Coalitional Game Theoretic Federated Learning

Masato Ota
Information Services
International-Dentsu, Ltd.
Tokyo, Japan

Yuko Sakurai
Graduate School of Engineering
Nagoya Institute of Technology
Nagoya, Japan
sakurai@nitech.ac.jp

Satoshi Oyama
Faculty of Information
Science and Technology
Hokkaido University
Sapporo, Japan
oyama@ist.hokudai.ac.jp

***Abstract*—This study approaches federated learning (FL) from the viewpoint of coalitional games with coalition structure generation (CSG). In conventional FL, even if each client has data from a different distribution, they still learn a single global model. However, the performance of each local model can degrade. To address such issues, we propose an algorithm in which clients form coalitions and the clients in the same coalition jointly train a specialized model for the coalition, namely a coalition model. We formulate the algorithm as a graphical coalition game given by a weighted undirected graph in which a node indicates a client and the weight of an edge indicates the synergy between two connected clients. Formulating FL as a CSG problem enables us to generate an optimal CS that maximizes the sum of synergies. We first define two types of synergy, i.e., that based on the average improvement in classification accuracy of two agents as they join the same coalition and that based on the cosine similarity between the gradients of the loss functions, which is intended to exclude adversaries having adversarial data from a set of non-adversaries. We conduct experiments to evaluate our algorithm, and the results indicate that it outperforms current algorithms.**

***Index Terms*—Machine Learning, Federated Learning, Coalitional Games, Coalition Structure Generation**

I. Introduction

Federated learning (FL) is a distributed machine learning technique proposed by Google in 2017 [1]. It enables multiple distributed clients to collaboratively learn a global model while not requiring them to send their training data to a server. Since its proposal, FL has been widely studied in various research fields. For example, Kairouz *et al.* (2021) summarized the advances in FL and discussed an extensive collection of open problems and challenges [2] .

While FL offers the strong advantage of protecting the privacy of the client's data, the performance of local models has been highlighted as an issue because the general goal with FL is to maximize the performance of the global model. As one approach to addressing this issue, existing works known as personalized FL focus on improving a local model for each client [3], [4]. As another approach, Clustered FL partitions a set of clients into clusters. Once partitioned, the clusters are trained independently but also in parallel [5]–[7].

In this paper, we propose formulating the FL framework as coalitional games. Coalitional game theory involves the study of coalition-structure generation (CSG) and payoff distribution. While coalitional game theory is a part of game theory, many researchers in artificial intelligence and multi-agent systems (MAS) have studied coalitional games for the last 20 years or so [8]. CSG (or coalition formation) is the problem to determine an optimal coalition structure so that agents are divided into coalitions to maximize the sum of coalition values, because the grand coalition is not always optimal. The payoff distribution involves how to divide the value of the coalition among agents, such as the Shapley value. In recent years, the Shapley value has attracted attention in the machine learning research field [9]. For example, it has been applied to fairly evaluate the effect of each data as a means to improve a model. By formulating FL as coalitional games, various FL problems can be separated into synergy definitions and coalition formation problems, and current theories of coalitional games can be applied.

We approach the FL framework from the perspective of graphical coalitional games, which provide a concise representation of such games. A graphical coalitional game is represented by a weighted undirected graph where a node indicates an agent and the weight of an edge indicates the degree of relationship/synergy between two agents when they belong to the same coalition. The value of a coalition is given by the sum of synergies the agents have in the coalition. In FL, each client can have data from a different data distribution; thus our goal is to generate a coalition structure in which clients with similar data form a coalition without exchanging their data with each other. We apply two types of synergy: that based on improvement in classification accuracy for two agents (ICA) and that based on cosine similarity between the gradients of the loss functions (COS). Our algorithm, called Coalition Formation for Federated Learning (CFFL), gradually improves specialized models in coalitions, called the coalition domain, by changing a coalition's structure in accordance with the update of the graph. This approach is taken because the synergy between any two agents is updated depending on the improvement of their local models. Figure 1 gives an overview of CFFL. Let us assume that client 1 has data from a specific dataset and clients 2 and 3 have data from another specific dataset. First, each client learns her local model (a) and sends her trained model to the server (b). After the server receives the trained model from each client, it creates a graph by calculating the synergies between

This work was partially supported by JSPS KAKENHI Grant Numbers JP18H03299, JP18H03337, and JP21K19833, and by JST CREST Grant Number JPMJCR21D1.

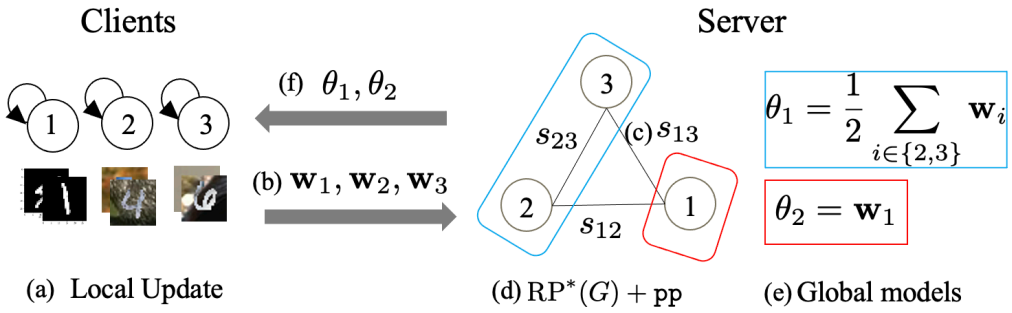

Fig. 1. Overview of the proposed algorithm

any two clients (c) then determines the coalition structure by solving a CSG problem (d). The server updates the coalition model for each coalition (e) and copies the model to the local models of the clients in the coalition (f). We formalize the CSG problem as an integer-linear programming (ILP) formulation and solve it by applying the current state-of-the-art algorithm [10]. We conducted experiments to show that CFFL with ICA-based synergy outperforms the current algorithms when no adversary exists. We also show that CFFL with COS-based synergy divides a set of clients, including adversaries who manipulate their training data, into sets of non-adversaries and adversaries then partitions the set of non-adversaries into coalitions depending on the data distributions.

The rest of this paper is organized as follows. In Section II, we present related work. In Section III, we introduce the coalition-formation problem and ILP to solve the coalition formation problem. In Section IV, we present CFFL with ICA- and COS-based synergy. Finally, in Section V, we discuss the experiment to evaluate the performance of CFFL.

II. Related Works

In a standard FL setting, independent and identically distributed (IID) data as well as other general machine-learning settings are assumed. When the IID assumption holds, the global model by aggregating local models from clients performs well.

The Federated Averaging (FedAvg) algorithm [1] is a well-known standard FL algorithm under the IID assumption. However, in a realistic problem setting, the IID assumption does not always hold. Unfortunately, under non-IID assumption, the performance of local models has been considered an issue because the goal of maximizing the performance of a global model does not mean good performance of a local model and vice versa. We consider FL under non-IID assumption.

As one approach to addressing the issue about the performance of local models, personalized FL focuses on improving a local model for each client. Dinh *et al.* (2022) proposed an algorithm for personalized FL called pFedMe using the Moreau envelope function that helps decompose a local model optimization from learning a global model [3]. Fallah *et al.* (2020) considered a data heterogeneous setting in which the data distributions for clients are not identical [4] . They formalized the problem as model-agnostic meta-learning and proposed the Personalized FedAvg (Per-FedAvg) algorithm on the basis of FedAvg. In Huang *et al.* (2021) proposed a federated attentive message passing (FedAMP) framework on the basis of the idea of the attentive message passing mechanism [11]. In FedAMP, each client has a personalized cloud model on the cloud server and collaborates with other clients with similar models by attentively passing her local model as a message.

Clustered FL partitions a set of clients into clusters. Once partitioned, the clusters are trained independently but also in parallel. Briggs *et al.* (2020) showed that learning a single joint model was often not optimal in the presence of certain types of non-IID data, thus presented a modification to FL by introducing a hierarchical clustering step to separate clusters of clients by the similarity of their local updates to the global model [5]. Ghosh *et al.* (2020) proposed the Iterative Federated Clustering Algorithm (IFCA), which alternately estimates the cluster identities of the clients and optimizes model parameters for the clients via gradient descent [6]. Sattler et al. (2020) proposed a federated multi-task learning framework to exploit the geometric properties of the FL loss surface to group a set of clients into clusters with jointly trainable data distributions [7]. Muhammad *et. al* (2020) proposed FedFast as an active aggregation method for making accurate distributed recommendations using deep neural networks and FL by using user-embedding clusters for propagating client updates in the cluster with similar clients [12].

III. Coalition Formation

A. Formalizing coalition formation

Coalitional game theory examines how self-interested agents form effective coalitions when they work together. Coalitional games are known by their characteristic function that takes a coalition as its input then returns the value of the coalition.

In this paper, we consider FL for coalitional games and create the optimal coalition structure for partitioning a set of clients into coalitions so that those clients who have data generated from the same distribution participate in the same coalition. By introducing coalitional games to the FL framework, we can use the abundant knowledge and techniques cultivated in coalitional games for the study of FL. However,

we have to resolve the important issue of how to represent the game.

To find the optimal coalition, we need to know the values of the characteristic functions for all possible combinations of coalitions, which requires setting FL for all possible combinations of coalitions. Unfortunately, this is computationally intractable because the number of all possible coalitions increases exponentially with the number of participants. On the other hand, MAS researchers have proposed concise representations for the characteristic function by adopting the assumption that an organizer/central entity knows the relationship among the participants, such as who are on good or bad terms.

However, we cannot apply such an assumption to the FL framework because a server has no information about the relationships among the clients. Thus we consider the graphical coalitional games established by Myerson [13], which are well-known for supplying a concise representation that requires no prior knowledge of the participants. Graphical coalitional games are represented by weighted undirected graphs in which a node indicates a client and the weight of an edge indicates the synergy between the two clients connected by the edge. An important issue is how to calculate synergy since the value of a coalition is determined on the basis of synergies. We define two types of synergy, ICA- and COS-based, in Section IV.

Let $N = \{1, 2, \ldots, n\}$ be a set of clients. The game is represented as an undirected weighted graph $G = (N, E, \mathbf{s})$. Each edge $(i, i') \in E$ is associated with weight $s_{i,i'} = s_{i',i} \in \mathbb{Q}$, where $\mathbb{Q}$ is the set of rational numbers. We regard the weight of (i, i') as zero, that is, $s_{i,i'} = 0$ when $(i, i') \notin E$. Let coalition $C \subseteq N$ denote a subset of clients. C value is the sum of weights among the clients in C.

Definition 1 (Value of Coalition): For any coalition of agents $C \subseteq N$, C value is given by

$$v(C) = \sum_{(i,i') \in E[C]} s_{i,i'}.$$

where $E[C] = \{(i, i') \in E \mid i, i' \in C\}$

Coalition structure CS is defined as a partition of N into disjoint and exhaustive coalitions.

Definition 2 (Coalition Structure): $CS = \{C_1, C_2, \ldots\}$ satisfies the following conditions:

$$\forall i, j \ (i \neq j), \ C_i \cap C_j = \emptyset, \ \bigcup_{C_i \in CS} C_i = N.$$

Denote the set of all coalition structures as $\Pi(N)$.
The value of CS, $V(CS)$, is given by $V(CS) = \sum_{C_j \in CS} v(C_j)$.

Coalition structure generation (CSG) involves partitioning a set of agents into coalitions to maximize the sum of the values. While it is generally assumed that the grand coalition (coalition of all agents) is the optimal coalition structure, this is not always satisfied in real-world settings.

Definition 3 (Coalition Structure Generation Problem (CSG)): The coalition structure generation problem is to obtain CS^* that satisfies

$$\forall CS, V(CS^*) \geq V(CS).$$

B. Integer Linear Programming formulation

We apply the current state-of-the-art Integer Linear Programming (ILP) formulation approach to solve CSG. This is because it can solve reasonably large problem instances by formulating a CSG as an ILP instance due to the recent advances in ILP solvers such as CPLEX and Gurobi. CSG for graphical coalitional games is identical to a clique partitioning problem that finds the optimal partition of a given weighted undirected graph, such that the sum of the weights within clusters is maximized.

Grötschel and Wakabayashi (1989) first proposed an ILP formulation, in which the goal was selecting a subset of edges such that the sum of their weights is maximized and satisfies triangle inequality constraints [14]. Decision variable $x_{i,j}$ is introduced for each edge $\{i, j\} \in E$, where $i < j$. $x_{i,j}$ equals 1 if i and j are in the same coalition and 0 otherwise. For notation simplicity, let $x_{\{i,j\}}$ denote $x_{i,j}$ when $i < j$; otherwise, $x_{j,i}$. Triangle inequality constraints are introduced for any $i, j, k \in N$. If i and j are in the same cluster and j and k are in the same cluster, then i and k must also be in the same cluster. This ILP formulation is defined as follows.

$$\max \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} s_{i,j} x_{i,j}$$

$$\begin{aligned} \text{s.t.} \quad & x_{i,j} + x_{j,k} - x_{i,k} \leq 1 \quad \forall (i,j,k) \in T \\ & x_{i,j} \in \{0, 1\} \qquad\qquad 1 \leq i < j \leq n, \end{aligned}$$

$$\text{where} \quad T = \{(i, j, k) \mid 1 \leq i < k \leq n, j \neq i, j \neq k\}.$$

The growth in the number of constraints creates a bottleneck for solving larger instances because the number of triangle inequality constraints is $\Theta(n^3)$. Miyauchi *et al.* (2018) proposed a concise ILP-based algorithm called RP*(G)+pp that reduces the number of triangle inequality constraints [10]. This algorithm has been proven to always find the optimal solution. RP*(G)+pp consists of two procedures. We first solve an ILP problem, called RP*(G) then apply post-processing, which we refer to as pp, since RP*(G) is incomplete and may fail to obtain the optimal solution. The pp decomposes the obtained partitions into weakly connected components by a depth-first search. In the search, we only examine the components connected by edges with positive weights. The pp runs in linear time with a size of G.

RP*(G):

$$\max \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} s_{i,j} x_{i,j}$$

$$\begin{aligned} \text{s.t.} \quad & x_{i,j} + x_{j,k} - x_{i,k} \leq 1 \quad \forall (i,j,k) \in T_{>0}, \\ & x_{i,j} \in \{0, 1\} \qquad\qquad 1 \leq i < j \leq n, \end{aligned}$$

$$\begin{aligned} \text{where} \quad T_{>0} = \{(i, j, k) : 1 \leq i < k \leq n, j \neq i, j \neq k, \\ w_{i,j} > 0 \vee s_{j,k} > 0\}. \end{aligned}$$

Algorithm 1 Coalition Formation for FL (CFFL)

Require: coalition model:$\theta_k^{(0)}$, Local model $\mathbf{w}_i^{(0)}$, Local dataset:$\mathcal{D}_i$, client: $i \in N$, Coalition structure: CS

for $t = 0, \ldots, T-1$ **do**
  for $i = 1, \ldots, n$ **do**
    Update $\mathbf{w}_i^{(t)} = \theta_k^{(t)}$ where $\theta_k^{(t)}$ is a coalition model of the coalition k client i joins
    $\mathbf{w}_i^{(t)}$ = Local Update($\mathbf{w}_i^{(t)}, \gamma, \tau, i$)
  end for
  Calc G_t: undirected weighted graph
  OPTION1 Improvement in classification accuracy
  OPTION2 Cosine similarity
  $CS = \{C_1, \ldots, C_k, \ldots\}$ from RP$^*(G_t)$+pp
  $\theta_k^{(t)} = \frac{1}{|C_k|} \sum_{i \in C_k} \mathbf{w}_i^{(t)}$, $C_k \in CS$
end for

Local Update($\mathbf{w}, \gamma, \tau, i$)
for each local epoch $q = 0, \ldots, \tau - 1$ **do**
  for batch $b \in \mathcal{D}_i$ **do**
    $\mathbf{w} = \mathbf{w} - \gamma \nabla L(\mathbf{w}, b)$
  end for
end for
return $\mathbf{w}$

IV. Proposed Algorithm

In this section, we present our algorithm that partitions a set of clients into coalitions in order to improve the local models of the clients. By grouping clients who have data from the same data distribution into the same coalition, they learn a good coalition model from the coalition and thus improve their local models.

A. CFFL Algorithm

As described in Section III, we formulate FL as a graphical coalitional game to find the optimal coalition structure. We propose two definitions for the synergy between two clients to make a weighted undirected graph for representing the relationship between the clients. The first definition of synergy is based on the improvement in the performance of the local models of the two clients, and the second one is given by the cosine similarity between the gradients of the loss functions of the two clients.

Our proposed algorithm is formally presented in Algorithm 1. $\theta_k^{(t)}$ denotes the coalition model for the k-th coalition in the t-th communication round. Similarly, $\mathbf{w}_i^{(t)}$ denotes the local model for client i at the t-th communication round. Let $\mathcal{D}_i$ denote the set of training data client i has. Let γ be the learning rate and let τ be the number of local epochs, which indicates how many times a client repeatedly uses the same data to update her local model. $L(\mathbf{w}_i, \mathcal{D}_i)$ denotes the empirical risk for client i.

The procedure of our algorithm is as follows: (1) A server sends the coalition model to each client and then a client copies the global model to her local model. In the initial state, all clients join a single coalition (grand coalition) with a single coalition (global) model. (2) A client learns a local model by using her training data for several epochs and then sends her trained local model to a server. (3) The server makes a weighted undirected graph by calculating the synergy between any two clients and then generates a coalition structure from RP$^*(G_t)$+pp. (4) The server generates a coalition model for each coalition by averaging the local models of the clients who belong to the same coalition. We carry out this procedure until the coalition models converge.

B. Synergy based on Improvement in Classification Accuracy

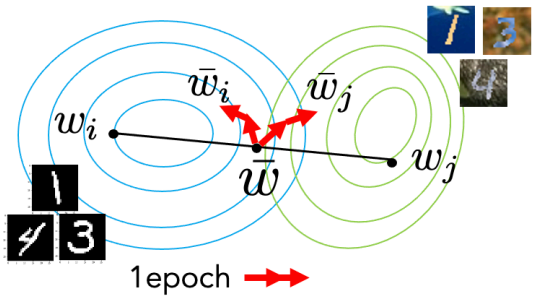

Fig. 2. Improvement in Classification Accuracy

Algorithm 2 Option1: Improvement in Classification Accuracy (ICA)

for each pair $(i, j) \in N, i < j$ **do**
  $\bar{\mathbf{w}}_{ij}^{(t)} = (\mathbf{w}_i^{(t)} + \mathbf{w}_j^{(t)})/2$
  $\bar{\mathbf{w}}_i, \bar{\mathbf{w}}_j = \bar{\mathbf{w}}_{ij}^{(t)}$
  for batch $b \in \mathcal{D}_i$ **do**
    $\bar{\mathbf{w}}_i = \bar{\mathbf{w}}_i - \gamma \nabla L(\bar{\mathbf{w}}_i, b)$
  end for
  for batch $b \in \mathcal{D}_j$ **do**
    $\bar{\mathbf{w}}_j = \bar{\mathbf{w}}_j - \gamma \nabla L(\bar{\mathbf{w}}_j, b)$
  end for
  $e_{i \to j} = \mathcal{P}(\bar{\mathbf{w}}_j, \mathcal{D}_j^{test}) - \mathcal{P}(\mathbf{w}_j^{(t)}, \mathcal{D}_j^{test})$
  $e_{j \to i} = \mathcal{P}(\bar{\mathbf{w}}_i, \mathcal{D}_i^{test}) - \mathcal{P}(\mathbf{w}_i^{(t)}, \mathcal{D}_i^{test})$
  $s_{ij} = (e_{i \to j} + e_{j \to i})/2$
end for
return s

The synergy based on the improvement in classification accuracy (ICA) is calculated from the mean improvement in local models for the two clients as shown in Figure 2. Let $\bar{\mathbf{w}}_{ij}$ be the mean of the parameters of the local models $\mathbf{w}_i^{(t)}, \mathbf{w}_j^{(t)}$ for clients i and j in round t. Let $\bar{\mathbf{w}}_i$ be parameters of the local model for client i, which are obtained by fine tuning (training for one epoch from the value $\bar{\mathbf{w}}_{ij}$) using the data D_i owned by client i. We calculate the improvement in classification accuracy of client i's local model for client i's test data D_i^{test}.

$$e_{j \to i} = \mathcal{P}\left(\bar{\mathbf{w}}_i, \mathcal{D}_i^{\text{test}}\right) - \mathcal{P}\left(\mathbf{w}_i^{(t)}, \mathcal{D}_i^{\text{test}}\right),$$

where $\mathcal{P}\left(\overline{\mathbf{w}}_i, \mathcal{D}_i^{\text{test}}\right)$ refers to the classification accuracy of model $\overline{\mathbf{w}}_i$ for her own test data D_i^{test} and is represented from 0 to 1. We define this type as follows.

$$s_{i,j} = \frac{(e_{i\to j} + e_{j\to i})}{2}.$$

The reason client i trains the model starting from $\bar{\mathbf{w}}_{i,j}$ for one epoch using her data is that if $e_{i\to j}$ is calculated using the mean model, the synergy might be a negative value. This is because if either of the local models of the two clients who have data from the same distribution is located around the local minimum value, it will become more distant from the local minimum value by averaging the local models of two clients. To avoid such a situation, we make clients i and j learn $\bar{\mathbf{w}}_i$ and $\bar{\mathbf{w}}_j$ for one epoch toward another local minimum value for specifying a local model.

C. Synergy based on Cosine Similarity

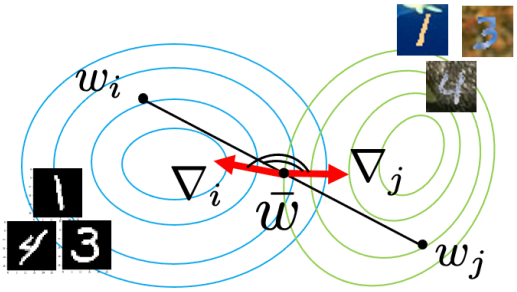

Fig. 3. Cosine Similarity

Algorithm 3 Option2:Cosine similarity (COS)

1: **for** each pair $(i, j) \in C, i < j$ **do**
2: $\quad \bar{\mathbf{w}}_{ij}^{(t)} = (\mathbf{w}_i^{(t)} + \mathbf{w}_j^{(t)})/2$
3: $\quad \nabla_i = L(\bar{\mathbf{w}}_{ij}^{(t)}, \mathcal{D}_i)$
4: $\quad \nabla_j = L(\bar{\mathbf{w}}_{ij}^{(t)}, \mathcal{D}_j)$
5: $\quad s_{ij} = \frac{\langle \nabla_i, \nabla_j \rangle}{\| \langle \nabla_i \| \| \nabla_j \|}$
6: **end for**
7: **return** s

The synergy based on cosine similarity between the gradients of the loss functions is shown in Figure 3 and defined as

$$s_{i,j} = \frac{\langle \nabla_i, \nabla_j \rangle}{\| \langle \nabla_i \| \| \nabla_j \|},$$

where we define the gradients of the loss functions as

$$\nabla_i = \nabla L\left(\bar{\mathbf{w}}_{ij}^{(t)}, \mathcal{D}_i\right).$$

For this synergy, we make the assumption that the minimized loss functions of the two clients are close to each other if they have data from the same data distribution. This indicates that COS tends to become 0 or negative if two clients have data from different distributions or one of the clients has adversarial data; thus, such clients are split into different coalitions.

Fig. 4. MT, SV, and MM datasets

V. Experimental Results

We conducted two types of experiments with and without adversary clients who have adversarial data. We first evaluated classification accuracy by varying the hyperparameters for FL when there exists no adversaries. We then evaluated whether CFFL can partition a set of clients into coalitions to exclude adversary clients.

We compared CFFL with the state-of-the-art clustered FL algorithm IFCA [6] and standard FL algorithm FedAvg [1] as well as baseline algorithms called local model and domain model.

IFCA: In this algorithm, each client estimates her cluster with the lowest loss function among the global models. The algorithm starts with n initial model parameters, which are randomly given, alternately estimates the cluster identities of the clients, and optimizes model parameters for the clusters via gradient descent.

FedAvg: Each client learns a single global model.

Local model: The model in each node executes gradient descent only on the available local data, and model averaging is not executed.

coalition model: This algorithm attempts to learn a single domain model from each distribution under the assumption that a server has information on which data distribution each client has. In other words, the server correctly classifies a set of clients into coalitions on the basis of different data distributions.

We assume 15 clients and 3 different data distributions. Here, each of the group of 5 clients has data from the same distribution. We created three different datasets that are based on MNIST [15] (MT dataset), SVHN [16] (SV dataset), and MNIST-M [17] (MM dataset) for the hand-written digit classification task shown in Figure 4. The MT and MM datasets have $60,000$ training images and $10,000$ test images with 10 classes. The SV dataset has $73,257$ training images and $26,032$ test images with 10 classes. In the prepossessing, the size of the image for each dataset was changed to 28×28 pixels and 3 channels. We assumed that the first five clients $\{1, \ldots, 5\}$ had data from the MT dataset, the next five clients $\{6, \ldots, 10\}$ had data from the SV dataset, and the remaining clients $\{11, \ldots, 15\}$ had data from the MM dataset.

Regarding the learning method for each client, we designed a simple convolutional neural network that takes three channels and 28×28 pixel images and passes them through two

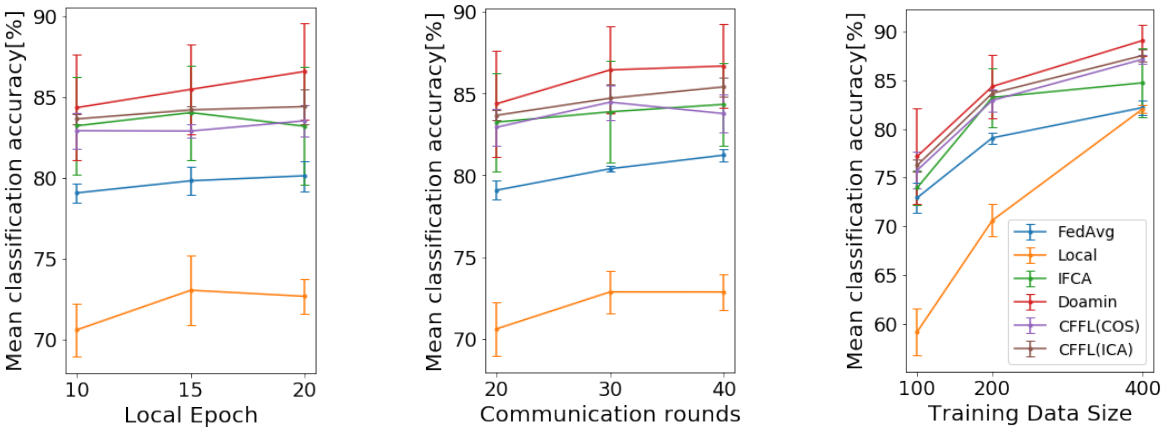

Fig. 5. Mean classification accuracy and standard deviation when hyperparameters are varied

TABLE I
MEAN CLASSIFICATION ACCURACY FOR EACH DATA DISTRIBUTION. DOM, ICA, COS, FA AND LOC RESPECTIVELY DENOTE DOMAIN MODEL, CFFL WITH ICA, CFFL WITH COS, FEDAVG, AND LOCAL DOMAIN.

Algorithm	MT	SV	MM
200-image training data			
Dom (84.3)	96.6 ± 0.59	74.3 ± 3.75	82.0 ± 3.14
ICA (**83.6**)	$\mathbf{96.6 \pm 1.61}$	74.0 ± 2.34	80.2 ± 3.21
COS (82.9)	96.3 ± 2.27	$\mathbf{74.4 \pm 1.61}$	77.9 ± 4.99
IFCA (83.2)	95.2 ± 0.86	71.1 ± 3.55	$\mathbf{83.2 \pm 2.87}$
FA (79.0)	95.8 ± 0.59	64.6 ± 5.37	65.1 ± 3.83
Loc (70.9)	91.4 ± 2.32	55.2 ± 1.53	81.0 ± 5.73
400-image training data			
Dom (89.0)	96.2 ± 1.40	80.7 ± 1.04	90.2 ± 0.65
ICA (**87.5**)	95.4 ± 1.25	77.9 ± 0.98	$\mathbf{89.1 \pm 1.10}$
COS (87.1)	95.1 ± 1.74	$\mathbf{78.7 \pm 2.22}$	87.5 ± 1.72
IFCA (84.7)	$\mathbf{95.8 \pm 1.12}$	74.2 ± 1.03	84.0 ± 1.52
FA (82.1)	95.1 ± 1.30	68.1 ± 1.15	83.2 ± 1.70
Loc (82.0)	91.6 ± 1.32	73.5 ± 1.98	81.0 ± 0.93

5×5 convolutional layers (10 and 20 channels) with rectified linear unit (ReLU) activation. Each convolutional layer is followed by a 2×2 max pooling layer. Finally, the network passes data through two fully connected dense layers (320 and 50 channels) with ReLU activation and provides output via Softmax classification over the 10 possible digit labels. We used stochastic gradient descent as the optimization algorithm in this study and set the learning rate to 0.01. We conducted experiments with 5 different random seeds (initial weight of a neural network).

We used a mixed integer programming package, Gurobi version 9.03, to solve RP$^*(G_t)$+pp in Algorithm 1.

A. Experimental results when no adversarial clients exist

We evaluated the classification accuracy for 15 clients by varying the hyperparameters for FL. As a baseline setting, we set the local-epoch size and number of communication rounds (iterations) to 10 and 20, respectively. We also assumed that each client had 250 images randomly chosen from a dataset. The data size was identical for every client, the same as in Ghosh et al.'s study [6]. The set of distributed data is divided into training data with 200 (80%) images and test data with 50 (20%) images. The ratio of the training-data size and the test-data size was fixed even for a different data size. When one parameter was varied, we used baseline parameters for the other two. Each client trained her local model by varying three parameters: local-epoch size was selected from $\{10, 15, 20\}$, number of communication rounds was selected from $\{20, 30, 40\}$, and training-data size was selected from $\{100, 200, 400\}$.

Figure 5 shows the mean classification accuracy and standard deviation over the set of clients at each parameter setting. The results indicate that we should partition the set of clients into coalitions on the basis of data distribution because the domain model performs the best among all algorithms. Note that the domain model is gold standard. FedAvg and the local model, which learns a single global model for all data distributions, perform poorly. Aside from the domain model, CFFL with ICA outperforms the other algorithms. When CFFL with ICA is applied, the clients who have data from the same distribution gather in a single coalition or are partitioned into several coalitions. Even if they are partitioned into several coalitions, those coalitions consist of the clients with the same data distribution. Thus, a client learns a coalition model in

TABLE II
COMPARISON AMONG ALGORITHMS WHEN ADVERSARIES EXISTS

Algorithm	No adversary	Targeted	Untargeted
Dom	84.3 ± 3.25	84.3 ± 3.25	84.3 ± 3.25
ICA	$\mathbf{83.6 \pm 0.32}$	82.5 ± 0.59	77.7 ± 1.05
COS	82.9 ± 1.10	$\mathbf{82.9 \pm 0.62}$	$\mathbf{82.3 \pm 0.72}$
IFCA	83.2 ± 3.00	81.9 ± 2.90	79.4 ± 3.49
FA	$79. \pm 0.57$	78.1 ± 1.28	77.3 ± 1.48
Loc	70.9 ± 2.57	70.9 ± 2.57	70.9 ± 2.57

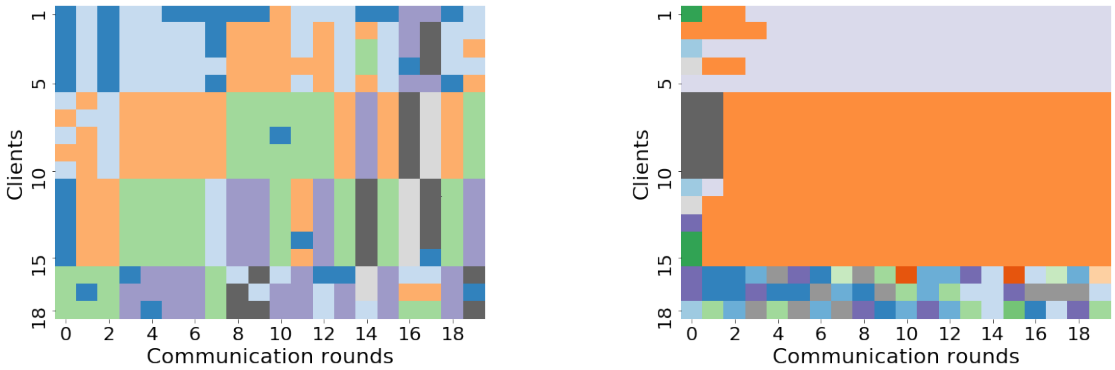

Fig. 6. Change of coalition structure for CFFL (COS) and IFCA

a coalition of clients with the same data distribution as her data distribution, and the performance of her local model is improved. On the other hand, CFFL with COS does not always outperform IFCA when the local epoch size or the number of communication rounds is varied, since the gradient direction is not stable when the data size is small. CFFL with ICA is stable against the changes in data size, and both CFFL with ICA and CFFL with COS perform better than IFCA.

Table I shows the mean classification accuracy for a subset of clients who have data from the same data distribution when a client has 200 and 400 images in the training data, respectively. The value next to the name of each algorithm indicates the mean classification accuracy for all clients. The performances for SV and MM are low depending on the complexity of an image feature. Compared with MT, SV and MM consist of more complicated background and RGB color model as shown in Fig. 4. For the MM dataset, CFFL with ICA performs well. While CFFL with ICA performs worse than IFCA for the 400-image training data, the difference is extremely small (0.4%). CFFL with COS generates the best coalition model for SV. For the MM dataset, while CFFL algorithms perform worse than IFCA for 200-image training data, CFFL with ICA performs the best for the 400-image training data.

B. Experimental results when adversaries exist

In this experiment, we evaluate whether CFFL can exclude an adversary who manipulates the training data to degrade performance of a global model. We consider two types of adversarial attacks: untargeted attacks and targeted attacks [2]. Untargeted attacks (model downgrade attacks) aim to reduce global accuracy or fully break the global model. Targeted attacks (backdoor attacks) aim to alter the behavior of the model on a minority of examples while maintaining good overall accuracy on all other examples.

In our setting, adversaries use incorrect labels as training data for untargeted attacks and adopt a label-flipping attack, in which labels of 1 and 7 are reversed, for targeted attacks. While adversaries manipulate the training data, they do not manipulate the test data. Thus, adversaries can degrade the performance of the global model while maintaining good classification accuracy for the evaluation by communicating with a server.

We add three adversaries, i.e., $\{16, 17, 18\}$, to the original 15 clients. The adversaries have an SV dataset and manipulate training data. The local epoch size is set to 10, the number of communication rounds (iterations) is set to 20, and the training data size of each client/adversary is set to 200.

Table II shows the mean classification accuracy and the standard deviation for three problem settings. As shown in the previous subsection, CFFL with ICA performs the best among the other algorithms except for the domain model when no adversary exists. When adversaries exist, CFFL with COS performs the best in both targeted and untargeted attacks. Cosine similarity works well for removing adversarial data.

Figure 6 shows how a coalition structure is generated by applying CFFL with COS and IFCA. The x-axis indicates the communication rounds and y-axis indicates the clients. In the initial state, all clients, including adversaries, have a single global model, thus join the same coalition. In the graphs, the result at the 0-th round is the result after conducting the first iteration. CFFL with COS sometimes generates a coalition

that consists of non-adversaries and adversaries but finally generates coalitions on the basis of the same data distribution and excludes adversaries. On the other hand, while IFCA splits non-adversaries and adversaries from the beginning, IFCA cannot partition non-adversaries having the SV and MM data distributions. Thus, the performance of IFCA is worse than CFFL with COS.

VI. Conclusions

We formulated FL as graphical coalitional games in which the value of a coalition is computed by the sum of synergies between any two clients who belong to the coalition. Formulating FL as a coalitional game enabled us to generate the optimal CS that maximizes the sum of synergies in a coalition. We applied two types of synergies to determine the value of a coalition: one based on the ICA between two clients and the other based on the COS between the gradients of the loss functions. Our algorithm called CFFL gradually improves a specialized model for every coalition (coalition model) by changing the CS along with the update of synergies. Experimental results indicate that CFFL with ICA performed well when adversaries did not exist, and CFFL with COS excluded adversaries and partitioned a set of non-adversaries into coalitions on the basis of data distributions. Our future work will include generalizing CFFL so that a client can join multiple coalitions.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS-2017)*, vol. 54, 2017, pp. 1273–1282.

[2] P. Kairouz, H. B. McMahan, and Contributors, "Advances and open problems in federated learning," *Foundations and Trends in Machine Learning*, vol. 14, no. 1, 2021.

[3] C. T. Dinh, N. H. Tran, and T. D. Nguyen, "Personalized federated learning with moreau envelopes," in *Advances in Neural Information Processing Systems 33, (NeurIPS-2020)*, 2020.

[4] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning: A meta-learning approach," in *Advances in Neural Information Processing Systems 33, (NeurIPS-2020)*, 2020.

[5] C. Briggs, Z. Fan, and P. Andras, "Federated learning with hierarchical clustering of local updates to improve training on non-iid data," in *Proceedings of 2020 International Joint Conference on Neural Networks (IJCNN-2020)*, 2020, pp. 1–9.

[6] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," in *Advances in Neural Information Processing Systems 33, (NeurIPS-2020)*, 2020.

[7] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–13, 2020.

[8] T. Rahwan, T. P. Michalak, M. Wooldridge, and N. R. Jennings, "Coalition Structure Generation: a Survey," *Artificial Intelligence*, vol. 229, pp. 139–174, 2015.

[9] A. Ghorbani and J. Zou, "Data shapley: Equitable valuation of data for machine learning," in *Proceeding of the 36th International Conference on Machine Learning (ICML-2019)*, 2019, pp. 2242–2251.

[10] A. Miyauchi, T. Sonobe, and N. Sukegawa, "Exact clustering via integer programming and maximum satisfiability," in *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI-18)*, 2018, pp. 1387–1394.

[11] Y. Huang, L. Chu, Z. Zhou, L. Wang, J. Liu, J. Pei, and Y. Zhang, "Personalized cross-silo federated learning on non-iid data," in *Proceeding of the 35th AAAI Conference on Artificial Intelligence (AAAI-2021)*, 2021, pp. 7865–7873.

[12] K. Muhammad, Q. Wang, D. OReilly-Morgan, E. Tragos, B. Smyth, N. Hurley, J. Geraci, and A. Lawlor, "Fedfast: Going beyond average for faster training of federated recommender systems," in *Proceeding of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD-2020)*, 2020, pp. 1234–1242.

[13] R. B. Myerson, "Graphs and cooperation in games," *Mathematics of Operations Research*, vol. 2, no. 3, pp. 225–229, 1977.

[14] M. Grötschel and Y. Wakabayashi, "A cutting plane algorithm for a clustering problem," *Mathematical Programming*, vol. 45, no. 1-3, pp. 59–96, 1989.

[15] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, vol. 86, no. 11, 1998, pp. 2278–2324.

[16] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[17] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Proceedings of the 32nd International Conference on Machine Learning (ICML-2015)*, vol. 37, 2015, pp. 1180–1189.