INDUSTRIAL AND SYSTEMS ENGINEERING

ISE

Using Synthetic Data to Mitigate Unfairness and Preserve Privacy through Single-Shot Federated Learning

CHIA-YUAN WU[1], FRANK E. CURTIS[1], AND DANIEL P. ROBINSON[1]

[1]Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA, USA

LEHIGH
UNIVERSITY.

Using Synthetic Data to Mitigate Unfairness and Preserve Privacy through Single-Shot Federated Learning

Chia-Yuan Wu[1], Frank E. Curtis[1], Daniel P. Robinson[1]

[1]Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA, USA
chw222@lehigh.edu, frank.e.curtis@lehigh.edu, daniel.p.robinson@lehigh.edu

Abstract

To address unfairness issues in federated learning (FL), contemporary approaches typically use frequent model parameter updates and transmissions between the clients and server. In such a process, client-specific information (e.g., local dataset size or data-related fairness metrics) must be sent to the server to compute, e.g., aggregation weights. All of this results in high transmission costs and the potential leakage of client information. As an alternative, we propose a strategy that promotes fair predictions across clients without the need to pass information between the clients and server iteratively and prevents client data leakage. For each client, we first use their local dataset to obtain a synthetic dataset by solving a bilevel optimization problem that addresses unfairness concerns during the learning process. We then pass each client's synthetic dataset to the server, the collection of which is used to train the server model using conventional machine learning techniques (that do not take fairness metrics into account). Thus, we eliminate the need to handle fairness-specific aggregation weights while preserving client privacy. Our approach requires only a single communication between the clients and the server, thus making it computationally cost-effective, able to maintain privacy, and able to ensuring fairness. We present empirical evidence to demonstrate the advantages of our approach. The results illustrate that our method effectively uses synthetic data as a means to mitigate unfairness and preserve client privacy.

1 Introduction

Fairness and privacy have become increasingly visible concerns as more machine learning (ML) tools are deployed in the real world. In the pursuit of highly accurate ML models, one often observes discriminatory outcomes when the training is performed using unfair/biased data. Discrimination here refers to decisions made against certain groups based on a sensitive feature, such as race, age, or gender (Mehrabi et al. 2021). For example, racial bias has been identified in the commercial risk assessment tool COMPAS (Dressel and Farid 2018) and in health systems (Obermeyer et al. 2019). Commercial facial recognition technologies have shown accuracy disparities by gender and skin type (Buolamwini and Gebru 2018). Additionally, age-related bias has been found in facial emotion detection systems (Kim et al. 2021). To mitigate such discriminatory practices, a collection of pre-processing, in-processing, and post-processing techniques

have been studied in a data-centralized setting (Mehrabi et al. 2021; Caton and Haas 2024; Wang, Shu, and Culotta 2021; Zafar et al. 2017; Lohia et al. 2019; d'Alessandro, O'Neil, and LaGatta 2017). These rely on access to the entire dataset, which in turn raises privacy leakage concerns. Thus, they are not applicable to decentralized data environments such as in the context of Federated Learning (FL).

FL has emerged as a promising paradigm that prioritizes data privacy. FL involves training client models using local data, sending the resulting client models to a server, then learning a server model by aggregating the client models using an iterative process, thereby avoiding the need to have direct access to client data (McMahan et al. 2017). Since the server model is indirectly derived from client data by using the client models as intermediaries, it is expected to lead to good representation and generalization across all clients (Pan et al. 2023). Separate from the typical distributed optimization setting, there are challenges in addressing fairness concerns in FL that include non-IID or unbalanced client datasets, massively distributed clients, and limited communication capacity between the clients and server (McMahan et al. 2017). If some client datasets are inherently unfair, perhaps reflecting demographic or regional disparities, then this can lead to unfair server models after aggregation steps are performed. Ensuring fairness across multiple independent client models is difficult in this situation.

When considering fairness in FL, prior studies have primarily focused on using aggregation to obtain a fair server model. However, these methods usually require frequent transmission of model-related information between clients and the server, which leads to significant communication costs. Furthermore, additional client information, such as dataset size or unfairness measures, is needed to calculate the server model aggregation weights, which in turn leads to potential information leakage. Addressing these limitations is critical for advancing fairness in the FL context.

To mitigate the aforementioned drawbacks, we propose an approach that only transfers *synthetic* client datasets to the server instead of client models. The framework is illustrated in Figure 1. Each synthetic client dataset is learned from the client's own training data with fairness considerations incorporated into the training process by solving a bilevel optimization problem. The server model is then trained on a combined (synthetic) dataset to minimize loss,
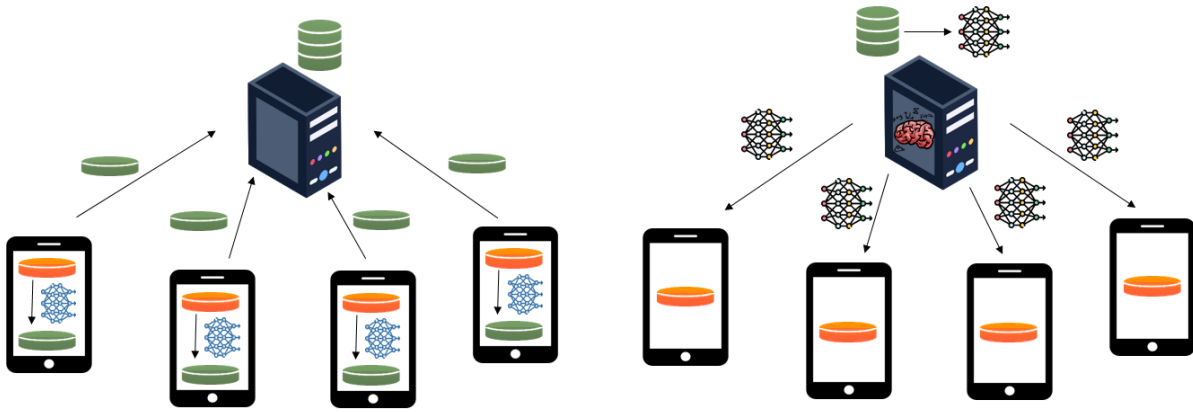
Figure 1: The framework of the proposed approach for addressing fairness in FL. The left side shows the actions of the clients. Each client trains its synthetic data (green) from the client training data (orange) while considering fairness issues, then passes this synthetic data to the server. The right side shows the server's action. The server trains a model on the combined client synthetic data, then passes it back to all the clients. The images used in the illustration are from (Flaticon 2023).

just like training a typical data-centralized machine learning model without explicit fairness concerns. Since the client training data is preserved on the client devices and no additional information needs to be sent to the server for calculating model aggregation weights, this approach prevents any client information leakage. Moreover, this approach is designed as a one-time model transmission between the clients and server. Once the server model is trained, the entire algorithm is completed when the server model is passed back to the clients. This greatly reduces transmission costs.

We summarize our contributions as follows:

- To the best of our knowledge, this is the first FL approach that involves only a single transmission between the clients and server while also addressing unfairness issues. Thus, our approach avoids high transmission costs while still producing a fair server model.

- Our approach prevents potential client information leakage by involving only a single transmission of *synthetic* data from each client to the server. Our experiments show that an accurate, but also fair server model can be trained even though the synthetic client data can differ significantly from the original client data.

- A major advantage of our approach is that the server model can be trained without fairness concerns. In a sense, the synthetic client data has already been generated to take fairness concerns into account, so all the server needs to do is aggregate the client data and train to maximize prediction accuracy.

- We validate our approach empirically on datasets from the literature. We provide concrete numerical evidence that our approach results in highly accurate and fair ML models without leaking private client information.

2 Background and Related Work

In this section, we discuss background and related work on group fairness measures, fairness in federated learning, bilevel optimization, and synthetic data generation.

Group Fairness Measures

In ML and FL, group fairness is a commonly used metric for determining whether a common group of people—as determined by a "sensitive feature" (e.g., gender, race, or age)—is treated fairly. One of the prominent group fairness measures is *statistical parity* (SP), also known in this context as *demographic parity*. SP defines fairness as the equal probability of predicting positive labels for different sensitive groups, i.e., for groups characterized by the value of a sensitive feature.

To define SP concretely, let X (resp., S) be a random vector representing nonsensitive (resp., sensitive) features and let Y be a binary-valued random variable representing a label, all defined over a combined probability space with probability measure $\mathbb{P}$. For convenience, let $A = (X, S)$. Let $y_\theta(\cdot)$ denote a binary classification function defined with weights θ. Then, as used in Caton and Haas (2024), the binary classifier $y_\theta(\cdot)$ is said to have SP if and only if

$$\mathbb{P}(y_\theta(A) = 1|S = 0) = \mathbb{P}(y_\theta(A) = 1|S = 1). \quad (1)$$

From a computational perspective, enforcing equation (1) is challenging because the probabilities lead to nonconvex functions. Zafar et al. (2019) used the decision boundary covariance as an approximation of the group conditional probabilities. In particular, the decision boundary covariance is defined as the covariance between the sensitive feature and the signed distance between the features and the decision boundary. Thus, letting $d_\theta(\cdot)$ denote this distance function corresponding to θ, the decision boundary covariance is

$$\begin{aligned}
&\text{Cov}(S, d_\theta(A)) \\
&= \mathbb{E}[(S - \mathbb{E}[S])d_\theta(A)] - \mathbb{E}[S - \mathbb{E}[S]]\mathbb{E}[d_\theta(A)] \\
&\approx \frac{1}{N}\sum_{i=1}^{N}(s_i - \bar{s})d_\theta(a_i)
\end{aligned} \quad (2)$$

where $\mathbb{E}[S - \mathbb{E}[S]] = 0$ (Zafar et al. 2017), $\{a_i\}_{i=1}^{N} = \{(x_i, s_i)\}_{i=1}^{N}$ is a sample of $\prod_{i=1}^{N} A = \prod_{i=1}^{N}(X, S)$, and $\bar{s} = (1/N)\sum_{i=1}^{N} s_i$ is the average of $\{s_i\}_{i=1}^{N}$.

In this study, we use a linear predictor, i.e., $d_\theta(a) = a^T\theta$. Using this choice, and keeping in mind our goal to obtain fair predictive models in terms of SP, we will present an optimization framework that enforces the constraint

$$\left| \frac{1}{N} \sum_{i=1}^{N} (s_i - \bar{s}) a_i^T \theta \right| \leq \epsilon, \tag{3}$$

where ϵ is a user-defined threshold that determines the level of unfairness that is to be tolerated. To evaluate the classifier y_θ in terms of SP, it is common to use the difference $P(y_\theta(A) = 1|S = 1) - P(y_\theta(A) = 1|S = 0)$, which motivates the definition of the following measure of statistical parity difference defined over $\{a_i\}_{i=1}^N$:

$$\text{SPD} \tag{4}$$
$$= \frac{|\{i : y_\theta(a_i) = 1 \cap s_i = 1\}|}{|\{i : s_i = 1\}|} - \frac{|\{i : y_\theta(a_i) = 1 \cap s_i = 0\}|}{|\{i : s_i = 0\}|}.$$

The closer SPD is to zero, the closer the predictor y_θ is to achieving perfect group fairness with respect to SP.

Fairness in Federated Learning

Proposed by McMahan et al. (2017), FL is a distributed optimization framework designed to address challenges related to data privacy and large-scale datasets. Given client objective functions $\{f_c\}$, the standard FL objective function is

$$\min_{\theta \in \mathbb{R}^n} f(\theta), \quad \text{where} \quad f(\theta) = \frac{1}{K} \sum_{c=1}^{K} f_c(\theta) \tag{5}$$

with K being the number of clients. This formulation aims to minimize the average (f) of the local client objectives ($\{f_c\}$), where θ is the model parameters (Pan et al. 2023). It is typical in ML that the client's objective function has the form $f_c(\theta) = \ell(\{x_{i,c}, y_{i,c}\}; \theta)$ for some loss function ℓ.

Many FL algorithms work by having the clients produce parameter values $\{\theta_c\}_{c=1}^K$, which are then passed to the server to obtain an estimate of a solution to (5). For example, the foundational FL algorithm *FedAvg* computes the server model update using the following aggregation formula:

$$\theta = \sum_{c=1}^{K} \frac{n_c}{n} \theta_c, \tag{6}$$

where n_c is the number of data points for client c for each index c and $n = \sum_{c=1}^{K} n_c$ (McMahan et al. 2017).

Some previous studies have focused on how to aggregate the model parameters in a manner that may be able to obtain a fair server model. For example, FairFed computes each client's aggregation weight as the difference in a fairness performance metric between the client model and the server model (Ezzeldin et al. 2023); FAIR-FATE incorporates momentum updates in the server model to prioritize clients with higher client fairness performance than the server model (Salazar et al. 2023); and FedGAN and Bias-Free FedGAN generate metadata from client generators on the server to ensure that a balanced training dataset is input to the server model (Rasouli, Sun, and Rajagopal 2020; Mugunthan et al.

2021). However, these algorithms rely on frequent communication of the model parameter values between the clients and the server. If client devices are slow or offline, the communication costs can be expensive and pose a significant limitation. Furthermore, in addition to the clients' model parameters, extra client information such as dataset size and the fairness measures on local models must be passed to the server to compute the aggregation weights, which can lead to potential leakage of client statistics (Ezzeldin et al. 2023).

Bilevel Optimization

Bilevel optimization is characterized by the nesting of one optimization problem within the other. One formulation of a bilevel optimization problem can be written as follows:

$$\min_{x \in \mathbb{R}^n, \, y \in \mathbb{R}^m} F_u(x, y)$$
$$\text{subject to } x \in X, \, y \in \arg\min_{y \in Y(x)} F_\ell(x, y), \tag{7}$$

where F_u and F_ℓ are the upper-level (or outer problem) and lower-level (or inner problem) objective functions, respectively. Note that allowable values for the upper-level problem variables is implicitly defined by the lower-level problem (Savard and Gauvin 1994; Giovannelli, Kent, and Vicente 2024). Due to the hierarchical structure, decisions made by the upper-level problem affect the outcomes of the lower-level problem (Sinha, Malo, and Deb 2017).

In this paper, we address fairness issues by formulating a bilevel optimization problem over clients. Our inner problem objective is to train a model with minimum loss from a given synthetic dataset. Our outer problem objective is to generate a synthetic dataset (one for each client) as the input to the inner problem. Moreover, when utilizing the model obtained from the inner problem and evaluating it on client training data, certain fairness constraints are enforced. The details of our approach are presented in the next section.

Synthetic Data Generation

Our approach is inspired by data distillation, which was first introduced in 2018 (Wang et al. 2018). Wang et al. (2018) presented the concept of distilling knowledge from a large training dataset into a small synthetic dataset. The idea is that the model trained on the synthetic dataset can perform approximately as well as one trained on the original large dataset. This method makes the training process more efficient and reduces the computational cost of model updates.

Synthetic data can also lead to benefits in the FL setting. In FL, the cost of communicating model parameter values between the server and clients is a major concern, especially when the model structure is complex and the number of model parameters is large. Goetz and Tewari (2020) and Hu et al. (2022) proposed transferring synthetic datasets to the server instead of the entire client model from the client side to reduce communication costs. The use of synthetic datasets helps in recovering the model update on the server to improve the server model updates (Goetz and Tewari 2020) or in recovering the client model directly to make server model aggregation more efficient (Hu et al. 2022).

The main contribution of these prior works is to make the training process efficient. They do not address fairness concerns. By contrast, we propose an extension of the previous works to improve fairness across clients in FL. While learning synthetic data for the clients, we not only distill the knowledge of their training data into the synthetic data but also mitigate unfairness in the training data at the same time. Consequently, when the model is trained by the server using the synthetic data from the clients, certain fairness criteria can be satisfied when the model is evaluated on the client's original training data. In addition, we can fully control the synthetic dataset size and the composition of each sensitive group and label. Thus, when the synthetic data is sent to the server, we can help prevent client information leakage.

3   Proposed Approach

In this section, we introduce our framework, which consists of two main parts: training synthetic datasets on the client side, then training a global model on the server. For simplicity, we focus on the case with a single sensitive feature (denoted in $\{0, 1\}$) and binary classification, although our approach could be generalized to other settings as well.

On the Client Side

We address fairness concerns on the client side through the use of SP as our fairness measure and the covariance decision boundary as a relaxation. Each client learns a synthetic dataset by solving the bilevel optimization problem in (8), stated below, which we now proceed to motivate.

Consider an arbitrary client. Assuming the client has N data points, for each $i \in \{1, ..., N\}$, we let $x_i \in \mathbb{R}^{n-1}$ be the ith nonsensitive feature vector, $s_i \in \{0, 1\}$ be the ith sensitive feature value, $y_i \in \{-1, 1\}$ be the label of the ith data point, $a_i = (x_i, s_i) \in \mathbb{R}^n$ be the complete ith feature vector, and $d_i = (x_i, s_i, y_i)$ be the complete ith data point. The goal of each client is to learn a synthetic dataset of size $N_s \leq N$. To denote the synthetic dataset, for each $i \in \{1, \ldots, N_s\}$, we let $\hat{d}_i = (\hat{x}_i, \hat{s}_i, \hat{y}_i)$ where $\hat{x}_i \in \mathbb{R}^{n-1}$, $\hat{s}_i \in \{0, 1\}$, $\hat{y}_i \in \{-1, 1\}$, and $\hat{a}_i = (\hat{x}_i, \hat{s}_i) \in \mathbb{R}^n$.

Multiple strategies can be used to define $\{\hat{s}_i, \hat{y}_i\}$. For example, if $N_s = N$, then one can simply choose $\{\hat{s}_i, \hat{y}_i\} = \{s_i, y_i\}$, whereas if $N_s < N$ once can choose the values for $\{\hat{s}_i, \hat{y}_i\}$ so that they have the same proportions as those for $\{s_i, y_i\}$. Details on how we choose these values for our numerical experiments are provided in Section 4.

Once $\{\hat{s}_i, \hat{y}_i\}$ has been chosen, the "ideal" problem solved by the client is defined, for user-defined tolerances $(\epsilon_1, \epsilon_2) \in [0, \infty)^2$ and loss function ℓ, as follows:

$$\min_{\{\hat{x}_i\}} \ell\big(\{d_i\}, \theta_{\text{ideal}}(\{\hat{d}_i\})\big)$$

$$\text{subject to } \left| \frac{1}{N} \sum_{i=1}^{N} (s_i - \bar{s}) a_i^T \theta_{\text{ideal}}(\{\hat{d}_i\}) \right| \leq \epsilon_1$$

$$\left| \frac{1}{N_s} \sum_{i=1}^{N_s} (\hat{s}_i - \bar{\hat{s}}) \hat{a}_i^T \theta_{\text{ideal}}(\{\hat{d}_i\}) \right| \leq \epsilon_2 \qquad (8)$$

where we define

$$\theta_{\text{ideal}}(\{\hat{d}_i\}) := \arg\min_{\theta} \ell(\{\hat{d}_i\}, \theta),$$

$$\bar{s} := \frac{1}{N} \sum_{i=1}^{N} s_i, \text{ and } \bar{\hat{s}} := \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{s}_i.$$

The loss function ℓ can be chosen based on the task being performed. In this paper, since we focus on binary classification, we select the logistic loss function, which defined over $\{d_i\}$ has the following form:

$$\ell(\{d_i\}, \theta) := \frac{1}{N} \sum_{i=1}^{N} \log(1 + e^{-y_i a_i^T \theta}), \qquad (9)$$

where $\theta \in \mathbb{R}^n$ represents the model parameters.

The aim of problem (8) is to learn synthetic nonsensitive features $\{\hat{x}_i\}$ in the synthetic dataset $\{\hat{d}_i\}$. The lower-level problem (or inner problem) is an unconstrained optimization problem that trains a model by minimizing the loss over the synthetic dataset. The upper-level problem (or outer problem) aims to minimize the loss of the model from the inner problem by evaluating it on the client's actual training dataset while satisfying fairness constraints.

Regarding the fairness constraints in (8), we want the covariance between the predictions and the sensitive features in the client training data and the synthetic data to fall within certain thresholds (i.e., ϵ_1 and ϵ_2). In this way, it keeps SPD close to zero to achieve good group fairness. The fairness constraints can be changed for other purposes as needed, for example, to achieve equal opportunity (Mehrabi et al. 2021).

On the Server Side

After each client learns its synthetic dataset, it is passed to the server. The server then trains a model on the combined synthetic dataset (i.e., the collection of synthetic datasets passed to it by the clients) defined using the same modeling formulation as in the client's inner optimization problem. Specifically, if we let $\hat{d}_{i,c}$ denote the ith data point for the cth client, then the server computes the model parameters for the global model by solving the following problem:

$$\theta(\{\hat{d}_{i,c}\}) := \arg\min_{\theta} \ell(\{\hat{d}_{i,c}\}, \theta). \qquad (10)$$

We stress that the server solves a standard ML problem to learn its model parameters, i.e., it does not make any attempt to account for fairness. The fact that it might "automatically" compute a fair model is a direct consequence of how each client computes its synthetic dataset in our framework.

Complete Algorithm

Instead of choosing nonnegative parameters ϵ_1 and ϵ_2, then solving the *constrained* optimization problem in (8), we choose to solve a penalty reformulation of (8) that takes the form of an *unconstrained* optimization problem, for the purposes of computational efficiency and convenience. Essentially, our penalty problem described below computes a solution to (8) for *implicitly* defined values of ϵ_1 and ϵ_2.

For a given choice of penalty parameters $(\rho_o, \rho_s) \in (0, \infty)^2$, regularization parameters $(\lambda_{\hat{x}}, \lambda_\theta) \in (0, \infty)^2$, and previously computed values for $\{(\hat{s}_i, \hat{y}_i)\}$, we define

$$\mathcal{P}(\{\hat{x}_i\})$$

$$:= \ell\left(\{d_i\}, \theta(\{\hat{d}_i\})\right) + \frac{\rho_o}{2N^2}\left(\sum_{i=1}^N (s_i - \bar{s})a_i^T\theta(\{\hat{d}_i\})\right)^2$$

$$+ \frac{\rho_s}{2N_s^2}\left(\sum_{i=1}^{N_s}(\hat{s}_i - \bar{\hat{s}})\hat{a}_i^T\theta(\{\hat{d}_i\})\right)^2 + \frac{\lambda_{\hat{x}}}{2(N_s \cdot n)^2}\sum_{i=1}^{N_s}\|\hat{x}_i\|_2^2$$

with

$$\theta(\{\hat{d}_i\}) := \arg\min_\theta \left\{\ell_r(\{\hat{d}_i\}, \theta) := \ell(\{\hat{d}_i\}, \theta) + \frac{\lambda_\theta}{2n^2}\|\theta\|_2^2\right\}. \tag{11}$$

The penalty reformulation of (8) that we solve is given by

$$\min_{\{\hat{x}_i\}} \mathcal{P}(\{\hat{x}_i\}). \tag{12}$$

In this formulation, ρ_o and ρ_s are the penalty parameters associated with fairness constraints on the client's original training data and synthetic data, respectively. Additionally, $\lambda_{\hat{x}}$ and λ_θ are regularization parameters for the outer and inner problem, respectively. Regularization terms are frequently adopted in ML modeling formulations for a variety of reasons. The regularization used on the nonsensitive features $\{\hat{x}_i\}$ in the outer problem help prevent the computation of excessively large values for $\{\hat{x}_i\}$, whereas the regularization term in the inner problem is introduced primarily to guarantee that the inner problem has a unique minimizer.

When required to solve the inner optimization problem (i.e., problem (11)), any solver designed for unconstrained optimization is applicable. In our testing, we choose the L-BFGS method with a strong Wolfe line search procedure.

To solve the outer optimization problem (12), we use a descent-type algorithm. Computing the gradient is nontrivial because of the inner optimization problem. That said, the gradient can be computed using the chain rule. Observe that

$$\mathcal{P}'(\{\hat{x}_i\}) = \frac{\partial\mathcal{P}(\{\hat{x}_i\})}{\partial\theta(\{\hat{d}_i\})} \cdot \frac{\partial\theta(\{\hat{d}_i\})}{\partial\{\hat{x}_i\}} + \frac{\lambda_{\hat{x}}}{(N_s \cdot n)^2}M(\{\hat{x}_i\}) \tag{13}$$

where $M(\{\hat{x}_i\})$ has ith row $\hat{x}_i^T$. Then, $\nabla\mathcal{P}(\{\hat{x}_i\}) = (\mathcal{P}'(\{\hat{x}_i\}))^T$. Computing $\partial\mathcal{P}(\{\hat{x}_i\})/\partial\theta(\{\hat{d}_i\})$ in (13) is straightforward while computing $\partial\theta(\{\hat{d}_i\})/\partial\{\hat{x}_i\}$ requires implicit differentiation to obtain

$$\frac{\partial^2\ell_r(\{\hat{d}_i\}, \theta(\{\hat{d}_i\}))}{\partial^2\theta} \cdot \frac{\partial\theta(\{\hat{d}_i\})}{\partial\{\hat{x}_i\}} = \frac{\partial^2\ell_r(\{\hat{d}_i\}, \theta(\{\hat{d}_i\}))}{\partial\theta\partial\{\hat{x}_i\}}, \tag{14}$$

which amounts to solving a linear system of equations.

With respect to the training performed on the server, we also include a regularization term to be consistent with the inner subproblem (11). Specifically, by letting $\hat{d}_{i,c}$ denote the ith synthetic data point computed by client c so that $\{\hat{d}_{i,c}\}$ denotes the combined synthetic dataset from all clients, the server model parameters are trained by solving

$$\theta(\{\hat{d}_{i,c}\}) := \arg\min_\theta \ell_r(\{\hat{d}_{i,c}\}, \theta). \tag{15}$$

Algorithm 1: Single-Shot Fair FL Method

1: **Client's Role:** Each client does the following steps.
2: Secures real training dataset $\{d_i\} = \{(x_i, s_i, y_i)\}_{i=1}^N$.
3: Chooses desired synthetic data size N_s.
4: Chooses $\{(\hat{s}_i, \hat{y}_i)\}$ (see earlier discussion).
5: Initializes synthetic data $\{\hat{x}_{i,0}\}$ as desired.
6: Chooses parameters $(\rho_o, \rho_s, \lambda_\theta, \lambda_{\hat{x}}) \in (0, \infty)^4$.
7: Chooses maximum number of iterations $k_{\max}$.
8: **for** each $k = 1, \ldots, k_{\max}$ **do**
9:    Compute $\theta(\{\hat{d}_{i,k}\})$ by solving (11).
10:    Compute $\frac{\partial\theta(\{\hat{d}_{i,k}\})}{\partial\{\hat{x}_{i,k}\}}$ by solving the linear system (14).
11:    Compute $\frac{\partial\mathcal{P}(\{\hat{x}_{i,k}\})}{\partial\theta(\{\hat{d}_{i,k}\})}$.
12:    Compute $\mathcal{P}'(\{\hat{x}_{i,k}\})$ as in (13).
13:    Set $\nabla\mathcal{P}(\{\hat{x}_{i,k}\}) \leftarrow (\mathcal{P}'(\{\hat{x}_{i,k}\}))^T$.
14:    Set $\{\hat{x}_{i,k+1}\}$ using $\{\hat{x}_{i,k}\}$ and $\nabla\mathcal{P}(\{\hat{x}_{i,k}\})$
15:      (e.g., using gradient descent or Adam).
16: **end for**
17: **Send to server:** $\{\hat{d}_i\} = \{(\hat{x}_i, \hat{s}_i, \hat{y}_i)\}_{i=1}^{N_s}$

18: **Server's Role:**
19: Obtain combined client synthetic data, i.e., $\{\hat{d}_{i,c}\}$.
20: Train server model by computing $\theta(\{\hat{d}_{i,c}\})$ from (15).
21: Communicate $\theta(\{\hat{d}_{i,c}\})$ to the clients.

Dataset	Law School	Dutch
Sensitive feature	race and gender	gender
Number of clients	2	4
N^{total}	20798	60420
N^{train}	8319	12084
$N^{train,small}$	800	1200
N^{test}	2080	3021

Table 1: For each dataset, N^{total} is the size of the original dataset, N^{train} and $N^{train,small}$ are the sizes of two randomly chosen training datasets (for each client), and N^{test} is the size of the testing dataset (for each client). The value for $N^{train,small}$ is chosen as approximately 10% of N^{train}.

The entire algorithm is shown in Algorithm 1. Client training starts at line 1, and server training begins at line 18. Unlike other conventional FL algorithms, which frequently communicate model parameters back and forth between the server and clients, our approach communicates parameter values in each direction a single time. Therefore, the transmission cost of our approach can be significantly less.

4 Experiments

In this section, we test through numerical experiments the performance of our approach. In particular, we examine how the penalty parameters and the size of the synthetic data affect the trade-off between accuracy and fairness metrics.

Experimental Setup

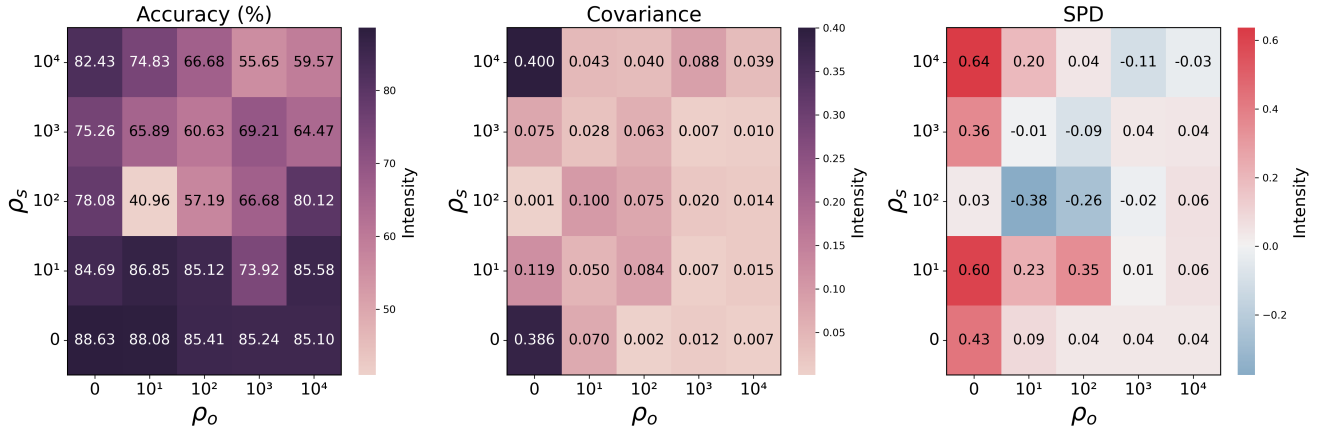**Computing environment.** The experiments reported in this paper were conducted on a machine equipped with an AMD

Figure 2: Accuracy, covariance estimate, and `SPD` fairness measure for different values of the penalty parameters ρ_o and ρ_s computed using the client testing data on the `Law School` dataset with race being the sensitive feature.

Dataset		Law School (race)		Law School (gender)		Dutch (gender)	
N_s	ρ_o	Accuracy(%)	SPD	Accuracy(%)	SPD	Accuracy(%)	SPD
N^{train}	0	88.44	0.4566	87.04	0.1320	78.21	0.2098
	10	87.91	0.0900	85.87	0.0150	76.85	0.0599
	100	86.20	0.0490	81.01	0.0151	76.56	0.0365
	1000	85.26	0.0400	73.89	0.0218	76.50	0.0344
	10000	85.22	0.0449	72.14	0.0224	74.33	0.0382
$N^{train,small}$	0	88.51	0.4362	87.24	0.1233	78.25	0.2103
	10	87.86	0.0816	85.50	0.0010	76.67	0.0478
	100	86.23	0.0547	78.17	-0.0014	76.62	0.0378
	1000	85.58	0.0375	77.74	-0.0032	76.70	0.0390
	10000	84.30	0.0412	73.37	0.0080	76.23	0.0496

Table 2: Accuracy and SPD performance measure values for various ρ_o and N_s values on the `Law School` and `Dutch` datasets. For all scenarios we set $\rho_s = 0$, and choose $\{(\hat{s}_i, \hat{y}_i)\}$ as described in section "Ability of Preserving Privacy".

Ryzen 9 7900X3D 12-Core CPU Processor. The FL framework was simulated using client parallel training implemented via the Python package `multiprocessing`.

Datasets. We present experiments for our approach on the binary classification datasets `Law School` (Wightman 1998) and `Dutch` (Van der Laan 2001), both of which are widely used real-world datasets used in the fairness-aware literature (Salazar et al. 2023). The `Law School` dataset consists of 11 features, including race and gender, which we treat as sensitive features in separate experiments. The goal is to predict whether a student will pass the bar exam. The `Dutch` dataset also has 11 features with gender identified as the sensitive feature. The task is to predict whether an individual's occupation is prestigious. Additional details on how we partition the datasets for our tests are provided in Table 1. We remark that our tests use a relatively small number of clients since they are for demonstration purposes only, although we do expect that our results representative real-world settings as well (i.e., when $K \gg 0$).

Hyperparameters and algorithm choices. We set the regularization parameter $\lambda_{\hat{x}} = 10^{-4}$. The maximum number of L-BFGS iterations used when solving the inner problem (11) is 100, and $k_{max} = 1000$ in Algorithm 1. The iterate update

in Line 14 of Algorithm 1 is performed using *Adam* (Kingma 2014). The penalty parameters ρ_o and ρ_s, regularization parameter λ_{θ}, and synthetic dataset size N_s of each client are discussed during each test that we present next.

Experimental Results

Ability to Mitigate Unfairness. To assess the ability of our method to mitigate unfairness, we first examine the trade-off between the accuracy lost and fairness gained by choosing different values for the penalty parameters ρ_o and ρ_s. These parameters correspond to the fairness constraints associated with the client's original training data and synthetic data, respectively. For each client, we set $N_s = N = N^{train}$ and $\{(\hat{s}_i, \hat{y}_i)\} = \{(s_i, y_i)\}$ so that we may clearly observe whether the synthetic datasets have the ability to mitigate unfairness. We conduct our tests on the `Law School` dataset, where race is the sensitive feature, with $\lambda_{\theta} = 10^{-4}$. The results are depicted in Figure 2. The values are computed across all clients' testing data.

One can view the bottom-left block in Figure 2 as a baseline since it corresponds to $\rho_o = \rho_s = 0$, i.e., no penalization on unfairness is enforced. Observing these metrics horizontally from left to right, we can see that the accuracy
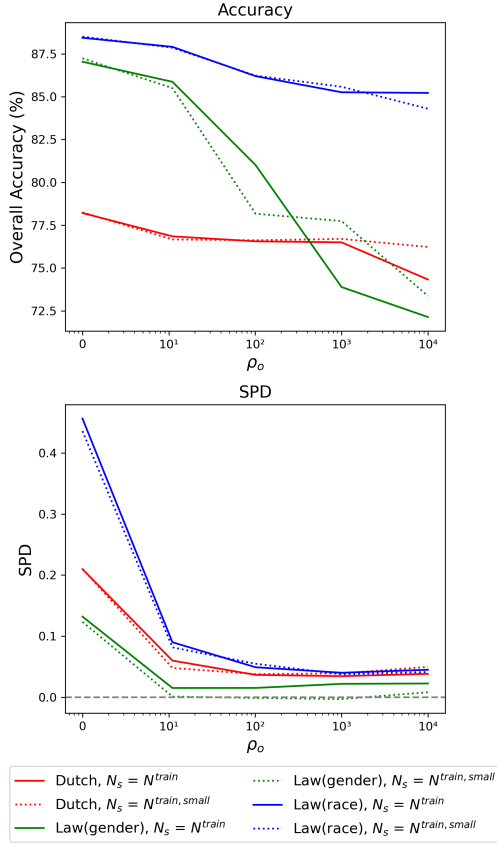
Figure 3: Accuracy and SPD performance trends for various values of ρ_o and two choices of synthetic data size N_s.

generally decreases, while the covariance and SPD measures approach zero thus reflecting a decrease in unfairness, although the trend is not perfectly monotone. Moreover, our results indicate that choosing $\rho_s > 0$ leads to results that are significantly less predictable and robust. Finally, based on these results, one might be tempted to choose $\rho_s = 0$ and $\rho_o = 10$, which leads to a mere 0.55% drop in accuracy and a significant improvement in the fairness measures.

In Figure 2, we can observe for the choice of $\rho_o = 10$ and $\rho_s = 100$ that the accuracy was significantly lower than the other settings. For experimental purposes, for this setting we computed the same quantities after adjusting λ_θ from 10^{-4} to 10^{-3}, which then achieved an accuracy of 69.38%, a covariance of 0.056, and an SPD of -0.10. This further underscores the diminished robustness of our approach in the case when $\rho_s > 0$. In conclusion, we recommend using the value $\rho_s = 0$ and tuning ρ_o appropriately.

Ability of Preserving Privacy. To evaluate the possible effectiveness of our method in preserving client privacy while mitigating unfairness, we conducted experiments related to controlling the size of the synthetic dataset. For all of the tests in this section, we select $\rho_s = 0$, which is motivated by the numerical results in the previous section, and consider multiple values for ρ_o to control the level of unfairness. We also set $\lambda_\theta = 10^{-3}$ for the results obtained in this section.

First, we ran our framework with $N_s = N = N^{train}$ for each client so that the synthetic datasets learned are of the same size as their training data. Here we select $\{(\hat{s}_i, \hat{y}_i)\} = \{(s_i, y_i)\}$, so that these *synthetic* data quantities are equal to their corresponding quantities in the training data (the synthetic quantities $\{\hat{x}_i\}$ are learned using our framework).

Second, we ran our framework with $N = N^{train}$ and $N_s = N^{train,small}$ (see Table 1) so that the synthetic datasets are significantly smaller than the training sets (the synthetic datasets are about 10% of the size of the training sets). Here, $\{(\hat{s}_i, \hat{y}_i)\}_{i=1}^{N_s}$ are computed to have equal numbers (25% of N_s) from each of the following sensitive feature and label pairs: $(1, 0)$, $(0, 1)$, $(1, 1)$, and $(0, 0)$. The quantities $\{\hat{x}_i\}_{i=1}^{N_s}$, which are learned in our framework, are initialized by sampling each feature from a normal distribution with mean and variance determined by the training data.

The results of our tests can be found in Table 2; see also the graphs in Figure 3. These empirical results show consistent performance trends for both accuracy and SPD, regardless of the synthetic dataset size or the initialization strategy; i.e., as we increase the penalty parameter ρ_o, we sacrifice some accuracy while significantly improving fairness. Moreover, the difference in accuracy obtained when passing the significantly smaller dataset of size $N_s = N^{train,small}$ is similar to the accuracy obtained when passing a synthetic dataset that is the same size as the training dataset. We also remind the reader that the baseline setting of $\rho_0 = 0$ corresponds to ignoring fairness concerns, which is clearly illustrated in the poor values of SPD. Finally, the values of $\min_{i,j} \|x_i - \hat{x}_j\|_2$ for all clients and datasets were in the range $[0.41, 1.03]$, thus showing that the synthetic datasets computed were significantly different from the training data. Overall, we believe these results provide evidence that our framework is able to offer good accuracy and acceptable unfairness measures for models obtained using relatively small synthetic datasets. Thus, our framework allows not only for the mitigation of unfairness but also the potential to protect client privacy through the use of small synthetic datasets.

5 Conclusion

In this study, we present a cost-effective communication strategy to mitigate unfairness and help preserve privacy across clients in an FL context. We propose to send a synthetic dataset from each client to the server instead of each client's model. The manner in which we propose computing the synthetic dataset takes unfairness into account by solving a bilevel optimization problem. This allows the server to train a conventional ML model without further fairness concerns on the collection of client-provided synthetic datasets. This approach not only simplifies the training process for the server but also significantly reduces communication costs by limiting it to a one-time transfer between the clients and the server. Our implementation on well-known datasets illustrates that our approach effectively mitigates unfairness without sacrificing too much accuracy. Even with synthetic datasets of 10% the size of the client training data, our strategy remains effective in producing accurate models that are fair and aid in maintaining privacy.

References

Buolamwini, J.; and Gebru, T. 2018. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*, 77–91. PMLR.

Caton, S.; and Haas, C. 2024. Fairness in machine learning: A survey. *ACM Computing Surveys*, 56(7): 1–38.

d'Alessandro, B.; O'Neil, C.; and LaGatta, T. 2017. Conscientious classification: A data scientist's guide to discrimination-aware classification. *Big data*, 5(2): 120–134.

Dressel, J.; and Farid, H. 2018. The accuracy, fairness, and limits of predicting recidivism. *Science advances*, 4(1): eaao5580.

Ezzeldin, Y. H.; Yan, S.; He, C.; Ferrara, E.; and Avestimehr, A. S. 2023. Fairfed: Enabling group fairness in federated learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, 7494–7502.

Flaticon. 2023. Illustration Icons. https://www.flaticon.com/. Accessed: 2023-06-02.

Giovannelli, T.; Kent, G. D.; and Vicente, L. N. 2024. Bilevel optimization with a multi-objective lower-level problem: Risk-neutral and risk-averse formulations. *Optimization Methods and Software*, 1–23.

Goetz, J.; and Tewari, A. 2020. Federated learning via synthetic data. *arXiv preprint arXiv:2008.04489*.

Hu, S.; Goetz, J.; Malik, K.; Zhan, H.; Liu, Z.; and Liu, Y. 2022. Fedsynth: Gradient compression via synthetic data in federated learning. *arXiv preprint arXiv:2204.01273*.

Kim, E.; Bryant, D.; Srikanth, D.; and Howard, A. 2021. Age bias in emotion detection: An analysis of facial emotion recognition performance on young, middle-aged, and older adults. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, 638–644.

Kingma, D. 2014. Adam: a method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lohia, P. K.; Ramamurthy, K. N.; Bhide, M.; Saha, D.; Varshney, K. R.; and Puri, R. 2019. Bias mitigation postprocessing for individual and group fairness. In *Icassp 2019-2019 ieee international conference on acoustics, speech and signal processing (icassp)*, 2847–2851. IEEE.

McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; and y Arcas, B. A. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, 1273–1282. PMLR.

Mehrabi, N.; Morstatter, F.; Saxena, N.; Lerman, K.; and Galstyan, A. 2021. A survey on bias and fairness in machine learning. *ACM computing surveys (CSUR)*, 54(6): 1–35.

Mugunthan, V.; Gokul, V.; Kagal, L.; and Dubnov, S. 2021. Bias-free fedgan: A federated approach to generate bias-free datasets. *arXiv preprint arXiv:2103.09876*.

Obermeyer, Z.; Powers, B.; Vogeli, C.; and Mullainathan, S. 2019. Dissecting racial bias in an algorithm used to manage the health of populations. *Science*, 366(6464): 447–453.

Pan, Z.; Wang, S.; Li, C.; Wang, H.; Tang, X.; and Zhao, J. 2023. Fedmdfg: Federated learning with multi-gradient descent and fair guidance. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 9364–9371.

Rasouli, M.; Sun, T.; and Rajagopal, R. 2020. Fedgan: Federated generative adversarial networks for distributed data. *arXiv preprint arXiv:2006.07228*.

Salazar, T.; Fernandes, M.; Araújo, H.; and Abreu, P. H. 2023. Fair-fate: Fair federated learning with momentum. In *International Conference on Computational Science*, 524–538. Springer.

Savard, G.; and Gauvin, J. 1994. The steepest descent direction for the nonlinear bilevel programming problem. *Operations Research Letters*, 15(5): 265–272.

Sinha, A.; Malo, P.; and Deb, K. 2017. A review on bilevel optimization: From classical to evolutionary approaches and applications. *IEEE transactions on evolutionary computation*, 22(2): 276–295.

Van der Laan, P. 2001. The 2001 census in the Netherlands: Integration of registers and surveys. In *CONFERENCE AT THE CATHIE MARSH CENTRE.*, 1–24.

Wang, T.; Zhu, J.-Y.; Torralba, A.; and Efros, A. A. 2018. Dataset distillation. *arXiv preprint arXiv:1811.10959*.

Wang, Z.; Shu, K.; and Culotta, A. 2021. Enhancing model robustness and fairness with causality: A regularization approach. *arXiv preprint arXiv:2110.00911*.

Wightman, L. F. 1998. LSAC National Longitudinal Bar Passage Study. LSAC Research Report Series. Technical report, ERIC.

Zafar, M. B.; Valera, I.; Gomez-Rodriguez, M.; and Gummadi, K. P. 2019. Fairness constraints: A flexible approach for fair classification. *The Journal of Machine Learning Research*, 20(1): 2737–2778.

Zafar, M. B.; Valera, I.; Rogriguez, M. G.; and Gummadi, K. P. 2017. Fairness constraints: Mechanisms for fair classification. In *Artificial intelligence and statistics*, 962–970. PMLR.