Mutual Information Driven Federated Learning

Md Palash Uddin, Yong Xiang, *Senior Member, IEEE*, Xuequan Lu, *Member, IEEE*, John Yearwood, *Senior Member, IEEE*, and Longxiang Gao, *Senior Member, IEEE*

Abstract—Federated Learning (FL) is an emerging research field that yields a global trained model from different local clients without violating data privacy. Existing FL techniques often ignore the effective distinction between local models and the aggregated global model when doing the client-side weight update, as well as the distinction of local models for the server-side aggregation. In this article, we propose a novel FL approach with resorting to mutual information (MI). Specifically, in client-side, the weight update is reformulated through minimizing the MI between local and aggregated models and employing Negative Correlation Learning (NCL) strategy. In server-side, we select top effective models for aggregation based on the MI between an individual local model and its previous aggregated model. We also theoretically prove the convergence of our algorithm. Experiments conducted on MNIST, CIFAR-10, ImageNet, and the clinical MIMIC-III datasets manifest that our method outperforms the state-of-the-art techniques in terms of both communication and testing performance.

Index Terms—Distributed learning, federated learning, parallel optimization, data parallelism, information theory, mutual information, communication bottleneck, data heterogeneity

1 Introduction

In THIS age of digital transformation, there are around 3 billion smartphone users and 7 billion connected Internet of Things (IoT) devices along with a huge number of other computing and embedded devices [1]. These end-edge devices even including standalone smart sensors in distributed networks generate immense amount of user private data everyday. Federated Learning (FL), as an emerging privacy-preserving Machine Learning (ML) technique, can train these private data on-device to enhance intelligence for many real-world applications, such as image analysis, sentiment analysis, clinical care activities (in-hospital mortality prediction or heart disease prediction), burglary analysis, activity analysis, and semantic location [2], [3].

FL is an iterative scheme of training a shared neural network model in parallel over clients' own private data and aggregating the trained models into a global trained model in the server. Specifically, the server-initialized global model is downloaded by the clients for training with their private data. After training, the clients upload their models to the server. The server aggregates the local models to generate an updated global model, which is subsequently downloaded by the clients for retraining again. This process of training and aggregation is repeated until reaching a learning convergence, depicted in Fig. 1.

FL is challenging due to the following two main issues: (1) communication overhead (communication between clients and server), and (2) data heterogeneity (e.g., non-independent and identically distributed). In specific, the well-established classical data center distributed ML strategies [4], [5], [6] would be infeasible, as they just consider Independent and Identically Distributed (IID) data over a small number of clients. [2], [7] and [8] confirmed these issues and provided detailed discussion on the differences between classical distributed ML and FL. The Federated Averaging (FA) method [7] uses classical Stochastic Gradient Descent (SGD) for model training at the client-side, and further averages the local trained models in the server. Federated Stochastic Variance Reduced Gradient (FSVRG) [8] adopts Stochastic Variance Reduced Gradient (SVRG) [9] for federated optimization with the expensive full gradient calculation. Nevertheless, these two methods are seminal FL works and consider little on the communication overhead and data heterogeneity (These approaches are provided in supplementary document A, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2020.3040981.). Recent research such as [10] proposes Momentum FA (MFA) which utilizes the momentum Gradient Descent (GD) instead of the conventional GD for accelerating the training speed of models in FA. The authors of [11] further improve their previous work [12], by proposing an updated layer-wise weight averaging for model aggregation, called Federated Matched Averaging (FMA) to replace the coordinate-wise weight averaging of the baseline FA, which requires more additional internal communications between the server and clients for transferring each layer's weights individually. However, these methods are not essentially designed to mitigate the above two issues, because they attempt to either update client-side model training (e.g., [8], [10]) or improve server-side model aggregation (e.g., [11]) at a time, rather than conducting both at the same time.

The existing FL methods often ignore the effective distinction between current local model and the server-sent global model for client-side weight update, i.e., the client

- *Md Palash Uddin, Yong Xiang, Xuequan Lu, and Longxiang Gao are with the Deakin Blockchain Innovation Lab, School of Information Technology, Deakin University, Geelong, VIC 3220, Australia. E-mail: {mpuddin, yong.xiang, xuequan.lu, longxiang.gao}@deakin.edu.au.*
- *John Yearwood is with the School of Information Technology, Deakin University, Geelong, VIC 3220, Australia. E-mail: john.yearwood@deakin.edu.au.*

Manuscript received 12 May 2020; revised 4 Sept. 2020; accepted 12 Oct. 2020. Date of publication 26 Nov. 2020; date of current version 11 Feb. 2021. (Corresponding author: Yong Xiang.)
Recommended for acceptance by P. Balaji, J. Zhai, and M. Si.
Digital Object Identifier no. 10.1109/TPDS.2020.3040981

replaces its current model by the downloaded global model for training on its private dataset at every communication round. Along with this, all local trained models uploaded to the server are considered indiscriminately for server-side model aggregation at every communication round. As data are often not evenly distributed among the clients and are often of non-IID type, the client's current trained model can moderately be utilized to minimize its training loss when training the server-sent global model. Also, the training performance of all local models might not be effective, that is, some local models can be outliers in respect to the overall training objective.

To address the aforementioned issues, we, therefore, propose a novel FL approach in this paper. In particular, we propose to update the model weights at client-side by exploiting the Mutual Information (MI) between the current local model and the shared global model. To enable effective selection of local trained models for aggregation, we further attempt to measure the MI between local models and the shared global model, and choose the local models based on the computed MI values. We also show the convergence analysis for our algorithm, and visualize the learning progresses through MI. Extensive experiments on IID and non-IID data have been conducted on different datasets (i.e., MNIST, CIFAR-10, ImageNet, and MIMIC-III) to validate the proposed method. Results show that it outperforms state-of-the-art techniques, in terms of communication speedup and testing performance. The main contributions of this paper are

- an MI-driven federated optimization approach,
- a novel model update algorithm for local models,
- a novel model aggregation algorithm, and
- a theoretical convergence analysis for our method

The remainder of this paper is formulated as follows. We provide the related works in Section 2. Section 3 describes the proposed FL approach systematically. Subsequently, we explain the theoretical convergence analysis of the proposed optimization strategy in Section 4. Section 5 analyzes and compares the experimental outcomes whereas Section 6 summarizes the observations and accomplishes the paper.

2 Related Work

In this section, we only review the most relevant researches to this work. FL dates back to several years ago, and has attracted increasing attention since then. The FA, i.e., vanilla FL algorithm, performed the classical SGD parallelism for client-side model training and averaged the local trained models for the server-side aggregation [7]. FSVRG was presented in [8] that aims to utilize a constant learning rate and keep tracking of the global model for federated optimization where all clients' participation is required to calculate the expensive full gradients using SVRG [9] at each communication round. Recently, [13] focused on optimizing the structure of Deep Neural Network (DNN) models used in FA training through the adaptation of genetic algorithms. [14] introduced FedProx which modifies FA substantially through adding a proximal term to client's local objective function. MFA was proposed in [10] which suggests to use the momentum GD instead of the classical

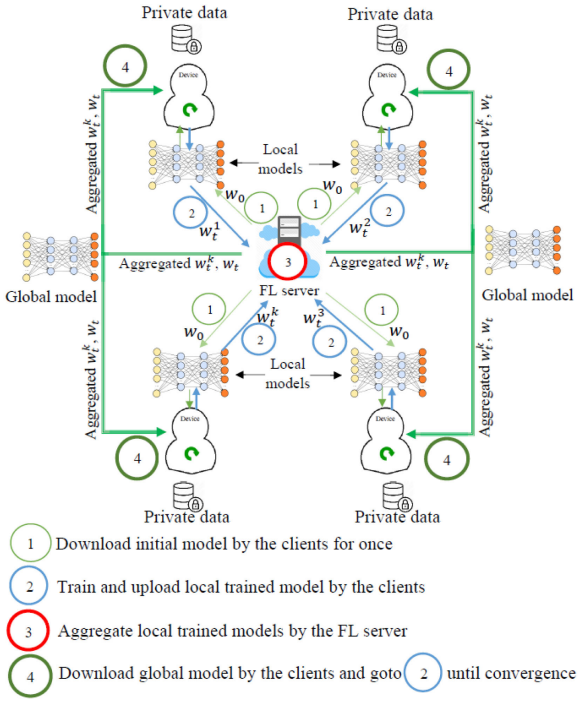


Fig. 1. Interaction among clients and server in FL paradigm. The clients train the downloaded global model ($w_t, t \geq 0$) using their own private data in parallel and upload the trained models to the FL server. The server aggregates these models ($w_t^k, t \geq 1$) and sends the aggregated model to the clients. This process repeats until reaching convergence.

GD and thus achieves faster training of the local models in comparison to the FA. A self-balancing FL approach named Astraea was proposed in [15] which relieves global imbalance through data augmentation and downsampling operations and local imbalance through creating mediators. It uses Kullback–Leibler divergence of clients' data for rescheduling the training process of the clients. The work in [12] was further improved by [11] which presents a layer-wise weight averaging algorithm called FMA for model aggregation instead of the coordinate-wise weight averaging of FA. As a result, FMA outperforms FA and FedProx, but demands more internal communications between the server and clients for transferring each layer's weights.

3 Proposed Approach

3.1 Approach Overview

Our approach consists of two steps: model weights update and aggregation. For the first step (called "MIFL-Update"), we propose to update model weights by exploiting the MI for current local model and the aggregated shared model. This is to discriminate their contributions. Regarding the second step (called "MIFL-Aggregation"), we propose to rank the local models based on the MI measurement and pick up the top effective models for aggregation. The overall approach termed as MIFL is shown in Fig. 2.

3.2 Model Weights Update

Unlike vanilla FA, both the shared global model and the existing local model at each client are emphasized for next weight update in MIFL-Update approach. The goal is to

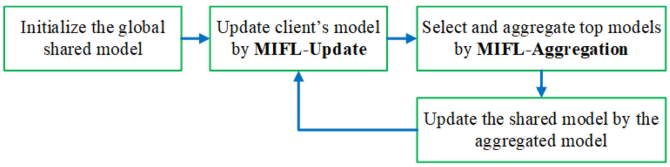


Fig. 2. Overview of the proposed MIFL.

optimally utilize individual local device's current trained model for minimizing overall training loss by the aggregated global model using the private dataset of each client. This can be defined as

$$\underset{w \in R^d}{min} f(w) = \sum_{k=1}^{K} \frac{N_k}{N} L_k(w), \tag{1}$$

where $L_k(w) = \frac{1}{N_k} \sum_{i \in M_k} l(w, i)$ represents the kth client loss calculated using the current model w over N_k samples. K is the total number of participating clients, and N is the total number of data samples of the clients. $N_k = |M_k|$ i.e., M_k is the set of data indices whose size is N_k, and $l(w, i)$ is the loss for the ith sample.

We reformulate the existing loss calculation strategy for conventional SGD algorithm [16] by exploiting the error measure between current local model and the aggregated global model. Specifically, we assume that the output F_k of current local network, k and the output F_g for the aggregated global network, g follow Gaussian functions with respective variances σ_k^2 and σ_g^2. Then, the MI between F_g and F_k, denoted by $MI(F_g; F_k)$, is defined as [17], [18], [19]

$$MI(F_g; F_k) = H(F_g) + H(F_k) - H(F_g, F_k), \tag{2}$$

where $H(F_g)$ is the entropy of F_g, $H(F_k)$ is the entropy of F_k, and $H(F_g, F_k)$ is the joint entropy of F_g and F_k, which are defined as

$$\left.\begin{array}{c} H(F_g) = \frac{1}{2}[1 + log(2\pi\sigma_g^2)] \\ H(F_k) = \frac{1}{2}[1 + log(2\pi\sigma_k^2)] \\ H(F_g, F_k) = 1 + log(2\pi) + \frac{1}{2} log[\sigma_g^2 \sigma_k^2 (1 - \rho_{gk}^2)] \end{array}\right\}. \tag{3}$$

In Eq. (3), the correlation coefficient, ρ_{gk} between F_g and F_k can be computed as

$$\rho_{gk} = \frac{E[(F_g - E[F_g])]E[(F_k - E[F_k])]}{\sigma_g^2 \sigma_k^2}, \tag{4}$$

where $E[.]$ denotes the mathematical expectation. Substituting Eq. (3) into Eq. (2), we get

$$MI(F_g; F_k) = -\frac{1}{2} log(1 - \rho_{gk}^2). \tag{5}$$

From Eq. (5), it can be observed that, i) when the outputs F_g and F_k are uncorrelated, the correlation ρ_{gk} is zero and their MI becomes very small; ii) when the outputs F_g and F_k become highly positively correlated, the correlation ρ_{gk} is close to 1 and their MI tends to be noticeably large. To enable the update using client's current local and aggregated global networks, the correlation coefficient ρ_{gk} between the current local network and the aggregated global network should be minimized, which is equivalent to minimizing the MI between them, to extract distinct information about the inputs [17], [18], [19]. Supposing that the aggregated global model represents local models of other clients, we formulate this minimization problem via introducing a correlation penalty term in the traditional loss function in Negative Correlation Learning (NCL) [17], [18], [19]. The final classical loss function E_g, a typical notion of $l(w)$, is therefore

$$\begin{aligned} E_g &= \frac{1}{N_k} \sum_{i=1}^{N_k} E_g(x_i) \\ &= \frac{1}{N_k} \sum_{i=1}^{N_k} [\frac{1}{2} (F_g(x_i) - y_i)^2 + \lambda \alpha_g(x_i)], \end{aligned} \tag{6}$$

where $E_g(x_i)$ denotes the loss function for the ith sample which is represented by sample-label pair as (x_i, y_i) and λ is a threshold $(0 \leq \lambda \leq 1)$. $\alpha_g(x_i)$ is the correlation penalty for the x_i^{th} sample that can be defined as

$$\alpha_g(x_i) = -\frac{1}{2} (F_g(x_i) - F_k(x_i))^2. \tag{7}$$

To enable effective training of FL local model at client-side through minimizing α_g, we take the partial derivative of Eq. (6) with respect to F_g, and obtain

$$\begin{aligned} \frac{\partial E_g(x_i)}{\partial F_g(x_i)} &= F_g(x_i) - y_i - \lambda(F_g(x_i) - F_k(x_i)) \\ &= (1 - \lambda)(F_g(x_i) - y_i) + \lambda(F_k(x_i) - y_i), \end{aligned} \tag{8}$$

where λ is a regularization parameter, which is used to dispose the strength of the penalty. Eq. (8) is then added to the standard SGD algorithm for minimization [17], [18], [19]. Note that we also provide the derivation for the cross-entropy based loss in our supplementary document B, available online.

Notice that we need to make a bit adjustment for the first training round of the clients. After downloading the initial or intermediate aggregated model, a client takes it as its local model for training with conventional loss calculation function of SGD, following the vanilla FA [7]. Our MIFL-Update approach is summarized in Algorithm 1.

3.3 Model Aggregation

Due to data heterogeneity or data discrepancy among clients, some local trained models can be similar to others, and some other models can be greatly different from others. Over-working models with excessively high performance and primitive models with excessively low performance are collectively called outlier models. It is obvious that local models with similar performance can increase redundancy in models' aggregation and also increase communication overhead significantly [2]. As such, we propose to quantitatively estimate the performance of all local trained models and rank their priorities before determining the models to be uploaded to the server for aggregation. Inspired by the input features ranking for different vision and visual tasks based on MI [20], [21], [22], [23], we consider the local models as features to

be selected for the aggregation task. This way finds an effective subset of local trained models for aggregation, and the communication cost among the clients and server can be thus reduced.

Algorithm 1. MIFL-Update

Input: Server-sent shared model, w_t at round t
Output: Updated model, w_{t+1}^k and MI, MI_{t+1}^k
procedure *MIFL-Update*(k, t, w_t) :
▷ Run on FL client k
  if kth's first update, **then**
    Set local model, $F_k := w_t$
    Set client's model, $F_g := w_{t+1}^k := w_t$ to be trained
    ß:= (Split training data into batches of size B)
    for each local epoch i from 1 to E **do**
      for each batch $b \in$ ß **do**
        $x(b)$:=Load data points of bth batch
        $y(b)$:=Load true labels for $x(b)$
        $\Omega := (F_g(x(b)) - y(b))$
        ▷ Classical loss calculation
        $w_{t+1}^k := w_{t+1}^k - \eta\Omega$
        ▷ Local model's weight update
        ▷ η is the learning rate
      end for
    end for
  else
    Set local model, $F_k := w_t^k$
    ▷ w_t^k is client's previous model
    Set client's model, $F_g := w_{t+1}^k := w_t$ to be trained
    ß:= (Split training data into batches of size B)
    for each local epoch i from 1 to E **do**
      for each batch $b \in$ ß **do**
        $x(b)$:=Load data points of bth batch
        $y(b)$:=Load true labels for $x(b)$
        $l_g := F_g(x(b) - y(b))$
        $l_k := F_k(x(b) - y(b))$
        $c_v := \frac{\sigma(l_g, l_k)}{\sum (l_g, l_k)}$
        if $l_g = l_k$ **then**
          $\lambda := \frac{1}{2}$
        else
          if $c_v <= \frac{1}{2}$ **then**
            $\lambda := c_v$
          else
            $\lambda := 1 - c_v$
          end if
        end if
        $\Omega := (1 - \lambda)(F_g(x(b)) - y(b)) +$
        $\lambda\,(F_k(x(b)) - y(b))$
        ▷ Modified loss calculation
        $w_{t+1}^k := w_{t+1}^k - \eta\Omega$
        ▷ Local model's weight update
        ▷ η is the learning rate
      end for
    end for
  end if
  $F_g := w_{t+1}^k$
  $MI_{t+1}^k := MI(F_g; F_k)$
  return w_{t+1}^k, MI_{t+1}^k
end procedure

Given the trained local models via Algorithm 1, the MI between the outputs of the local trained network and the previous global network is calculated via Eq. (5), where the correlation coefficient ρ_{gk} is

$$\rho_{gk} = \frac{\sum_{i=1}^{N_k}(F_g(x_i) - E[F_g(x_i)])(F_k(x_i) - E[F_k(x_i)])}{\sqrt{\sum_{i=1}^{N_k}(F_g(x_i) - E[F_g(x_i)])^2 (F_k(x_i) - E[F_k(x_i)])^2}}. \tag{9}$$

The MI between the outputs of the local model and global model is uploaded to the server. The server ranks the local trained models' MI values in an increasing order to identify the outlier models (over-working and primitive models). A lower MI means a better learning as it forces the network to have different weights with different inputs. The models with excessively low MI values are over-working, while the models with too high MI are thus primitive. The server picks up K_{Top} top effective models (say, $w_{t+1}^k, k \in [1, K_{Top}]$) by eliminating the over-working and the primitive models for aggregation. The aggregation manner is then similar to vanilla FA, shown in Eq. (10)

$$w_{t+1} = \sum_{k=1}^{K_{Top}} \frac{N_k}{N_{Top}} w_{t+1}^k, \tag{10}$$

where w_{t+1} is the aggregated global model and N_{Top} represents the total number of samples of K_{Top} clients. Notice that the selection of K_{Top} local models does not affect the number of total parameters of the global model, and it influences the values of the parameters of the global model. This is because the outlier models are not used for calculating the global model, to alleviate the misleading of the overall training process and the delay of achieving the convergence.

Algorithm 2. MIFL-Aggregation

Input: Clients' models, w_{t+1}^k and MI, MI_{t+1}^k, and targeted accuracy, $T_{accuracy}$
Output: Aggregated model, w_{t+1}
procedure *MIFL-Aggregation* () :
▷ Run on FL server
  Initialize w_0
  for each round $t := 0, 1, 2, \ldots, T-1$ **do**
    if $T_{accuracy}$ not achieved **then**
      Choose $U := \lceil (C \times K) \rceil$ clients, $0 < C \leq 1$
      for client $k \in U$ in parallel **do**
        $(w_{t+1}^k, MI_{t+1}^k) :=$ *MIFL-Update* (k, t, w_t)
      end for
      K_{Top} := Compute the set of effective clients pruning outliers based on MI_{t+1}^k
      $w_{t+1} = \sum_{k=1}^{K_{Top}} \frac{N_k}{N_{Top}} w_{t+1}^k$
      ▷ Aggregation of effective models
      Send w_{t+1} to outlier clients
    else
      break
    end if
  end for
end procedure

After aggregation, the global model is sent to the newly selected clients to train this aggregated shared model along with their respective local models using the MIFL-Update algorithm (Algorithm 1). As with FA [7], a small fraction of total

clients is randomly selected as the new clients. However, this updated global model is also delivered to those clients which are over-working and primitive in this round, for replacing their previous global model of the previous round. Similarly, the clients' updated models are ranked and selected for the next aggregation. This process of training and aggregation is repeated until a targeted learning convergence is reached. Algorithm 2 shows the model aggregation procedure.

4 Convergence Analysis

To analyse the convergence of our proposed federated optimization algorithm (MIFL), we focus on the IID linear regression task where each sample is represented by (x_i, y_i) and the associated classical loss is calculated as $l(w,i) = (y_i - x_i^\top w)^2$. Notice that non-IID data appear to be diverse, such as class imbalance, size imbalance and feature imbalance. As a result, it involves various factors and still requires researchers' significant efforts on modelling the non-IID data setting in FL. That is why we do not carry out the convergence analysis on the non-IID data, and we believe that the convergence analysis on non-IID data will be a promising research direction in FL context in future.

4.1 Setting

Suppose that the conventional loss function, $l : \mathbb{R}^d \to \mathbb{R}$ in linear regression is defined as follows:

$$l(w) = \frac{1}{2}\mathbb{E}_{(x,y)\sim\mathbb{P}}(y - w^\top x)^2, \tag{11}$$

where $\mathbb{P}$ denotes a fixed distribution on $\mathbb{R}^d \times \mathbb{R}$, $\mathbb{P}_X$ represents the marginal distribution on $\mathbb{R}^d$ and $\mathbb{P}(.|x)$ represents a conditional distribution of $\mathbb{R}$ given $x \in \mathbb{R}^d$. In the FL setup on IID data distribution, each client is assumed to have access to independent samples from $\mathbb{P}$ and the goal is to minimize the overall loss l. In our proposed MIFL, the MIFL-Update algorithm calculates $l(w) = (1-\lambda)(F_g(x-y)) + \lambda(F_k(x-y))$ through an adaptive weighted-combination (by λ) of losses of a client's current local model (F_k) and the aggregated global model (F_g). While *Client_Update()* of vanilla FA calculates $l(w) = (F_g(x-y))$ using F_g only. Besides, the MIFL-Aggregation algorithm of MIFL selects priority models to be effectively aggregated in each round while *Server_Computation()* of vanilla FA takes all models equally for aggregation.

4.2 Assumptions

The first assumption is about the uniformity of clients' upload time [24], the second is about marginal $\mathbb{P}_X$ and the last one is about conditional $\mathbb{P}(.|x)$ [25].

4.2.1 Assumption 1

Each selected client uploads its trained model within a predefined allocated time to the server for aggregation.

4.2.2 Assumption 2

(i) For few $R>0$, $\mathbb{E}_{x\sim\mathbb{P}_X}[||x||^2 xx^\top] \preceq R^2\mathbb{E}[xx^\top]$ exists, where the representation $A \preceq B$ signifies that $B - A$ is positive semi-definite for symmetric matrices $A, B \in \mathbb{R}^{d\times d}$. Particularly, this relationship exists when $||x|| \leq R$ in the marginal distribution $\mathbb{P}_X$. (ii) The lowest eigenvalue $\mu = \Lambda_{min}\mathbb{E}_{x\sim\mathbb{P}_X}[xx^\top]$ of the hessian $\mathbb{E}_{x\sim\mathbb{P}_X}[xx^\top]$ of l is strictly positive.

In linear regression, (i) is a common assumption [25], while (ii) implies that l is μ-strongly convex, where a condition number κ is defined as $\kappa = \frac{R^2}{\mu}$.

4.2.3 Assumption 3

There holds a vector $w_* \in \mathbb{R}^d$ for which for all $x \in \mathbb{R}^d$, $y(x)$ distributed through $\mathbb{P}(.|x)$ complies with $y(x) = x^\top w_* + \zeta$. ζ is a random variable complying: (i) ζ is not dependant on x, (ii) $\mathbb{E}\zeta = 0$, and (iii) $\mathbb{E}\zeta^2 \leq \sigma^2$ for some $\sigma^2 < \infty$.

If all three assumptions hold, then w_* implies that it is an unique minimizer of the overall loss function l. Moreover, it is clear from Assumption 3 that as ζ is independent of x, it represents that a well-specified model is produced.

4.3 Convergence Result

We now attempt to express the convergence of MIFL with tail-averaged SGD as *MIFL-Update()*, while the aggregation approach *MIFL-Aggregation()* is expected to produce a result in conv$\{w^1, \ldots, w^K\}$ (K is the total number of clients).

4.3.1 Proposition

Considering l from Eq. (11), we suppose that the above assumptions exist. Also, we suppose the aggregation procedure *MIFL-Aggregation()* $\in$ conv$\{w^1, \ldots, w^K\}$ for all $w^1, \ldots, w^K \in \mathbb{R}^d$, and $E > (8\log 2)\kappa\log\kappa$. Then, the aggregation output w_T using K_{Top} clients' models, $\eta = (1/2R^2)$ and the amount of SGD iterations $E_t = E$ for all t, satisfy the following [25], [26]

$$\mathbb{E}l(w_T) - l(w_*) \leq 2^{-T}(l(w_0) - l(w_*)) + \frac{16d\sigma^2}{E}. \tag{12}$$

Alternatively, if we consider local SGD epochs $E_t = 2^t E$, then we have

$$\mathbb{E}l(w_T) - l(w_*) \leq 2^{-T}(l(w_0) - l(w_*)) + \frac{8d\sigma^2 T}{2^{T-1}E}. \tag{13}$$

4.3.2 Proof

Suppose that ℓ_t is sigma algebra produced by w_t and w_t^k is the updated model of each selected client. The output model of *MIFL-Update()* having batch size $B = 1$ complies with the following guarantee [25]

$$\mathbb{E}[l(w_t^k)|\ell_t] - l(w_*) \leq 2e^{-\frac{E_t}{8\kappa\log\kappa}}(l(w_t) - l(w_*)) + \frac{8d\sigma^2}{E_t}. \tag{14}$$

As $E_t > (8\log 2)\kappa\log\kappa$ for each t, we get

$$\mathbb{E}[l(w_t^k)|\ell_t] - l(w_*) \leq \frac{1}{2}(l(w_t) - l(w_*)) + \frac{8d\sigma^2}{E_t}. \tag{15}$$

Since our aggregation strategy *MIFL-Aggregation()* picks a point in the convex hull of $\{w_t^1, \ldots, w_t^k\}$, it satisfies the convexity of l as follows:

$$l(w_{t+1}) \leq l(w_t^k), \tag{16}$$

for each client k, $k \in K_{Top}$. If we take the expectation of Eq. (15) over ℓ_t and use Eq. (16), we then get

$$\mathbb{E}l(w_{t+1}) - l(w_*) \leq \frac{1}{2}(\mathbb{E}l(w_t) - l(w_*)) + \frac{8d\sigma^2}{E_t}. \quad (17)$$

To this end, we unfold this sum over t global epochs and obtain

$$\mathbb{E}l(w_T) - l(w_*) \leq 2^{-T}(l(w_0) - l(w_*)) + 8d\sigma^2\left(\frac{1}{E_{T-1}} + \frac{1}{2E_{T-2}} + \cdots + \frac{1}{2^{T-1}E_0}\right). \quad (18)$$

When $E_t = E$ for each t, then the series in the second term of right hand side of Eq. (18) is upperly bounded by $2/E$, which finishes the proof of Eq. (12). On the other hand, when $E_t = 2^t E$, then the second term sums to $8d\sigma^2 T/(2^{T-1}E)$, which thus proves Eq. (13).

4.4 Discussion

- It can be seen that the bounds in the proposition do not depend on the number of fraction of clients (C) participated in the optimization, which entails that a large number of clients may be engaged for training convergence. Particularly, the bounds convey that the model aggregation in convex hull cannot be worse than executing same optimization method with a single client having $\frac{1}{U^{th}}$ of the entire data.
- When Assumption 3 does not exist i.e., if the model is mis-specified, similar convergence can be achieved with a tiny learning rate η and a great number of local iterations as $E_t = 2^t E$. This is practical because the computation cost of local epochs is relatively inexpensive in comparison to the communication [25].
- It is evident from [27] that combining losses is faster at reaching convergence. The proposed MIFL-Update also benefits from this since it combines losses from both client's current local and global models to calculate $l(w)$ (Eq. (11)). Since MIFL-Aggregation picks the most effective local models using MI values for the aggregation, Eq. (16) must hold for the proposed method, and we can conclude that our method converges.

5 Experimental Results

5.1 Experimental Settings

To better interpret the effectiveness of our proposed MIFL, we inherit and modify a bit the experimental settings in [7]. In particular, we use similar model architectures i.e., Multi-Layer Perceptron (MLP) and Convolutional Neural Network (CNN) with same specifications for similar image datasets. We first consider to extend parallelism using different values of client fraction C and then increase computations in each client. In contrast to [7], we evaluate a real-world clinical critical care dataset-MIMIC–III [28] and a large image classification dataset–ImageNet [29]. Besides, we visualize the training and testing progresses for analysing the learning behaviour from the MI perspective and consider noise free outputs for calculating the MI between the local model and global model.

TABLE 1
Datasets

Dataset	Training set	Testing set	Labels	Channels	Resolutionor features
MNIST	60K	10K	10	1	28×28
CIFAR-10	50K	10K	10	3	32×32
ImageNet	1.2M	50K	1000	3	224×224
MIMIC-III	22.5K	7.5K	2	1	1473

Datasets. The widely used image classification datasets MNIST, CIFAR-10, and ImageNet along with MIT's health care dataset MIMIC-III have been utilized to validate the superiority of the proposed MIFL over vanilla FA. The parametric information of the datasets is provided in Table 1. There are 26 tables in MIMIC-III such as patients' admissions, diagnosis codes. Among them, we extract three raw tables (ADMISSIONS, PATIENTS and PRESCRIPTIONS) for our experiments. We first form two new tables from the three ones. The first table named PERSONAL_INFORMATION is made through an inner-join of ADMISSIONS and PATIENTS, and contains AGE_GROUP, GENDER and MORTALITY (survival status) of all patients. The second table called SUBJECT_DRUG_TABLE is extracted from PRESCRIPTIONS, and contains the usage of drugs during first 48 hours of stay (STARTDATE-ENDDATE=48 hours) of each patient. The two tables are further joined on SUBJECT_ID to prepare a dataset of 30,000 samples, from which we randomly select 22,500 samples for the training set and the rest is the test set. In this dataset, DRUGS involving 1,473 different drug features (e.g., D5W, Heparin Sodium, Nitro-glycerine, Insulin, Atropine Sulphate etc.) prescribed to the patients are used as data points while MORTALITY is the class label to be predicted.

Model Structure. Similar to the baseline FA [7], we employ two commonly used DNN model categories i.e., MLP and CNN with similar architecture. For the MNIST digit recognition task, we use 2-hidden layers of MLP (termed as 2NN MNIST), with 200 units in each hidden layer and ReLu activation (network parameters: 199,210), as well as CNN with two 5×5 convolutional layers, a fully connected layer with 128 units having ReLu activation (network parameters: 454,922) and a final 10-class softmax output layer for both models. The numbers of channels in the convolutional layers are respectively 32 and 64, and each convolutional layer is followed by a 2×2 max pooling. For MIMIC-III, 3-hidden layers of MLP (denoted as 3NN MIMIC) with respectively 200, 100 and 50 neurons are implemented with the ReLu activation, and finally a single-neuron sigmoid output layer is employed (network parameters: 320,051). For the CIFAR-10 dataset, we use CNN with three 3×3 convolutional layers, a fully connected layer with 64 units involving ReLu activation and a final 10-class softmax output layer (network parameters: 122,570). The numbers of channels in the convolutional layers are 32, 64 and 64, respectively, while each convolutional layer is followed by a 2×2 max pooling. Finally for ImageNet dataset, we employ the mostly used network architecture called

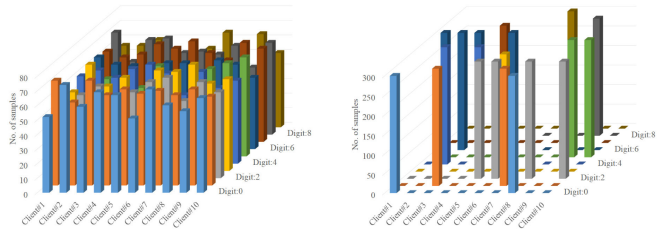


Fig. 3. Data distribution on MNIST for 10 typical clients used in the experiments. (a) IID case and (b) Non-IID case.

ResNet-50 (denoted as ResNet ImageNet; network parameters: 25,557,032) [30].

Data Distribution. To validate the FL algorithms, it is important to discuss the data distributions (IID and non-IID) over the participating clients. Aligned with [7], the MNIST dataset is assumed to be both IID and non-IID, CIFAR-10 and ImageNet IID, and MIMIC-III non-IID. For the IID MNIST, the entire 60,000 training samples are shuffled and then partitioned into 100 clients, each of which has 600 samples of all classes. For the non-IID MNIST, the training samples are sorted by digit labels and divided into 200 fragments of size 300. After that, 2 fragments are assigned to each of 100 clients that signify skewed non-IID manner as each client has only samples of two classes. An instance of IID and non-IID distribution for a set of 10 clients is depicted in Fig. 3. For the IID CIFAR-10 and ImageNet, the entire training datasets are divided among 100 clients so that each client has 500 and 12,000 training examples, respectively. However, for the non-IID MIMIC, the training set is sorted according to AGE_GROUP and GENDER, and is equally split and assigned to 100 clients.

Hyperparameters Tuning. There are some model-oriented hyperparameters, e.g., learning rate (η), local batch size (B) and local epochs (E), some existing federated optimization hyperparameters like fraction of clients (C), and our hyperparameter (λ in Eq. (8)). For η, we simply set it to 0.1 and 0.001 for image datasets and MIMIC dataset respectively, based on the naive-approach recommendation and some recent works such as [13] and [3]. B, E and C are tuned consistently for both existing FL algorithms and ours, to allow a better understanding of different methods. λ is empirically tuned based on the models' performance (observing the error values of global and local models), which would induce a decent contribution of a client's local model in training the global model.

Evaluation Metric. The mitigation of communication overhead is evidenced by the reduction of communication rounds to reach a targeted test-set accuracy for any data distribution setups; meanwhile, the mitigation of data heterogeneity issue is achieved implicitly. Particularly, the non-IID data signifies an instance of data heterogeneity difficulty, where the training data have been assigned among the clients in an imbalanced way. In general, the non-IID data create more difficulties for the training algorithms and thus demand comparatively more training rounds to reach the convergence than IID data. A potential way to prove that the non-IID data problem (heterogeneity) has been mitigated is to provide faster training convergence. Therefore, the reduction of communication rounds also serves a proof for mitigating heterogeneity. As the proposed MIFL-Update and MIFL-Aggregation algorithms respectively follow the local model training and global model aggregation procedures of the baseline FA, our MIFL has no additional overall overhead in comparison to the vanilla-FA. Note that the modified loss function in MIFL involves the additional output calculation of client's current model to train the server-sent global model, which may increase negligible computation according to our empirical observations.

5.2 Parallelism

We first consider to increase multi-client parallelism through experimenting on different values of client fraction (C) for the proposed MIFL and existing FA optimization approaches. The results for both 2NN and CNN models over MNIST are illustrated in Table 2. By pre-setting a targeted accuracy for the test set, we record the number of communication rounds. We generate a testing-accuracy curve for each combination of parameters, enabling the curve almost monotonically updating through the best testing result produced over all prior rounds. The number of communication rounds is calculated where the curve crosses the pre-set accuracy. This can be better perceived through the reference of Fig. 4 in which the targeted accuracies are marked by grey lines. $B = \infty$ represents that all samples (full local batch size i.e., 600, 500 and 225 samples per client for MNIST, CIFAR-10 and MIMIC datasets respectively) for a client is used as a mini-batch for training in each round.

It is noticed that there is still a speedup by the proposed MIFL over the baseline FA when B=∞ [7]. MIFL endows a noticeable speedup for a small batch size (B=10), especially in the non-IID case when $C \geq 0.1$. Based on the experimental observations, we fix $C = 0.1$ in the rest of experiments, which usually obtains a tradeoff between the convergence rate and the computational efficiency. For the IID MNIST, our optimization reduces the amount of communication rounds and achieves speedups of (1.20-1.70×) and (1.05-1.82×) using 2NN with $E = 1$ and $E = 5$, respectively, and (2.00-3.34×) using CNN with $E = 5$. Similarly, the speedups are (1.00-1.74×) and (1.00-1.70×) using 2NN with $E = 1$ and $E = 5$, respectively, and (1.33-3.15×) using CNN with $E = 5$ for the non-IID MNIST. In general, it can be observed that our approach can enable noticeable speedups for communication. This is because our both weight update and aggregation reduce the computation.

5.3 Increasing Computation in Each Client

We now increase computation in each client by fixing C=0.1 in every communication round. This is accomplished by varying either B or E, or both. Some core results are illustrated in Fig. 4 (Some other results are provided in supplementary document C.1, available online.). Table 3 shows the results of communication rounds. We calculate u, the expected number of updates for each client in each round, as $u = (\mathbb{E}[N_k]/B)E = NE/(KB)$, and arrange the table's rows with this value. It can be seen that varying B and E for increasing u is effective, as the communication rounds are generally reduced while reaching convergence (i.e., reaching the pre-set accuracy). For CNN MNIST, the speedup

TABLE 2
Impact of Fraction of Clients C

Data distribution	B	Method	2NN MNIST, E=1: targeted test-set accuracy: 97%				
			C				
			0.0	0.1	0.2	0.5	1.0
IID	∞	FA	1298	1092	1172	1579	1931
		MIFL	1081 (1.20×)	792 (1.38×)	881 (1.33×)	1137 (1.39×)	1138 (1.70×)
	10	FA	82	58	53	51	50
		MIFL	59 (1.39×)	46 (1.26×)	39 (1.36×)	40 (1.28×)	39 (1.28×)
Non-IID	∞	FA	4387	1652	1594	1854	- (86.54% at 1929)
		MIFL	3142 (1.40×)	1329 (1.24×)	1145 (1.39×)	1349 (1.37×)	1535 (Immeasurable×)
	10	FA	3500	261	197	182	183
		MIFL	3492 (1.00×)	150 (1.74×)	129 (1.53×)	115 (1.58×)	141 (1.30×)

(a)

Data distribution	B	Method	2NN MNIST, E=5 Targeted test-set accuracy: 97% C					CNN MNIST, E=5 Targeted test-set accuracy: 99% C				
			0.0	0.1	0.2	0.5	1.0	0.0	0.1	0.2	0.5	1.0
IID	∞	FA	1109	871	812	843	1224	258	248	288	183	181
		MIFL	610 (1.82×)	538 (1.62×)	540 (1.50×)	591 (1.43×)	1092 (1.12×)	78 (3.31×)	81 (3.06×)	96 (3.00×)	63 (2.90×)	66 (2.74×)
	10	FA	397	47	39	34	34	21	18	17	20	24
		MIFL	376 (1.06×)	33 (1.42×)	26 (1.50×)	23 (1.48×)	23 (1.48×)	11 (2.00×)	7 (2.57×)	9 (1.89×)	8 (2.50×)	12 (2.00×)
Non-IID	∞	FA	1196	1096	1147	1396	1466	1172	1007	825	1038	980
		MIFL	965 (1.24×)	896 (1.22×)	854 (1.34×)	993 (1.41×)	1132 (1.30×)	686 (1.71×)	320 (3.15×)	350 (2.36×)	417 (2.49×)	402 (2.44×)
	10	FA	3497	134	117	103	96	1091	239	201	295	180
		MIFL	3496 (1.00×)	90 (1.49×)	73 (1.60×)	61 (1.69×)	58 (1.66×)	552 (1.98×)	157 (1.52×)	120 (1.68×)	129 (2.29×)	136 (1.32×)

(b)

(a) 2NN MNIST with E=1. (b) 2NN MNIST and CNN MNIST, with E=5. Each cell specifies the communication round numbers required to produce the targeted test-set accuracy. Speedups of our MIFL in parentheses are calculated based on the corresponding cell results of FA.

ranges from (1.51-3.09×) for the IID and (1.18-3.15×) for the non-IID setup, while 3NN MIMIC-III for the non-IID case produces a speedup of (1.08-2.02×). These speedups confirm that our MIFL has a great potential to be deployed in real-world applications. Although we primarily concern about the generalization performance of the proposed MIFL with the targeted test-set accuracy, it also shows the effectiveness at largely decreasing training loss, illustrated in Fig. 5 (Some other plots are provided in supplementary document C.2, available online.).

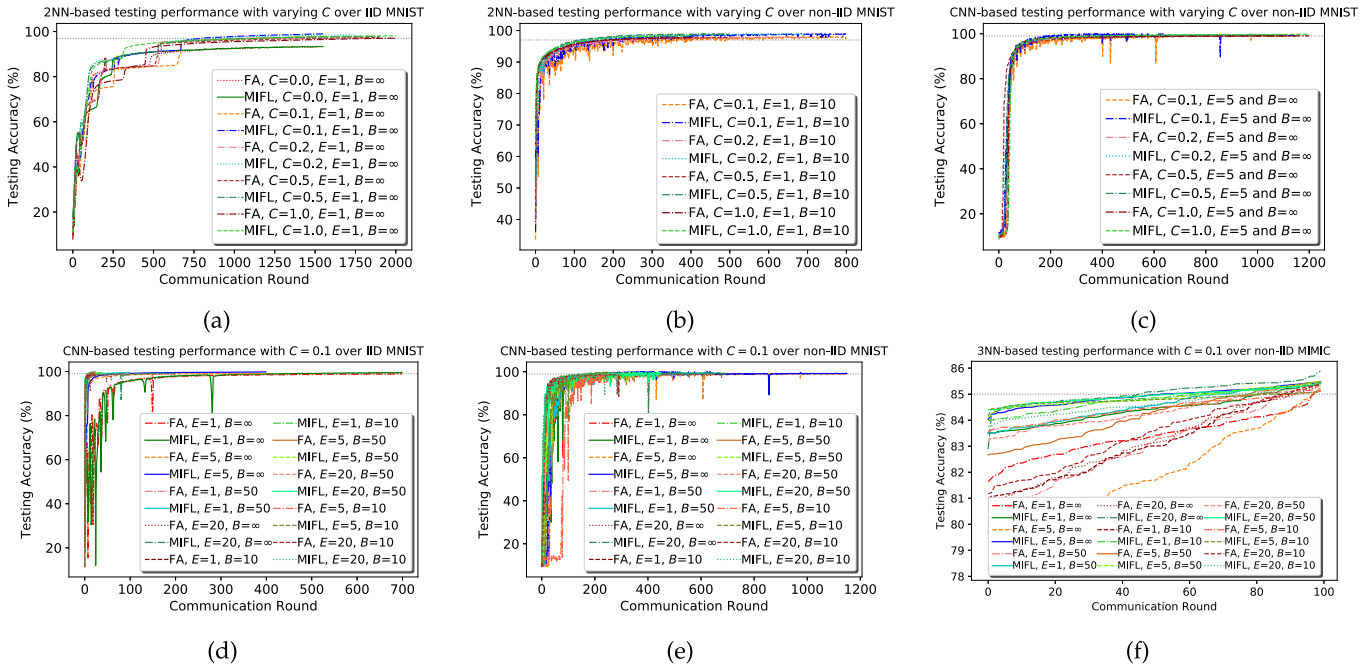


Fig. 4. Test-set accuracies versus communication rounds. (a) Varying C, $E = 1$ and $B = \infty$ over IID MNIST using 2NN. (b) Varying C, $E = 1$ and $B = 10$ over non-IID MNIST using 2NN. (c) Varying C, $E = 5$ and $B = \infty$ using CNN over non-IID MNIST. (d) C = 0.1, and varying E and B using CNN over IID MNIST. (e) C = 0.1, and varying E and B using CNN over non-IID MNIST. (f) C = 0.1, and varying E and B using 3NN over non-IID MIMIC. The targeted accuracies are marked by grey lines.

TABLE 3
Numbers of Communication Rounds Required to Reach the Targeted Test-Set Accuracy for Both MIFL and FA

CNN MNIST, targeted test-set accuracy: 99%					3NN MIMIC-III, targeted test-set accuracy: 85%		
E, B, u	FA		MIFL		E, B, u	FA	MIFL
	IID	Non-IID	IID	Non-IID		Non-IID	Non-IID
$1, \infty, 1$	700	417	431 (1.62×)	318 (1.31×)	$1, \infty, 1$	98	82 (1.20×)
$5, \infty, 5$	248	1007	81 (3.06×)	320 (3.15×)	$5, \infty, 5$	98	53 (1.85×)
1, 50, 12	57	530	38 (1.50×)	311 (1.70×)	1, 50, 4.5	93	69 (1.35×)
$20, \infty, 20$	187	633	108 (1.73×)	229 (2.76×)	$20, \infty, 20$	93	46 (2.02×)
1, 10, 60	25	269	16 (1.56×)	193 (1.39×)	1, 10, 22.5	92	64 (1.44×)
5, 50, 60	28	214	17 (1.65×)	181 (1.18×)	5, 50, 22.5	79	66 (1.20×)
20, 50, 240	31	393	16 (1.94×)	190 (2.07×)	20, 50, 90	81	56 (1.45×)
5, 10, 300	18	239	7 (2.57×)	157 (1.52×)	5, 10, 112.5	92	85 (1.08×)
20, 10, 1200	12	275	6 (2.00×)	139 (1.98×)	20, 10, 450	89	75 (1.19×)

The value of $u = EN/(KB)$ signifies the expected number of local updates per communication round.

5.4 Visualization of Training and Testing Progresses

We also calculate and visualize the MI, $MI(Y;\hat{Y})$, between the ground truth information (Y) and the logits (outputs, $\hat{Y}$) produced by every global model at each communication round using both training and testing sets. Some results are depicted in Fig. 6 (Some other results are provided in Appendix C.3, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2020.3040981.), which illustrates that the proposed MIFL with presented local models' update and aggregation strategies produces better MI values than those by the vanilla FA. The training MI values in the non-IID setup, are fluctuating, because the data is more challenging to be converged. The way of calculating MI could be an potential attempt towards interpreting the black-box of DNN in the FL setup.

5.5 MI-Based Loss Calculation for Model Training and Model Pruning for Model Aggregation

In the proposed federated optimization procedure, client's current local model is considered along with current aggregated global model to calculate the final loss to be adjusted by the global model on client's own private data. Doing this for all selected clients, the overall training loss can be significantly reduced by MIFL. This impact is shown in Fig. 5 and in supplementary document (C.2), available online. It can be observed that the combination of losses in MIFL is converged faster than the loss of the standalone global model in vanilla FA. Besides this, the proposed federated optimization introduces the pruning of primitive and over-working outlier models before aggregation in each round. We pick 2.5 percent of the selected clients' models per round as primitive and over-working separately, based on the MI values between the updated local

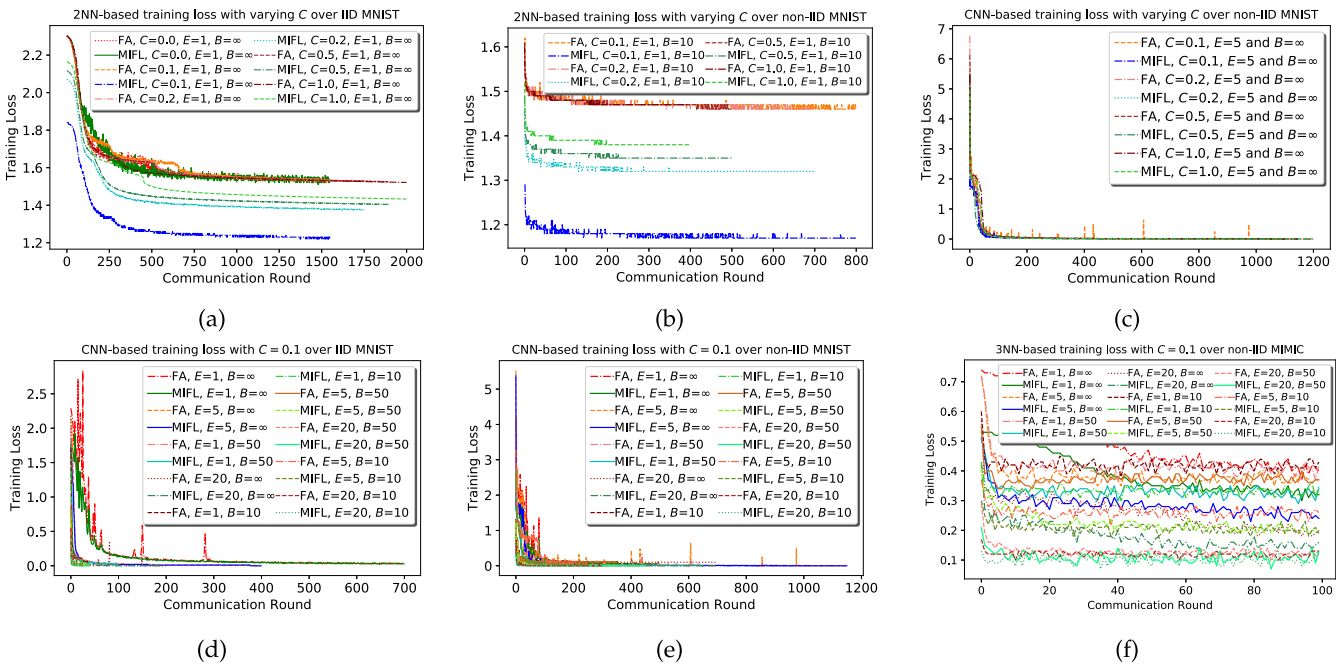


Fig. 5. Training losses versus communication rounds. (a) Varying C, $E = 1$ and $B = \infty$ using 2NN over IID MNIST. (b) Varying C, $E = 1$ and $B = 10$ using 2NN over non-IID MNIST. (c) Varying C, $E = 5$ and $B = \infty$ using CNN over non-IID MNIST. (d) C = 0.1, and varying E and B using CNN over IID MNIST. (e) C = 0.1, and varying E and B using CNN over non-IID MNIST. (f) C = 0.1, and varying E and B using 3NN over non-IID MIMIC.

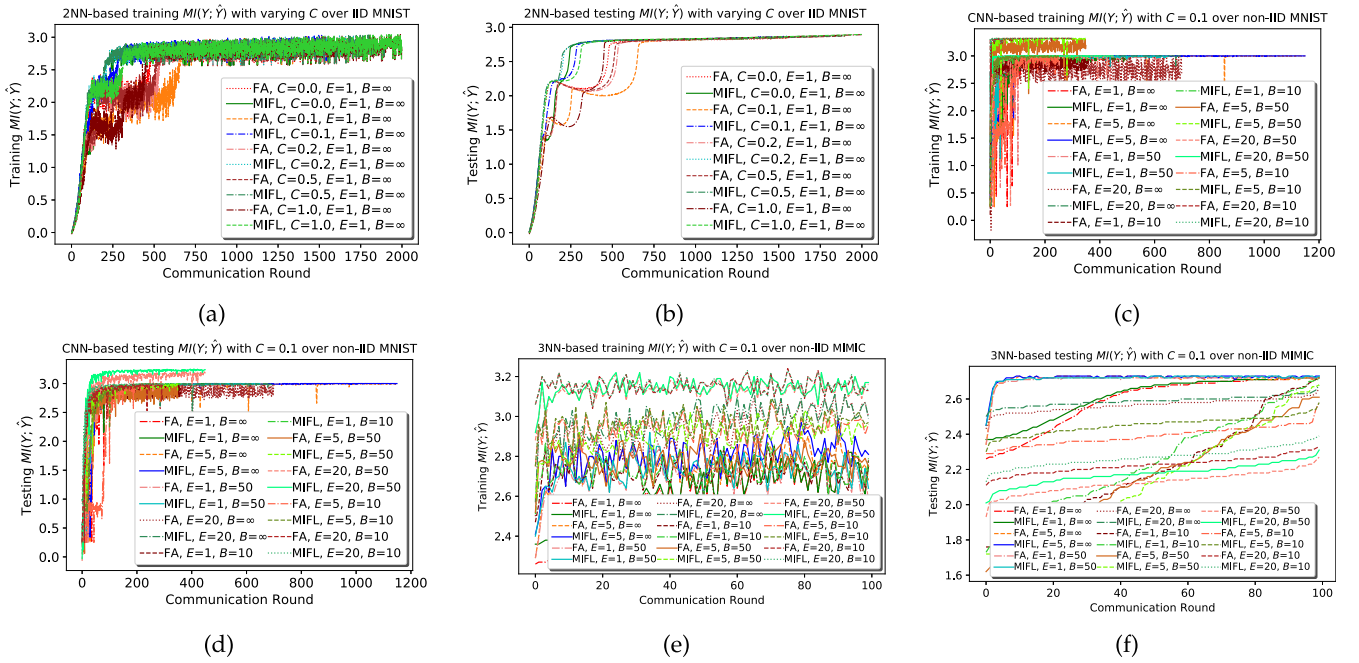


Fig. 6. Training and testing $MI(Y;\hat{Y})$ with increasing communication rounds. (a) and (b): varying C, $E=1$ and $B=\infty$ using 2NN over IID MNIST (training and testing). (c) and (d): C = 0.1, and varying E and B using CNN over non-IID MNIST (training and testing). (e) and (f): C = 0.1, and varying E and B using 3NN over non-IID MIMIC (training and testing).

model and the previous aggregated model. The primitive models are identified by the highest MI values and the over-working models by the lowest MI values. The proposed MIFL is remarkably improved by this model pruning strategy, as illustrated in Fig. 7. From the curves of both IID and non-IID cases, it can be seen that the MI values of clients' local models first increase or decrease from low or high initial values, respectively, and then attempt to convergence through tiny fluctuations.

5.6 Experiments on CIFAR-10 and ImageNet

Based on the experiments and observations on MNIST, we also consider complex image classification task on the CIFAR-10 and ImageNet datasets. Unlike FA using testing accuracy for comparisons, we set the targeted test-set accuracies to 70-80 percent for CIFAR-10 and 60-70 percent (Top-1) for ImageNet, since we validate our proposed optimization algorithm using the standard CNN and ResNet models, respectively. Table 4 shows the required numbers of communication rounds to achieve the targeted test-set accuracies, along with corresponding speedups $(1.09-3.32\times)$ for CIFAR-10 and $(1.14-1.66\times)$ for ImageNet. Fig. 8 shows the testing accuracies and training losses with the increase of communication rounds. (MI-based learning visualization plots are provided in supplementary document C.4, available online.).

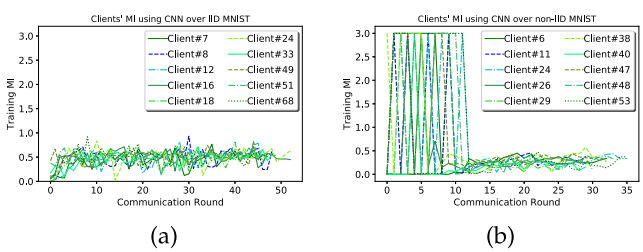


Fig. 7. MI values between clients' updated models and corresponding previous global model using CNN over MNIST training rounds of the clients (C = 0.1 and E=5). (a) IID case with B=600. (b) Non-IID case with B=10.

5.7 Extended Comparison

In addition to the above results, we also compare the FSVRG [8], MFA [10], and FMA [11] with our MIFL. Similar to above experiments, we set a targeted test-set accuracy, and run the aforementioned FL methods and MIFL on both IID and non-IID MNIST with the same CNN architecture and on non-IID MIMIC-III with same MLP architecture. We set C=0.1 for all methods except for FSVRG, as it requires all clients for training and full gradient calculation at each round, and set E={1, 5} and B=50. We provide the results in Table 5, and in Fig. 9

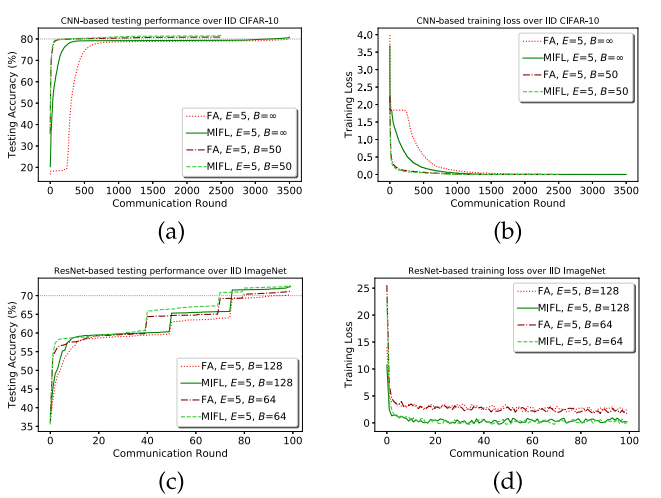


Fig. 8. CNN and ResNet-50 over CIFAR-10 and ImageNet, respectively. (a) and (b): testing accuracies and training losses for CIFAR-10. (c) and (d): testing accuracies and training losses for ImageNet.

TABLE 4
Numbers of Communication Rounds Required to Reach the Targeted Test-Set Accuracies With CNN Over CIFAR-10 and ResNet-50 Over ImageNet Using C = 0.1 and E=5

CNN CIFAR-10					ResNet-50 ImageNet				
B	Method	Targeted test-set accuracy			B	Method	Targeted test-set accuracy (Top-1)		
		70%	75%	80%			60%	65%	70%
∞	FA	432	517	3497	128	FA	51	76	96
	MIFL	130 (3.32×)	187 (2.76×)	3215 (1.09×)		MIFL	38 (1.34×)	51 (1.49×)	76 (1.26×)
50	FA	17	32	337	64	FA	41	68	81
	MIFL	13 (1.31×)	27 (1.19×)	207 (1.63×)		MIFL	32 (1.28×)	41 (1.66×)	71 (1.14×)

TABLE 5
Numbers of Communication Rounds Required to Reach a Targeted Test-Set Accuracy for Our MIFL and the Compared FL Techniques

	CNN MNIST, targeted test-set accuracy: 99%										3NN MIMIC-III, targeted test-set accuracy: 85%				
E, B	FA		FSVRG		MFA		FMA		MIFL		FA	FSVRG	MFA	FMA	MIFL
	IID	Non-IID	IID	Non-IID	IID	Non-IID	IID	Non-IID	IID	Non-IID	Non-IID	Non-IID	Non-IID	Non-IID	Non-IID
1, 50	57	530	49 (1.16×)	392 (1.35×)	50 (1.14×)	406 (1.31×)	51 (1.12×)	415 (1.28×)	38 (1.50×)	311 (1.70×)	93	89 (1.04×)	89 (1.04×)	90(1.03×)	69 (1.35×)
5, 50	28	214	26 (1.08×)	199 (1.08×)	23 (1.22×)	193(1.11×)	24 (1.17×)	194(1.10×)	17 (1.65×)	181 (1.18×)	79	70 (1.13×)	73 (1.08×)	74(1.07×)	66 (1.20×)

TABLE 6
Impact of Each Component (MIFL-Update and MIFL-Aggregation) of MIFL With E={1, 5} and B={10, 50}

	2NN MNIST, targeted test-set accuracy: 97%								3NN MIMIC-III, targeted test-set accuracy: 85%			
E, B	FA		MIFL-Update		MIFL-Aggregation		MIFL		FA	MIFL-Update	MIFL-Aggregation	MIFL
	IID	Non-IID	IID	Non-IID	IID	Non-IID	IID	Non-IID			Non-IID	
1, 50	89	213	73 (1.22×)	178 (1.20×)	80 (1.11×)	188 (1.13×)	60 (1.48×)	157 (1.36×)	93	75 (1.24×)	83 (1.12×)	69 (1.35×)
5, 50	68	173	54 (1.26×)	151 (1.15×)	59 (1.15×)	162 (1.07×)	46 (1.48×)	133 (1.30×)	79	72 (1.10×)	73 (1.08×)	66 (1.20×)
5, 10	47	134	39 (1.21×)	107 (1.25×)	41 (1.15×)	118 (1.14×)	33 (1.42×)	90 (1.49×)	92	89 (1.03×)	90 (1.02×)	85 (1.08×)

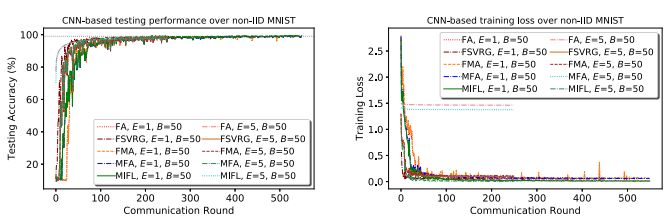
(a)

	CNN MNIST, targeted test-set accuracy: 99%								CNN CIFAR, targeted test-set accuracy: 80%			
E, B	FA		MIFL-Update		MIFL-Aggregation		MIFL		FA	MIFL-Update	MIFL-Aggregation	MIFL
	IID	Non-IID	IID	Non-IID	IID	Non-IID	IID	Non-IID			IID	
1, 50	57	530	47 (1.21×)	346 (1.53×)	48 (1.19×)	458 (1.16×)	38 (1.50×)	311 (1.70×)	546	350 (1.56×)	386 (1.41×)	278 (1.96×)
5, 50	28	214	23 (1.22×)	193 (1.11×)	25 (1.12×)	195 (1.10×)	17 (1.65×)	181 (1.18×)	337	248 (1.36×)	281 (1.20×)	207 (1.63×)
5, 10	18	239	12 (1.50×)	188 (1.27×)	15 (1.20×)	213 (1.12×)	7 (2.57×)	157 (1.52×)	292	223 (1.31×)	254 (1.15×)	179 (1.63×)

(b)

(a) 2NN MNIST and 3NN MIMIC-III. (b) CNN MNIST and CNN CIFAR.

(other results are provided in supplementary document C.5, available online.). It manifests that the MIFL still requires relatively less communications to achieve the targeted accuracy than the compared approaches.

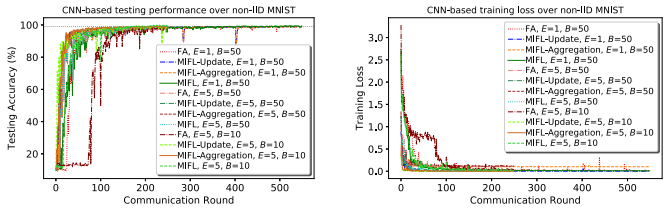


(a) (b)

Fig. 9. CNN over non-IID MNIST for the comparison of the proposed MIFL with other FL methods. (a) Testing accuracies. (b) Training losses.

5.8 Ablation Studies

We perform algorithm-level and data distribution-level ablation studies to investigate the influence of each component of our proposed MIFL. First, we remove the second component (MIFL-Aggregation) of MIFL and run the first component (MIFL-Update) on both IID and non-IID data independently, and run the aggregation scheme of vanilla FL. Then, we remove the first component (MIFL-Update) of MIFL, use the model update scheme of vanilla FL and run the second component (MIFL-Aggregation) on both IID and non-IID data individually. Additionally, we provide the results produced by our entire method (MIFL) and the vanilla FL. The results are provided in Table 6. Also, some accuracy and training loss plots are provided in Fig. 10, and other results plots are given in supplementary document C.6, available online. From the results, it can be clearly seen



(a) (b)

Fig. 10. CNN over non-IID MNIST for the ablation study. (a) Testing accuracies. (b) Training losses.

that MIFL-Update and MIFL-Aggregation can both outperform the vanilla FL in reducing communication rounds, and together they boost the communication reduction.

6 Conclusion

We have introduced a novel federated optimization approach in this work. We first present an MI-driven client-side loss calculation for model training with considering both client's local model and the aggregated global model at each communication round. The final loss from these two models leads to faster training convergence. Second, we propose a server-side outlier model pruning strategy for model aggregation, based on the MI values between clients' trained models and the previous aggregated model. Top effective local models are thus selected for better aggregation and faster learning convergence. We also present the convergence proof for our proposed optimization algorithm MIFL. We have conducted extensive experiments to validate our method, and demonstrated that it outperforms state-of-the-art techniques, in terms of communication rounds for reaching a targeted test-set accuracy.

It is thus obvious from the observations that the adoption of MI in context of client-side model training and server-side outlier model pruning could be an interesting potential research dimension. Moreover, the presented MI-based shared global model's behaviour analysis mechanism signifies that the information bottleneck principle [31] might be utilized to unfold the 'blackbox' of shared DNN model learning. In addition, as in baseline FL, there is still a direction to employ the use of additional privacy preserving schemes, such as differential privacy [32], [33], encryption strategy [34] and secure multi-party computation [35].

References

[1] K. Lueth, *State of the IoT 2018: Number of IoT Devices now at 7B-Market Accelerating*. [Online]. Available: https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/

[2] J. Park, S. Samarakoon, M. Bennis, and M. Debbah, "Wireless network intelligence at the edge," *Proc. IEEE*, vol. 107, no. 11, pp. 2204–2239, Nov. 2019.

[3] L. Huang, Y. Yin, Z. Fu, S. Zhang, H. Deng, and D. Liu, "LoAdaBoost: Loss-based AdaBoost federated machine learning with reduced computational complexity on IID and non-IID intensive care data," *PLoS One*, vol. 15, no. 4, pp. 1–16, 2020.

[4] J. Dean *et al.*, "Large scale distributed deep networks," in *Proc. Int. Conf. Neural Inform. Process. Syst.*, 2012, pp. 1223–1231.

[5] D. Povey, X. Zhang, and S. Khudanpur, "Parallel training of deep neural networks with natural gradient and parameter averaging," in *Proc. Int. Conf. Learning Representations*, 2015, pp. 1–28.

[6] S. Zhang, A. Choromanska, and Y. LeCun, "Deep learning with elastic averaging SGD," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 685–693.

[7] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, 2017, pp. 1273–1282.

[8] J. Konečny, H. B. McMahan, and P. Richtarik, "Federated optimization: Distributed machine learning for on-device intelligence," *Comput. Res. Repository*, vol. abs/1610.02527, pp. 1–38, 2016.

[9] R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2013, pp. 315–323.

[10] W. Liu, L. Chen, Y. Chen, and W. Zhang, "Accelerating federated learning via momentum gradient descent," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 8, pp. 1754–1766, Aug. 2020.

[11] H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos, and Y. Khazaeni, "Federated learning with matched averaging," in *Proc. Int. Conf. Learn. Representations*, 2020, pp. 1–16.

[12] M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenewald, T. N. Hoang, and Y. Khazaeni, "Bayesian nonparametric federated learning of neural networks," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 7252–7261.

[13] H. Zhu and Y. Jin, "Multi-objective evolutionary federated learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 4, pp. 1310–1322, Apr. 2020.

[14] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proc. Mach. Learn. Syst. Conf.*, 2020, pp. 429–450.

[15] M. Duan, D. Liu, X. Chen, R. Liu, Y. Tan, and L. Liang, "Self-balancing federated learning with global imbalanced data in mobile systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 1, pp. 59–71, Jan. 2021.

[16] M. Li, T. Zhang, Y. Chen, and A. J. Smola, "Efficient mini-batch training for stochastic optimization," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2014, pp. 661–670.

[17] Y. Liu and X. Yao, "Simultaneous training of negatively correlated neural networks in an ensemble," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 29, no. 6, pp. 716–725, Dec. 1999.

[18] Y. Liu and X. Yao, "Ensemble learning via negative correlation," *Neural Netw.*, vol. 12, pp. 1399–1404, 1999.

[19] X. Yao and Y. Liu, "Evolving neural network ensembles by minimization of mutual information," *Int. J. Hybrid Intell. Syst.*, vol. 1, pp. 12–21, Sep. 2004.

[20] R. Battiti, "Using mutual information for selecting features in supervised neural net learning," *IEEE Trans. Neural Netw.*, vol. 5, no. 4, pp. 537–550, Jul. 1994.

[21] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 8, pp. 1226–1238, Aug. 2005.

[22] P. A. Estevez, M. Tesmer, C. A. Perez, and J. M. Zurada, "Normalized mutual information feature selection," *IEEE Trans. Neural Netw.*, vol. 20, no. 2, pp. 189–201, Feb. 2009.

[23] S. Cang and H. Yu, "Mutual information based input feature selection for classification problems," *Decis. Support Syst.*, vol. 54, pp. 691–698, Dec. 2012.

[24] R. Zhang and J. T. Kwok, "Asynchronous distributed ADMM for consensus optimization," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. II-1701–II-1709.

[25] P. Jain, P. Netrapalli, S. M. Kakade, R. Kidambi, and A. Sidford, "Parallelizing stochastic gradient descent for least squares regression: Mini-batching, averaging, and model misspecification," *J. Mach. Learn. Res.*, vol. 18, no. 223, pp. 1–42, 2018.

[26] P. Jain, S. M. Kakade, R. Kidambi, P. Netrapalli, V. K. Pillutla, and A. Sidford, "A Markov chain theory approach to characterizing the minimax optimality of stochastic gradient descent (for least squares)," in *Proc. IARCS Annu. Conf. Foundations. Softw. Technol. Theoretical. Comput. Sci.*, 2017, pp. 2:1–2:10.

[27] H. Hajiabadi, D. Molla-Aliod, R. Monsefi, and H. S. Yazdi, "Combination of loss functions for deep text classification," *Int. J. Mach. Learn. Cybern.*, vol. 11, pp. 751–761, 2020.

[28] A. E. Johnson *et al.*, "MIMIC-III, a freely accessible critical care database," *Sci. Data*, vol. 3, 2016, Art. no. 160035.

[29] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[31] N. Tishby and N. Zaslavsky, "Deep learning and the information bottleneck principle," in *Proc. IEEE Inf. Theory Workshop*, 2015, pp. 1–5.
[32] N. Agarwal, A. T. Suresh, F. Yu, S. Kumar, and H. B. Mcmahan, "cpSGD: Communication-efficient and differentially-private distributed SGD," in *Proc. Int. Conf. Neural Inform. Process. Syst.*, 2018, pp. 7575–7586.
[33] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Differentially private asynchronous federated learning for mobile edge computing in urban informatics," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 2134–2143, Mar. 2020.
[34] L. Lyu *et al.*, "Towards fair and privacy-preserving federated deep models," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 11, pp. 2524–2541, Nov. 2020.
[35] K. Bonawitz *et al.*, "Practical secure aggregation for privacy-preserving machine learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2017, pp. 1175–1191.

Md Palash Uddin received the BSc degree in computer science and engineering from Hajee Mohammad Danesh Science and Technology University (HSTU), Bangladesh, and the MSc degree in computer science and engineering from the Rajshahi University of Engineering & Technology, Rajshahi, Bangladesh. He is currently working toward the PhD degree with the School of Information Technology, Deakin University, Geelong, Australia. He is also an academic faculty member with HSTU, Bangladesh. His research interests include machine learning, blockchain, and remote sensing image analysis.

Yong Xiang (Senior Member, IEEE) received the PhD degree in electrical and electronic engineering from the University of Melbourne, Melbourne, Australia. He is a professor with the School of Information Technology, Deakin University, Australia. His research interests include information security and privacy, signal and image processing, data analytics and machine intelligence, Internet of Things, and blockchain. He has published five monographs, more than 165 refereed journal articles, and numerous conference papers in these areas. He is the senior area editor of the *IEEE Signal Processing Letters*, and associate editor of the *IEEE Communications Surveys and Tutorials*, and guest editor of the *IEEE Transactions on Industrial Informatics*. He was the associate editor of the *IEEE Signal Processing Letters* and the *IEEE Access*, and the guest editor of the *IEEE Multimedia*. He has served as an honorary chair, general chair, program chair, TPC chair, symposium chair, and track chair for many conferences, and was invited to give keynotes at a number of international conferences.

Xuequan Lu (Member, IEEE) received the PhD degree from Zhejiang University, Hangzhou, China, in June 2016. He is a lecturer (assistant professor) with Deakin University, Australia. He spent more than two years as a research fellow in Singapore. His research interests mainly fall into the category of visual computing, for example, geometry modeling, processing and analysis, animation/simulation, 2D data processing, and analysis. For more information please visit: http://www.xuequanlu.com.

John Yearwood (Senior Member, IEEE) is head of the School of Information Technology, Deakin University and was previously executive dean, faculty of Science and Technology, and director, Centre of Informatics and Applied Optimization, Federation University, Australia. He was instrumental in setting up the Internet Commerce Security Laboratory with Westpac, IBM and the Victorian State Government as a joint industry-focused and data-driven laboratory on cyber security in the financial sector. He has held a number of ARC grants and was a QEII fellow working on computational narrative and argumentation in decision science. His work in data mining and computational intelligence has led to the development of new machine learning and hybrid learning algorithms for artificial neural networks, as well as new data and text mining and pattern recognition approaches. His work in decision science has developed the use of argumentation structures for the modeling of knowledge and collaborative decision making in complex domains. He has published more than 200 journal and refereed conference papers including two books. He is currently a CI on the ARC funded Discovery Project 'Enhancing and supporting deliberation in multi-disciplinary team decision-making'. He is editor-in-chief of the *Journal of Research & Practice in Information Technology* and a reviewer for a large number of journals and competitive research grant programs including the Australian Research Council grant program, the NHMRC grant program, and for the Dutch Government in the assessment of their NWO/ToKeN2000.

Longxiang Gao (Senior Member, IEEE) received the PhD degree in computer science from Deakin University, Geelong, Australia. He is currently a senior lecturer and co-founder of Deakin Blockchain Innovation Lab, School of Information Technology, Deakin University. Before joined Deakin University, he was a post-doctoral research fellow with IBM Research & Development, Australia. His research interests include fog/edge computing, blockchain, data analysis, and privacy protection. He has more than 80 publications, including patent, monograph, book chapter, journal, and conference papers. Some of his publications have been published in the top venue, such as the *IEEE Transactions on Mobile Computing*, *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Internet of Things Journal*, *IEEE Transactions on Dependable and Secure Computing*, *IEEE Transactions on Vehicular Technology*, *IEEE Transactions on Computational Social Systems*, *IEEE Transactions on Industrial Informatics*, and *IEEE Transactions on Network Science and Engineering*. He has being a chief investigator (CI) for more than 20 research projects (the total awarded amount is more than $5million), from pure research project to contracted industry research. He is active in IEEE Communication Society. He has served as the TPC co-chair, publicity co-chair, organization chair, and TPC member for many international conferences. He is an associate editor of the *IEEE Access* and the assessor of Australian Research Council (ARC) Projects.