

---

# Gradient-Driven Rewards to Guarantee Fairness in Collaborative Machine Learning

---

Xinyi Xu<sup>1,5\*</sup>, Lingjuan Lyu<sup>2\*</sup>, Xingjun Ma<sup>3</sup>, Chenglin Miao<sup>4</sup>,  
Chuan Sheng Foo<sup>5</sup>, and Bryan Kian Hsiang Low<sup>1</sup>

<sup>1</sup>Department of Computer Science, National University of Singapore, Republic of Singapore

<sup>2</sup>Sony AI, <sup>3</sup>School of Information Technology, Deakin University, Australia

<sup>4</sup>Department of Computer Science, University of Georgia, USA

<sup>5</sup>Institute for Infocomm Research, A\*STAR, Republic of Singapore

<sup>1</sup>{xuxinyi, lowkh}@comp.nus.edu.sg, <sup>2</sup>lingjuan.lv@sony.com

<sup>3</sup>danxjma@gmail.com, <sup>4</sup>cmiao@uga.edu, <sup>5</sup>foo\_chuan\_sheng@i2r.a-star.edu.sg

## Abstract

Collaborative machine learning provides a promising framework for different agents to pool their resources (e.g., data) for a common learning task. In realistic settings where agents are self-interested and not altruistic, they may be unwilling to share data or model without adequate rewards. Furthermore, as the data/model shared by the agents may differ in quality, designing rewards which are fair to them is important so that they do not feel exploited and discouraged from sharing. In this paper, we adopt *federated learning* as a gradient-based formalization of collaborative machine learning, propose a novel cosine gradient Shapley value to evaluate the agents' uploaded model parameter updates/gradients, and design theoretically guaranteed fair rewards in the form of better model performance. Compared to existing baselines, our approach is more efficient and does not require a validation dataset. We perform extensive experiments to demonstrate that our proposed approach achieves better fairness and predictive performance.

## 1 Introduction

In *collaborative machine learning* (CML), multiple agents (e.g., researchers, organizations, companies) pool their resources (e.g., data) together for a common learning task. It spans a wide variety of real-world applications such as digital healthcare [48], clinical trial research [13, 23], wake word detection for smart voice assistants [27], and next word prediction on mobile devices [15].

*Federated learning* (FL) provides one natural formalization of CML [18, 40, 42, 55, 59]. In FL, the agents perform local model training (e.g., via stochastic gradient descent) and share their resulting model parameter updates/gradients [39, 54, 57]. An important distinction of our work here from the common FL literature is that we allow the agents to be self-interested and so, they are not necessarily cooperative like the worker nodes in distributed learning. The implication is that in order to achieve competitive predictive performance for the learning task, it is imperative to incentivize/reward the agents for contributing/sharing information in the form of gradients [46, 47, 51].

Our work here adopts FL as the particular formalization of gradient-based CML to investigate and design a fair reward mechanism. So, the (self-interested) agents who contribute more would not feel exploited, i.e., if these agents are rewarded equally with others who contributed much less. This is often termed as fairness in cooperative game theory [41], mechanism design [4], and computational social choice [11]. To design such a fair reward mechanism, we need to address three main questions.

---

\*Equal contribution.

For the first question, *what is a suitable notion of fairness?* The *Shapley value* (SV) [49] from cooperative game theory is an appealing choice and has been used in ML [14] and FL [53, 54]. However, the effectiveness of existing SV-based methods is limited by the validation process which requires an auxiliary dataset and more time overhead [19, 54]. To address these challenges, we propose to consider the vector alignment between the shared/uploaded gradients by the agents as it circumvents the need for validation and implicitly captures model performance [12]. We therefore propose our SV-based definition using the cosine similarity between gradients called the *cosine gradient Shapley value* (CGSV) which can be efficiently and accurately approximated.

For the subsequent question, *what shall we use as reward?* Various choices such as monetary rewards from a pre-allocated budget [62] or the total revenue generated from FL [9, 10] have been proposed. Unfortunately, though it is natural to consider monetary incentives, these methods may not be readily practical due to the open problem of the denomination between data/gradient and dollar value [1, 45]. Instead, we propose to consider the downloaded gradients from the server as the reward to agents. In this manner, we can avoid the need of external resources and the ensuing practical challenges.

For the last question, *how should rewards be allocated to guarantee fairness?* Using the downloaded gradients as rewards, we propose a mechanism which controls the quality of the gradients that the agents download according to their uploaded model parameter update/gradients via a *sparsifying gradient* trick [2, 7]. This mechanism allows us to provide fairness guarantees in the form of model performance such that the agent who contributes/uploads the best model parameter updates/gradients receives a model with the best performance (i.e., lowest training loss) [51].

In comparison with the related works, our proposed fair gradient reward mechanism has the following practical advantages: (a) Evaluation without an auxiliary dataset: Our fair gradient reward mechanism does *not* require any auxiliary dataset to evaluate the quality of the agents' gradients, as opposed to existing methods [36, 53, 54], hence saving the additional costs from procuring and processing this dataset [51]. (b) Inherent rewards: Our fair gradient reward mechanism does not need external resources and circumvents the challenge of determining the form and denomination between an external resource such as money and gradients/data points [1, 45]. (c) Training-time rewards: Our fair gradient reward mechanism dynamically determines the appropriate rewards (i.e., fair) to agents and realizes them during training instead of post-hoc [47, 51, 62], hence saving post-processing costs.

In summary, the contributions of our work here to FL include the following:

- We propose a novel *cosine gradient Shapley value* (CGSV) to fairly evaluate the quality of uploaded gradients without any auxiliary dataset and present an efficient approximation with bounded error.
- Based on CGSV, we design a novel training-time fair gradient reward mechanism to determine and realize the rewards (i.e., the downloaded gradients by the agents) with theoretical fairness guarantees (i.e., allocate models of lower training losses to agents who upload better gradients).
- We empirically demonstrate the effectiveness of our fair gradient reward mechanism on multiple benchmark datasets in terms of fairness, predictive performance, and time overhead.

## 2 Related Work

**Reward design and choice in CML.** In related fields such as FL [29, 35, 37, 46, 57, 60, 63], Bayesian CML [51], and data sharing [13, 23, 47], designing appropriate rewards to encourage collaboration (e.g. sharing data, gradient, or other information) is a non-trivial practical problem. A useful solution concept should provide a formalization of fairness, a suitable form and denomination of reward, and a principled way to guarantee fairness via a carefully constructed reward design mechanism. Previous approaches have considered monetary rewards from a pre-allocated budget [62, 63] or the total revenue generated from collaboration [9, 10], or simply an abstract and quantifiable form of reward [46, 47]. These methods usually face the practical challenge of determining a suitable denomination between money and data/gradients [1, 45]. The work of [63] has explored a different avenue of using a reverse auction to guarantee truthfulness in its mechanism instead of fairness.

**Fairness notions.** Due to several desirable properties including symmetry and null player, the *Shapley value* (SV) [49] from cooperative game theory is commonly considered as a theoretically principled fairness formalization [4, 11, 41]. Existing SV-based methods consider fairness in the sense of rewarding agents according to their contributions [19, 53, 54]. However, they typically encounter two practical challenges: significant time incurred from computing the SV using validation (i.e.,

further exacerbated by the long training time [14, 19, 54]) and additional cost to procure the auxiliary validation dataset [36, 51]. In contrast, the work of [30] has adopted an egalitarian notion of fairness by aiming to equalize the final individual performance among agents, which is fundamentally different from SV.

Different from the fairness definition in [30], we adopt the fairness notion formalized by SV [14, 19, 51, 53, 54]. A novelty of our approach is in our application of SV: While previous works use the validation accuracy [14, 19, 53, 54], we leverage the cosine similarity between gradient vectors in gradient-based ML models [12]. We thus propose *cosine gradient Shapley value* (CGSV) to fairly measure the quality of the uploaded gradients of the agents. Based on CGSV, we design a gradient reward mechanism with fairness guarantees (Sec. 3.5) and empirically show that it outperforms the adapted variants of some existing approaches in important aspects, including predictive model performance and time overhead (Sec. 4.2).

### 3 Fair Gradient Reward Mechanism

#### 3.1 Vanilla Federated Learning (FL) Problem Setting and Notations

The vanilla FL problem [54, 57] involves a set  $\mathcal{N} := \{i\}_{i=1,\dots,N}$  of  $N$  *honest* agents learning a  $D$ -dimensional vector  $\mathbf{w} \in \mathbb{R}^D$  of model parameters to minimize a loss function  $\mathbf{F}(\mathbf{w})$  that can be additively decomposed into  $N$  local differentiable loss functions  $\mathbf{F}_i(\mathbf{w})$  defined using the local dataset  $\mathcal{D}_i$  of agent  $i \in \mathcal{N}$  and weighted by its importance  $p_i \geq 0$  (e.g., proportional to  $|\mathcal{D}_i|$ ). That is,  $\mathbf{F}(\mathbf{w}) := \sum_{i \in \mathcal{N}} p_i \mathbf{F}_i(\mathbf{w})$  where  $\sum_{i \in \mathcal{N}} p_i = 1$ . We call  $\mathcal{N}$  the grand coalition; a coalition  $\mathcal{S} \subseteq \mathcal{N}$  is then a subset of the grand coalition  $\mathcal{N}$  of  $N$  agents. In iteration  $t = 0$ , every agent  $i \in \mathcal{N}$  starts with the same initialized parameter vector  $\mathbf{w}_{i,0} := \mathbf{w}_0$  as the server. In iteration  $t > 0$ , every agent  $i \in \mathcal{N}$  calculates a parameter update  $\Delta \mathbf{w}_{i,t} := -\eta_t \nabla \mathbf{F}_i(\mathbf{w}_{i,t-1})$  with step size  $\eta_t$  and gradient  $\nabla \mathbf{F}_i(\mathbf{w}_{i,t-1})$  w.r.t. parameter vector  $\mathbf{w}_{i,t-1}$  and uploads it to a *trusted* server who normalizes and aggregates all agents' parameter updates as follows:

$$\mathbf{u}_{i,t} := \Gamma \Delta \mathbf{w}_{i,t} / \|\Delta \mathbf{w}_{i,t}\|, \quad \mathbf{u}_{\mathcal{N},t} := \sum_{i \in \mathcal{N}} r_{i,t-1} \mathbf{u}_{i,t} \quad (1)$$

where  $\Gamma$  is a normalization coefficient used to prevent gradient explosion [32, 44] and the importance coefficient  $r_{i,t-1}$  will be described later in Sec 3.4. So, we call (1) the *gradient aggregation step*. The *gradient download step* then follows where every agent  $i \in \mathcal{N}$  downloads the aggregated parameter update/gradient  $\mathbf{u}_{\mathcal{N},t}$  (1) from the server (as reward) for updating its model parameters  $\mathbf{w}_{i,t} := \mathbf{w}_{i,t-1} + \mathbf{u}_{\mathcal{N},t}$  to the same  $\mathbf{w}_t := \mathbf{w}_{t-1} + \mathbf{u}_{\mathcal{N},t}$  as the server. That is,  $\mathbf{w}_{i,t} = \mathbf{w}_t$  for all  $i \in \mathcal{N}$  and  $t \in \mathbb{Z}^+ \cup \{0\}$ . We define  $\mathbf{u}_{\mathcal{S},t}$  for any coalition  $\mathcal{S} \subseteq \mathcal{N}$  in a similar way as  $\mathbf{u}_{\mathcal{N},t}$  (1). For brevity, we omit  $t$  from our notations in Secs. 3.2 and 3.3 since we only refer to iteration  $t$ .

#### 3.2 Cosine Gradient Shapley Value (CGSV) for Fairness

In the gradient aggregation step (1), the quality/value of coalition  $\mathcal{S}$ 's (normalized) aggregated parameter update/gradient  $\mathbf{u}_{\mathcal{S}}$  can be measured by its *cosine similarity*  $\cos(\mathbf{u}_{\mathcal{S}}, \mathbf{u}_{\mathcal{N}}) := \langle \mathbf{u}_{\mathcal{S}}, \mathbf{u}_{\mathcal{N}} \rangle / (\|\mathbf{u}_{\mathcal{S}}\| \|\mathbf{u}_{\mathcal{N}}\|)$  to the grand coalition  $\mathcal{N}$ 's aggregated parameter update/gradient  $\mathbf{u}_{\mathcal{N}}$  [12, 28, 34]. We use this cosine similarity measure as our *gradient valuation function*  $\nu(\mathcal{S}) := \cos(\mathbf{u}_{\mathcal{S}}, \mathbf{u}_{\mathcal{N}})$ . Intuitively, if the direction of  $\mathbf{u}_{\mathcal{S}}$  aligns more closely with that of  $\mathbf{u}_{\mathcal{N}}$ , then its quality/value  $\nu(\mathcal{S})$  is higher. Using  $\nu$ , the contribution  $\phi_i$  of agent  $i \in \mathcal{N}$  is defined based on the notion of *Shapley value* (SV) [49] which measures its expected marginal contribution when joining the other agents preceding it in any permutation and satisfies certain desirable fairness properties [5], such as null player (i.e., an agent with no marginal contribution has zero SV), symmetry (i.e., agents with identical marginal contributions have equal SVs), among others, as formally discussed in Appendix A.1:

**Definition 1 (Cosine gradient Shapley value (CGSV)).** Let  $\Pi_{\mathcal{N}}$  be a set of all possible permutations of  $\mathcal{N}$  and  $\mathcal{S}_{\pi,i}$  be the coalition of agents preceding agent  $i$  in permutation  $\pi \in \Pi_{\mathcal{N}}$ . The CGSV of agent  $i \in \mathcal{N}$  is defined as

$$\phi_i := (1/N!) \sum_{\pi \in \Pi_{\mathcal{N}}} [\nu(\mathcal{S}_{\pi,i} \cup \{i\}) - \nu(\mathcal{S}_{\pi,i})]. \quad (2)$$

If  $\phi_i$  is negative, then it follows from the weighted sum of parameter updates/gradients in (1) that  $\mathbf{u}_i$  points in an opposite direction to some other parameter updates/gradients and hence has negative cosine similarities to them. In practice, due to the noisy training arising from the use of *stochastic*

gradient descent (SGD) and/or a highly non-convex loss function,  $\phi_i$  may at times be negative even for an honest agent  $i$ . When the number of such cases is limited, training via SGD can still converge to yield a competitive predictive performance, as empirically validated in [12].

### 3.3 Efficient Approximation of CGSV

Since evaluating agent  $i$ 's CGSV  $\phi_i$  (2) exactly incurs  $\mathcal{O}(2^N D)$  time and is thus costly, we propose an efficient approximation by directly measuring the cosine similarity of its (normalized) parameter update/gradient  $\mathbf{u}_i$  to the grand coalition  $\mathcal{N}$ 's aggregated parameter update/gradient  $\mathbf{u}_{\mathcal{N}}$ , which reduces the incurred time by a factor of  $2^N$  and has a bounded error from  $\phi_i$  (Theorem 1):

$$\phi_i \approx \psi_i := \cos(\mathbf{u}_i, \mathbf{u}_{\mathcal{N}}). \quad (3)$$

**Theorem 1 (Approximation Error).** *Let  $I \in \mathbb{R}^+$ . Suppose that  $\|\mathbf{u}_i\| = \Gamma$  and  $|\langle \mathbf{u}_i, \mathbf{u}_{\mathcal{N}} \rangle| \geq 1/I$  for all  $i \in \mathcal{N}$ . Then,  $\phi_i - L_i \psi_i \leq I\Gamma^2$  where the multiplicative factor  $L_i$  can be normalized away.*

Its proof is in Appendix A.2. From Theorem 1, the approximation error is bounded and decreases quadratically with normalization coefficient  $\Gamma$ . However,  $\Gamma$  cannot be reduced to be arbitrarily small, which may cause  $|\langle \mathbf{u}_i, \mathbf{u}_{\mathcal{N}} \rangle| \geq 1/I$  not to hold. It also does not hold when  $\mathbf{u}_i$  is orthogonal to  $\mathbf{u}_{\mathcal{N}}$  or is close to the zero vector, hence implying the quality of that agent  $i$ 's parameter update/gradient is not high enough. So, every agent is encouraged to contribute a parameter update/gradient of sufficiently high quality in order to ensure the quality of the approximation  $\psi_i$  (Theorem 1).

We have performed a simple experiment to compare the quality of our approximation  $\psi_i$  with that of a sampling-based  $(\epsilon, \delta)$ -approximation  $\bar{\phi}_i$  [38], the latter of which is widely used by existing works in data valuation and CML/FL [14, 19, 51, 54]. In this experiment, we have drawn  $N$  random  $D$ -dimensional vectors from a standard multivariate normal distribution to simulate  $\mathbf{u}_1, \dots, \mathbf{u}_N$  and calculated the resulting exact CGSVs  $\phi := (\phi_i)_{i=1, \dots, N}$ , our approximation  $\psi := (\psi_i)_{i=1, \dots, N}$ , and the sampling-based  $(0.1, 0.1)$ -approximation  $\bar{\phi} := (\bar{\phi}_i)_{i=1, \dots, N}$ . Fig. 1 shows the results for  $\ell_1$  error,  $\ell_2$  error, and the incurred time averaged over 10 runs: Our approximation  $\psi$  performs better in all three metrics with varying  $D$  (right figure) and the performance gap widens with an increasing number  $N$  of agents (left figure).

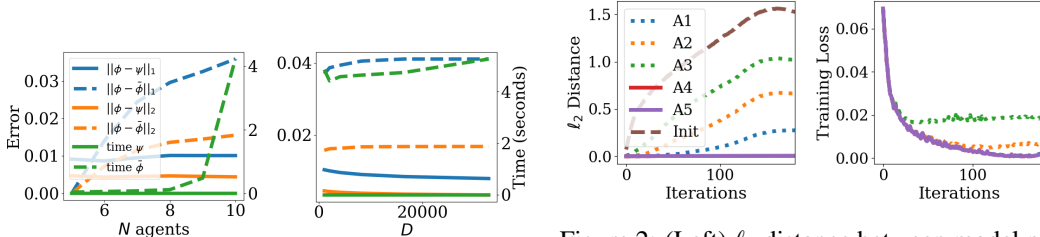


Figure 1: Comparison of  $\ell_1$  error (blue),  $\ell_2$  error (orange), and incurred time (green) (i.e., averaged over 10 runs) between our approximation  $\psi$  (solid lines) vs. a sampling-based approximation  $\bar{\phi}$  (dashed lines) [38] of the exact CGSVs  $\phi$  with (left) varying number  $N$  of agents and  $D = 1024$ , and (right) varying vector dimension  $D$  and  $N = 10$ . For all metrics, lower is better.

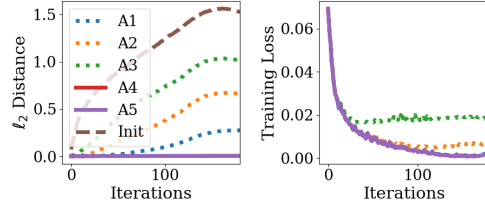


Figure 2: (Left)  $\ell_2$  distance between model parameters of agent  $i = 1, \dots, 5$  (abbreviated to  $A_i$ ) vs. that of the server, and (right) corresponding training loss for an FL problem with  $N = 5$  agents using local MNIST datasets of 600 images each to collaboratively learn 2-layer CNN parameters where the datasets of A1 (blue), A2 (orange), and A3 (green) have 20%, 40%, and 60% randomly corrupted labels, respectively. The brown line denotes  $\ell_2$  distance between  $w_0$  (initialization) vs. server's model parameters.

### 3.4 Server-Side Training-Time Gradient Reward Mechanism

We will now describe the exact details of the gradient aggregation and download steps performed by the server to implement our proposed fair gradient reward mechanism:

**Gradient Aggregation Step.** With a specified normalization coefficient  $\Gamma$  and an initialized coefficient  $r_{i,0}$ , the server performs normalization and aggregation of all agents' parameter updates into

$\mathbf{u}_{\mathcal{N},t}$  using (1), as previously discussed in the FL problem setting (Sec. 3.1). Then, the server computes our approximation  $\psi_{i,t}$  (3) of the CGSV  $\phi_{i,t}$  (2) and updates (and normalizes) the importance coefficient  $r_{i,t}$  in iteration  $t$  via a moving average of  $\psi_{i,t}$  given the relative weight  $\alpha$  on  $r_{i,t-1}$  from previous iteration  $t - 1$ :

$$r_{i,t} := \alpha r_{i,t-1} + (1 - \alpha) \psi_{i,t}, \quad r_{i,t} \leftarrow r_{i,t} / \sum_{i' \in \mathcal{N}} r_{i',t} \quad (4)$$

where  $r_{i,0} := 0$ . Note that  $r_{i,t}$  (4) is used for deriving the sparsified gradient (5) in the gradient download step as well as the aggregation of all agents' parameter updates into  $\mathbf{u}_{\mathcal{N},t+1}$  (1) in iteration  $t + 1$ . The use of a moving average of  $\psi_{i,t}$  to compute  $r_{i,t}$  (4) provides a smoothed estimate without abrupt fluctuations and reduces the effect of noisy training due to the use of SGD in practice [30, 54]. It also allows a flexible weighting over the iterations of the entire training process: In particular, setting  $\alpha < 1$  can effectively mitigate the noise from random initialization of model parameters  $\mathbf{w}_0$  because the weight on  $\psi_{i,t'}$  in earlier iteration  $t' < t$  decays exponentially with  $t$  [53].

**Gradient Download Step.** Recall from the *vanilla* FL problem setting (Sec. 3.1) that in each iteration  $t$ , this step involves all agents downloading an identical aggregated parameter update/gradient  $\mathbf{u}_{\mathcal{N},t}$  (1) from the server (as reward) for updating their model parameters to the same  $\mathbf{w}_t$  (as the server), which is expected to converge to yield a competitive predictive performance [8, 31]. However, such *equal* rewards to all agents is unfair and will discourage any agent from uploading/contributing a parameter update/gradient of higher quality [36, 60] when it can afford to. To ensure fairness, each agent should download some form of aggregated parameter update/gradient as reward that is commensurate to the quality/value of its uploaded/contributed parameter update/gradient. Consequently, an agent who uploads/contributes higher-quality parameter updates/gradients over the entire training process should eventually be rewarded with converged model parameters whose resulting training loss (and hence predictive performance) is closer to that of the server (Theorem 2).

To achieve this, we adopt the trick of *sparsifying*<sup>2</sup> the aggregated parameter update/gradient  $\mathbf{u}_{\mathcal{N},t}$  downloaded by agent  $i$  as reward in each iteration  $t$ . Specifically, we zero out fewer of its smallest components (hence higher-quality gradient reward) when the importance coefficient  $r_{i,t}$  (4) (i.e., moving average of the approximate CGSV  $\psi_{i,t}$ ) is larger:

$$\mathbf{v}_{i,t} := \text{mask}(\mathbf{u}_{\mathcal{N},t}, q_{i,t}), \quad q_{i,t} := \lfloor D \tanh(\beta r_{i,t}) / \max_{i' \in \mathcal{N}} \tanh(\beta r_{i',t}) \rfloor \quad (5)$$

where  $\text{mask}(\mathbf{u}, q)$  retains the largest  $\max(0, q)$  components (in magnitude) of  $\mathbf{u}$  and zeros out all of its other components [2, 58], and  $\beta \geq 1$  specifies the degree of altruism: Greater altruism  $\beta$  gives any agent with a smaller  $r_{i,t}$  a larger improvement in the quality of its gradient reward, i.e., a larger reduction in the sparsity of its downloaded  $\mathbf{v}_{i,t}$  as reward. In the extreme case of  $\beta = \infty$ , we recover the *vanilla* FL problem setting (Sec. 3.1) where all agents are rewarded equally with  $\mathbf{u}_{\mathcal{N},t}$  (i.e., best-quality gradient reward  $\mathbf{v}_{i,t} = \mathbf{u}_{\mathcal{N},t}$  for all  $i \in \mathcal{N}$  with no sparsification), albeit with importance coefficients  $r_{i,t}$  possibly differing across agents  $i \in \mathcal{N}$  and dynamically updated over iteration  $t \in \mathbb{Z}^+$ . Hence, increasing  $\beta$  from 1 to  $\infty$  trades off fairness for equality in gradient rewards by being more altruistic to any agent with a smaller  $r_{i,t}$ ; we empirically show the effect of varying  $\beta$  on training loss in Fig. 7 of Sec. 4.2. Note the agent  $i^* := \arg\max_{i' \in \mathcal{N}} \tanh(\beta r_{i',t})$  with the largest possible  $r_{i^*,t}$  does not benefit from such altruism since it already downloads the best-quality gradient reward (i.e.,  $\mathbf{u}_{\mathcal{N},t}$ ) according to (5).

Suppose that there exists a known threshold  $\underline{r} > 0$  s.t.  $r_{i,t} \geq \underline{r}$  for all  $i \in \mathcal{N}$  and  $t \in \mathbb{Z}^+$  and we want to limit the sparsity of any downloaded  $\mathbf{v}_{i,t}$  or, equivalently, ensure the minimum quality of any gradient reward: Specifically, given a predefined threshold  $c \in (0, 1]$ , we want to guarantee  $q_{i,t} \geq \lfloor D \times c \rfloor$  holds for all  $i \in \mathcal{N}$  and  $t \in \mathbb{Z}^+$ . By setting  $\beta$  s.t.  $\tanh(\beta \underline{r}) \geq c$ , it follows from (5) and  $\max_{i' \in \mathcal{N}} \tanh(\beta r_{i',t}) \leq 1$  that  $\tanh(\beta r_{i,t}) / \max_{i' \in \mathcal{N}} \tanh(\beta r_{i',t}) \geq \tanh(\beta r_{i,t}) \geq \tanh(\beta \underline{r}) \geq c$  and hence  $q_{i,t} \geq \lfloor D \times c \rfloor$  ensues. By using the property that  $\tanh(\beta \underline{r}) = (\exp(2\beta \underline{r}) - 1) / (\exp(2\beta \underline{r}) + 1)$ ,  $\beta \geq \ln((1 + c) / (1 - c)) / (2\underline{r})$  can be derived and used for setting  $\beta$ . It further informs us that reducing the sparsity of any downloaded  $\mathbf{v}_{i,t}$  or, equivalently, improving the minimum quality of any gradient reward (i.e., by increasing  $c$ ) requires greater altruism  $\beta$  to be introduced, while improving the minimum quality of uploaded/contributed parameter updates/gradients by any agent over the entire training process (hence larger  $\underline{r}$ ) eases the need of introducing greater altruism  $\beta$ .

To see why the sparsifying gradient trick (5) can ensure fairness, we illustrate its effect in an FL problem with  $N = 5$  agents using local MNIST datasets of 600 images each to collaboratively learn

<sup>2</sup>Sparsifying a parameter update/gradient vector means zeroing out some of its components and leaving the others unchanged [7, 32].

the parameters of a 2-layer *convolutional neural network* (CNN) where the datasets of agents 1, 2, and 3 have 20%, 40%, and 60% randomly corrupted labels, respectively. The uploaded/contributed parameter updates/gradients thus decrease in quality from agents 1 to 3 (i.e.,  $\psi_{1,t} = 0.194$ ,  $\psi_{2,t} = 0.088$ , and  $\psi_{3,t} = 0.043$  on average) due to increasingly noisy labels in their datasets, while agents 4 and 5 upload/contribute parameter updates/gradients of high quality (i.e.,  $\psi_{4,t} = 0.331$  and  $\psi_{5,t} = 0.342$  on average) due to uncorrupted labels in their datasets. Consequently, agents 1 to 3 have increasing sparsity (resp., 34.9%, 67.6%, and 83.0% on average) while agents 4 and 5 have little/no sparsity (resp., 3.5% and 1.1% on average) in their downloaded  $\mathbf{v}_{i,t}$  as rewards ( $\beta = 1$ ). Fig. 2 shows that the converged model parameters of agents 1 to 3 grow in  $\ell_2$  distance from that of the server (hence increasing training loss) while agents 4 and 5 have the closest converged model parameters (hence lowest training loss).

We provide the pseudocodes performed by the server and agent  $i \in \mathcal{N}$  in each iteration  $t$  below. We will discuss in Sec 4.2 how to set the hyperparameters  $\Gamma$ ,  $\alpha$ , and  $\beta$ , in (1), (4), and (5) respectively in our experiments.

---

Server ( $t$ )

---

1: **for all**  $i \in \mathcal{N}$  **do**  
2:   Download  $\Delta \mathbf{w}_{i,t}$  from agent  $i$   
3:   ▷ **Gradient Aggregation Step**  
4:   Compute  $\mathbf{u}_{i,t}$  and  $\mathbf{u}_{\mathcal{N},t}$  (1)  
5:   **for all**  $i \in \mathcal{N}$  **do**  
6:     Compute  $\psi_{i,t}$  (3) and  $r_{i,t}$  (4)  
7:     ▷ **Gradient Download Step**  
8:     **for all**  $i \in \mathcal{N}$  **do**  
9:       Compute  $\mathbf{v}_{i,t}$  (5) for download by agent  $i$

---



---

Agent ( $i, t$ )

---

1: Upload  $\Delta \mathbf{w}_{i,t} = -\eta_t \nabla \mathbf{F}_i(\mathbf{w}_{i,t-1})$  to server  
2: Download  $\mathbf{v}_{i,t}$  from server  
3: Update  $\mathbf{w}_{i,t} = \mathbf{w}_{i,t-1} + \mathbf{v}_{i,t}$

---

### 3.5 Fairness Guarantee

We have previously discussed the intuition underlying our notion of fairness in Sec. 3.4 that an agent who uploads/contributes higher-quality parameter updates/gradients over the entire training process should eventually be rewarded with converged model parameters whose resulting training loss (and hence predictive performance) is closer to that of the server. Note that the importance coefficient  $r_{i,t}$  (4) measures the overall quality of the parameter updates/gradients uploaded/contributed by agent  $i$  over the entire training process till iteration  $t$ . Our main result below guarantees a notion of fairness that under some conditions on loss function  $\mathbf{F}$  and the server’s model parameters  $\mathbf{w}_t$ , if an agent  $i$  has a larger importance coefficient  $r_{i,t}$  and model parameters  $\mathbf{w}_{i,t-1}$  closer to that of the server (i.e.,  $\mathbf{w}_{t-1}$ ) than another agent by at least  $2\|\mathbf{v}_{i,t}\|$  in previous iteration  $t - 1$ , then it is rewarded with model parameters  $\mathbf{w}_{i,t}$  incurring smaller training loss  $\mathbf{F}(\mathbf{w}_{i,t})$  in iteration  $t$ :

**Theorem 2 (Fairness in Training Loss).** *Let  $\delta_{i,t} := \|\mathbf{w}_t - \mathbf{w}_{i,t}\|$ . Suppose that  $\mathbf{w}_t$  is near to a stationary point of  $\mathbf{F}$  for  $t \geq t^* \in \mathbb{Z}^+$  and some regularity conditions on  $\mathbf{F}$  hold. For all  $i, i' \in \mathcal{N}$  and  $t \geq t^*$ , if  $r_{i,t} \geq r_{i',t}$  and  $\delta_{i',t-1} - \delta_{i,t-1} \geq 2\|\mathbf{v}_{i,t}\|$ , then  $\mathbf{F}(\mathbf{w}_{i,t}) \leq \mathbf{F}(\mathbf{w}_{i',t})$ .*

Its proof is in Appendix A.3. Our experiments in Appendix B.3 will empirically verify the fairness guarantee in Theorem 2 (and fairness in test accuracy) without needing to impose its conditions.

## 4 Experiments and Discussion

### 4.1 Experimental Settings

**Datasets.** We perform extensive experiments on image classification datasets like MNIST [26] and CIFAR-10 [21] and text classification datasets like *movie review* (MR) [43] and *Stanford sentiment treebank* (SST) [20]. We use a 2-layer *convolutional neural network* (CNN) for MNIST [25], a 3-layer CNN for CIFAR-10 [22], and a text embedding CNN for MR and SST [20].

**Baselines.** We consider several existing FL baselines such as FedAvg [39],  $q$ -FFL [30], CFFL [36], and an *extended contribution index* (ECI) method from [53] utilizing validation accuracy-based SV and setting  $q_{i,t}$  for  $i \in \mathcal{N}$  in (5) to be proportional to the agents’ CIs. CFFL also utilizes the validation accuracy but is more efficient by using a leave-one-out approach instead of SV, while  $q$ -FFL aims at

Table 1: Average test accuracy (%) achieved by the agents collaborating via our fair gradient reward mechanism with varying degrees of altruism  $\beta$  vs. tested baselines on all datasets. Each value in brackets denotes the highest test accuracy achieved by any agent.

No. Agents	MNIST						CIFAR-10			MR	SST
	10			20			10			5	5
	UNI	POW	CLA	UNI	POW	CLA	UNI	POW	CLA	POW	POW
Standalone	91 (91)	88 (92)	53 (92)	91 (91)	89 (92)	48 (90)	46 (47)	43 (49)	31 (44)	47(56)	31(34)
FedAvg	93 (94)	92 (94)	53 (93)	93 (93)	92 (94)	49 (92)	48 (48)	47 (50)	32 (47)	51(63)	33(35)
q-FFL	85 (91)	27 (45)	44 (64)	88 (91)	48 (53)	40 (59)	41 (46)	36 (36)	22 (28)	12(18)	23(25)
CFFL	90 (92)	85 (90)	34 (44)	91 (93)	88 (91)	39 (46)	39 (41)	35 (45)	22 (40)	44(53)	31(32)
ECI	94 (94)	92 (94)	53 (94)	94 (94)	92 (94)	49 (92)	49 (49)	47 (51)	31 (46)	56(61)	33(34)
DW	93 (94)	92 (94)	53 (93)	93 (93)	92 (94)	49 (92)	48 (48)	47 (50)	32 (47)	51(62)	33(35)
RR	94 (95)	<b>95 (95)</b>	64 (72)	94 (95)	94 (95)	50 (56)	47 (59)	49 (51)	26 (29)	<b>63(65)</b>	<b>36(36)</b>
Ours (EU)	94 (94)	94 (94)	54 (94)	94 (94)	94 (94)	49 (92)	49 (49)	49 (51)	32 (46)	54(59)	34(36)
Ours ( $\beta = 1$ )	96 (97)	94 (95)	74 ( <b>95</b> )	95 (96)	<b>96 (97)</b>	65 (93)	61 ( <b>62</b> )	60 ( <b>62</b> )	35 ( <b>54</b> )	62( <b>76</b> )	35(36)
Ours ( $\beta = 1.2$ )	94 (95)	<b>95 (95)</b>	<b>75 (95)</b>	96 (96)	<b>96 (97)</b>	65 (93)	61 ( <b>62</b> )	60 ( <b>62</b> )	35 ( <b>54</b> )	62(75)	34( <b>37</b> )
Ours ( $\beta = 1.5$ )	<b>97 (97)</b>	<b>95 (95)</b>	<b>75 (95)</b>	<b>96 (97)</b>	94 (95)	65 (93)	61 ( <b>62</b> )	59 ( <b>62</b> )	35 ( <b>54</b> )	62(74)	35( <b>37</b> )
Ours ( $\beta = 2$ )	96 (96)	<b>95 (96)</b>	73 (94)	<b>97 (97)</b>	95 (96)	<b>66 (95)</b>	<b>62 (62)</b>	<b>61 (62)</b>	<b>36 (54)</b>	62(75)	35(37)

achieving egalitarian fairness by equalizing the local training losses of the agents. Furthermore, we implement simple FL baselines based on *round robin* (RR), *dataset weighted download* (DW), and *Euclidean distance* (EU). RR is commonly adopted in mechanism design to ensure fairness [6, 33] and also used in FL to schedule gradient downloads [50, 64]. For DW (EU),  $q_{i,t}$  for  $i \in \mathcal{N}$  in (5) are set to be proportional to the agents’ local dataset sizes (negative Euclidean distance of their unnormalized parameter updates from that of the server). We also include *standalone* agents as a baseline, i.e., each agent trains its CNN using only its local dataset without involving FL.

**Performance Metrics.** To measure fairness, we consider the *scaled Pearson correlation coefficient*<sup>3</sup>  $\rho := 100 \times \text{pearsonr}(\varphi, \xi) \in [-100, 100]$  between the test accuracies  $\varphi$  achieved by the agents when standalone [36] vs. that  $\xi$  achieved by them when collaborating via a gradient reward mechanism in FL after the entire training process has ended at iteration  $t = T$ . The corresponding experimental results will be reported in Sec. 4.2. To empirically verify the fairness guarantee in Theorem 2, we have also reported in Appendix B.3 results on the fairness metric  $\rho$  between the importance coefficients  $\varphi := (r_{i,T})_{i=1,\dots,N}$  (4) (i.e., measuring overall qualities of the parameter updates/gradients uploaded/contributed by the agents) vs. test accuracies (or negative training losses)  $\xi$  achieved by them. We consider other performance metrics like predictive performance (i.e., average and highest test accuracies achieved by the agents) and time overhead of the tested gradient reward mechanisms.

**Data Partitions among Agents.** We carefully construct two heterogeneous data partitions by varying the agents’ local dataset sizes and corresponding numbers of distinct classes. For **imbalanced dataset sizes** (POW), we follow a power law to partition the entire dataset among the agents. For MNIST, we partition the entire dataset of size  $\{3000, 6000, 12000\}$ , respectively, among  $\{5, 10, 20\}$  agents s.t. each agent has a randomly sampled local dataset of size 600 on average [39]. The size of the local dataset increases from the first to the last agent. Since the local dataset sizes vary significantly (i.e., superlinearly) among the agents, the agents with larger local datasets are expected to achieve better predictive performance. For **imbalanced class numbers** (CLA), we vary the number of distinct classes in the local datasets of the agents, while keeping their sizes fixed at 600. For this setting, we only consider MNIST and CIFAR-10 datasets and partition classes in a “linspace” manner as both contain 10 classes. To illustrate, for MNIST with 5 agents, agents 1, 2, 3, 4, 5 own local datasets with 1, 3, 5, 7, 10 classes, respectively; so, agent 1 (5) has a local dataset with 1 (10) class(es). Similarly, the agents with local datasets containing more classes are expected to achieve better predictive performance. We also include the simplest setting of the uniform/homogeneous data partition (UNI) where the agents are expected to achieve comparable predictive performance.

Additional details of the experimental settings are described in Appendix B.1.

## 4.2 Experimental Results

**Predictive Performance.** Table 1 shows results of the average and highest test accuracies achieved by the agents collaborating via our fair gradient reward mechanism vs. tested baselines on all datasets. Our fair gradient reward mechanism generally outperforms the tested baselines on both metrics, especially for heterogeneous data partitions and on the MR dataset. On MNIST, for the

<sup>3</sup>The Pearson correlation coefficient has been applied to a similar use case in [19].

Table 2: Fairness metric  $\rho \in [-100, 100]$  achieved by our fair gradient reward mechanism with varying degrees of altruism  $\beta$  vs. tested baselines on all datasets. Higher value means greater fairness.

No. Agents	MNIST						CIFAR-10			MR	SST
	10			20			10			5	5
	UNI	POW	CLA	UNI	POW	CLA	UNI	POW	CLA	POW	POW
FedAvg	-45.60	55.24	24.12	0.85	-32.58	40.83	18.47	97.48	98.75	48.68	57.50
q-FFL	-44.73	39.00	22.38	-22.01	38.71	48.07	-17.64	51.33	94.06	56.43	-75.92
CFFL	83.57	91.80	81.24	82.52	94.70	85.71	78.25	72.55	81.31	96.85	93.34
ECI	85.26	<b>99.83</b>	<b>99.98</b>	80.95	<b>99.41</b>	95.21	75.85	79.50	99.55	97.69	95.00
DW	89.15	98.93	65.34	86.94	99.63	35.21	-23.14	91.97	45.45	<b>99.20</b>	97.12
RR	83.77	71.17	-26.75	-18.64	25.47	95.86	30.67	0.70	90.67	44.16	-25.11
Ours (EU)	84.25	98.25	99.82	80.55	97.77	<b>99.97</b>	78.25	94.24	94.95	97.58	93.21
Ours ( $\beta = 1$ )	94.03	95.74	94.54	84.47	96.39	97.23	<b>98.80</b>	<b>98.78</b>	<b>99.89</b>	96.01	98.20
Ours ( $\beta = 1.2$ )	94.75	97.28	96.23	90.52	97.72	95.21	91.07	91.59	99.82	96.12	<b>98.47</b>
Ours ( $\beta = 1.5$ )	<b>96.34</b>	86.99	95.37	82.68	90.94	98.75	93.55	93.78	95.89	95.32	97.88
Ours ( $\beta = 2$ )	94.66	91.20	95.38	<b>96.90</b>	91.33	94.32	89.80	88.78	93.39	92.22	95.74

CLA data partition among 10 agents, our fair gradient reward mechanism achieves average (highest) test accuracy of 75% (95%) at  $\beta = 1.5$ , while the best-performing ECI baseline achieves only that of 53% (94%). On CIFAR-10, for the CLA data partition among 10 agents, our fair gradient reward mechanism achieves average (highest) test accuracy of 36% (54%) at  $\beta = 2$ , while the best-performing DW baseline achieves only that of 32% (47%). On the MR dataset, our fair gradient reward mechanism achieves average (highest) test accuracy of 62% (76%) at  $\beta = 1$ , while the best-performing RR baseline achieves that of 63% (65%). Its better performance may be attributed to the adaptive re-weighting in the gradient aggregation step (1) via  $r_{i,t}$ , which can dynamically account for the heterogeneity in the agents’ local datasets [31]. While EU performs comparably to both FedAvg and ECI (i.e., difference in average test accuracies between EU vs. FedAvg/ECI is less than 3%), it does not perform better than our fair gradient reward mechanism (e.g., on MNIST, for the CLA data partition among 10 agents, the difference in average test accuracies between EU vs. our fair gradient reward mechanism at  $\beta = 1.5$  is more than 20%) because unlike cosine similarity, Euclidean distance fails to capture the directional difference between gradients, which is important since the negative gradients are pointing in the direction of lower loss. Importantly,  $q$ -FFL aims to equalize the local training losses w.r.t. the agent’s local datasets, which may be suboptimal for heterogeneous data partitions like POW and CLA. We provide further results in Appendix B.5 empirically comparing the predictive performances of our fair gradient reward mechanism vs.  $q$ -FFL.

**Fairness.** To empirically verify the fairness guarantee in Theorem 2, Table 2 shows results on the fairness metric  $\rho$  achieved by our fair gradient reward mechanism vs. tested baselines on all datasets. From Table 2, our fair gradient reward mechanism achieves a high degree of fairness of above 80, while the commonly used FedAvg performs suboptimally s.t. it produces the lowest degree of fairness of  $-45.6$ . On MNIST, for the POW data partition among 10/20 agents and the CLA data partition among 10 agents, ECI outperforms our fair gradient reward mechanism, albeit at a much higher time overhead of over 100 times and with additional information from an auxiliary dataset. CFFL underperforms our fair gradient reward mechanism and ECI as it adopts the leave-one-out approach which seems less accurate than SV in valuing the contributions of the agents [19]. Both  $q$ -FFL and RR promote egalitarian fairness instead of our notion of fairness via SV and hence do not perform optimally. DW achieves high degrees of fairness only for the POW data partition because it uses the agents’ local dataset sizes to determine their gradient rewards. Fig. 3 illustrates an intuitive trend of the predictive performances achieved by 10 agents collaborating via our fair gradient reward mechanism for homogeneous and heterogeneous data partitions among the agents on MNIST and CIFAR-10: For the UNI data partition, all agents achieve comparable predictive performance. Their predictive performances vary more (most) for the POW (CLA) data partition, hence demonstrating that our fair gradient reward mechanism can distinguish the contributions of the agents and reward them with sparsified gradients fairly.

We have performed an additional experiment to understand our fair gradient reward mechanism for homogeneous and heterogeneous data partitions among 3 agents on MNIST and CIFAR-10 where for POW and CLA, agent 1 (3) uploads/contributes parameter updates/gradients of lowest (highest) quality over the entire training process. Fig. 4 shows how  $r_{i,t}$  for agent  $i = 1, 3$  varies over iterations  $t$ : Interestingly, for the CLA data partition, though agent 3 (brown solid line) is initially mistaken to provide a low contribution, the dynamic update of  $r_{3,t}$  (4) allows its true contribution to be recognized quickly. Fig. 5 (Fig. 6) shows how the  $\ell_2$  distance between the downloaded sparsified gradient  $v_{i,t}$  (5)



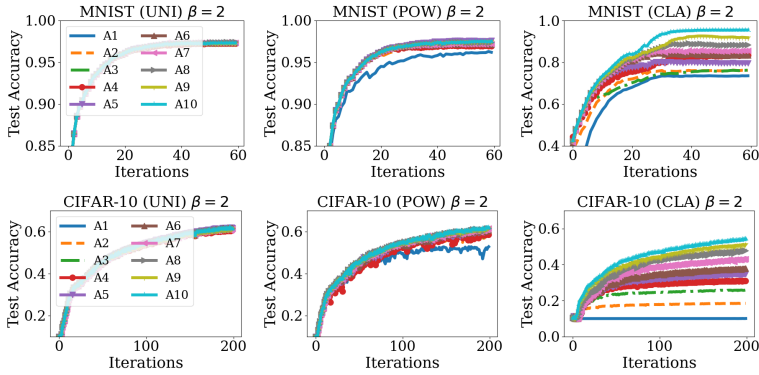


Figure 3: Test accuracy achieved by agent  $i = 1, \dots, 10$  (abbreviated to  $A_i$ ) collaborating via our fair gradient reward mechanism at  $\beta = 2$  for the UNI (left), POW (middle), and CLA (right) data partitions among the 10 agents on MNIST (top) and CIFAR-10 (bottom). Their predictive performances vary least, more, and most for the respective UNI, POW, and CLA data partitions.

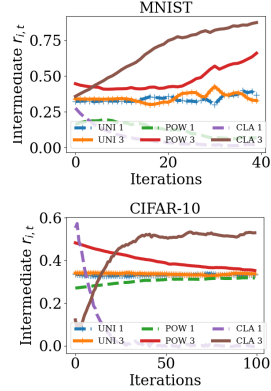


Figure 4: Graphs of  $r_{i,t}$  (4) for agent  $i = 1, 3$  vs. iteration  $t$  for UNI, POW, and CLA data partitions among 3 agents on MNIST and CIFAR-10.

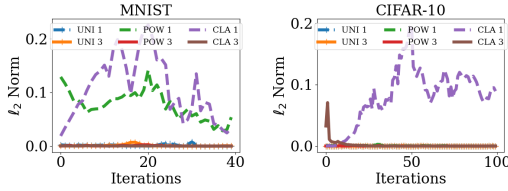


Figure 5: Graphs of  $\ell_2$  distance between down-loaded  $v_{i,t}$  (5) of agent  $i = 1, 3$  and aggregated layer’s model parameters of agent  $i = 1, 3$   $u_{\mathcal{N},t}$  (1) vs. iteration  $t$  for UNI, POW, and CLA data partitions among 3 agents on MNIST (left) and CIFAR-10 (right).

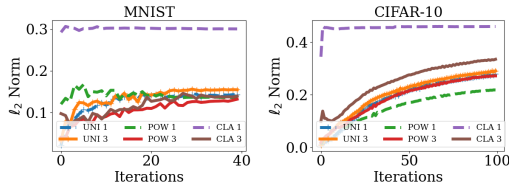


Figure 6: Graphs of  $\ell_2$  distance between last layer’s model parameters of agent  $i = 1, 3$  and that of the server vs. iteration  $t$  for UNI, POW, and CLA data partitions among 3 agents on MNIST (left) and CIFAR-10 (right).

of agent  $i = 1, 3$  and aggregated parameter update/gradient  $u_{\mathcal{N},t}$  (1) (last layer’s model parameters of agent  $i = 1, 3$  and that of the server) varies over iterations  $t$ : In particular, for the CLA data partition, agent  $i = 1$  ( $i = 3$ ) who uploads/contributes parameter updates/gradients of lowest (highest) quality over the entire training process downloads  $v_{i,t}$  as reward that is further from (closer to)  $u_{\mathcal{N},t}$ , hence training last layer’s model parameters to be further from (closer to) that of the server. Such results further validate that in Fig. 2 previously.

Lastly, Fig. 7 confirms that for the CLA data partition among 10 agents on MNIST, increasing the degree of altruism  $\beta$  leads to all agents downloading higher-quality gradient rewards  $v_{i,t}$  (5) and thus incurring smaller training loss. In particular, agent 1 (abbreviated to A1 and represented by a blue solid line) who uploads/contributes parameter updates/gradients of lowest quality over the entire training process benefits most as  $\beta$  increases, as explained previously in Sec. 3.4. Additional results w.r.t. test loss are reported in Appendix B.4.

**Time Overhead.** Table 3 compares the time overhead (seconds) of our fair gradient reward mechanism vs. tested baselines on all datasets; the ratio between the time overhead vs. training time is given in brackets. Our fair gradient reward mechanism is much more efficient than ECI and CFFL which also consistently achieve fairness. In particular, our fair gradient reward mechanism incurs a small time overhead of at most  $0.4 \times$  of the training time, while ECI incurs a significant time overhead of up to  $140 \times$  of the training time due to the calculation of the CI incurring  $\mathcal{O}(2^N)$  time, even with the permutation sampling-based approximation [38, 54] for 10/20 agents. CFFL incurs at most  $2 \times$  of the training time (i.e., 5-6 times longer than ours) from the additional validation in each iteration.

**Hyperparameters.** We find that  $\alpha \in [0.8, 1]$  (i.e., relative weight on  $r_{i,t-1}$  in (4)),  $\beta \in [1, 2]$  (i.e., degree of altruism in (5)) and  $\Gamma \in [0.1, 1]$  (i.e., normalization coefficient in (1)) are effective in

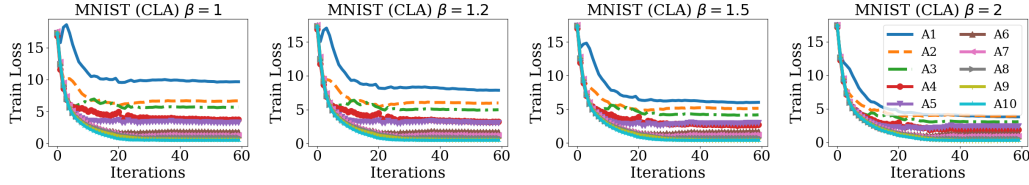


Figure 7: Training losses incurred by agent  $i = 1, \dots, 10$  (abbreviated to  $A_i$ ) collaborating via our fair gradient reward mechanism with varying degrees of altruism  $\beta = 1.0, 1.2, 1.5, 2$  for the CLA data partition on MNIST.

Table 3: Time overhead (seconds) of our fair gradient reward mechanism vs. tested baselines on all datasets. Each value in brackets denotes the ratio between the time overhead vs. training time.

No. Agents	MNIST			CIFAR-10		MR	SST
	5	10	20	5	10	5	5
FedAvg	1.17 (7e-3)	1.05 (1e-2)	4.29 (1e-2)	1.66 (7e-3)	7.41 (1e-2)	1.3 (1e-4)	1.31 (6e-4)
q-FFL	6.14 (4e-2)	4.97 (5e-2)	91.20 (0.3)	97.28 (0.4)	58.94 (7e-2)	90.01 (8e-3)	82.85 (4e-2)
CFFL	32.15 (0.2)	21.79 (0.3)	500.03 (1.6)	570.12 (2.0)	302.44 (0.4)	479.12 (0.2)	487.71 (2e-1)
ECI	2377.33 (16)	11937.80 (141)	23749.06 (74)	3571.75 (15)	58835.83 (84)	422.85 (4e-2)	801.20 (0.4)
DW	<b>0.89 (6e-3)</b>	<b>0.79 (9e-3)</b>	<b>1.60 (5e-3)</b>	<b>1.21 (5e-3)</b>	<b>5.29 (7e-3)</b>	<b>0.99 (1e-5)</b>	0.98 (5e-4)
RR	<b>0.89 (6e-3)</b>	0.82 (9e-3)	<b>1.60 (5e-3)</b>	3.31 (1e-2)	5.41 (7e-3)	1.01 (5e-4)	<b>0.99 (5e-4)</b>
Ours (EU)	<b>0.89 (6e-3)</b>	0.81 (9e-3)	1.61 (5e-3)	1.22 (5e-3)	5.33 (7e-3)	1.01 (5e-4)	<b>0.99 (5e-4)</b>
Ours (Cosine)	6.34 (4e-2)	4.94 (5e-2)	94.30 (0.3)	98.39 (0.4)	54.94 (7e-2)	89.81 (8e-3)	82.87 (4e-2)

achieving competitive predictive performance and fairness. In our experiments, we set  $\alpha = 0.95$ ,  $\beta = [1, 1.2, 1.5, 2]$ , and  $\Gamma = 0.5$  for MNIST,  $\Gamma = 0.15$  for CIFAR-10, and  $\Gamma = 1$  for SST and MR.

## 5 Conclusion and Future Work

We introduce a novel formulation called *cosine gradient Shapley value* (CGSV) to fairly evaluate the quality/value of the uploaded/contributed model parameter updates/gradients by the agents in *federated learning* (FL)/gradient-based *collaborative machine learning* (CML) and use it to design their corresponding rewards in the form of downloaded gradients. Our approach ensures that the agents who upload better gradients can download better gradients, which in turn leads to better local models with lower training losses. We theoretically and empirically demonstrate the effectiveness of our approach on both predictive performance and fairness. Moreover, our approach is much more efficient than existing baselines and non-restrictive, i.e., it requires only slight calculations by the server and does not require an auxiliary dataset.

Our approach provides high flexibility for the trade-off between equitable and strictly fair rewards via a hyperparameter  $\beta$  controlling the degree of altruism. Interestingly, a higher altruism degree can sometimes lead to better predictive performance, which naturally raises the following question: Can we achieve both optimally or is there some inescapable trade-off between fairness and performance? For future work, it would be interesting to consider the notion of fairness when there are some adversaries. We would also consider generalizing our work and fairness guarantee to other forms of CML (e.g., model fusion [16, 17, 24]) and collaborative Bayesian optimization [52].

## Acknowledgments and Disclosure of Funding

This research is supported by the National Research Foundation, Singapore under its AI Singapore Programme (Award No: AISG2-RP-2020-018). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore. Xinyi Xu is supported by the Institute for Infocomm Research of Agency for Science, Technology and Research (A\*STAR).

## References

- [1] A. Agarwal, M. Dahleh, and T. Sarkar. A marketplace for data: An algorithmic solution. In *Proc. EC*, 2019.

- [2] D. Alistarh, T. Hoeffler, M. Johansson, S. Khirirat, N. Konstantinov, and C. Renggli. The convergence of sparsified gradient methods. In *Proc. NeurIPS*, 2018.
- [3] M. Bahir, M. And, B. Peleg, M. Maschler, and B. Peleg. A characterization, existence proof and dimension bounds for the kernel of a game. *Pacific Journal of Mathematics*, 18(2), 1966.
- [4] E. Balkanski and Y. Singer. Mechanisms for fair attribution. In *Proc. EC*, 2015.
- [5] G. Chalkiadakis, E. Elkind, and M. Wooldridge. Computational aspects of cooperative game theory. In R. J. Brachman, W. W. Cohen, and T. G. Dietterich, editors, *Synthesis Lectures on Artificial Intelligence and Machine Learning*. Morgan & Claypool Publishers, 2011.
- [6] C. Chen, W. Wang, and B. Li. Round-robin synchronization: Mitigating communication bottlenecks in parameter servers. In *Proc. IEEE INFOCOM*, 2019.
- [7] C. Y. Chen, J. Choi, D. Brand, A. Agrawal, W. Zhang, and K. Gopalakrishnan. ADaComp: Adaptive residual gradient compression for data-parallel distributed training. In *Proc. AAAI*, 2018.
- [8] M. Chen, B. Mao, and T. Ma. Fedsta: A staleness-aware asynchronous federated learning algorithm with non-iid data. *Future Generation Computer Systems*, 120:1–12, 2021.
- [9] Z. Chen, Z. Liu, K. L. Ng, H. Yu, Y. Liu, and Q. Yang. A gamified research tool for incentive mechanism design in federated learning. In Q. Yang, L. Fan, and H. Yu, editors, *Federated Learning*, volume 12500 of *Lecture Notes in Computer Science*, pages 168–175. Springer, Cham, 2020.
- [10] M. Cong, H. Yu, X. Weng, and S. Yiu. A game-theoretic framework for incentive mechanism design in federated learning. In Q. Yang, L. Fan, and H. Yu, editors, *Federated Learning*, volume 12500 of *Lecture Notes in Computer Science*, pages 205–222. Springer, Cham, 2020.
- [11] V. Conitzer and T. Sandholm. Computing shapley values, manipulating value division schemes, and checking core membership in multi-issue domains. In *Proc. AAAI*, 2004.
- [12] W. Dai, Y. Zhou, N. Dong, H. Zhang, and Eric P. Xing. Toward understanding the impact of staleness in distributed machine learning. In *Proc. ICLR*, 2019.
- [13] J. M. Drazen, S. Morrissey, D. Malina, M. B. Hamel, and E. W. Champion. The importance — and the complexities — of data sharing. *New England Journal of Medicine*, 375(12):1182–1183, 2016.
- [14] A. Ghorbani and J. Y. Zou. Data Shapley: Equitable valuation of data for machine learning. In *Proc. ICML*, 2019.
- [15] A. Hard, K. Rao, R. Mathews, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage. Federated learning for mobile keyboard prediction. arXiv:1811.03604, 2018.
- [16] Q. M. Hoang, T. N. Hoang, B. K. H. Low, and C. Kingsford. Collective model fusion for multiple black-box experts. In *Proc. ICML*, pages 2742–2750, 2019.
- [17] T. N. Hoang, C. T. Lam, B. K. H. Low, and P. Jaillet. Learning task-agnostic embedding of multiple black-box experts for multi-task model fusion. In *Proc. ICML*, pages 4282–4292, 2020.
- [18] Y. Hu, D. Niu, J. Yang, and S. Zhou. Fdml: A collaborative machine learning framework for distributed features. In *Proc. SIGKDD*, page 2232–2240, 2019.
- [19] R. Jia, D. Dao, B. Wang, F. A. Hubis, N. Hynes, N. M. Gürel, B. Li, C. Zhang, D. Song, and C. J. Spanos. Towards efficient data valuation based on the Shapley value. In *Proc. AISTATS*, 2019.
- [20] Y. Kim. Convolutional neural networks for sentence classification. In *Proc. EMNLP*, 2014.
- [21] A. Krizhevsky. Learning multiple layers of features from tiny images. Master’s thesis, Department of Computer Science, University of Toronto, 2009.

- [22] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *Proc. NeurIPS*, 2012.
- [23] H. M. Krumholz and J. Waldstreicher. Toward fairness in data sharing. *New England Journal of Medicine*, 375(5):405–407, 2016.
- [24] C. T. Lam, T. N. Hoang, B. K. H. Low, and P. Jaillet. Model fusion for personalized learning. In *Proc. ICML*, pages 5948–5958, 2021.
- [25] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel. Hand-written digit recognition with a back-propagation network. In *Proc. NeurIPS*, 1990.
- [26] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.
- [27] D. Leroy, A. Coucke, T. Lavril, T. Gisselbrecht, and J. Dureau. Federated learning for keyword spotting. In *Proc. ICASSP*, 2019.
- [28] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein. Visualizing the loss landscape of neural nets. In *Proc. NeurIPS*, 2018.
- [29] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.
- [30] T. Li, M. Sanjabi, A. Beirami, and V. Smith. Fair resource allocation in federated learning. In *Proc. ICLR*, 2020.
- [31] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang. On the convergence of FedAvg on non-iid data. In *Proc. ICLR*, 2020.
- [32] Y. Lin, Y. Wang, S. Han, W. J. Dally, and H. Mao. Deep gradient compression: Reducing the communication bandwidth for distributed training. In *Proc. ICLR*, 2018.
- [33] R. Lipton, E. Markakis, E. Mossel, and A. Saberi. On approximately fair allocations of indivisible goods. In *Proc. EC*, 2004.
- [34] E. Lorch. Visualizing Deep Network Training Trajectories with PCA. In *Proc. ICML*, 2016.
- [35] L. Lyu, Y. Li, K. Nandakumar, J. Yu, and X. Ma. How to democratise and protect AI: Fair and differentially private decentralised deep learning. *IEEE Transactions on Dependable and Secure Computing*, 2020.
- [36] L. Lyu, X. Xu, Q. Wang, and H. Yu. Collaborative fairness in federated learning. In Q. Yang, L. Fan, and H. Yu, editors, *Federated Learning*, volume 12500 of *Lecture Notes in Computer Science*, pages 189–204. Springer, Cham, 2020.
- [37] L. Lyu, J. Yu, K. Nandakumar, Y. Li, X. Ma, J. Jin, H. Yu, and K. S. Ng. Towards fair and privacy-preserving federated deep models. *IEEE Transactions on Parallel and Distributed Systems*, 31(11):2524–2541, 2020.
- [38] S. Maleki, L. Tran-Thanh, G. Hines, T. Rahwan, and A. Rogers. Bounding the estimation error of sampling-based Shapley value approximation with/without stratifying. arXiv:1306.4265, 2013.
- [39] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proc. AISTATS*, 2017.
- [40] B. McMahan and D. Ramage. Federated learning: Collaborative machine learning without centralized training data. <https://ai.googleblog.com/2017/04/federated-learning-collaborative.html>. Accessed: 2021-10-08.
- [41] T. P. Michalak, P. L. Szczepański, T. Rahwan, A. Chrobak, S. Brânzei, M. Wooldridge, and N. R. Jennings. Implementation and computation of a value for generalized characteristic function games. *ACM Transactions on Economics and Computation*, 2(4), 2014.

- [42] D. Ng, X. Lan, M. M.-S. Yao, W. P. Chan, and M. Feng. Federated learning: a collaborative effort to achieve better medical imaging models for individual sites that have small labelled datasets. *Quantitative Imaging in Medicine and Surgery*, 11(2), 2020.
- [43] B. Pang and L. Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proc. ACL*, 2005.
- [44] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *Proc. ICML*, 2013.
- [45] R. Raskar, P. Vepakomma, T. Swedish, and A. Sharan. Data markets to support AI for all: Pricing, valuation and governance. arXiv:1905.06462, 2019.
- [46] A. Richardson, A. Filos-Ratsikas, and B. Faltings. Budget-bounded incentives for federated learning. In Q. Yang, L. Fan, and H. Yu, editors, *Federated Learning*, volume 12500 of *Lecture Notes in Computer Science*, pages 176–188. Springer, Cham, 2020.
- [47] A. Richardson, A. Filos-Ratsikas, and B. Faltings. Incentivizing and rewarding high-quality data via influence functions. In *Proc. ICML Workshop on Incentives in Machine Learning*, 2020.
- [48] N. Rieke, J. Hancox, W. Li, F. Milletari, H. R. Roth, S. Albarqouni, S. Bakas, M. N. Galtier, B. A. Landman, K. Maier-Hein, S. Ourselin, M. Sheller, R. M. Summers, A. Trask, D. Xu, M. Baust, and M. J. Cardoso. The future of digital health with federated learning. *npj Digital Medicine*, 3(119), 2020.
- [49] L. S. Shapley. A value for  $n$ -person games. In H. W. Kuhn and A. W. Tucker, editors, *Contributions to the Theory of Games*, volume 2, pages 307–317. Princeton Univ. Press, 1953.
- [50] R. Shokri and V. Shmatikov. Privacy-preserving deep learning. In *Proc. ACM SIGSAC*, 2015.
- [51] R. H. L. Sim, Y. Zhang, M. C. Chan, and B. K. H. Low. Collaborative machine learning with incentive-aware model rewards. In *Proc. ICML*, 2020.
- [52] R. H. L. Sim, Y. Zhang, B. K. H. Low, and P. Jaillet. Collaborative Bayesian optimization with fair regret. In *Proc. ICML*, pages 9691–9701, 2021.
- [53] T. Song, Y. Tong, and S. Wei. Profit allocation for federated learning. In *Proc. IEEE International Conference on Big Data*, 2019.
- [54] T. Wang, J. Rausch, C. Zhang, R. Jia, and D. Song. A principled approach to data valuation for federated learning. In Q. Yang, L. Fan, and H. Yu, editors, *Federated Learning*, volume 12500 of *Lecture Notes in Computer Science*, pages 153–167. Springer, Cham, 2020.
- [55] K. Wei, J. Li, C. Ma, M. Ding, and H. V. Poor. Differentially private federated learning: Algorithm, analysis and optimization. In *Federated Learning Systems: Towards Next-Generation AI*, pages 51–78. Springer International Publishing, Cham, 2021.
- [56] E. Winter. The Shapley value. In R. Aumann and S. Hart, editors, *Handbook of Game Theory with Economic Applications*, volume 3, chapter 53, pages 2025–2054. Elsevier B.V., 2002.
- [57] X. Xu and L. Lyu. A reputation mechanism is all you need: Collaborative fairness and adversarial robustness in federated learning. In *Proc. ICML Workshop on Federated Learning for User Privacy and Data Confidentiality*, 2021.
- [58] Z. Yan, D. Xiao, M. Chen, J. Zhou, and W. Wu. Dual-way gradient sparsification for asynchronous distributed deep learning. In *Proc. ICPP*, 2020.
- [59] Q. Yang, Y. Liu, T. Chen, and Y. Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology*, 10(2), 2019.
- [60] Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, and H. Yu. *Federated Learning*. Morgan & Claypool Publishers, 2019.

- [61] H. P. Young. Monotonic solutions of cooperative games. *International Journal of Game Theory*, 14(2):65–72, 1985.
- [62] H. Yu, Z. Liu, Y. Liu, T. Chen, M. Cong, X. Weng, D. Niyato, and Q. Yang. A fairness-aware incentive scheme for federated learning. In *Proc. AIES*, 2020.
- [63] J. Zhang, Y. Wu, and R. Pan. Incentive mechanism for horizontal federated learning based on reputation and reverse auction. In *Proc. TheWebConf*, page 947–956, 2021.
- [64] S. Zhang, A. Choromanska, and Y. LeCun. Deep learning with elastic averaging SGD. In *Proc. NeurIPS*, 2015.

## Checklist

1. For all authors...
  - (a) Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope? [Yes] Section 3 formalizes our fairness framework and solution while Section 4 provides the theoretical guarantees.
  - (b) Did you describe the limitations of your work? [Yes] See Section 4 where we explicitly point out that the noise in estimation and generalizability are potential limitations.
  - (c) Did you discuss any potential negative societal impacts of your work? [N/A]
  - (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
  - (a) Did you state the full set of assumptions of all theoretical results? [Yes] The assumptions are stated and interpreted when the results are presented.
  - (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix A for complete proof of all theoretical results.
3. If you ran experiments...
  - (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
  - (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 5 for training details and Appendix B for additional information on hyperparameters.
  - (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
  - (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix B for computational resources used.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
  - (a) If your work uses existing assets, did you cite the creators? [Yes] See Section 5 under the paragraph for Datasets.
  - (b) Did you mention the license of the assets? [N/A]
  - (c) Did you include any new assets either in the supplemental material or as a URL? [No]
  - (d) Did you discuss whether and how consent was obtained from people whose data you’re using/curating? [N/A]
  - (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
  - (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
  - (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
  - (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]