta-Per và FedMeta. Kết quả cuối cùng và khả năng hội tụ của FedMeta-Per cao hơn đáng kể so với FedMeta trên người dùng cục bộ. Trên người dùng mới, mặc dù FedMeta-Per đạt giá trị cao hơn, khác biệt trong quá trình hội tụ là không quá lớn.

Khả năng cá nhân hoá. Quan sát độ lệch chuẩn của các thuật toán `FedMeta`, `FedAvg` và `FedPer` trong Bảng 4, có thể thấy độ lệch chuẩn trong kết quả của `FedMeta` nhỏ hơn nhiều so với các thuật toán còn lại. Chứng tỏ tính cá nhân hoá của hệ thống FL đã được cải thiện rất nhiều lần nhờ vào việc huấn luyện theo các thuật toán ML cũng như việc thực hiện fine-tune trên tập support để nắm bắt đặc điểm của một tập dữ liệu mới. Tuy nhiên, như đã đề cập ở trên, khả nằng này hoàn toàn có thể được cải thiện. Bằng các đề xuất mới trong quá trình kiểm thử, `FedMeta-Per` giúp làm giảm hơn nữa độ lệch chuẩn trên các thang đo. Cụ thể, sau khi kết hợp việc fine-tune mô hình trên tập support của dữ liệu cục bộ và việc duy trì lớp cá nhân hoá tại mỗi máy khách, đối với người dùng cục bộ, độ lệch chuẩn giảm từ 4% đến 14% trên CIFAR-10 và từ 1% đến 5% trên MNIST trong khi các giá trị trung bình vẫn vượt trội. Trên người dùng mới, các giá trị trung bình cao hơn nhưng độ lệch chuẩn trong một vài trường hợp không thực sự nhỏ hơn các thuật toán trước đó. Điều này là do các lớp phần riêng trong mạng học sâu của người dùng mới chưa thực sự khớp với người dùng



(a) MNIST, local client

(b) MNIST, new client

(c) CIFAR-10, local client

(d) CIFAR-10, new client

Fig. 2: Độ chính xác trên điểm dữ liệu của FedMeta-Per, FedAvg và FedPer. Khả năng hội tụ cũng như độ chính xác đạt được của FedMeta-Per là vượt trội hơn so với các thuật toán FedAvg và FedPer.

đó. Tuy nhiên, theo thời gian, khi người dùng mới tham gia vào một hoặc một vài bước huấn luyện cục bộ, họ sẽ tự xây dựng được các lớp phần riêng tốt cho bản thân, dẫn đến việc các thang đo đạt giá trị tương đương người dùng cũ.

6 Conclusion

Qua quá trình nghiên cứu, chúng tôi kết hợp các thuật toán ML và các kỹ thuật PL vào hệ thống FL, nhằm cải thiện độ chính xác cũng như khả năng cá nhân hoá cho từng người dùng của hệ thống trên dữ liệu non-IID. Bằng thực nghiệm, chúng tôi chứng minh được tính hiệu quả của thuật toán đề xuất `FedMeta-Per` so với các thuật toán `FedAvg`, `FedPer`, `FedMeta` trên 50 người dùng với hai tập dữ liệu CIFAR-10 và MNIST. Trong đó, có thể giải thích việc đạt được kết quả cao dựa vào hai yếu tố mang tính thừa kế: (1) - Khả năng thích ứng nhanh trên tập dữ liệu mới của thuật toán đề xuất thừa hưởng từ các thuật toán ML, (2) - Khả năng cá nhân hóa cao cho từng người dùng kế thừa từ các lớp cá nhân hóa của PL và việc fine-tune dữ liệu của ML.

Trong tương lai, các thuật toán ML theo hướng tối ưu hai cấp độ (FO-MAML [6], iMAML [18], Reptile) có thể được tích hợp thêm vào hệ thống. Việc tìm kiếm và phân cụm người dùng sao cho mỗi người dùng tìm được bộ tham số phần riêng tốt nhất cũng nên được nghiên cứu thêm để tăng hiệu suất của hệ thống. Ngoài ra, vấn đề về bảo mật thông tin giữa máy chủ và máy khách cũng nên được xét đến để hệ thống trở nên thực tiễn hơn.

References

1. Arivazhagan, M.G., Aggarwal, V., Singh, A.K., Choudhary, S.: Federated learning with personalization layers. arXiv preprint arXiv:1912.00818 (2019)
2. Caldas, S., Duddu, S.M.K., Wu, P., Li, T., Konečnỳ, J., McMahan, H.B., Smith, V., Talwalkar, A.: Leaf: A benchmark for federated settings. arXiv preprint arXiv:1812.01097 (2018)
3. Chen, F., Luo, M., Dong, Z., Li, Z., He, X.: Federated meta-learning with fast convergence and efficient communication. arXiv preprint arXiv:1802.07876 (2018)
4. Deng, L.: The mnist database of handwritten digit images for machine learning research. IEEE Signal Processing Magazine **29**(6), 141–142 (2012)
5. Fallah, A., Mokhtari, A., Ozdaglar, A.: Personalized federated learning: A meta-learning approach. arXiv preprint arXiv:2002.07948 (2020)
6. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: International Conference on Machine Learning. pp. 1126–1135. PMLR (2017)
7. Harlow, H.F.: The formation of learning sets. Psychological review **56**(1), 51 (1949)
8. Hospedales, T., Antoniou, A., Micaelli, P., Storkey, A.: Meta-learning in neural networks: A survey. arXiv preprint arXiv:2004.05439 (2020)
9. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
10. Jiang, Y., Konečnỳ, J., Rush, K., Kannan, S.: Improving federated learning personalization via model agnostic meta learning. arXiv preprint arXiv:1909.12488 (2019)
11. Khan, W.Z., Ahmed, E., Hakak, S., Yaqoob, I., Ahmed, A.: Edge computing: A survey. Future Generation Computer Systems **97**, 219–235 (2019)
12. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
13. Li, T., Sahu, A.K., Talwalkar, A., Smith, V.: Federated learning: Challenges, methods, and future directions. IEEE Signal Processing Magazine **37**(3), 50–60 (2020)
14. Li, Z., Zhou, F., Chen, F., Li, H.: Meta-sgd: Learning to learn quickly for few-shot learning. arXiv preprint arXiv:1707.09835 (2017)
15. Liang, P.P., Liu, T., Ziyin, L., Allen, N.B., Auerbach, R.P., Brent, D., Salakhutdinov, R., Morency, L.P.: Think locally, act globally: Federated learning with local and global representations. arXiv preprint arXiv:2001.01523 (2020)
16. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial intelligence and statistics. pp. 1273–1282. PMLR (2017)
17. Nichol, A., Achiam, J., Schulman, J.: On first-order meta-learning algorithms. arXiv preprint arXiv:1803.02999 (2018)

18. Rajeswaran, A., Finn, C., Kakade, S.M., Levine, S.: Meta-learning with implicit gradients. Advances in neural information processing systems **32** (2019)
19. Yin, X., Zhu, Y., Hu, J.: A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions. ACM Computing Surveys (CSUR) **54**(6), 1–36 (2021)
20. Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., Chandra, V.: Federated learning with non-iid data. arXiv preprint arXiv:1806.00582 (2018)
21. Zhu, H., Xu, J., Liu, S., Jin, Y.: Federated learning on non-iid data: A survey. arXiv preprint arXiv:2106.06843 (2021)

A Experimental Details

A.1 Datasets

Ký hiệu $C_{train} = \{c_1^{train}, ..., c_{50}^{train}\}$ là tập máy khách dùng trong huấn luyện, $C_{test} = \{c_1^{test}, ..., c_{50}^{test}\}$ là tập máy khách dùng trong kiểm thử, N là tổng số điểm dữ liệu, ta có số lượng dữ liệu huấn luyện và kiểm tra lần lượt là:

$$N_{train} = \sum_{i=1}^{50} \left|c_i^{train}\right| = 0.75N$$

$$N_{test} = \sum_{i=1}^{50} \left|c_i^{test}\right| = 0.25N$$

Trong cài đặt ML, ký hiệu $c_i^{train} = \{\mathcal{D}_{train(i)}^{support}, \mathcal{D}_{train(i)}^{query}\}$, $c_i^{test} = \{\mathcal{D}_{test(i)}^{support}, \mathcal{D}_{test(i)}^{query}\}$. Ta có số lượng dữ liệu chứa trong tập support và query của tất cả các máy khách lần lượt là:

$$N_{train/test(i)}^{support} = \left|\mathcal{D}_{train/test(i)}^{support}\right| = 0.2\left|c_i^{train/test}\right|$$

$$N_{train/test(i)}^{query} = \left|\mathcal{D}_{train/test(i)}^{query}\right| = 0.8\left|c_i^{train/test}\right|$$

Người dùng $c_j^{test} \in C_{test}$ được gọi là người dùng cục bộ nếu tồn tại người dùng $c_i^{train} \in C_{train}$ sao cho $p\left((x,y) \in c_j^{test}\right) = p\left((x,y) \in c_i^{train}\right)$. Ngược lại, c_j^{test} là người dùng mới nếu $p\left((x,y) \in c_j^{test}\right) \neq p\left((x,y) \in c_i^{train}\right)$ với mọi $c_i^{train} \in C_{train}$.

A.2 Model Architecture

Nghiên cứu sử dụng hai mô hình đơn giản để rút trích đặc trưng và phân lớp dữ liệu cho tập dữ liệu CIFAR-10 và MNIST.

CIFAR-10. Mô hình nhận các ảnh đầu vào có kích thước $(32 \times 32 \times 3)$. Hai lớp tích chập (kernel có kích thước (5×5), số chanel lần lượt là 6 và 16) được sử dụng để rút trích đặc trưng. Theo sau mỗi lớp tích chập là một lớp `MaxPooing` có kích thước (2×2). Phần phân lớp gồm ba lớp tuyến tính có đầu ra lần lượt là 120, 84 và 10. Các hàm kích hoạt được sử dụng là `ReLU` và `Softmax`.

MNIST. Mô hình nhận các ảnh đầu vào đã được làm phẳng có kích thước (1×784). Sử dụng hai lớp tuyến tính có đầu ra lần lượt là 100 và 10. Các hàm kích hoạt được sử dụng là `ReLU` và `Softmax`.

A.3 Hyper-parameters Searching

Các siêu tham số của hệ thống FL của nghiên cứu bao gồm: số máy khách tham gia huấn luyện trong một bước huấn luyện toàn cục ($\#clients/round$), số bước huấn luyện cục bộ ($\#epochs$), số bước huấn luyện toàn cục ($\#rounds$), lượng dữ liệu trong một batch dữ liệu ($batch\_size$), số lớp phần riêng đối với các thuật toán sử dụng kỹ thuật PL ($\#per\_layers$) và các siêu tham số học sử dụng trong tối ưu mạng học sâu bằng kỹ thuật mini batch gradient descent.

Để phù hợp cho phần cứng của các máy khách có cấu hình yếu trong kịch bản Horizontal FL, nghiên cứu giới hạn $\#epochs = 1$ và $batch\_size = 32$. Từ việc khảo sát các thí nghiệm FL của các nghiên cứu gần đây, số máy khách tham gia huấn luyện toàn cục được được chọn lần lượt là 2, 5 và 10 máy. Trong đó, $\#clients/round = 5$ cho kết quả cao hơn và tiêu tốn chi phí tính toán ở một mức chấp nhận được.

Đối với các mạng học sâu được cài đặt, việc duy trì một lớp phần chung và một lớp phần riêng cho mạng neuron MNIST là tất nhiên. Với mạng học sâu dùng cho tập CIFAR-10, $\#per\_layers \in \{1, 2, 3\}$ (tính từ lớp tuyến tính cuối cùng). Kết quả chạy thực nghiệm cho thấy, việc sử dụng lớp tuyến tính cuối cùng làm phần riêng và các lớp học sâu còn lại làm phần chung cho kết quả tốt nhất.

Ngoại trừ siêu tham số học của từng thuật toán, các siêu tham số kể trên đều được giữ cố định trong quá trình huấn luyện. Bảng 5 trình bày tóm tắt các giá trị siêu tham số này.

Table 5: Fixed hyper-parameters in FL system

	#clients/round	#epochs	#rounds	batch_size	#per_layers
MNIST	5	1	300	32	1
CIFAR-10			600		

Các siêu tham số học được tìm kiếm trong khoảng $(10^{-5}, 0.01)$ cho từng thuật toán. Kết quả tìm kiếm được trình bày trong Bảng 6. Các ô để trống biểu thị việc không tìm được siêu tham số để mô hình hội tụ.

Table 6: Tuning learning rate for algorithms

	CIFAR-10	MNIST
FedAvg, FedAvgMeta	-	10^{-5}
FedPer, FedPerMeta	-	10^{-5}
FedMeta(MAML) (α, β)	$(0.01, 0.001)$	$(0.001, 0.001)$
FedMeta(Meta-SGD)(α, β)	$(0.001, 0.001)$	$(0.001, 5 \times 10^{-4})$
FedMeta-Per(MAML)(α, β)	$(0.001, 0.005)$	$(0.001, 0.001)$
FedMeta-Per(Meta-SGD)(α, β)	$(0.01, 0.01)$	$(0.001, 5 \times 10^{-4})$