Incentive-Aware Decentralized Data Collaboration

YATONG WANG*, University of Electronic Science and Technology of China, China
YUNCHENG WU†, National University of Singapore, Singapore
XINCHENG CHEN, National University of Singapore, Singapore
GANG FENG, University of Electronic Science and Technology of China, China
BENG CHIN OOI, National University of Singapore, Singapore

Data collaboration enables multiple parties to pool data for deriving meaningful data insights. However, data misuse and unlawful data collection have led to precautionary measures being imposed by individual organizations to guide against data leakage and abuse. As a response, decentralized federated learning (DFL) has emerged as an attractive paradigm to facilitate data collaboration while being amenable to privacy-preserving data and knowledge sharing, cost reduction, and prediction accuracy improvement. Unfortunately, the participating parties in DFL tend to be heterogeneous with skew datasets and uneven capabilities. Inevitably, training and transmission costs, and the presence of 'free-riders' pose challenges to the adoption and participation of DFL. The absence of centralized parameter servers further exacerbates the problem of evaluating the contribution of each individual party. Therefore, an effective incentive mechanism is essential to promote data collaboration.

In this paper, we propose a novel Incentive-aware Decentralized fEderated leArning (IDEA) framework for facilitating data collaboration. Specifically, we first design a customizable reward scheme for heterogeneous parties to optimize their respective objectives such as higher model accuracy, communication efficiency, and computational efficiency. To reward fairly to deserving parties while offering flexibility, we propose a novel multi-agent reinforcement learning (MARL) incentive mechanism, which enables heterogeneous parties to learn their own optimal collaboration policy. We then design an efficient decentralized data collaboration algorithm that supports the customizable reward scheme based on individual objective-specific collaboration policy. We theoretically prove that the algorithm achieves a Nash equilibrium, which ensures the fairness of the corresponding rewards for parties. We conduct extensive experiments to evaluate the performance of our proposed framework against four baselines on five real-world datasets. The results show that IDEA outperforms state-of-the-art methods in terms of effectiveness, efficiency, and accumulated reward.

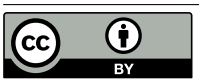
CCS Concepts: • **Computing methodologies** → **Distributed computing methodologies**; *Machine learning*; • **Information systems** → Information systems applications.

Additional Key Words and Phrases: data collaboration, decentralized learning, incentive mechanism

ACM Reference Format:
Yatong Wang, Yuncheng Wu, Xincheng Chen, Gang Feng, and Beng Chin Ooi. 2023. Incentive-Aware Decentralized Data Collaboration. *Proc. ACM Manag. Data* 1, 2, Article 158 (June 2023), 27 pages. https://doi.org/10.1145/3589303

*This work was done when this author was a visiting student at National University of Singapore.
†Yuncheng Wu is the corresponding author.

Authors' addresses: Yatong Wang, University of Electronic Science and Technology of China, China, wangyatong@std.uestc.edu.cn; Yuncheng Wu, National University of Singapore, Singapore, wuyc@comp.nus.edu.sg; Xincheng Chen, National University of Singapore, Singapore, e0838447@u.nus.edu; Gang Feng, University of Electronic Science and Technology of China, China, fenggang@uestc.edu.cn; Beng Chin Ooi, National University of Singapore, Singapore, ooibc@comp.nus.edu.sg.



2836-6573/2023/6-ART158
https://doi.org/10.1145/3589303

1 INTRODUCTION

Digitization and digitalization are being increasingly adopted by companies and organizations to transform themselves for either improved profitability, efficiency or sustainability [39]. Data is a key to the success of such transformation. For example, risk analysts may have to identify fraudulent users using the bank databases [72], and clinicians may use large collections of electronic medical records to facilitate the diagnosis of diseases in order to improve patient care [44, 71]. However, some organizations may not have sufficiently large and/or useful datasets, and are therefore keen to share data for analytics purposes. Such practice is prevalent in certain industries such as healthcare and emerging digital banking, in which a party[1] may share data with both vertically [33, 58] or horizontally [3] relevant parties. Nevertheless, there has been a parallel development in data protection, and various protection acts have been subsequently legislated [1, 53]. This deters any direct exchange of raw data.

To facilitate data collaboration without violating data privacy protection acts [25], decentralized federated learning (DFL) [12, 13, 31, 38, 70, 73] has been emerging as a conducive paradigm for distributed knowledge sharing, accurate statistical models, and cost reduction. Specifically, multiple parties in DFL can collaboratively learn an effective machine learning (ML) model by using their local datasets and exchanging model parameters with their neighboring parties. This paradigm can circumvent exposing raw data to privacy risks. However, parties may be reluctant to participate in data collaboration due to the lack of effective incentive mechanisms, even though it can improve the model performance compared to training individually. The reason is two-fold. First, participation in collaborative learning incurs training and transmission costs, and consequently, free-riders [11] may intentionally obtain the other parties' model parameters without doing their part. Second, a malicious party may intentionally provide useless or harmful model parameters to degrade the model performance or even cause the failure of convergence [14, 68]. Understandably, these behaviors may discourage honest participants from diligently providing high-quality data, and consequently, defeat the whole purpose of data collaboration. An effective incentive mechanism is therefore of vital importance to data collaboration, where all parties are incentivized to contribute to the DFL process, with the goal of achieving improved accuracy, higher efficiency, and lower cost.

However, there are two key challenges in designing an effective incentive-aware DFL algorithm for supporting real-world data collaboration. The first challenge stems from the heterogeneity of participating parties in data collaboration. In particular, the datasets of parties are typically skewed [26], i.e., with unbalanced qualities and quantities, or with non-independently and identically distribution (Non-IID). Moreover, parties typically equip with uneven capabilities such as computational and communication capacities. These heterogeneities of parties make the design space of the decentralized collaboration strategy more complex as an individual party may prefer a personalized strategy that is optimal from its own perspective. The second challenge is the difficulty of measuring the contribution of each party in a fully decentralized setting. We note that various incentive mechanisms have been proposed for data collaboration [8, 30, 48, 62, 64, 65]. However, these designs require the existence of a centralized server to collect the information of all the parties, and thus they are not applicable to DFL. Moreover, it is impractical to derive a closed-form expression of the contribution evaluation, e.g., Shapley value [48], in the DFL framework. For example, the Shapley value measures a party's marginal contribution by iterating over all the possible coalitions and calculating the difference in model performance when this party is in each coalition or not. The evaluation is intractable as it requires a huge number of re-training for the calculation.

[1]We shall use the term party to refer to an organization joining data collaboration.

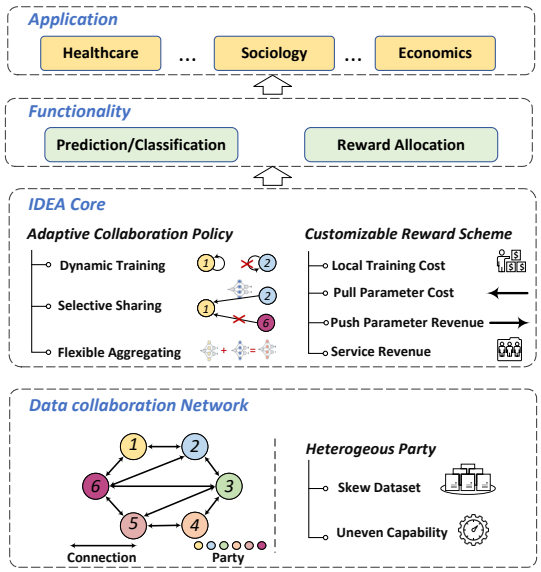


Fig. 1. Overview of IDEA data collaboration framework.

In this paper, we propose a novel Incentive-aware Decentralized fEderated leArning (IDEA) framework to address these challenges with the aim to facilitate data collaboration. To tackle the heterogeneity of participating parties, we present a customizable reward scheme, which enables each party to define its objectives in data collaboration flexibly. For example, parties with limited communication capacity may expect to pull other parties' parameters less frequently while achieving high model accuracy. Then, the parties can tailor the corresponding weights in our scheme for this objective. In particular, the conventional DFL setting [15, 26, 28, 30] can be deemed as a special case of our framework, where the parties solely focus on the model accuracy. To effectively incentivize all parties to contribute to the collaboration, we propose a novel incentive model to facilitate heterogeneous parties to learn their optimal collaboration policy. For example, the collaboration policy of a party could be used to determine whether to train its model locally and whether to pull model parameters from a specific neighbor at each training step. We formulate the policy learning problem as a stochastic game, where the objective of each party is to maximize its reward during the collaboration process. Consequently, we introduce a multi-agent reinforcement learning (MARL) method, where each party acts as an intelligent agent to determine its collaboration actions. Lastly, we propose an efficient decentralized data collaboration algorithm that enables individual parties to train their model and selectively exchange parameters in accordance with their objective-specific collaboration policies.

We present an overview of our proposed IDEA data collaboration framework for heterogeneous parties in Figure 1. Its incentive model supports customizable incentive functions and adaptive collaboration policy learning for each party. At the functionality layer, our framework supports prediction/classification and reward allocation, which can be applied to a wide spectrum of application areas, such as healthcare, sociology, and economics analytics.

In summary, we make the following contributions in this paper.

- We propose an incentive-aware DFL framework, which allows individual parties to customize their incentive functions and facilitates their participation and contribution to data collaboration.

To the best of our knowledge, we are the first to propose an incentive model to support fully decentralized data collaboration.
- We present a multi-agent reinforcement learning method to learn the optimal collaboration policy for individual parties. We address the inherent issues in the policy learning, such as high-dimensional action space and overestimation, with a mean-field representation method and dual neural network-based approximation method, respectively.
- We theoretically prove that the learned collaboration policy achieves the Nash equilibrium and provide a convergence analysis of the IDEA collaboration algorithm. Moreover, we describe several exemplary adversarial behaviors and discuss how IDEA can mitigate them.
- We conduct extensive experiments on five real-world datasets to evaluate the performance of our proposed framework. The results demonstrate that IDEA outperforms state-of-the-art baselines in terms of model accuracy, efficiency, and calculated reward.

The remaining of this paper is organized as follows. We introduce the preliminaries in Section 2. Section 3 presents IDEA framework. In Section 4, we elaborate on the design of the incentive mechanism. We introduce the IDEA-based DFL training algorithm in Section 5. The experimental evaluation is presented in Section 6. We review existing works in Section 7, followed by conclusions in Section 8.

2 PRELIMINARIES

2.1 Decentralized Federated Learning

Decentralized federated learning (DFL) enables multiple parties $\mathcal{V}$ to collaboratively learn a machine learning (ML) model by using their local datasets and exchanging model parameters with one-hop neighboring parties in an iterative manner. Let t denote the iteration step of conventional DFL, which is composed of local training, gossiping, and averaging.

- **Local training.** In the local training phase, each party i first samples a mini-batch $\{\xi_m^i\}_{m=1}^M$ from its local dataset $\mathcal{D}_i$, where M is the batch size. Then, party i calculates the parameter gradients:

$$\triangle \boldsymbol{\theta}_i(t) = \sum\nolimits_{m=1}^{M} \nabla f_i(\boldsymbol{\theta}_i; \xi_m^i) \qquad \forall i \in \mathcal{V}, \tag{1}$$

  where $\boldsymbol{\theta}_i$ and $f_i(\cdot)$ denote the parameters and loss function of party i's local ML model, respectively. Then, the party updates the local model by:

$$\boldsymbol{\theta}_i(t+\frac{1}{2}) = \boldsymbol{\theta}_i(t) - \beta \triangle \boldsymbol{\theta}_i(t) \qquad \forall i \in \mathcal{V}, \tag{2}$$

  where β is the learning rate.
- **Gossiping.** In the gossiping phase, each party i communicates with a subset $\widehat{\mathcal{N}}(i) \subset \mathcal{N}(i)$ of its neighbors $\mathcal{N}(i)$ to pull their model parameters $\boldsymbol{\theta}_j(t+\frac{1}{2})$, where $\forall j \in \widehat{\mathcal{N}}(i)$.
- **Averaging.** In the averaging phase, each party i updates its local model by a weighted average of the locally trained model and pulled models:

$$\boldsymbol{\theta}_i(t+1) = (1 - \sum_{j \in \widehat{\mathcal{N}}(i)} \rho_i)\boldsymbol{\theta}_i(t+\frac{1}{2}) + \sum_{j \in \widehat{\mathcal{N}}(i)} \rho_i \boldsymbol{\theta}_j(t+\frac{1}{2}) \quad \forall i \in \mathcal{V}, \tag{3}$$

  where ρ_i is the average rate for aggregating neighbors' models.

The DFL training process repeats until convergence or the maximum number of iteration steps is reached.

2.2 Multi-agent Reinforcement Learning

MARL has recently witnessed notable success in solving the Markov game in multiple fields, e.g., finance, communication networks, and social science [69]. The reinforcement learning agents learn concurrently by interacting with the environment via a trial and error approach, where they receive rewards for their actions. The feedback guides the agent to the improvement of its future policy. The environment/problem in the multi-agent system is usually modeled as a Markov game, also called a stochastic game. Specifically, a Markov game is defined by a tuple $\{\mathcal{V}, \mathcal{S}, \{\mathcal{A}_i\}_{i\in\mathcal{V}}, \mathcal{P}, \{R_i\}_{i\in\mathcal{V}}, \gamma\}$, where $\mathcal{V} = \{1, \cdots, i, \cdots, |\mathcal{V}|\}$ denotes the set of $|\mathcal{V}|$ agents (parties); $\mathcal{S}$ denotes the state space observed by all agents; $\mathcal{A}_i$ is the action space of agent i, and $\mathcal{A} := \mathcal{A}_1 \times \cdots \times \mathcal{A}_{|\mathcal{V}|}$ is the joint action space; $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ denotes the transition probability from any state $\boldsymbol{s} \in \mathcal{S}$ to any state $\boldsymbol{s'} \in \mathcal{S}$ by a joint action $\mathbf{a} \in \mathcal{A}$; $R_i : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the immediate reward received by agent i for executing the action; γ is the discount factor. To solve the Markov game, the goal of the agents in MARL is to find a Nash equilibrium joint policy $\boldsymbol{\pi}^* = (\pi_1^*, \cdots, \pi_{|\mathcal{V}|}^*)$, which is a mapping from the state space $\mathcal{S}$ to the distribution of the action space $\mathcal{A}$ for maximizing the long-term reward of individual parties: $\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R_i(\boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{s}_{t+1}) | \boldsymbol{a}_t \sim \boldsymbol{\pi}^*]$, where $\boldsymbol{s}_t$, $\boldsymbol{a}_t$, and $\boldsymbol{s}_{t+1}$ denote the current state, executed action, and next state at iteration step t.

3 IDEA FRAMEWORK

In this section, we present our proposed Incentive-aware Decentralized fEderAted learning (IDEA) framework. We start by denoting the collaborating parties in DFL as an undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$. Let $|\mathcal{V}|$ be the number of parties and $i \in \mathcal{V}$ be the i-th party. The set of one-hop neighbors of party i is denoted by $\mathcal{N}(i)$. The parties are heterogeneous as each party $i \in \mathcal{V}$ owns a specific local dataset $\mathcal{D}_i$, and let $|\mathcal{D}_i|$ denote its dataset size. Further, each party i is also equipped with uneven computational capability denoted by p_i^{cpt} and uneven communication capability denoted by p_i^{cmt}, where p_i^{cpt} and p_i^{cmt} are related to the data processing rate and transmission rate respectively. In this paper, we consider that the $|\mathcal{V}|$ parties have the same type of neural network (NN) model with parameters $\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_i, \cdots, \boldsymbol{\theta}_{|\mathcal{V}|}$, where $\boldsymbol{\theta}_i$ denotes party i's model parameter. Note that this is a practical setting of DFL in many real-world scenarios, such as Internet of things [27] and smart healthcare [36], as the parties typically need to handle different groups of customers (e.g., patients in different regions). Thus, each party prefers to learn a personalized model regarding its own private dataset.

To motivate the parties to undertake a learning task in a collaborative way, IDEA enables each party to customize its reward scheme, and based on which, a specific collaboration policy is designed for each party so as to achieve a high reward during the data collaboration process. In the following, we first introduce our adaptive collaboration policy in Section 3.1 and present the customizable reward scheme in Section 3.2.

3.1 Adaptive Collaboration Policy

For effective data collaboration, DFL should support an adaptive training policy to accommodate the objectives and constraints of each party. However, most existing DFL algorithms utilize fixed training strategies, which are not only inefficient for heterogeneous parties, but also insensitive to the incentives for individual parties. For example, in D-PSGD [28], each party randomly communicates with one of its neighbors at each iteration step, which may restrict the potential for parties with high communication capabilities. Likewise, in CDSGD [20], each party pulls the model parameters from all of its neighbors at each iteration step, which consumes huge communication resources. Thus, it may hinder some parties with fewer communication resources from participating in the collaborative training process. Moreover, most existing DFL works [16, 23, 25, 45, 47, 55, 73] require

each party to perform local model training at each iteration step, which may not be acceptable or feasible to some parties. The reason is two-fold. First, local model training incurs some computational costs, and the performance of the local model relies heavily on the quality/quantity of the local dataset. Therefore, the parties that lack computational capabilities and/or data, may prefer pulling model parameters from neighbors rather than local training. Second, as the number of local training iterations increases, the local model may converge to the local optimum and can hardly be improved significantly. Therefore, local model training is thus not cost-efficient.

Motivated by these observations, as opposed to fixed training, we propose an adaptive collaboration policy for each party in IDEA, which determines whether to conduct local model training and which neighbors' model parameters to be pulled from at each iteration step. Specifically, we define the collaboration policy of each party i as follows.

Definition 1. *Collaboration Policy. The stochastic collaboration policy of party i is denoted by π_i, which outputs a probability distribution over the specific collaboration action $\boldsymbol{a}_i(t)$:*

$$\boldsymbol{a}_i(t) \sim \pi_i = [a_{i,0}(t), a_{i,1}(t), \cdots, a_{i,j}(t), \cdots, a_{i,|\mathcal{N}(i)|}(t)], \tag{4}$$

where $a_{i,0}(t)$ is the local training indicator such that $a_{i,0}(t) = 1$ if party i trains the local model at time t and $a_{i,0}(t) = 0$ otherwise; and $a_{i,j}(t)$ is the pull indicator such that $a_{i,j}(t) = 1$ if the party i pulls the model parameters from its neighboring party $j \in \mathcal{N}(i)$ at iteration step t, and $a_{i,j}(t) = 0$ otherwise.

With this adaptive collaboration policy, each party can flexibly adjust its training strategy during the collaboration process to maximize its expected reward. In particular, in the DFL local training phase (see Section 2.1), each party i decides whether to train the local model according to $a_{i,0}(t)$ at iteration step t. In the gossiping phase, each party i only pulls the model parameters from neighboring party j if $a_{i,j}(t) = 1$. Accordingly, the model averaging calculation in the averaging phase is performed only on the pulled model parameters. We shall present the detailed training algorithm based on collaboration policy in Section 5.1.

3.2 Customizable Reward Scheme

To facilitate data collaboration and encourage parties to participate in the DFL training process, we propose a customizable reward scheme in the IDEA framework, which enables each party to define its own incentive function. Specifically, we design the incentive function with four items: local training cost, pull parameter cost, push parameter revenue, and service revenue. In the following, we elaborate on the four items, respectively.

Local training cost. When a party decides to train its model locally, i.e., $a_{i,0}(t) = 1$, it is inevitable to consume computational resource and power. Therefore, we define the local training cost as follows.

Definition 2. *Local training cost. The local training cost $c_i^l(t)$ of party i at iteration step t is defined as*

$$c_i^l(t) = a_{i,0}(t) \cdot c_i^{train} \cdot |\mathcal{D}_i| / p_i^{cpt} \quad \forall i \in \mathcal{V}, \tag{5}$$

where c_i^{train} denotes the unit cost of local model training.

Specifically, the local training cost of each party i is proportional to its dataset size $|\mathcal{D}_i|$ and inversely proportional to the computational capability p_i^{cpt} [65].

Pull parameter cost. To mitigate free-rider problem [11] and ensure fairness, party i should pay its neighboring party j for pulling its model parameters (i.e., $a_{i,j}(t) = 1$). Let $\mathbf{Pr} = [pr_i]_{i \in \mathcal{V}} \in \mathbb{R}^{|\mathcal{V}|}$ be the model parameter price vector of parties, and we define the model parameter price pr_i of

party i as:

$$pr_i(t) = c_i^{com} \frac{S}{p_i^{cmt}} + c_i^{acc} \log(1 + acc_i(t)), \qquad \forall i \in \mathcal{V} \tag{6}$$

where S is the size of model parameters, c_i^{com} is the unit communication cost, and c_i^{acc} is the priority metric of the accuracy $acc_i(t)$ of party i. To ensure the strictly concave relationship between accuracy and model parameter price, we introduce a commonly used $\log(\cdot)$ function in Eqn 6. Based on the price vector, each party i has a pull parameter cost for the parameters it requests from its neighboring parties, which is defined as follows.

Definition 3. *Pull parameter cost. The pull parameter cost $c_i^p(t)$ of party i at iteration step t is expressed as*

$$c_i^p(t) = \sum\nolimits_{j \in \mathcal{N}(i)} a_{i,j}(t) pr_j(t) \qquad \forall i \in \mathcal{V}. \tag{7}$$

Push parameter revenue. Conversely, party i may be also requested by its neighbor $j \in \mathcal{N}(i)$ to push its model parameters $\theta_i(t + \frac{1}{2})$ to j, i.e., $a_{j,i}(t) = 1$. Thus, party i can receive revenue from neighboring parties for the pushed model parameters, which is defined as follows.

Definition 4. *Push parameter revenue. The push parameter revenue $r_i^p(t)$ of party i at time t is expressed as*

$$r_i^p(t) = \sum\nolimits_{j \in \mathcal{N}(i)} a_{j,i}(t) pr_i(t) \qquad \forall i \in \mathcal{V}. \tag{8}$$

Service revenue. Another important objective of each party i is to improve its local model accuracy so that it is able to offer a higher plan service level agreement (SLA) to its customers. Succinctly, the higher the accuracy of its local model is, the higher the service revenue the party gains.

Definition 5. *Service revenue. The service revenue of party i at iteration step t is expressed as*

$$r_i^s(t) = \log(1 + acc_i(t)) \ \ \forall i \in \mathcal{V}, \tag{9}$$

where Eqn 9 is continuously increasing with the validation accuracy $acc_i(t)$ of party i and strictly concave.

Incentive function. In summary, the incentive function $R_i(t)$ of party i at iteration step t with regards to the above four items can be given by:

$$R_i(t) = \beta_s r_i^s(t) + \beta_p r_i^p(t) - \beta_c c_i^p(t) - \beta_l c_i^l(t) \qquad \forall i \in \mathcal{V}, \tag{10}$$

where the β_s, β_p, β_c, and β_l are the weights of corresponding items. A noteworthy aspect is that each participating party can customize its incentive function by adjusting the weights in Eqn 10. The goal of each party is to maximize its incentive function by learning an adaptive collaboration policy.

The customizable reward scheme enables the parties to define their incentive functions based on the needs for supporting their individual applications. We would like to highlight that our proposed IDEA is extensible by design in that objectives of most existing DFL approaches can be realized by adjusting some parameters in IDEA. For example, DFL approaches [16, 25, 45, 47] aim to improve the model accuracy of parties without considering costs, which is a common objective in the healthcare and digital banking applications. IDEA can adapt to such scenarios by setting the weights β_p, β_c, and β_l of Eqn 10 to 0. In this manner, the parties only need to focus on improving service revenue by seeking high accuracy. As a further illustration on the extensibility of IDEA, let us consider federated edge computing scenarios [35, 54] and communication systems [73], where communication efficiency is a key issue in DFL training. In this case, the weights β_p and β_l of Eqn 10 can just be set to 0 to adapt to the communication capability constraints while improving the model accuracy.

4 INCENTIVE MODEL DESIGN

In this section, we present our incentive model design for the IDEA framework. We first model the incentive mechanism in Section 4.1. We then propose a MARL approach in Section 4.2.

4.1 Incentive Mechanism Modeling

Motivated by game theoretical modeling [22] of multi-agent/party systems and the stochastic nature of collaboration policy (see Definition 1), the incentive mechanism in our model can be modeled as a Markov game $\{\mathcal{V}, \mathcal{S}, \{\mathcal{A}_i\}_{i\in\mathcal{V}}, \mathcal{P}, \{R_i\}_{i\in\mathcal{V}}, \gamma\}$, as described in Section 2.2. Specifically, in our model, each party $i \in \mathcal{V}$ acts as an intelligent agent learning an optimal collaboration policy. The state space $\mathcal{S}$ describes the whole system environment, which includes the parties' accuracy on their respective local models. Thus, the state of the parties at iteration step t can be expressed as

$$\boldsymbol{s}(t) = [acc_{i,c}(t-1)]_{i\in\mathcal{V},c\in C}, \tag{11}$$

where $acc_{i,c}(t-1)$ is the accuracy of party i on class c at iteration step $t-1$ and C is the set of classes for the dataset. Furthermore, the action $\boldsymbol{a}_i(t) \in \mathcal{A}_i$ of party i at iteration step t can be represented by Eqn 4 defined in the collaboration policy. After executing the actions, the state $\boldsymbol{s}(t)$ then transits to the next state $\boldsymbol{s}(t+1)$ with transition probability p, which satisfies the constraint:

$$\sum\nolimits_{\boldsymbol{s}(t+1)\in\mathcal{S}} p(\boldsymbol{s}(t+1)|\boldsymbol{s}(t), \boldsymbol{a}_1(t), \cdots, \boldsymbol{a}_{|\mathcal{V}|}(t)) = 1. \tag{12}$$

After all the parties take actions, each party i will receive an immediate reward, say $R_i(t)$, as a consequence of taking a specific action. We define the reward $R_i(t)$ as the incentive function in Eqn 10 customized by the party, which relates to the service revenue $r_i^s(t)$, push parameter revenue $r_i^p(t)$, pull parameter cost $c_i^p(t)$, and local training cost $c_i^l(t)$. Note that the parties may have different rewards due to their heterogeneity. This raises a question: what kind of incentive mechanism is optimal and fair? Nash equilibrium is a standard solution in game theory [10], where no party can improve its reward by unilaterally deviating from the collaboration policy, ensuring the fairness of the incentive mechanism. Therefore, we focus on the incentive mechanism in the context of Nash equilibrium. Our goal here is therefore to design an effective incentive mechanism by learning an optimal collaboration policy $\boldsymbol{\pi} = (\pi_1, \cdots, \pi_{|\mathcal{V}|})$ for each party to maximize its own long-term average reward with Nash equilibrium. The objective function of each party i under the incentive mechanism can be described as

$$\max U^i(\pi_i|\pi_{-i}) = \mathbb{E}_{\pi_i|\pi_{-i}}[\sum\nolimits_{t=0}^{\infty} \gamma^t R_i(t)], \tag{13}$$

where γ is a discounted factor, π_i denotes the stochastic collaboration policy of party i, $-i$ represents the indices of all parties in $\mathcal{V}$ except party i, and π_{-i} is the joint collaboration policy of all parties except party i. We define the value-function of party i as $V^i : \mathcal{S} \to \mathbb{R}$ under the joint collaboration policy as

$$V^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}) := \mathbb{E}[\sum\nolimits_{t>=0} \gamma^t R_i(\boldsymbol{s}(t), \boldsymbol{a}(t), \boldsymbol{s}(t+1))|\boldsymbol{a}_i(t) \sim \pi_i(\cdot|\boldsymbol{s}(t)), \boldsymbol{s}(0) = \boldsymbol{s}], \tag{14}$$

where value-function $V^i_{\pi_i,\pi_{-i}}(\boldsymbol{s})$ is the expected cumulative discounted reward; $\boldsymbol{a}(t) = [\boldsymbol{a}_1(t), \cdots, \boldsymbol{a}_{|\mathcal{V}|}(t)] \in \mathcal{A}$ is the joint action taken by all parties. Accordingly, the action-value function (Q-function) under the joint collaboration policy $\boldsymbol{\pi}$ of party i is expressed by

$$Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}, \boldsymbol{a}) := \mathbb{E}[\sum_{t>=0} \gamma^t R_i(\boldsymbol{s}(t), \boldsymbol{a}(t), \boldsymbol{s}(t+1))|\boldsymbol{a}_i(t) \sim \pi_i(\cdot|\boldsymbol{s}(t)), \boldsymbol{s}(0) = \boldsymbol{s}, \boldsymbol{a}(0) = \boldsymbol{a}]. \tag{15}$$

The optimal performance of each party is controlled by not only its own policy, but also the choices of all other parties of the Markov game. Therefore, the Nash equilibrium is defined as follows.

Definition 6. *A Nash equilibrium of the Markov game is a joint collaboration policy* $\boldsymbol{\pi}^* = (\pi_1^*, \cdots, \pi_{|\mathcal{V}|}^*)$*, such that for any* $\boldsymbol{s} \in \mathcal{S}$ *and* $i \in \mathcal{V}$*:* $V^i_{\pi_i^*, \pi_{-i}^*}(\boldsymbol{s}) \geq V^i_{\pi_i, \pi_{-i}^*}(\boldsymbol{s})\ \forall \pi_i$.

Nash equilibrium characterizes an equilibrium point $\boldsymbol{\pi}^*$, from which none of the parties has any reward to deviate. In other words, for any party $i \in \mathcal{V}$, the policy π_i^* is the best-response of π_{-i}^*. As a standard learning goal for MARL, Nash equilibrium always exists for finite-space infinite-horizon discounted Markov game.

4.2 Collaboration Policy Learning

There are two main solutions for solving the Markov game problem, namely game theory (GT) based methods [22] and multi-agent reinforcement learning based (MARL) methods [69]. Traditional GT-based methods such as Stackelberg game [40] typically require all parties to have perfect information, i.e., all parties must have full knowledge of the current environment, including the other parties' collaboration policies. However, the perfect information is not available, as each party's collaboration policy should be hidden from the other parties. Although extensive-form games [9] in computational GT allow for the explicit representation of imperfect information in the multi-agent system, they become inefficient when the action space is large because they mainly solve the problem by identifying a game tree [15]. Recently, MARL has been found great potential in solving Markov game problems and is recognized as the key to emerging technologies in decentralized frameworks. Therefore, we design a MARL-based solution in this paper.

To design an efficient and effective incentive mechanism for IDEA, we have to address the following two key challenges. First, the dimension of joint actions may be high, making the estimation of the Q-function of each party fairly complicated [60]. Second, each party learns the collaboration policy in each iteration step even when there are insufficient experience samples at the beginning of the training process. This will lead to an overestimation of the Q-function and degrade the quality of the learned collaboration policy. In the following, we introduce our incentive mechanism design that addresses the two challenges.

4.2.1 Mean Field Representation of Q-function. To reduce the dimension of joint actions in the standard Q-function in Eqn 15, we first factorize the standard Q-function $Q^i_{\pi_i, \pi_{-i}}(\boldsymbol{s}, \boldsymbol{a})$ of party i into the sum of pairwise local interactions [6], which is expressed by

$$Q^i_{\pi_i, \pi_{-i}}(\boldsymbol{s}, \boldsymbol{a}) = \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} Q^i_{\pi_i, \pi_{-i}}(\boldsymbol{s}, \boldsymbol{a}_i, \boldsymbol{a}_j), \tag{16}$$

where $Q^i_{\pi_i, \pi_{-i}}(\boldsymbol{s}, \boldsymbol{a}_i, \boldsymbol{a}_j)$ is the pairwise Q-function of the party i and its neighbor $j \in \mathcal{N}(i)$. Pairwise representation significantly reduces the complexity of interactions between agents and the dimension of joint action, while being proved to preserve global interactions implicitly [6]. However, factorized pairwise Q-functions still require interactions between every pair of parties, which limits the scalability of the data collaboration system. Therefore, we further approximate the Q-function by employing the mean field theory [5]. In particular, the dimension of joint actions in standard Q-function (i.e., Eqn 15) grows proportionally with respect to the number of parties. The main idea of the mean field representation is to study the interactions between one representative party and the population distribution, which greatly reduces the dimension of action space of the Q-function. Specifically, the dimension can be reduced from w.r.t. the number of parties to w.r.t. the number of neighbors by employing mean field theory. The Q-function after using the mean field theory is defined as follows.

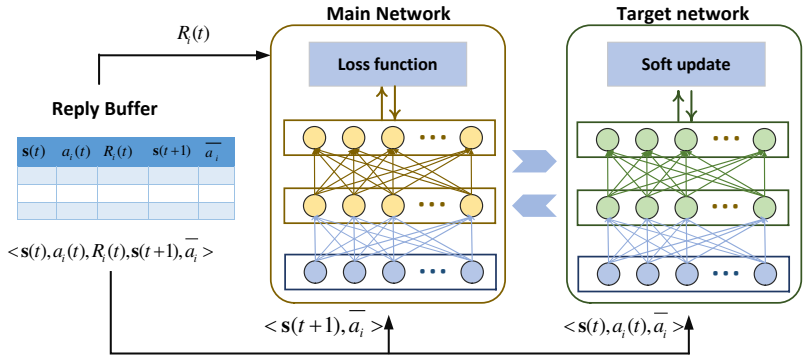


Fig. 2. The dual neural networks for individual agents.

Definition 7. *Mean field Q-function* $Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}, \boldsymbol{a}_i, \bar{\boldsymbol{a}}_i)$ *[60] is an approximation of the standard Q-function* $Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}, \boldsymbol{a})$ *with respect to the mean action of party i's neighbors* $\bar{\boldsymbol{a}}_i = \frac{1}{|\mathcal{N}(i)|}\sum_{j\in\mathcal{N}(i)} \hat{\boldsymbol{a}}_j$*, where* $\hat{\boldsymbol{a}}_j$ *is the one-hot action* $\boldsymbol{a}_j$ *of party j.*

Consequently, the standard Q-function $Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}, \boldsymbol{a})$ can be represented by the mean field Q-function with a bounded remainder $b \in [-2M, 2M]$, when pairwise Q-functions are M-smooth:

$$Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}, \boldsymbol{a}) = \frac{1}{|\mathcal{N}(i)|}\sum_{j\in\mathcal{N}(i)} Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}, \boldsymbol{a}_i, \boldsymbol{a}_j) = Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}, \boldsymbol{a}_i, \bar{\boldsymbol{a}}_i) + b \approx Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}, \boldsymbol{a}_i, \bar{\boldsymbol{a}}_i). \tag{17}$$

In practice, the remainder b acts as a small fluctuation near zero. Therefore, with the representation of standard Q-function, the dimension of joint actions and interactions between parties can be significantly reduced. According to Bellman's equation [50], the mean field Q-function can be expressed as

$$\begin{aligned} Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}(t), \boldsymbol{a}_i(t), \bar{\boldsymbol{a}}_i(t)) = (1-\gamma) Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}(t), \boldsymbol{a}_i(t), \bar{\boldsymbol{a}}_i(t)) + \\ \gamma[R_i(\boldsymbol{s}(t), \boldsymbol{a}(t), \boldsymbol{s}(t+1)) + \bar{V}^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}(t+1))], \end{aligned} \tag{18}$$

where the mean field value function $\bar{V}^i_{\pi_i,\pi_{-i}}(s(t+1))$ of party i is:

$$\bar{V}^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}(t+1)) = \sum_{\boldsymbol{a}_i} \pi_i(\boldsymbol{a}_i|\boldsymbol{s}(t+1), \bar{\boldsymbol{a}}_i) * \mathbb{E}_{\pi_i,\pi_{-i}}[Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}(t+1), \boldsymbol{a}_i(t+1), \bar{\boldsymbol{a}}_i(t+1))] \tag{19}$$

To balance the exploitation and exploration, we employ the widely used Boltzmann exploration policy [60] for each agent i:

$$\pi(\boldsymbol{a}_i|\boldsymbol{s}, \bar{\boldsymbol{a}}_i) = \frac{\exp(\xi Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}, \boldsymbol{a}_i, \bar{\boldsymbol{a}}_i))}{\sum_{\boldsymbol{a}'_i\in\mathcal{A}_i} \exp(\xi Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}, \boldsymbol{a}'_i, \bar{\boldsymbol{a}}_i)}, \tag{20}$$

where ξ is the temperature of the Boltzmann exploration policy.

4.2.2 Dual Neural Networks based Approximation. In conventional Q-learning methods, a Q-table should be constructed to store all possible mean-field Q-function $Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}(t), \boldsymbol{a}_i(t), \bar{\boldsymbol{a}}_i(t))$ according to Eqn 18. However, constructing Q-table becomes inefficient due to the high-dimensional action space. Therefore, we adopt multi-layers neural networks to approximate the mean field Q-function. Specifically, the update rule in Eqn 18 can be reformulated as parameterized adjustment. In the multi-agent reinforcement learning incentive mechanism design, we tackle the overestimation issue by employing the dual neural networks method to approximate the mean field Q-function for each

agent i. As illustrated in Figure 2, we introduce the parameterized main network $Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}, \boldsymbol{a}_i, \bar{\boldsymbol{a}}_i; \boldsymbol{\omega}_i)$ and parameterized target network $Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}, \boldsymbol{a}_i, \bar{\boldsymbol{a}}_i; \boldsymbol{\omega}_i^-)$ in our incentive mechanism, where $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_i^-$ are the parameters of main network and target network, respectively. The idea of the dual neural network is to mitigate overestimation by decomposing the action selection and action evaluation operations into two interactive networks. In addition, we introduce a reply buffer in the dual neural networks to store the experiences $\langle \boldsymbol{s}(t), \boldsymbol{a}_i(t), R_i(t), \boldsymbol{s}(t+1), \bar{\boldsymbol{a}}_i(t)\rangle$, which are used to network training. Specifically, agent i samples a mini-batch $\{\varsigma_k^i\}_{k=1}^K$ of experiments from its replay buffer B, where K is the batch size. Therefore, the loss function of party i for training the main network is:

$$\mathcal{L}_i(\boldsymbol{\omega}_i) = \frac{1}{K}\sum\nolimits_{\varsigma_k^i}[Y_i(t) - Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}(t), \boldsymbol{a}_i(t), \bar{\boldsymbol{a}}_i(t); \boldsymbol{\omega}_i]^2, \tag{21}$$

where $Y_i(t) = R_i(\boldsymbol{s}(t), \boldsymbol{a}(t), \boldsymbol{s}(t+1)) + \bar{V}^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}(t+1); \boldsymbol{\omega}_i^-)$ is the target mean field value of the target network. Then, the parameter $\boldsymbol{\omega}_i$ of main network can be updated by:

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_i - \xi_t \nabla \mathcal{L}_i(\boldsymbol{\omega}_i), \tag{22}$$

where ξ_t is the learning rate of the main network. The target network parameters $\boldsymbol{\omega}_i^-$ can be soft updated periodically with the parameter of main network $\boldsymbol{\omega}_i$:

$$\boldsymbol{\omega}_i^- = \tau\boldsymbol{\omega}_i + (1-\tau)\boldsymbol{\omega}_i^-, \tag{23}$$

where τ is the soft update factor of the target network. With two independent mean field Q-function estimation networks, IDEA is able to achieve unbiased mean field Q-function estimation by selecting actions with the opposite network, which helps to mitigate the overestimation problem. With the effective dual neural networks based approximation of mean field Q-function, the objective-specific collaboration policy can be calculated by using Eqn 20.

4.2.3 Complexity Analysis. The complexity of the collaboration policy learning module in each iteration is $O(|\mathcal{A}_i| + Kmn_w)$. The first term is the complexity of collaboration policy generation in Eqn 20, where $|\mathcal{A}_i|$ is the number of actions of party i. The second term denotes the complexity of dual neural networks training process, where m, n_w are the number of features and the number of parameters of dual neural networks respectively. We will also experimentally analyze the overhead of this module in Section 6.

5 ALGORITHM AND ANALYSIS

In this section, we first present the IDEA collaboration algorithm, and prove its fairness guarantee and convergence in Section 5.1, Section 5.2, and Section 5.3, respectively. We then discuss the IDEA collaboration algorithm in terms of adversarial behaviors, limitations, and extensibility in Sections 5.4 and 5.5.

5.1 IDEA Collaboration Algorithm

Given the incentive mechanism with the adaptive collaboration policy, we propose our IDEA collaboration algorithm. Algorithm 1 describes the iteration steps on each party $i \in \mathcal{V}$. At the beginning, the party initializes the DFL model parameters $\boldsymbol{\theta}_i$, the parameters of dual neural networks $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_i^-$, the mean action $\bar{\boldsymbol{a}}_i$, and the reply buffer B (line 1). For each iteration step t, the party first observes the current environment by Eqn 11 (line 3), then calculates the collaboration policy according to Eqn 20 (line 4).

Based on the collaboration policy, the party executes the local training phase (lines 5-7). Note that different from existing DFL algorithms where each party performs local training at each iteration

step, party i decides to train the local model only when $a_{i,0}(t) = 1$ in the collaboration policy. Thus, the update formula in the local training phase is:

$$\boldsymbol{\theta}_i(t+\frac{1}{2}) = \boldsymbol{\theta}_i(t) - a_{i,0}(t)\beta\triangle\boldsymbol{\theta}_i(t) \tag{24}$$

where $\triangle\boldsymbol{\theta}_i(t)$ is the gradient vector calculated by Eqn 1 and β is the learning rate. Next, each party i executes the gossiping phase (lines 8-11) to pull the model $\boldsymbol{\theta}_j(t+\frac{1}{2})$ from its neighbor $j \in \mathcal{N}(i)$ if $a_{i,j}(t) = 1$ in the collaboration policy. We assume that the neighbors are pre-determined for each party. Similarly, it also pushes its model $\boldsymbol{\theta}_i(t+\frac{1}{2})$ to the neighbor $j \in \mathcal{N}(i)$ if $a_{j,i}(t) = 1$. After that, the party aggregates the pulled model parameters according to the following formula in the averaging phase (line 12).

$$\boldsymbol{\theta}_i(t+1) = (1 - \sum_{j\in\widehat{\mathcal{N}}(i)} a_{i,j}(t)\rho_i)\boldsymbol{\theta}_i(t+\frac{1}{2}) + \sum_{j\in\mathcal{N}(i)} a_{i,j}(t)\rho_i\boldsymbol{\theta}_j(t+\frac{1}{2}), \tag{25}$$

where ρ_i is the weight for aggregating the pulled model parameters.

In the following, the party calculates the reward according to Eqn 10 (lines 13-14). Then, the party observes the state of the next step, computes the mean action based on Eqn 7, and stores the experience data in the reply buffer (lines 15-17). Finally, the party trains the dual neural networks in Section 4.2.2 (lines 18-23). The parties then synchronize their states and actions with their neighboring parties. After that, the parties enter the next iteration step and repeat these executions until convergence or the maximum number of iteration steps T is reached.

5.2 Fairness Guarantee

Next, we theoretically prove the fairness of the proposed incentive mechanism in Section 4 and used in Algorithm 1.

Assumption 1. *To analyze the fairness of the incentive mechanism, we make the following commonly used assumptions[60]:*

(1) The learning rate ξ_t in Eqn 22 of agents decreases over time such that $\sum_{t=0}^{\infty}\xi_t = \infty$ and $\sum_{t=0}^{\infty}(\xi_t)^2 < \infty$.
(2) Each action of agents can be visited infinitely often while can be chosen with a non-zero probability.
(3) The collaboration policy $\boldsymbol{\pi}$ of parities is greedy in the limit with infinite exploration, and the set of possible states is finite.
(4) In the learning process, the Nash equilibrium $\boldsymbol{\pi^*} = (\pi_1^*, \cdots, \pi_{|\mathcal{V}|}^*)$ is recognized either as 1) the global optimum or 2) a saddle point expressed as:1. $\mathbb{E}_{\boldsymbol{\pi}^*}[V^i_{\boldsymbol{\pi}^*}(\boldsymbol{s})] \geq \mathbb{E}_{\pi_i}[V^i_{\boldsymbol{\pi}}(\boldsymbol{s})], \forall\boldsymbol{\pi}$; 2. $\mathbb{E}_{\boldsymbol{\pi}^*}[V^i_{\boldsymbol{\pi}^*}(\boldsymbol{s})] \geq \mathbb{E}_{\pi_i}\mathbb{E}_{\pi^*_{-i}}[V^i_{\pi_i,\pi^*_{-i}}(\boldsymbol{s})], \forall\boldsymbol{\pi_i}$ and $\mathbb{E}_{\boldsymbol{\pi}^*}[V^i_{\boldsymbol{\pi}^*}(\boldsymbol{s})] \leq \mathbb{E}_{\pi_i^*}\mathbb{E}_{\pi_{-i}}[V^i], \forall\boldsymbol{\pi_{-i}}$.

Lemma 1. *In a Markov stochastic game, if Assumption 1 stands, the Q-value of agents computed by the update rule in Eqn 18 converges to the Nash Q-value $\boldsymbol{Q}_* = [Q_*^1, \cdots, Q_*^{|\mathcal{V}|}]$.*

Proof. Please see Theorem 1 in [60] for detailed derivation. We include it here to stay self-contained. □

Theorem 1. *Given any collaborative learning task formulated as Eqn 13, the proposed IDEA algorithm can achieve a Nash equilibrium of the Markov game while guaranteeing the fairness of the incentive mechanism under Assumption 1.*

Algorithm 1 IDEA collaboration (Party i)

Require: $\mathcal{G}$, $\mathcal{D}_i$, p_i^{cpt}, p_i^{cmt}, T, β, ρ, γ, ξ, τ, K.
Ensure: $\{R_i(1), \cdots, R_i(T)\}$, π_i^*, and $\boldsymbol{\theta_i}$

```
Initialize θ_i , ω_i, ω_i^-, ā_i, and B
for t = 1, 2, ··· , T do
    s(t) ← Observe the current state by Eqn 11
    a_i(t) ← Get the collaboration policy according to Eqn 20
    if a_{i,0}(t) = 1 then          /*Local Training*/
        θ_i(t + 1/2) ← Perform local model training by Eqn 24
    end if
    for j ∈ N(i) do                 /*Gossiping*/
        θ_j(t + 1/2) ← Pull the model from neighbor j if a_{i,j}(t) = 1
        Push θ_i(t + 1/2) to neighbor j if a_{j,i}(t) = 1
    end for
    θ_i(t + 1) ← Average the models by Eqn 25   /*Averaging*/
    c_i^l, c_i^p, r_i^p, r_i^s ← Calculate the items by Eqns 5, 7, 8, 9
    R_i(t) ← Calculate the reward by Eqn 10
    s(t + 1) ← Observe the next state
    ā_i(t) ← Compute the mean action by Definition 7
                /* Store Experiences in Buffer*/
    B ← B ∪ ⟨s(t), a_i(t), R_i(t), s(t + 1), ā_i(t)⟩
    if t ≥ K then                   /*Dual Neural Network Training*/
        Sample K experiences from the reply buffer B
        Train Q^i_{π_i,π_{-i}}(s, a_i, ā_i; ω_i) by minimizing Eqn 21
        Train Q^i_{π_i,π_{-i}}(s, a_i, ā_i; ω_i^-) by Eqn 23
    end if
end for
```

Proof. The learning rate of the main network is $\xi_t = \frac{1}{n(\boldsymbol{a_i},t)+1}$, where $n(\boldsymbol{a_i}, t)$ is the total number that action $\boldsymbol{a_i}$ is selected until iteration step t. Therefore, $\sum_{t=0}^{\infty} \frac{1}{n(\boldsymbol{a_i},t)+1} = \infty$ and $\sum_{t=0}^{\infty} \frac{1}{(n(\boldsymbol{a_i},t)+1)^2} < \infty$. Hence, IDEA algorithm satisfies the first condition in Assumption 1. Then the second condition is apparently satisfied as a stochastic Boltzmann exploration policy is employed. In the following, we need to show that the collaboration policy $\boldsymbol{\pi}$ is greedy in the limit with infinite exploration. For Assumption 1's third condition, we begin with Eqn 20: $\pi(\boldsymbol{a}_i|\boldsymbol{s}, \bar{\boldsymbol{a}}_i) = \frac{\exp(\xi Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s},\boldsymbol{a}_i,\bar{\boldsymbol{a}}_i))}{\sum_{\boldsymbol{a}'_i \in \mathcal{A}_i} \exp(\xi Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s},\boldsymbol{a}'_i,\bar{\boldsymbol{a}}_i))}$. The policy $\boldsymbol{\pi}$ is greedy with respect to $Q^i_{\pi_i,\pi_{-i}}(\boldsymbol{s}, \boldsymbol{a}'_i, \bar{\boldsymbol{a}}_i)$, which means that the probability of selecting a action with a higher Q-value is higher than of other actions. Moreover, as temperature ξ decaying asymptotically zero, the non-dominant actions still can be selected. Hence, the policy $\boldsymbol{\pi}$ satisfies the third condition of Assumption 1. Under the fourth condition of Assumption 1, the update rule in Eqn 18 forms a contraction mapping on the complete space from $\boldsymbol{Q}$ to the fixed point being the Nash Q-value Q^* of the entire Markov game [17], which shows that

$$\gamma \parallel \Delta(\boldsymbol{s}, a) \parallel_\infty = \gamma \parallel \bar{Q}(s(t), \boldsymbol{a}(t)) - Q_* \parallel_\infty \geq \parallel R_i(s(t), \boldsymbol{a}(t), s(t+1)) + \gamma \bar{V}^i_{\boldsymbol{\pi}}(s(t+1)) - Q_* \parallel_\infty . \tag{26}$$

Therefore, following Theorem 1 in [18], $\Delta(\boldsymbol{s}, a)$ converges to zero with probability 1, and hence IDEA algorithm can achieve a Nash equilibrium of the Markov game. □

5.3 Convergence Analysis

Now we investigate the convergence performance of IDEA. By comparing IDEA with the training policy without the incentive design in [29], we have the following theorem.

Theorem 2. *Let $acc_i(\boldsymbol{\theta}_i(t+1))$ and $acc_i(\boldsymbol{\theta}'_i(t+1))$ be the accuracy achieved by IDEA and [29] at iteration step $t+1$, respectively. After IDEA reaches the Nash equilibrium, $acc_i(\boldsymbol{\theta}_i(t+1)) \geq acc_i(\boldsymbol{\theta}'_i(t+1)), \forall i, \forall t$ when β_p, β_c, β_l are 0, and γ limits to 0.*

Proof. Let $g(\boldsymbol{\theta}(t), \xi(t))$ be the objective function in terms of the parties' parameters and samples at iteration step t. And let $\partial g(\boldsymbol{\theta}(t), \xi(t)) = \left\{ \sum_{m=1}^{M} \nabla f(\boldsymbol{\theta}_i(t), \xi_m^i(t)) \right\}_{i=1}^{|\mathcal{V}|}$, where $f(\cdot)$ is the loss function and $\xi_m^i(t)$ is a mini-batch sample from party i at iteration step t. Then, at iteration step $t+1$, party i can obtain the model parameter $\boldsymbol{\theta}_i(t+1)$ according to the model update policy of IDEA in Eqn 25, and the reward received is $R_i(t+1)$. In [29], party i does local training and pulls all its neighbors' model parameters, then updates the model by $\boldsymbol{\theta}'_i(t+1) = \boldsymbol{\theta}'_i(t)W - \beta \partial g(\boldsymbol{\theta}(t), \xi(t))$, where W is the connection matrix of parties and β is the learning rate. Similarly, we can calculate the corresponding reward, denoted as $R'_i(t+1)$. According to Theorem 1, IDEA can achieve a Nash equilibrium, and as required by the Nash equilibrium, $R_i(t+1) \geq R'_i(t+1)$ holds when γ limits to 0. Based on the designs of service revenue (Eqn 9) and incentive function (Eqn 10) that defines the positive relationship between accuracy and reward when β_p, β_c, β_l are 0, we have $acc_i(\boldsymbol{\theta}_i(t+1)) \geq acc_i(\boldsymbol{\theta}'_i(t+1)), \forall i, \forall t$. □

In essence, Theorem 2 illustrates that [29] is a lower bound of the model update strategy in IDEA. In other words, the model parameters of each party will converge with a rate no less than $O(\frac{1}{\sqrt{T}})$, where T is the number of iteration steps.

5.4 Adversarial Behaviors

In a decentralized collaboration system, some parties may be malicious and aim to explore arbitrage opportunities to increase their profits. In the following, we present three exemplary adversarial behaviors and discuss how IDEA can mitigate them.

First, a malicious party i may use irrational settings, such as an absurdly high price, to increase revenue when the other parties pull its model parameters. However, since IDEA can converge to Nash equilibrium (NE) given any settings, the learned collaboration policies will be adjusted to converge to a new NE when the input settings are changed. Hence, this behavior may cause other parties to pull party i's parameters less frequently given the new collaboration policies. Second, a malicious party i may misreport its states; for example, overstating its accuracy during training to attract the other parties to pull its parameters. Nevertheless, IDEA will calculate feedback (aka. reward) in Eqn 10 according to the pulled parameters, which is used to approximate the implicit relationship between the state and Q-function and then adjust the collaboration policy. Therefore, if party i's reported accuracy is incorrect or is not useful (e.g., in the Non-IID scenario, party i's accuracy may not be beneficial for the others' accuracy improvement), the other parties can down-weight these values and rarely pull party i's parameters in future iterations. Third, a malicious party i may be reluctant to train its local model as it may not have sufficient high-quality data and computational capability, and then just pull the parameters from one neighbor (say party j) and forward it to other neighbors to gain profits. Note that if party i is a key node in the collaboration network, it should receive some reward for its connectivity. Otherwise, the other neighbors may already pull party j's parameters as party j's accuracy is better in the current iteration, and thus it is hard for party i to arbitrage due to the pull parameter cost. Overall, IDEA is effective in mitigating these adversarial behaviors to ensure fairness. We shall experimentally study the impact of these behaviors in IDEA in Section 6.

5.5 Limitation and Extensibility Discussion

We note that IDEA only pulls a subset of neighboring parties' model parameters in each iteration. In the IID scenario, which is often not the case in DFL applications, the accuracy of IDEA may not be as ideal as those approaches that collect all parties' information, such as Allreduce. Notwithstanding, the communication efficiency of IDEA remains much higher, as will be discussed in Section 6.3. We also note that the training of MARL collaboration policy has a certain computational and communication overhead, as will be presented in Figure 7, even though it is lightweight compared to the whole iteration. Further, in this paper, we only consider static collaboration topologies, which assume the neighbors are pre-determined for each party in Algorithm 1. Thus, it does not support dynamic collaboration networks where parties can leave and join at any time. Nevertheless, IDEA is extensible to dynamic scenarios with the incorporation of meta-learning so that when each party faces a new task (i.e., a dynamic network setting), the meta-learning module can learn the relationship between the new task and historical tasks to generate a collaboration policy for the party. This however requires an in-depth study. Moreover, though we consider fully decentralized topologies in IDEA, it can be easily extended to support various DFL architectures. For example, we can incorporate IDEA into the coordinator-based DFL architecture [7, 56], where a quorum-based selection of coordinators can implement the MARL collaboration policy modules.

6 EXPERIMENTS

In this section, we detail the experimental setup, baselines, datasets, and experimental evaluations.

6.1 Experimental Setup

6.1.1 Datasets and Models. We utilize five real-world datasets with diversified feature types and applications in the experimental evaluation, so as to demonstrate the robustness of the IDEA framework.

- **Adult Dataset** [41] is drawn from the United States Census Bureau data and uses personal information to predict whether an individual will earn 50, 000 per year. It is a structured dataset consisting of 48842 samples with 14 attribute fields in 2 classes.
- **Connect-4 Dataset** [42] is based on the outcome of Connect-4 games containing unforced positions after 8 moves and the eventual winner. This dataset is made up of 67557 structured data samples represented by 42 features in 2 classes. The percentage of the majority class of the dataset is 75.4%. Therefore, we use this dataset to evaluate IDEA in the unbalanced dataset setting.
- **CIFAR-10 Dataset** [24] is one of the most widely used image datasets for classification tasks. It consists of 60000 samples with 3072 features in 10 classes, including airplanes, cars, birds, etc.
- **OrganSMNIST (Organ) Dataset** [59] is a medical dataset based on abdominal clinical computed tomography. In this dataset, we aim to demonstrate the meaningful usage of IDEA in the field of healthcare analytics. The dataset comprises 25221 samples with 784 numerical features in 11 classes.
- **Sensorless Drive Diagnosis (Diag) Dataset** [43] is related to process automation systems. The data is extracted from electric current drive signals, which results in 11 different conditions for detecting defective equipment. The dataset consists of 58509 samples with 49 numerical features with 11 classes.

Dataset Partition. We apply two data partition settings for each dataset: 1) *IID setting*, where sub-datasets are uniformly distributed from the global dataset for each party. 2) *Non-IID setting*, where sub-datasets follow the Dirichlet distribution for each party's class distribution [63]. Note that in both partitions, the number of samples for each party is unbalanced and drawn from the

Log-Normal distribution with variance [2]. We split the sub-dataset of each party in 7:1:2 for training, validation, and testing, respectively.

Local Models. For Adult, Connect-4, and Diag datasets, we employ the multilayer perceptron (MLP) based deep neural networks (DNN) as the local model for each party to capture the nonlinear feature interaction. For CIFAR-10 and OrganSMNIST datasets, we adopt the widely used VGG-19 [49] as the local model for each party.

6.1.2 Data Collaboration Network. We adopt irregular topologies generated by power-law [4], which are widely used in healthcare analysis [32], traffic forecasting [67], and wireless networks [46] for decentralized data collaboration. We also use topologies with different sparsity values $[0, 0.5, 0.75]$ to evaluate the performance, where the topology is fully connected when the sparsity value is 0, and the higher the value, the fewer communication links in the network. We vary the number of parties from 4 to 24. Unless stated otherwise, the number of parties is set to 8 as default.

In terms of hardware heterogeneity, we consider that different parties are equipped with different computational and communication capacities. In the experiments, the computational and communication capacities of parties are randomly assigned in the range of $[0, 1]$. The unit cost of local model training c_i^{train} is uniformly distributed in $[1e\text{-}4, 3e\text{-}4]$ [61], which is the cost of computing one sample data. The priority metric c_i^{acc} is set to 1 by default, which can be used to adjust the parameter price in practice.

6.1.3 Baselines. IDEA is designed as an extensible DFL framework to satisfy objective-specific data collaboration. Accordingly, we evaluate four variants, namely **IDEA-AC**, **IDEA-CPE**, **IDEA-CME**, and **IDEA-ALL**. We compare IDEA with four baselines, which are imposed on the same network configurations for a fair comparison. The baselines and our variants are as follows.

- **Solo** is a baseline in which each party locally trains its model using the local dataset without interacting with other parties. It gives the accuracy lower bound without data collaboration.
- **D-PSGD** [28] is a decentralized parallel stochastic gradient descent algorithm for solving large-scale machine learning problems where each party randomly communicates with one of its neighbors at each iteration step.
- **CDSGD** [20] enables data parallelization as well as decentralized computation in collaborative deep learning, where each party communicates with all of its neighbors at each iteration step.
- **Allreduce** [19] is a three-phase all-reduce algorithm for aggregating multiple parties' model parameters, where each party can obtain all parties' model parameters at each iteration step.
- **IDEA-AC** focuses on improving each party's training model accuracy. Specifically, we set β_s as 1, while β_p, β_c, and β_l are set to 0 in Eqn 10.
- **IDEA-CPE** focuses on improving the computational efficiency for individual parties. Specifically, β_s and β_l in Eqn 10 are set to 1, while β_p and β_c are set to 0.
- **IDEA-CME** focuses on improving the communication efficiency of individual parties. Consequently, β_s and β_c in Eqn 10, are set to 1, while β_p and β_l are set to 0.
- **IDEA-ALL** considers all four aspects, including service revenue, push parameter revenue, pull parameter cost, and local training cost, aiming to receive a fair reward. Accordingly, we set β_s, β_p, β_c and β_l in Eqn 10 to 1.

For ease of reference for our subsequent discussion on the experimental results, we summarize the characteristics of the proposed IDEAs and four baselines in Table 1.

6.1.4 Evaluation Metrics. To comprehensively evaluate the performance of the IDEA framework, we employ the following metrics:

- **Best Mean Testing Accuracy (BMTA)** is the average of the best accuracy on the testing dataset among the participating parties.

Table 1. Characteristics of IDEA vs. four baselines.

Methods	Collaborative	Partial Data Sharing	Fairness Guarantee
Solo	×	-	-
D-PSGD	✓	✓	×
CDSGD	✓	✓	×
Allreduce	✓	×	×
IDEA	✓	✓	✓

Table 2. BMTA with standard deviation on all parties.

Settings	Solo	D-PSGD	CDSGD	Allreduce	**IDEA-AC**
Adult-IID	83.5±0.8	85.4±1.2	84.2±0.8	**86.7±0.0**	86.3±1.3
Adult-N-IID	89.3±7.1	89.7±7.0	89.5±7.2	89.4±7.2	**92.4±5.0**
Connect-IID	84.3±2.3	86.6±2.3	86.4±1.9	**89.1±0.0**	87.1±1.0
Connect-N-IID	81.8±10.6	82.9±10.1	83.7±9.8	86.3±8.5	**88.5±6.9**
CIFAR-IID	66.2±3.2	80.4±3.4	75.8±3.2	**85.0±0.0**	84.2±1.0
CIFAR-N-IID	74.6±5.9	81.4±4.2	75.6±9.4	82.8±3.3	**86.5±3.5**
Organ-IID	88.9±2.8	96.7±3.6	94.3±2.8	**100.0±0.0**	97.8±0.4
Organ-N-IID	90.2±5.3	93.2±4.6	92.6±3.9	95.4±2.1	**97.1±1.9**
Diag-IID	59.6±12.3	71.8±7.3	66.4±4.2	**93.0±0.0**	90.8±2.4
Diag-N-IID	64.0±14.3	71.4±6.8	70.0±6.4	79.6±6.9	**87.5±6.6**

- **Computational Efficiency (CPE)** refers to the average test accuracy improvement achieved by a method compared to Solo in a given local iteration step. It is defined as $CPE = \sum_i \frac{acc_i^* - acc_{i,Solo}^*}{LT_i}$, where acc_i^* denotes the best test accuracy of party i achieved by a method; $acc_{i,solo}^*$ indicates the best test accuracy of party i given the Solo algorithm; LT_i is the number of local training actions of party i during the whole training phase.
- **Communication Efficiency (CME)** refers to the average test accuracy improvement achieved by a method compared to Solo for one communication round. It can be expressed as $CME = \sum_i \frac{acc_i^* - acc_{i,Solo}^*}{CR_i}$, where acc_i^* and $acc_{i,solo}^*$ are the same as those in CPE, and CR_i is the average number of communication rounds of party i with its neighbors during the whole training phase.
- **Execution Time** is used to evaluate the average training time per iteration of the methods and to illustrate the computation overhead of the MARL module.
- **Communication Cost** is used to measure the volume of data transmitted per iteration and to evaluate the communication overhead of the MARL module.
- **Pulled Frequency** is the average number of each party's parameters being pulled by other parties over the number of total training iterations.

6.1.5 Implementation Details. We use the SGD optimizer to train the local models with batch size 128, momentum 0.9, and weight decay 1e-4. The learning rate starts from 0.01 and decays with a factor of 10 when iteration step t is $\frac{T}{3}$ or $\frac{2T}{3}$. The Q networks are 4-layer MLPs with 64 neurons per hidden layer and trained with K 10, B 100, γ 0.8, and τ 0.5. We implement the models using PyTorch 1.12.0 with CUDA 11.3. For the time and communication cost evaluations, we conduct experiments on AWS EC2 with up to 24 g4dn.xlarge instances across two regions, where each instance equips 1 NVIDIA T4 GPU, and 4vCPUs with 32G memory. The rest experiments are conducted on a server with Xeon(R) Silver 4214R CPU @ 2.40GHz (14 cores), 128G memory, 8 GeForce RTX 3090.

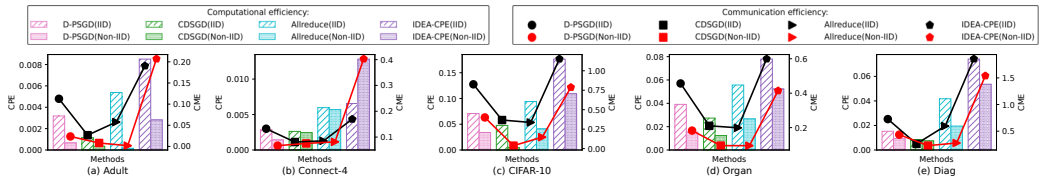


Fig. 3. Comparison of computational and communication efficiency.

6.2 Effectiveness

Baseline comparison w.r.t. BMTA. We first compare IDEA-AC with the baseline methods regarding the model effectiveness. Table 2 summarizes the experimental results measured by BMTA with standard deviation on the five datasets under IID and Non-IID settings. We make the following three key observations.

First, all the decentralized federated learning approaches, i.e. D-PSGD, CDSGD, All-reduce, and IDEA-AC, achieve noticeably higher BMTA than the Solo approach. It validates the effectiveness of the collaborative learning framework. *Second*, IDEA-AC can significantly outperform D-PSGD and CDSGD baselines in term of mean BMTA of all parties. The reason is that D-PSGD and CDSGD adopt fixed collaboration policies, which communicate with a random neighbor or with all neighbors to pull the model parameters for aggregation. This may lead a party to pull useless or even harmful parameters, degrading its local model accuracy. In contrast, IDEA-AC adaptively adjusts the collaboration policy, enabling each party to take actions with aim of maximizing the incentive function related to model accuracy. Note that different parties' BMTA may vary greatly w.r.t. their local datasets, leading to a large standard deviation of BMTA and thus accuracy overlaps among different approaches. We provide an in-depth BTMA comparison of each party in Table 3 later. *Third*, regarding the comparison to the Allreduce baseline, our observation is two-fold. For one thing, under the IID setting, Allreduce performs slightly better than IDEA-AC. This is because, as summarized in Table 1, Allreduce requires perfect data sharing between all parties, i.e., each party can obtain all of the other parties' model parameters at each iteration step. Therefore, Allreduce can maintain a consistent model at each iteration step, which promotes the convergence of Allreduce by avoiding the jitter caused by model asynchronization. Nevertheless, Allreduce requires more communication costs per iteration than IDEA-AC, as the party in Allreduce should communicate with others at each iteration (as will be shown in Section 6.3). For another, under the Non-IID setting, IDEA-AC is superior to Allreduce. It shows that the global model in Allreduce that averages all parties' local models would deviate from the actual optimum of heterogeneous parties.

In-depth comparison w.r.t. BTMA on each party. We further conduct an in-depth comparison of BMTA on each party as shown in Tables 3 and 4, using Adult and Connect-4 under IID and Non-IID settings, respectively. From Table 3, we observe that the BMTA is consistent with the analysis in Table 2, where IDEA-AC outperforms D-PSGD, CDSGD, and Solo for most parties.

Table 3. BMTA of Adult under IID setting.

Party	Solo	D-PSGD	CDSGD	Allreduce	**IDEA-AC**
0	82.81	83.98	83.59	**86.72**	86.59
1	82.81	85.94	84.38	86.72	**87.09**
2	82.42	85.16	83.59	**86.72**	83.43
3	84.77	86.72	83.98	86.72	**87.99**
4	84.77	86.72	84.38	86.72	**87.48**
5	83.59	85.94	85.94	**86.72**	86.48
6	83.59	85.55	84.38	**86.72**	85.71
7	83.20	83.20	83.20	**86.72**	85.90

Also, different parties have different BMTA as these solutions do not maintain a global model for all parties. Thus, there is a standard deviation w.r.t. the mean BMTA of all parties (see Table 2), making the mean BMTA overlaps among different approaches. The results of the Non-IID setting for Connect-4 dataset are shown in Table 4. We can observe that the BMTA of all the parties improves under decentralized federated learning approaches (i.e., D-PSGD, CDSGD, All-reduce, and IDEA-AC) compared to Solo. Further, IDEA-AC outperforms the baselines for all the parties. These observations are consistent with the results in Table 2, demonstrating the effectiveness of the proposed IDEA framework. Moreover, we find that the parties' performance in terms of BMTA varies significantly due to their heterogeneities, and those with worse performance under Solo achieve larger performance improvements under the IDEA-AC. For example, the three worst performing parties (i.e., party 4, party 0, and party 7) achieve 10.4%, 10.55%, and 12.22% BMTA improvements under IDEA-AC, which are significantly higher than those of the remaining parties. Clearly, the parties with insufficient or low-quality data can greatly improve their local models' accuracy by data collaboration through IDEA, which may thus encourage more parties to join in DFL based on the proposed IDEA framework.

6.3 Efficiency

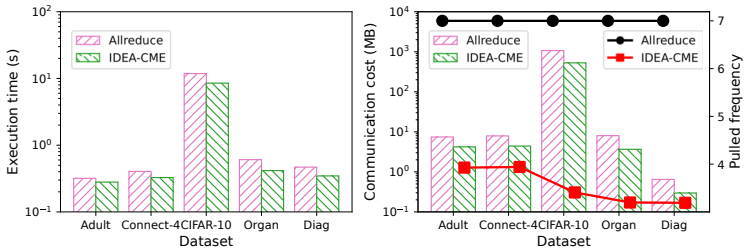
To evaluate the IDEA's efficiency, we first compare IDEA-CPE and IDEA-CME with the five baselines w.r.t. CPE and CME, as shown in Figure 3. Note that the greater the value is, the higher efficiency a method obtains. We observe that IDEA-CPE and IDEA-CME significantly outperform the baselines with regard to CPE and CME, respectively, on five datasets in both IID and Non-IID settings. The results demonstrate that the proposed collaboration policy can dynamically adapt to various scenarios. Specifically, in terms of CPE, since the D-PSGD, CDSGD, and Allreduce approaches conduct local DFL training in each iteration, their CPEs are proportional to the model accuracy in Table 2. In contrast, IDEA-CPE achieves comparable or even higher model accuracy without local DFL training in every iteration, which therefore results in higher CPE. For CME, although Allreduce's model accuracy is slightly better than IDEA under the IID setting (see Table 2), it requires more communication rounds to synchronize the model parameters in each iteration step. As a consequence, its CME is lower than that of IDEA-CME. As an aside, as summarized in Table 1, Allreduce requires perfect parameter sharing between all parties, where each party needs to pull all of the other parties' model parameters at each iteration step.

We next compare the efficiency w.r.t. the average training time, average communication cost, and average pulled frequency of IDEA-CME and Allreduce in a distributed cluster on AWS EC2. We use a fully connected collaboration network with 8 workers under the IID setting. The results are shown in Figure 4. We can observe from Figure 4(b) that IDEA's communication cost is much lower than Allreduce. For example, IDEA's average communication cost on CIFAR-10 is 521.36MB, while that of Allreduce is around 1070.91MB. This is because Allreduce requires each party to pull other parties' parameters in each iteration, and thus the pulled frequency is 7. In contrast, IDEA's

Table 4. BMTA of Connect-4 under Non-IID setting.

Party	Solo	D-PSGD	CDSGD	Allreduce	**IDEA-AC**
0	73.2	75.7	75.16	81.48	**83.75**
1	99.45	**100.00**	**100.00**	**100.00**	**100.00**
2	79.51	80.12	79.69	84.46	**85.77**
3	74.16	75.56	77.73	78.91	**80.46**
4	73.02	75.00	79.09	81.55	**85.24**
5	80.95	81.73	83.77	87.68	**88.53**
6	99.67	**100.00**	**100.00**	**100.00**	**100.00**
7	74.11	75.11	74.33	76.46	**84.51**

average pulled frequency on CIFAR-10 is 3.41, greatly reducing the communication cost. Besides, IDEA is faster than Allreduce on all five datasets (see Figure 4(a)) as each party transmits fewer model parameters, saving the overall training time.



(a) Execution time (b) Comm. cost and pulled freq.

Fig. 4. The comparison of IDEA and Allreduce in terms of execution time, communication cost, and pulled frequency.

6.4 Reward Evaluation

We now evaluate the rewards administered by IDEA-ALL for the parties involved to validate the fairness of our incentive mechanism.

Comparison of calculated rewards. We further compare IDEA-ALL with the baselines in terms of the average reward with standard deviation over all parties. We employ the same incentive function for all the baselines. Figure 5 presents the results on the five datasets. We can observe that the average reward of IDEA-ALL consistently and significantly outperforms the baselines on all datasets. For example, IDEA-ALL achieves up to 1.30x, 1.42x, and 1.35x higher rewards compared to D-PSGD, CDSGD, and Allreduce, respectively. The rationale is that IDEA-ALL's collaboration policy is learned to maximize each party's long-term reward while the baselines adopt fixed policies, which may not be good choices for parties.

Ablation study. For the two designs presented in Section 4.2 for improving IDEA's performance, namely the mean field (MF) representation and the dual neural networks (Dual NN) approximation, we shall now study their effects on the reward and Q-value estimation. Figure 6 shows the results on CIFAR-10 under IDEA (using both MF and Dual NN) and three variants: (1) standard Q-function + Dual NN; (2) MF + Single NN; and (3) standard Q-function + Single NN. We can see from Figure 6(a) that IDEA converges faster than the standard Q-function + Dual NN variant, demonstrating that the MF representation can accelerate the convergence process. This is because MF reduces the action space of the Q-function, and thus the Boltzmann exploration policy will be more efficient. Furthermore, Figure 6(b) shows that IDEA can mitigate the overestimation issue by comparing it with the MF + Single NN variant. The rationale is that by introducing a target network, IDEA can decompose the action selection and evaluation operations into two interactive networks. In this way, the main network will be more robust by learning from the target network.

6.5 Scalability

We further investigate the scalability of our framework with respect to the number of parties and the sparsity of the network.

Effects of number of parties. We first evaluate the execution time and communication cost of the whole iteration and the proposed MARL module w.r.t. $|\mathcal{V}| \in \{4, 8, 12, 16, 20, 24\}$ on CIFAR-10. From Figure 7(a), we can see that the average iteration time increases as the parties need to conduct more local training iterations and communications to learn an effective model when there are more

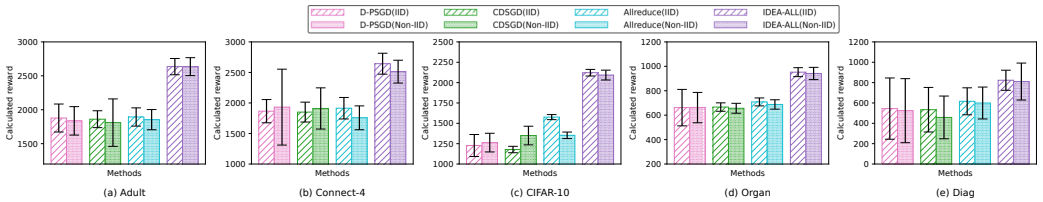


Fig. 5. Comparison based on calculated reward.

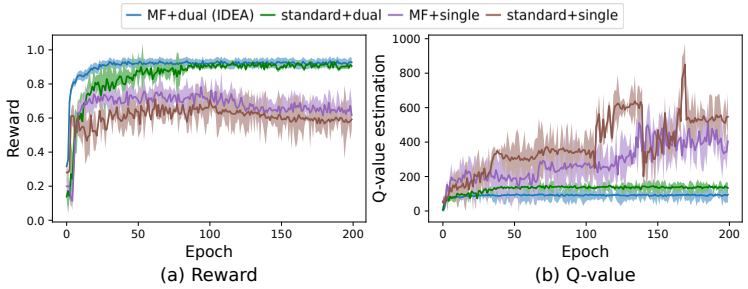


Fig. 6. The ablation study of the incentive mechanism.

parties. Meanwhile, the MARL training time increases since the collaboration policy becomes more complex given a larger $|\mathcal{V}|$. Similarly, Figure 7(b) compares the average communication costs. We observe that both costs increase with $|\mathcal{V}|$ while the MARL's cost only occupies a small percentage of the overall cost. The reason is that the size of actions and states is much smaller compared to the exchanged DFL model parameters.

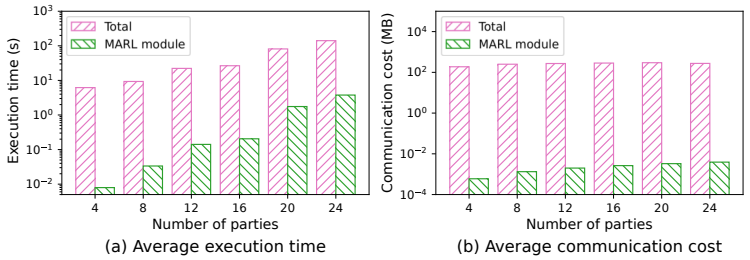
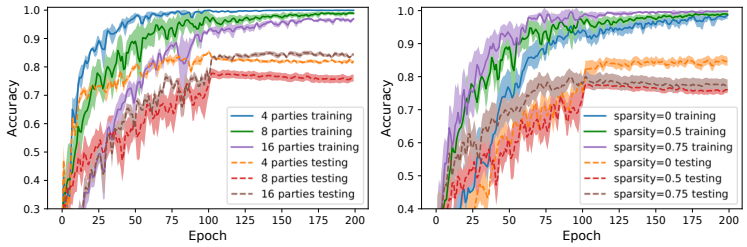


Fig. 7. Effects of the number of parties on execution time and communication cost.

We then vary the number of parties $|\mathcal{V}| \in \{4, 8, 16\}$ and evaluate the mean accuracy of all the parties. As shown in Figure 8(a), the convergence rate slows down when the number of parties increases, which is consistent with the results in [20]. The reason is that when $|\mathcal{V}|$ increases, the parties need more communication rounds to exchange the model parameters for model averaging and synchronization. Therefore, the inconsistency of models results in a slight fluctuation of the performance. Nevertheless, IDEA-ALL is able to scale to more parties within an acceptable number of iterations.

Effects of network sparsity. We also evaluate IDEA-ALL with varying network sparsity to study its scalability. Specifically, we conduct experiments with three network sparsity: 0, 0.5, and 0.75. The larger the value, the more sparse the network. In particular, 0 indicates the fully connected network. The results of the mean accuracy for all parties are shown in Figure 8(b). We observe that with the increasing sparsity of the network, the convergence rate speeds up. The rationale is that

reducing the connectivity level of parties makes the parties' stochastic interaction matrix sparser. Thus, it takes fewer iterations to finish the training. Nonetheless, IDEA can easily scale to denser networks with reasonable overhead.



(a) Effect of the number of parties (b) Effect of network sparsity

Fig. 8. Training and test accuracy of IDEA-ALL with respect to the number of parties and network sparsity.

6.6 Effects of Incentive Function

We now compare the four IDEA variants with respect to BMTA, CPE, and CME. The results are reported in Table 5. We observe that IDEA-AC performs the best in terms of BTMA. This is expected as IDEA-AC focuses solely on improving the model accuracy while IDEA-CPE, IDEA-CME, and IDEA-ALL also consider other factors in the incentive function Eqn 10. Similarly, IDEA-CPE and IDEA-CME perform better than the other variants in terms of CPE and CME, respectively. The results demonstrate the effectiveness of the incentive function in realising the intended objective of each party's collaboration policy as a whole. It further confirms the generality and flexibility of IDEA, and hence its extensibility for data collaboration across different application domains.

Table 5. IDEA variants Comparison on Connect-4 (Non-IID).

Methods	IDEA-AC	IDEA-CPE	IDEA-CME	IDEA-ALL
BMTA	**88.53**	87.85	87.79	87.67
CPE	1.203E-2	**1.283E-2**	1.211E-2	1.198E-2
CME	3.933E-1	3.921E-1	**4.029E-1**	3.874E-1

6.7 Adversarial Behaviors

Finally, we evaluate IDEA under the adversarial behaviors discussed in Section 6.6, where a malicious party may use irrational settings, misreport its accuracy, or act as a 'free rider' without local training, to seek arbitrage opportunities. For ease of analysis, we use the fully-connected network topology with four parties $\{0, 1, 2, 3\}$, where party 3 is malicious while other parties are honest.

Irrational settings. Figure 9 shows the reward and pulled frequency of parties w.r.t. different parameter price settings (Eqn. 6) of party 3. Specifically, we use four settings $c_3^{acc} \in \{1, 4, 7, 10\}$. The higher the value, the higher party 3's parameter price. Note that $c_3^{acc} = 1$ is the baseline, which is the default value. We observe that the other parties' rewards and pulled frequencies are relatively stable when c_3^{acc} increases. However, party 3's reward gradually decreases. The reason is that the collaboration policy learning modules on honest parties down-weight the action for pulling party 3's parameters. It can be inferred from Figure 9(b), where party 3's pulled frequency decreases greatly. This indicates that IDEA is effective in dealing with irrational parameter price settings.

Misreport accuracy. Figure 10 shows the reward and pulled frequency of parties w.r.t. different misreported accuracies (Eqn 11) of party 3. Similarly, we consider four settings, where $acc_{3,c} + x\%$

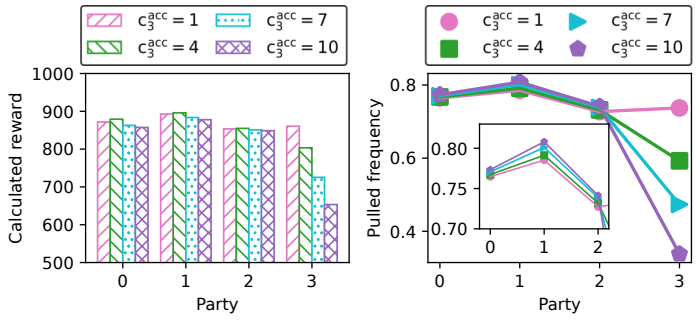


Fig. 9. The comparison w.r.t. irrational settings.

denotes that party 3 overstates its accuracy by $x\% \in \{0\%, 2\%, 4\%, 8\%\}$. We can see that all parties' rewards (including that of party 3) only change slightly, and the pulled frequencies are also stable compared to those with irrational settings. The rationale is that each party will validate the other parties' accuracies by calculating the reward in IDEA. Thus, the collaboration policy is slightly affected even if a malicious party overstates its training accuracies.

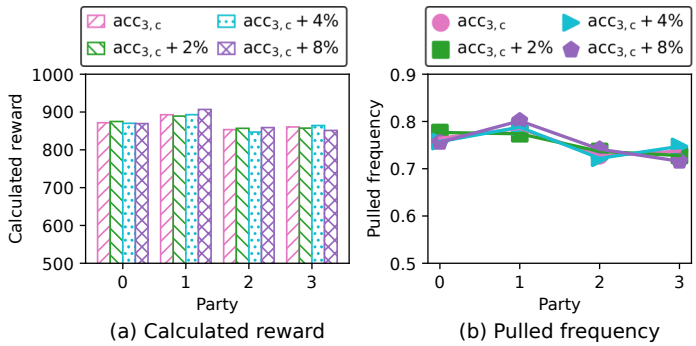


Fig. 10. The comparison w.r.t. misreported accuracy.

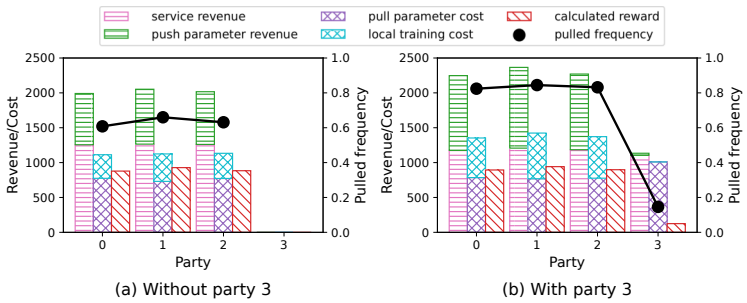


Fig. 11. The comparison without local training.

Without local training. Figure 11 presents the results of the adversarial behavior without local training. Figure 11(a) shows the baseline of 3-party collaboration without party 3, where the parties' rewards are 878.24, 927.84, and 883.51, respectively. Figure 11(b) shows the results after party 3 joins, where the rewards of previous parties increase to 894.08, 941.88, and 897.88, respectively.

This is mainly due to the increasing pull requests from party 3, which can be verified by the pulled frequency and push parameters revenue in Figure 11(b). Moreover, party 3 also receives a reward after collaboration because it can obtain service revenue with a more accurate model. Besides, we find that the former parties' service revenues slightly decrease, and their local train costs increase. This is because party 3 is a straggler without local training, which slows down the collaboration process. Nevertheless, their overall rewards are increased for pushing the parameters to party 3.

7 RELATED WORK

DFL with heterogeneous parties. DFL is receiving increasing attention, where multiple heterogeneous parties are connected with irregular topologies. Existing works mainly investigate two heterogeneities in DFL: (i) data heterogeneity, where the parties hold different sizes of datasets or different data distributions [16, 25, 37, 45, 47]; and (ii) hardware heterogeneity, where the parties have different computational or communication capabilities [23, 55, 73]. However, these works only focus on one type of heterogeneity and consider a fixed collaboration policy, which is insufficient to handle various real-world scenarios. Moreover, they do not take the incentive for data collaboration into consideration. In comparison, we consider both data heterogeneity and hardware heterogeneity and devise an incentive-aware framework for DFL.

Incentive mechanism for data collaboration. In real-world applications, parties may be reluctant to join a collaborative learning process without reasonable and fair incentives. Therefore, a line of works has pursued various incentive mechanisms to support data collaboration [51]. Most existing works are driven by parties' data contribution [48, 64, 66] and reputation [21, 52] to calculate reasonable incentives for parties. Moreover, many incentive mechanisms are designed based on conventional game-theory approaches, such as Stackelberg game [40] and non-cooperative game [57]. However, these mechanisms only consider the centralized setting, where a centralized server is responsible for evaluating the incentives. Thus, they are not applicable to DFL. Our proposed framework does not rely on a centralized server, while allowing individual parties to customize their incentive models to gain fair rewards during data collaboration. In [34], a set of blockchain nodes replaces the central server to evaluate each party's contribution. However, it employs a fixed collaboration policy for all parties. In contrast, we propose an effective MARL method to learn an adaptive policy for individual parties in decentralized data collaboration.

8 CONCLUSIONS

In this paper, we have proposed IDEA, an incentive-aware decentralized federated learning framework to facilitate decentralized data collaboration. IDEA allows heterogeneous parties to customize their incentive functions and learn an objective-specific collaboration policy. We present a multi-agent reinforcement learning incentive model to enable the parties to maximize their long-term rewards. We theoretically analyze the fairness and convergence of the proposed framework. To the best of our knowledge, this is the first work investigating the incentive mechanism for fully decentralized data collaboration. The experimental results demonstrate that IDEA achieves high model accuracy and efficiency, outperforming state-of-the-art approaches.

ACKNOWLEDGMENTS

This work is supported by the National Key Research and Development Program of China under grant 2020YFB1806804, Huawei cooperation project under grant TC20210316002, and the Singapore Ministry of Education Academic Research Fund Tier 3 under MOEs official grant number MOE2017-T3-1-007.

REFERENCES

[1] 2018. California Consumer Privacy Act. Bill No. 375 privacy: personal information: businesses. https://leginfo.legislature.ca.gov/. (2018).

[2] Durmus Alp Emre Acar, Yue Zhao, Ramon Matas, Matthew Mattina, Paul Whatmough, and Venkatesh Saligrama. 2020. Federated Learning Based on Dynamic Regularization. In *ICLR*.

[3] Ergute Bao, Yizheng Zhu, Xiaokui Xiao, Yin Yang, Beng Chin Ooi, Benjamin Tan, and Khin Mi Mi Aung. 2022. Skellam Mixture Mechanism: a Novel Approach to Federated Learning with Differential Privacy. *Proc. VLDB Endow.* 15, 11 (2022), 2348–2360.

[4] Albert-László Barabási and Réka Albert. 1999. Emergence of scaling in random networks. *science* 286, 5439 (1999), 509–512.

[5] Albert-László Barabási, Réka Albert, and Hawoong Jeong. 1999. Mean-field theory for scale-free random networks. *Physica A: Statistical Mechanics and its Applications* 272, 1-2 (1999), 173–187.

[6] Lawrence E Blume. 1993. The statistical mechanics of strategic interaction. *Games and economic behavior* 5, 3 (1993), 387–424.

[7] Christopher Briggs, Zhong Fan, and Peter Andras. 2020. Federated learning with hierarchical clustering of local updates to improve training on non-IID data. In *2020 International Joint Conference on Neural Networks (IJCNN)*. 1–9. https://doi.org/10.1109/IJCNN48605.2020.9207469

[8] Zhenan Fan, Huang Fang, Zirui Zhou, Jian Pei, Michael P. Friedlander, Changxin Liu, and Yong Zhang. 2022. Improving Fairness for Data Valuation in Horizontal Federated Learning. In *ICDE*. 2440–2453.

[9] Gabriele Farina, Andrea Celli, Alberto Marchesi, and Nicola Gatti. 2022. Simple uncoupled no-regret learning dynamics for extensive-form correlated equilibrium. *J. ACM* 69, 6 (2022), 1–41.

[10] Arlington M Fink. 1964. Equilibrium in a stochastic n-person game. *Journal of science of the hiroshima university, series ai (mathematics)* 28, 1 (1964), 89–93.

[11] Yann Fraboni, Richard Vidal, and Marco Lorenzi. 2021. Free-rider attacks on model aggregation in federated learning. In *AISTATS*. PMLR, 1846–1854.

[12] Fangcheng Fu, Yingxia Shao, Lele Yu, Jiawei Jiang, Huanran Xue, Yangyu Tao, and Bin Cui. 2021. VF^{2}Boost: Very Fast Vertical Federated Gradient Boosting for Cross-Enterprise Learning. In *SIGMOD*. 563–576.

[13] Fangcheng Fu, Huanran Xue, Yong Cheng, Yangyu Tao, and Bin Cui. 2022. BlindFL: Vertical Federated Machine Learning without Peeking into Your Data. In *SIGMOD*. 1316–1330.

[14] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. 2020. Inverting gradients-how easy is it to break privacy in federated learning? *NeurIPS* 33 (2020), 16937–16947.

[15] Sergiu Hart. 1992. Games in extensive and strategic forms. *Handbook of game theory with economic applications* 1 (1992), 19–40.

[16] István Hegedűs, Gábor Danner, and Márk Jelasity. 2019. Gossip learning as a decentralized alternative to federated learning. In *DAIS*. Springer, 74–90.

[17] Junling Hu and Michael P Wellman. 2003. Nash Q-learning for general-sum stochastic games. *Journal of machine learning research* 4, Nov (2003), 1039–1069.

[18] Tommi Jaakkola, Michael Jordan, and Satinder Singh. 1993. Convergence of stochastic iterative dynamic programming algorithms. *Advances in neural information processing systems* 6 (1993).

[19] Xianyan Jia, Shutao Song, Wei He, Yangzihao Wang, Haidong Rong, Feihu Zhou, Liqiang Xie, Zhenyu Guo, Yuanzhou Yang, Liwei Yu, et al. 2018. Highly scalable deep learning training system with mixed-precision: Training imagenet in four minutes. *arXiv preprint arXiv:1807.11205* (2018).

[20] Zhanhong Jiang, Aditya Balu, Chinmay Hegde, and Soumik Sarkar. 2017. Collaborative deep learning in fixed topology networks. *Advances in Neural Information Processing Systems* 30 (2017).

[21] Jiawen Kang, Zehui Xiong, Dusit Niyato, Yuze Zou, Yang Zhang, and Mohsen Guizani. 2020. Reliable federated learning for mobile networks. *IEEE Wireless Communications* 27, 2 (2020), 72–80.

[22] Daphne Koller and Avi Pfeffer. 1997. Representations and solutions for game-theoretic problems. *Artificial intelligence* 94, 1-2 (1997), 167–215.

[23] Anastasia Koloskova, Sebastian Stich, and Martin Jaggi. 2019. Decentralized stochastic optimization and gossip algorithms with compressed communication. In *ICML*. PMLR, 3478–3487.

[24] Alex Krizhevsky. 2009. Learning Multiple Layers of Features from Tiny Images. (2009), 32–33. https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf

[25] Chengxi Li, Gang Li, and Pramod K Varshney. 2021. Decentralized federated learning via mutual knowledge transfer. *IEEE Internet of Things Journal* 9, 2 (2021), 1136–1147.

[26] Qinbin Li, Yiqun Diao, Quan Chen, and Bingsheng He. 2022. Federated Learning on Non-IID Data Silos: An Experimental Study. In *ICDE*. 965–978.

[27] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. 2020. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine* 37, 3 (2020), 50–60.

[28] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. 2017. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. *NIPS* 30 (2017).

[29] Xiangru Lian, Ce Zhang, Huan Zhang, Cho Jui Hsieh, Wei Zhang, and Ji Liu. 2017. Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent. (2017).

[30] Wei Yang Bryan Lim, Zehui Xiong, Chunyan Miao, Dusit Niyato, Qiang Yang, Cyril Leung, and H Vincent Poor. 2020. Hierarchical incentive mechanism design for federated machine learning in mobile networks. *IEEE Internet of Things Journal* 7, 10 (2020), 9575–9588.

[31] Yejia Liu, Weiyuan Wu, Lampros Flokas, Jiannan Wang, and Eugene Wu. 2021. Enabling SQL-based Training Data Debugging for Federated Learning. *Proc. VLDB Endow.* 15, 3 (2021), 388–400.

[32] Songtao Lu, Yawen Zhang, and Yunlong Wang. 2020. Decentralized Federated Learning for Electronic Health Records. In *2020 54th Annual Conference on Information Sciences and Systems (CISS)*. 1–5. https://doi.org/10.1109/CISS48834.2020.1570617414

[33] Xinjian Luo, Yuncheng Wu, Xiaokui Xiao, and Beng Chin Ooi. 2021. Feature Inference Attack on Model Predictions in Vertical Federated Learning. In *ICDE*. 181–192.

[34] Shuaicheng Ma, Yang Cao, and Li Xiong. 2021. Transparent contribution evaluation for secure federated learning on blockchain. In *2021 IEEE 37th International Conference on Data Engineering Workshops (ICDEW)*. IEEE, 88–91.

[35] Viraaji Mothukuri, Prachi Khare, Reza M. Parizi, Seyedamin Pouriyeh, Ali Dehghantanha, and Gautam Srivastava. 2022. Federated-Learning-Based Anomaly Detection for IoT Security Attacks. *IEEE Internet Things J.* 9, 4 (2022), 2545–2554.

[36] Dinh C Nguyen, Quoc-Viet Pham, Pubudu N Pathirana, Ming Ding, Aruna Seneviratne, Zihuai Lin, Octavia Dobre, and Won-Joo Hwang. 2022. Federated learning for smart healthcare: A survey. *ACM Computing Surveys (CSUR)* 55, 3 (2022), 1–37.

[37] Noa Onoszko, Gustav Karlsson, Olof Mogren, and Edvin Listo Zec. 2021. Decentralized federated learning of deep neural networks on non-iid data. *arXiv preprint arXiv:2107.08517* (2021).

[38] Beng Chin Ooi, Gang Chen, Mike Zheng Shou, Kian-Lee Tan, Anthony Tung, Xiaokui Xiao, James Wei Luen Yip, Bingxue Zhang, and Meihui Zhang. 2023. The Metaverse Data Deluge: What Can We Do About It?. In *ICDE*.

[39] Beng Chin Ooi, Kian-Lee Tan, Sheng Wang, Wei Wang, Qingchao Cai, Gang Chen, Jinyang Gao, Zhaojing Luo, Anthony K. H. Tung, Yuan Wang, Zhongle Xie, Meihui Zhang, and Kaiping Zheng. 2015. SINGA: A Distributed Deep Learning Platform. In *Proceedings of the 23rd Annual ACM Conference on Multimedia Conference, MM '15*. 685–688.

[40] Shashi Raj Pandey, Nguyen H Tran, Mehdi Bennis, Yan Kyaw Tun, Zhu Han, and Choong Seon Hong. 2019. Incentivize to build: A crowdsourcing framework for federated learning. In *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 1–6.

[41] UCI Machine Learning Repository. 2017. Adult dataset. In *https://archive.ics.uci.edu/ml/datasets/Adult*.

[42] UCI Machine Learning Repository. 2017. Connect-4 dataset. In *http://archive.ics.uci.edu/ml/datasets/connect-4*.

[43] UCI Machine Learning Repository. 2017. Sensorless drive diagnosis dataset. In *https://archive.ics.uci.edu/ml/datasets/dataset+for+sensorless+drive+diagnosis*.

[44] Nicola Rieke, Jonny Hancox, Wenqi Li, Fausto Milletari, Holger Roth, Shadi Albarqouni, Spyridon Bakas, Mathieu N. Galtier, Bennett A. Landman, Klaus H. Maier-Hein, Sébastien Ourselin, Micah J. Sheller, Ronald M. Summers, Andrew Trask, Daguang Xu, Maximilian Baust, and M. Jorge Cardoso. 2020. The Future of Digital Health with Federated Learning. *CoRR* abs/2003.08119 (2020).

[45] Abhijit Guha Roy, Shayan Siddiqui, Sebastian Pölsterl, Nassir Navab, and Christian Wachinger. 2019. Braintorrent: A peer-to-peer environment for decentralized federated learning. *arXiv preprint arXiv:1905.06731* (2019).

[46] Stefano Savazzi, Monica Nicoli, and Vittorio Rampa. 2020. Federated learning with cooperating devices: A consensus approach for massive IoT networks. *IEEE Internet of Things Journal* 7, 5 (2020), 4641–4654.

[47] Stefano Savazzi, Monica Nicoli, Vittorio Rampa, and Sanaz Kianoush. 2020. Federated Learning with Mutually Cooperating Devices: A Consensus Approach Towards Server-Less Model Optimization. In *ICASSP*. 3937–3941. https://doi.org/10.1109/ICASSP40776.2020.9054055

[48] Rachael Hwee Ling Sim, Yehong Zhang, Mun Choon Chan, and Bryan Kian Hsiang Low. 2020. Collaborative machine learning with incentive-aware model rewards. In *ICML*. PMLR, 8927–8936.

[49] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[50] Gerald Tesauro et al. 1995. Temporal difference learning and TD-Gammon. *Commun. ACM* 38, 3 (1995), 58–68.

[51] Xuezhen Tu, Kun Zhu, Nguyen Cong Luong, Dusit Niyato, Yang Zhang, and Juan Li. 2022. Incentive mechanisms for federated learning: From economic and game theoretic perspective. *IEEE Transactions on Cognitive Communications and Networking* (2022).

[52] Muhammad Habib ur Rehman, Khaled Salah, Ernesto Damiani, and Davor Svetinovic. 2020. Towards blockchain-based reputation-aware federated learning. In *INFOCOM Workshops*. IEEE, 183–188.
[53] Paul Voigt and Axel von dem Bussche. 2017. *The EU General Data Protection Regulation (GDPR): A Practical Guide* (1st ed.). Springer Publishing Company, Incorporated.
[54] Han Wang, Luis Muñoz-González, David Eklund, and Shahid Raza. 2021. Non-IID data re-balancing at IoT edge with peer-to-peer federated learning for anomaly detection. In *WiSec*. 153–163.
[55] Jianyu Wang and Gauri Joshi. 2021. Cooperative sgd: A unified framework for the design and analysis of local-update sgd algorithms. *Journal of Machine Learning Research* 22, 213 (2021), 1–50.
[56] Jiayi Wang, Shiqiang Wang, Rong-Rong Chen, and Mingyue Ji. 2020. Local averaging helps: Hierarchical federated learning and convergence analysis. *arXiv preprint arXiv:2010.12998* (2020).
[57] Jiasi Weng, Jian Weng, Hongwei Huang, Chengjun Cai, and Cong Wang. 2021. Fedserving: A federated prediction serving framework based on incentive mechanism. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 1–10.
[58] Yuncheng Wu, Shaofeng Cai, Xiaokui Xiao, Gang Chen, and Beng Chin Ooi. 2020. Privacy Preserving Vertical Federated Learning for Tree-based Models. *Proc. VLDB Endow.* 13, 11 (2020), 2090–2103.
[59] Jiancheng Yang, Rui Shi, and Bingbing Ni. 2021. MedMNIST Classification Decathlon: A Lightweight AutoML Benchmark for Medical Image Analysis. In *IEEE 18th International Symposium on Biomedical Imaging (ISBI)*. 191–195.
[60] Yaodong Yang, Rui Luo, Minne Li, Ming Zhou, Weinan Zhang, and Jun Wang. 2018. Mean field multi-agent reinforcement learning. In *ICML*. PMLR, 5571–5580.
[61] Zhaohui Yang, Mingzhe Chen, Walid Saad, Choong Seon Hong, Mohammad Shikh-Bahaei, H Vincent Poor, and Shuguang Cui. 2020. Delay minimization for federated learning over wireless communication networks. *arXiv preprint arXiv:2007.03462* (2020).
[62] Han Yu, Zelei Liu, Yang Liu, Tianjian Chen, Mingshu Cong, Xi Weng, Dusit Niyato, and Qiang Yang. 2020. A sustainable incentive scheme for federated learning. *IEEE Intelligent Systems* 35, 4 (2020), 58–69.
[63] Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Nghia Hoang, and Yasaman Khazaeni. 2019. Bayesian nonparametric federated learning of neural networks. In *ICML*. PMLR, 7252–7261.
[64] Rongfei Zeng, Shixun Zhang, Jiaqi Wang, and Xiaowen Chu. 2020. Fmore: An incentive scheme of multi-dimensional auction for federated learning in mec. In *ICDCS*. IEEE, 278–288.
[65] Yufeng Zhan, Peng Li, Zhihao Qu, Deze Zeng, and Song Guo. 2020. A learning-based incentive mechanism for federated learning. *IEEE Internet of Things Journal* 7, 7 (2020), 6360–6368.
[66] Yufeng Zhan, Jiang Zhang, Peng Li, and Yuanqing Xia. 2019. Crowdtraining: Architecture and incentive mechanism for deep learning training in the internet of things. *IEEE Network* 33, 5 (2019), 89–95.
[67] Chenhan Zhang, Shuyu Zhang, JQ James, and Shui Yu. 2021. FASTGNN: A topological information protected federated learning approach for traffic speed forecasting. *IEEE Transactions on Industrial Informatics* 17, 12 (2021), 8464–8474.
[68] Jiale Zhang, Junjun Chen, Di Wu, Bing Chen, and Shui Yu. 2019. Poisoning attack in federated learning using generative adversarial nets. In *TrustCom/BigDataSE*. IEEE, 374–380.
[69] Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. 2021. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of Reinforcement Learning and Control* (2021), 321–384.
[70] Zhebin Zhang, Dajie Dong, Yuhang Ma, Yilong Ying, Dawei Jiang, Ke Chen, Lidan Shou, and Gang Chen. 2021. Refiner: A Reliable Incentive-Driven Federated Learning System Powered by Blockchain. *Proc. VLDB Endow.* 14, 12 (2021), 2659–2662.
[71] Kaiping Zheng, Shaofeng Cai, Horng Ruey Chua, Melanie Herschel, Meihui Zhang, and Beng Chin Ooi. 2022. DyHealth: Making Neural Networks Dynamic for Effective Healthcare Analytics. *Proc. VLDB Endow.* 15, 12 (2022), 3445–3458.
[72] Wenbo Zheng, Lan Yan, Chao Gou, and Fei-Yue Wang. 2020. Federated Meta-Learning for Fraudulent Credit Card Detection. In *IJCAI*. 4654–4660.
[73] Pan Zhou, Qian Lin, Dumitrel Loghin, Beng Chin Ooi, Yuncheng Wu, and Hongfang Yu. 2021. Communication-efficient decentralized machine learning over heterogeneous networks. In *ICDE*. IEEE, 384–395.

Received October 2022; revised January 2023; accepted February 2023