Proceedings of Machine Learning Research 279:173–198, 2024 Algorithmic Fairness Through the Lens of Metrics and Evaluation

Privacy-Preserving Group Fairness in Cross-Device Federated Learning

Sikha Pentyala SIKHA@UW.EDU
University of Washington, Tacoma, WA, USA
Mila, Montréal, Quebec, CA

Nicola Neophytou NICOLA.NEOPHYTOU@MILA.QUEBEC
University of Montréal & Mila
Montréal, Quebec, CA

Anderson Nascimento ANDCLAY@UW.EDU
University of Washington, Tacoma, WA, USA

Martine De Cock MDECOCK@UW.EDU
University of Washington, Tacoma, WA, USA
Ghent University, Belgium

Golnoosh Farnadi FARNADIG@MILA.QUEBEC
McGill University & Mila
Montréal, Quebec, CA

Editors: Miriam Rateike, Awa Dieng, Jamelle Watson-Daniels, Ferdinando Fioretto, Golnoosh Farnadi

Abstract

Group fairness ensures that the outcome of machine learning (ML) based decision making systems are not biased towards a certain group of people defined by a sensitive attribute such as gender or ethnicity. Achieving group fairness in Federated Learning (FL) is challenging because mitigating bias inherently requires using the sensitive attribute values of all clients, while FL is aimed precisely at protecting privacy by *not* giving access to the clients' data. As we show in this paper, this conflict between fairness and privacy in FL can be resolved by combining FL with Secure Multiparty Computation (MPC) and Differential Privacy (DP). To this end, we propose a privacy-preserving approach to calculate group fairness notions in the cross-device FL setting. Then, we propose two bias mitigation pre-processing and post-processing techniques in cross-device FL under formal privacy guarantees, without requiring the clients to disclose their sensitive attribute values. Empirical evaluations on real world datasets demonstrate the effectiveness of our solution to train fair and accurate ML models in federated cross-device setups with privacy guarantees to the users.

Keywords: Group fairness, Privacy, Federated Learning, Secure Multiparty Computation

1. Introduction

Machine learning (ML) models are widely adopted in decision making systems, directly affecting quality of life, including in healthcare, justice, education, surveillance, human resources, and advertising. The use of ML in such impactful domains has raised valid concerns regarding fairness of models that make biased predictions, resulting in discrimination of individuals, disparity in allocation of resources, and inequity in quality of service. Discrimination by ML models has been reported in applications for recidivism prediction (Angwin et al., 2016; Larson et al., 2016), credit card approval (Vigdor, 2019), advertising (Sweeney, 2013;

Ali et al., 2019), and job matching (Wall and Schellmann, 2021), among others (Buolamwini and Gebru, 2018). *Group fairness*, a widely considered notion of fairness that has been imposed by regulatory bodies in AI applications (e-CFR, 1981), aims at ensuring that the ML model predictions are not biased towards a certain group of people, as defined by a sensitive attribute such as race, gender, or age (Verma and Rubin, 2018). Developing group-fair ML models in the centralized learning setup where a single entity has access to all data has been well studied in the literature (Dwork et al., 2012; Hardt et al., 2016; Pessach and Shmueli, 2020). In a lot of applications however, data originates from many different clients who – out of privacy concerns – may not wish to, or may legally not be allowed to, disclose their data to a central entity.

Federated learning (FL) (McMahan et al., 2017), a collaborative training paradigm where each client performs training on its own data and shares only model parameters or gradients instead of raw data with a central aggregator, is gaining popularity as a privacy-enhancing technology (PET) (Hard et al., 2018; Kairouz et al., 2021; Paulik et al., 2021; Shenwai, 2022). Many FL applications naturally exhibit a *cross-device federated setup* with horizontally partitioned data, where each client in the federation has one particular value for the sensitive attribute (e.g. a female client, using a mobile recommending application, will have user-item interaction data belonging only to the female group). Methods for training group-fair ML models in the *centralized* paradigm rely on knowledge of sensitive attribute values across the entire dataset, for instance to compute aggregated statistics for data rebalancing, or to compute a bias correction term in the loss function. Extending these methods to FL is non-trivial. *Fair model training in FL* has been identified as challenging due to the intrinsic conflict between bias mitigation algorithms' need to perform computations over the data of all clients, and FL's aims to preserve data privacy by *not* giving access to client data (Zhou et al., 2021). The emerging literature on group-fair FL works around this by assuming a *cross-silo* set-up in which each client has data for *multiple* sensitive attribute values (e.g. each client represents a bank with data about female as well as male customers) (Mohri et al., 2019; Abay et al., 2020; Du et al., 2021; Cui et al., 2021) and/or by sacrificing some privacy for fairness. Recently proposed methods ((Kanaparthy et al., 2021; Papadaki et al., 2021; Yue et al., 2021)) send unprotected model parameters and/or other unprotected information such as fairness metrics to a central aggregator, thereby leaking information about clients' data (Boenisch et al., 2021). FairFed (Ezzeldin et al., 2021) employs techniques to protect the individual updates but the aggregations at the central aggregator are not protected and can reveal information about the clients' data when under attack. In this paper, to the best of our knowledge, we propose the first method for training group-fair ML models in cross-device FL under complete and formal privacy guarantees, protecting both the sent updates or values as well as the aggregated information.

Our approach. Building on the literature on fair ML in the centralized paradigm (Kamiran and Calders, 2012; Hardt et al., 2016; Yan et al., 2020) we propose a *privacy-preserving pre-processing* technique for bias mitigation through training sample reweighing (`PrivFairFL-Pre`), and a *privacy-preserving post-processing* technique that identifies fair classification thresholds for different groups (`PrivFairFL-Post`). The core of these techniques are Secure Multiparty Computation (MPC) protocols for collecting aggregated statistics of the label and sensitive attribute value distributions across the federation. Unlike a traditional central aggregator in FL, the *computing servers* that execute these MPC protocols never see the personal values of the clients in an unencrypted manner. We further protect any output of such computations that needs to be made public through perturbations to provide Differential Privacy (DP) guarantees. To do so, the computing servers use MPC to generate the necessary Laplacian noise and add it to the outputs of the computations to satisfy DP requirements, thereby having an MPC protocol effectively play the role of a trusted curator implementing global DP. As such, our solution combines the best of multiple PETs, namely FL, MPC, and DP, to train fair and high-utility ML models, as we show through an empirical analysis.

Our main novel contributions are:

- MPC protocols for (1) collecting ROC curves and statistics of label and sensitive attribute value distributions across a federation of clients in a fully privacy-preserving manner, and for (2) the publication of said ROC curves and statistics under DP guarantees.
- Pre- and post-processing based algorithms `PrivFairFL-Pre` and `PrivFairFL-Post` that leverage these statistics for training group-fair ML models in the FL paradigm, without requiring the clients to disclose the values of their sensitive attributes.
- An empirical evaluation of our proposed `PrivFairFL` methods to demonstrate that group fairness can be achieved in cross-device FL without leaking sensitive user information.

2. Preliminaries

Notations. We consider a federation with M clients, in which each client k ($k = 1 \dots M$) holds a dataset D_k with n_k training samples. $<X_{ik}, s_{ik}, y_{ik}>$ represents the i^{th} training sample held by the k^{th} client where s_{ik} is the value of a sensitive attribute, y_{ik} is the value of a class label, and X_{ik} is the set of remaining feature values. $N = \sum_{k=1}^{M} n_k$ denotes the total number of samples. S denotes the sensitive attribute and Y denotes the class label. For the sake of simplicity, we focus on the case where S represents a binary sensitive attribute that takes u (representing the unprivileged group) and p (representing the privileged group) as its values, and Y takes 0 and 1 as its values. Our proposed methods, however, can be extended to the case of multiple sensitive attributes, including non-binary, and multi-class classification, as outlined in the Appendix. We use $C(s, y)$ to denote the count of all training samples from all the clients with sensitive attribute value s and class label y:

$$C(s,y) = \sum_{k=1}^{M} \sum_{i=1}^{n_k} \#\{<X_{ik}, s_{ik}, y_{ik}> \mid s_{ik} = s \wedge y_{ik} = y\} \tag{1}$$

For example, $C(u, 1)$ is the number of training samples from all clients belonging to the unprotected group and with class label 1. The model parameters of client k are denoted as θ_k and the aggregated model parameters are represented by θ. The model θ makes predictions $\hat{Y}$.

Federated Learning (FL). FL is a collaborative learning paradigm where the clients together train an ML model in coordination with a central server while keeping the training data private and decentralized. The clients train the model on their own data locally and send only the gradients or model parameters θ_k to the central server for aggregation. This is done iteratively for a defined number of rounds to get a global model θ that has achieved the learning objective on the combined training data of the clients. This enables the clients to keep their data within the premises providing first-level privacy to the clients. It is well-known that FL alone does not offer formal privacy guarantees, and that it leaves the clients vulnerable to information leakage (Kairouz et al., 2021). MPC-based methods (Bonawitz et al., 2017) and DP-based methods (Abadi et al., 2016) have been proposed to be used in combination with FL to provide stronger privacy guarantees.

Differential Privacy (DP). DP guarantees plausible deniability regarding an instance being in a dataset, hence offering privacy guarantees. DP techniques ensure that a randomized algorithm $\mathcal{A}$ behaves similarly on a dataset D and a neighboring dataset D' that differs in a single entry,[1] i.e. $\mathcal{A}$ generates a similar output probability distribution on D and D' (Dwork and Roth, 2014). $\mathcal{A}$ can for instance be an algorithm that takes as input a dataset D of training examples and outputs an ML model; alternatively $\mathcal{A}$ can return counts or other statistics about D. A randomized algorithm $\mathcal{A}$ is called (ϵ, δ)-DP if for all pairs of neighboring sets D and D', and for all subsets S of $\mathcal{A}$'s range,

$$\mathrm{P}(\mathcal{A}(D) \in S) \leq e^{\epsilon} \cdot \mathrm{P}(\mathcal{A}(D') \in S) + \delta. \tag{2}$$

where ϵ is the privacy budget or privacy loss and δ is the probability of violation of privacy. The smaller these values, the stronger the privacy guarantees. When $\delta = 0$, then $\mathcal{A}$ is said to be ϵ-DP. An (ϵ, δ)-DP randomized algorithm $\mathcal{A}$ is commonly created out of an algorithm $\mathcal{A}^*$ by adding noise that is proportional to the *sensitivity* of $\mathcal{A}^*$, in which the sensitivity measures the maximum impact a change in the underlying dataset can have on the output of $\mathcal{A}^*$. The post-processing property of DP guarantees that if $\mathcal{A}$ is ϵ-DP, then $g(\mathcal{A})$ is also ϵ-DP where g is an arbitrary function. In other words, any arbitrary computations performed on DP output preserves DP without any effect on the privacy budget ϵ.

The traditional DP paradigm – *global DP* – assumes that the entire dataset resides with a central curator who computes the noise. Alternatively, if the data originates from multiple data holders, then they can add noise before sending their information to a trusted curator for further processing. Approaches based on this *local DP* paradigm tend to have lower utility while of course offering better privacy than requiring data holders to disclose their information in an unprotected manner to a trusted curator. In `PrivFairFL` (Sec. 4) we replace the trusted curator by an MPC protocol that can be run on untrusted servers, achieving the same utility as in the global DP paradigm but without requiring the clients to disclose their sensitive attribute values to anyone.

Secure Multiparty Computation (MPC). MPC is an umbrella term for cryptographic approaches that allow two or more parties (servers) to jointly compute a specified output from their private information in a distributed fashion, without revealing anything beyond the output of the computation to each other (Cramer et al., 2015). MPC protocols are designed to prevent and detect attacks by an adversary A who can corrupt one or more parties to learn the private information or to cause incorrect computations. These protocols can be mathematically proven to guarantee privacy and correctness. We follow the universal composition theorem that allows modular design where the protocols remain secure even if

1. Throughout this paper we consider event-level DP, i.e. an entry or instance corresponds to one training example of one client in the federation, and D holds the data of all clients combined, i.e. $D = \cup_{k=1}^{M} D_k$.

composed with other or the same MPC protocols (Canetti, 2000). For details, see Evans et al. Evans et al. (2018).

All computations in MPC are commonly done on integers modulo q, i.e., in the ring $\mathbb{Z}_q$. In `PrivFairFL`, the clients first convert any real-valued inputs into a fixed point representation[2] and then encrypt their private data (all integers) by splitting them into so-called secret-shares. These secret-shares are then distributed to a set of servers (computing parties) that run MPC protocols and perform computations over these secret shares. Proper design of the MPC protocols ensures that no server learns anything about the input on its own and that nothing about the input is revealed to any subset of colluding (malicious) servers that can be corrupted. For example, in the replicated secret-sharing scheme with 3 servers by Araki et al. (Araki et al., 2016), a private value $x \in \mathbb{Z}_{2^f}$ is secret-shared among computing parties P_1, P_2, and P_3 by picking uniformly random numbers (shares) $x_1, x_2, x_3 \in \mathbb{Z}_{2^f}$ such that $x_1 + x_2 + x_3 = x \mod 2^f$, and distributing (x_1, x_2) to P_1, (x_2, x_3) to P_2, and (x_3, x_1) to P_3. Note that, while the secret-shared information can be trivially revealed by combining shares from any two servers, no single server can obtain any information about x given its shares. In the remainder of this paper, we use $[\![x]\!]$ as a shorthand for a secret sharing of x, regardless of which secret-sharing scheme is used. In the replicated secret-sharing scheme sketched above, $[\![x]\!] = ((x_1, x_2), (x_2, x_3), (x_3, x_1))$.

Group Fairness. Group fairness measures how balanced the predicted outcomes are across the groups defined by the sensitive attribute S. Many different notions of group fairness have been proposed in the literature (Dwork et al., 2012; Feldman et al., 2015; Hardt et al., 2016; Kleinberg et al., 2016; Donini et al., 2018). We consider popular statistical notions of fairness that rely on computing the true positive rate (TPR_p and TPR_u), false positive rate (FPR_p and FPR_u), and the number of true positives (TP_p and TP_u) for the privileged (p) and unprivileged groups (u) respectively. *Disparate Impact (DI)* measures discrimination in the predictions by computing the recall (TPR) for the groups (Speicher et al., 2018). A value $\mathrm{DI} = 1$ indicates discrimination-free predictions. We report the degree of discrimination using $|1 - \mathrm{DI}|$. *Equal opportunity (EOP)* considers the fairness of the predictions from the perspective of TPR (Hardt et al., 2016). This metric focuses only on the positive or advantaged outcome. We report this metric as equal opportunity difference (ΔEOP) by computing the difference between the TPR of the two groups. *Equalized odds (EODD)* considers the predictions fair if $\hat{Y}$ and S are independent conditional on Y (Hardt et al., 2016). This implies that both TPR and FPR are equal between the groups. We report this metric as average odds difference (ΔEODD). *Statistical parity (SP)* considers the predictions as fair if the number of positive predictions are the same for the two groups (Dwork et al., 2012). We report statistical parity difference (ΔSP) as the difference between the ratio of positive outcomes per group.

Formally:

$$\begin{aligned}
|1 - \mathrm{DI}| &= |1 - \max(\mathrm{TPR}_u/\mathrm{TPR}_p, \mathrm{TPR}_p/\mathrm{TPR}_u)| && (3)\\
\Delta\mathrm{EOP} &= |\mathrm{TPR}_p - \mathrm{TPR}_u| && (4)\\
\Delta\mathrm{EODD} &= 0.5 \cdot (|\mathrm{TPR}_p - \mathrm{TPR}_u| + |\mathrm{TPR}_p - \mathrm{TPR}_u|) && (5)\\
\Delta\mathrm{SP} &= |(\mathrm{TP}_p/\mathrm{N}_p) - (\mathrm{TP}_u/\mathrm{N}_u)| && (6)
\end{aligned}$$

2. Real values are represented using l bits in total with d bits for decimal precision.

where N_p and N_u denote the number of samples for privileged and unprivileged group respectively. The lower the values of the metrics, the fairer the predictions made by the model.

3. Related Work

Group fairness in centralized learning. Existing unfairness mitigation techniques can be categorized into three categories based on the stage in the ML pipeline they are incorporated into. *Pre-processing techniques* are applied to the training data to create a less biased dataset (Kamiran and Calders, 2012; Zemel et al., 2013; Feldman et al., 2015; Calmon et al., 2017; Yan et al., 2020). They either modify the raw data by changing the sensitive attribute and/or the class label, or assign weights to the samples based on their label and/or sensitive attribute. Reweighing techniques make the loss function penalize incorrect predictions based on the assigned weights of the samples to learn a fair predictor across groups. *In-processing techniques* are employed during the training phase of a model (Kamishima et al., 2012; Zhang et al., 2018; Agarwal et al., 2018; Kearns et al., 2018; Celis et al., 2019). They usually modify the optimization problem by either adding a regularizer to the objective function or constraints to the optimization formulation. *Post-processing techniques* are applied to the predicted labels to generate fairer predictions either by flipping the labels or finding an optimal threshold (Kamiran et al., 2012; Hardt et al., 2016). Unlike in-processing, the pre- and post-processing techniques are independent of the notions of fairness, learning objective, and the model being trained. `PrivFairFL` extends pre- and post-processing to FL.

Group fairness in FL. Extending bias mitigation techniques from the centralized paradigm to FL is challenging due to an intrinsic conflict between fair model training and FL (Zhou et al., 2021): (i) evaluating the fairness of a model, or mitigating bias, requires access to the data of all clients; and (ii) FL aims at preserving data privacy by *not* giving such access. As Tab. 1 shows, methods for training group-fair models have been proposed for cross-silo setups (each client has data for multiple sensitive attribute values), and for cross-device setups (each client has data for only one sensitive attribute value). Nearly all methods are based on in-processing, hence tailored to a specific training algorithm, model architecture, and fairness notion (Cui et al., 2021; Zhang et al., 2021; Du et al., 2021). Furthermore, as the "privacy" column in Tab. 1 indicates, most of the emerging literature on group-fair FL tries to work around the conflict between (i) and (ii) by sacrificing privacy for fairness. Information leaks can occur when the clients send updated model parameters, gradients, fairness metrics, or the values of sensitive attributes to the aggregator, or by analyzing the aggregated outputs. For example, during FL training, assuming that an adversary A has the model from the previous round and the gradient updates from the current round, A can infer a private training example (Kairouz et al., 2021). Current works do not take into account such information leaks in FL (Papadaki et al., 2021; Yue et al., 2021; Kanaparthy et al., 2021; Hong et al., 2021). A can also analyze the aggregated outputs to infer knowledge about a particular client. FairFed (Ezzeldin et al., 2021) and Rodríguez-Gálvez et al. (2021) employ SecAgg (Bonawitz et al., 2017) to protect data leaks from information sent by the clients, but fail to protect the aggregated values. Though Rodríguez-Gálvez et al. (2021) use a combination of MPC and DP, they reveal the aggregated gradients to the computing parties and add noise in-the-clear to publish DP aggregates to the FL aggregator. Similarly, Zhang et al. (2020) protect only the discrimination indices sent by clients and fail to protect any aggregated output and gradient updates.

Table 1: Related work on Group Fairness in FL

Paper	**Scenario**		**Privacy**		**Mitigation Alg.**		
	silo	dev	DP	MPC	Pre	In	Post
Abay (Abay et al., 2020)	✓		✓	✗	✓	✓	✗
AgnosticFair (Du et al., 2021)	✓		✗	✗	✗	✓	✗
FCFL (Cui et al., 2021)	✓		✗	✗	✗	✓	✗
Zhang (Zhang et al., 2021)	✓		✗	✗	✗	✓	✗
FairFL (Zhang et al., 2020)	✓		✗	✓	✗	✓	✗
FPFL (Padala et al., 2021)	✓		✓	✗	✗	✓	✗
Rodriguez (Rodríguez-Gálvez et al., 2021)	✓	✗	✓	✓	✗	✓	✗
Kanaparthy (Kanaparthy et al., 2021)		✓	✗	✗	✗	✓	✗
GI-FAIR (Yue et al., 2021)		✓	✗	✗	✗	✓	✗
FADE (Hong et al., 2021)		✓	✗	✗	✗	✓	✗
FairFed (Ezzeldin et al., 2021)	✓	✓	✗	✓	✗	✓	✗
Papadaki (Papadaki et al., 2021)	✓	✓	✗	✗	✗	✓	✗
PrivFairFL	*	✓	✓	✓	✓	*	✓

Private aggregation techniques. Statistics about the underlying data distribution are commonly used to improve the ML model learning process. Various works have shown that statistics about the data distribution across clients in FL can improve the utility of models or make them more fair, especially for clients with imbalanced or non-i.i.d. data (Duan et al., 2019; Du et al., 2021). The challenge is to collect the statistics without infringing upon the clients' privacy. Solutions for aggregation with MPC to protect the private input data have been recently employed in FL to train group-fair models (Ezzeldin et al., 2021; Zhang et al., 2020). However, the output of such MPC-based aggregations can still leak information about the client's data. In `PrivFairFL` we go a step further by adding noise to the aggregated values to provide DP guarantees. Existing works for aggregation do so by having the clients participate in the noise generation. Such solutions are not resilient to malicious clients, and require extensive communication between clients and the aggregator (Ács and Castelluccia, 2011; Bindschaedler et al., 2017). Our proposal correctly generates the noise in a secure way, inside MPC protocols that are resilient to corruptions by semi-honest and malicious adversaries (see Sec. 4).

4. Methodology

Computational setting. There are M clients who each have a dataset D_k as described in Sec. 2. There are $r \geq 2$ computing parties (servers) who can execute MPC protocols and perform other computations to aid the clients in model training. We propose two strategies for bias mitigation, namely one that is applied before model training (`PrivFairFL-Pre`) and one that is applied after model training (`PrivFairFL-Post`). Both strategies are independent of the model training phase, which means that they can be combined with any technique for model training in FL, including DP-SGD (Abadi et al., 2016). While all r servers participate in the MPC protocols for bias mitigation, the aggregation of the weights or gradients during model training can either be performed by one of the servers – which then acts like the traditional central aggregator in traditional FL – or it can be implemented as a straightforward MPC protocol ran by all the servers. The methods described in this section are generic and work with either aggregation setup during the model training phase. Each strategy is based on the clients in the federation sending encrypted shares of their information to the r computing parties in an "MPC as a service" setting. These parties (1) run MPC protocols to perform computations for bias mitigation, including an MPC subprotocol to add secret-shared noise to the secret-shared results of those computations to provide the desired DP guarantees, and (2) publish the outcomes of this process back to the clients. The two strategies differ in when the MPC protocols are executed (before or after model training), what the input and output is, and what computations are being performed. Performing bias mitigation inside MPC protocols yields the same utility and fairness as one could achieve with global DP, with the added advantage that the clients do not need to disclose their data to a trusted curator.

Protocol 1: π_{LAP} for secure sampling from Laplacian distribution

Input: Scale b
Output: Secret-shared value $[\![x]\!]$ drawn from Laplace distribution with mean 0 and scale b

$[\![u]\!] \leftarrow \pi_{\mathsf{GR-RANDOM}}(-0.5, 0.5)$
$[\![\mathrm{sgn}_u]\!] \leftarrow \pi_{\mathsf{GTE}}([\![u]\!], 0)$
$[\![\mathrm{abs}_u]\!] \leftarrow \pi_{\mathsf{MUL}}([\![u]\!], [\![\mathrm{sgn}_u]\!]))$
$[\![\ln_u]\!] \leftarrow \pi_{\mathsf{LN}}(1 - 2 \cdot [\![\mathrm{abs}_u]\!])$
$[\![x]\!] \leftarrow -b \cdot \pi_{\mathsf{MUL}}([\![\ln_u]\!], [\![\mathrm{sgn}_u]\!])$
return $[\![x]\!]$

4.1. Privacy Preserving Pre-processing for Bias Mitigation

In `PrivFairFL-Pre`, the unified training dataset D is debiased by assigning weights to the samples based on the values of S and/or Y. Such global assignment of weights on the unified data addresses the heterogeneous data distributions in cross-device setups. A reweighing technique from the centralized learning domain is to assign a training instance with sensitive attribute value s and label y a weight of $1/C(s, y)$, with $C(s, y)$ the total number of examples in the training data with that sensitive attribute and label value (Yu, 2021). We adopt this idea in protocol π_{RW}, and explain how π_{RW} can be extended to other weight balancing techniques in the Appendix.

In FL, each client k has its local dataset D_k, and needs to obtain weights for the instances in D_k that are based on counts $C(s, y)$ across the entire federation. To obtain these weights, each client k starts by counting the number of negative and positive instances in D_k. We denote these local counts as $LC(0, k)$ and $LC(1, k)$ respectively, for $k = 1 \ldots M$. Next each client encrypts this information by splitting it into secret shares and sending it to the computing parties P_i $(i = 1 \ldots r)$,[3] along with secret shares of $T(k)$ which denotes whether client k belongs to the protected group $T(k) = 1$ or to the unprotected group $T(k) = 0$. None of the computing parties (servers) can derive any information about the local counts or sensitive attribute values from the secret shares received.

3. We implemented our solutions for $r = 2, 3$ and 4 (see Sec. 5) but they are general and work with any number of computing parties.

Protocol 2: π_{RW} for privacy-preserving reweighing of training data

Input : Total number M of clients; secret-shares of vector T of length M denoting whether client k belongs to the protected group ($T(k) = 1$) or to the unprotected group ($T(k) = 0$); secret-shares of $2 \times M$ matrix LC with local counts of number of negative instances $LC(0, k)$ and positive instances $LC(1, k)$ for client k; ϵ privacy budget allotted for bias mitigation

Output : Secret-shares of sample weights for negative and positive instances in protected and unprotected groups

1 **for** s in $\{p, u\}$ and y in $\{0, 1\}$ **do**
2   Set $C(s, y)$ to 0.
3 **end**
4 **for** $k \leftarrow 1$ **to** M **do**
5   **for** y in $\{0, 1\}$ **do**
6         $[\![LCPr(y)]\!] \leftarrow \pi_{\mathsf{MUL}}([\![T(k)]\!], [\![LC(y, k)]\!])$
7         $[\![C(p, y)]\!] \leftarrow [\![C(p, y)]\!] + [\![LCPr(y)]\!]$
8         $[\![C(u, y)]\!] \leftarrow [\![C(u, y)]\!] + [\![LC(y, k)]\!] - [\![LCPr(y)]\!]$
9   **end**
10 **end**
11 **for** s in $\{p, u\}$ and y in $\{0, 1\}$ **do**
12     $[\![C(s, y)]\!] \leftarrow [\![C(s, y)]\!] + \pi_{\mathsf{LAP}}\ (1/\epsilon)$
13 **end**
14 Set N' to 0.
15 **for** s in $\{p, u\}$ and y in $\{0, 1\}$ **do**
16     $[\![N']\!] \leftarrow [\![N']\!] + [\![C(s, y)]\!]$
17 **end**
18 **for** s in $\{p, u\}$ and y in $\{0, 1\}$ **do**
19     $[\![W(s, y)]\!] \leftarrow \pi_{\mathsf{DIV}}\ ([\![N']\!], 4 \cdot [\![C(s, y)]\!])$
20 **end**
21 **return** $[\![W(s, y)]\!]$

In Line 1–10 in Prot. 2, the computing parties compute secret shares of $C(s, y)$, for all values of s and y, from the secret shares of the clients' local counts $LC(y, k)$. As is common in MPC protocols, we use multiplication instead of control flow logic for conditional assignments. In this way, the number and the kind of operations executed by the parties does not depend on the actual values of the inputs, so it does not leak information that could be exploited by side-channel attacks. For instance, we rewrite "**if** $T(k) = 1$ **then** $C(p, y) \leftarrow C(p, y) + LC(y, k)$ **else** $C(p, y) \leftarrow C(p, y)$" as "$C(p, y) \leftarrow C(p, y) + T(k) \cdot LC(y, k)$" (Line 7–8). Similarly, '**if** $T(k) = 0$ **then** $C(u, y) \leftarrow C(u, y) + LC(y, k)$ **else** $C(u, y) \leftarrow C(u, y)$" is rewritten as "$C(u, y) \leftarrow C(u, y) + (1 - T(k)) \cdot LC(y, k)$". The last expression can be rewritten as $C(u, y) + LC(y, k) - T(k) \cdot LC(y, k)$, allowing us to take advantage of the already computed multiplication in Line 6 to arrive at Line 8. The counts $C(s, y)$ depend on disjoint subsets of the unified dataset D (4 such disjoint subsets, given that the sensitive attribute and the class label are binary). The results of the counts can be made public under ϵ-DP guarantee with the Laplace mechanism, i.e. by adding Laplace noise with magnitude $1/\epsilon$ (noting that the sensitivity of each count query is 1, and that the parallel composition property of DP is applicable). To add such noise, on Line 12, the parties call protocol π_{LAP} to obtain a secret-shared value sampled from the Laplace distribution with mean 0 and median $1/\epsilon$. Pseudocode for π_{LAP} is provided separately in Prot. 1 and explained in the Appendix. All the remaining computation in Prot. 2 depend only on the now noisy counts. Because of DP's post-processing property, the outcomes of those computations also satisfy ϵ-DP. The computing parties then execute the MPC-protocol for division π_{DIV} to compute secret shares of the weights on Line 19. The multiplication with constant 4 (4 is the number of sensitive attribute values times the number of label values) and with N' serve to rescale the weights so that they sum up to N', which is the sum of the noisy counts $C(s, y)$ for all s and y (and likely differs from the real total number N of training examples). At the end of the protocol, the computing parties hold secret shares of the computed weights protected under DP guarantees, which they then publish them so that the clients can use them for model training. π_{RW} not only protects the values of the clients' labels and sensitive attributes but also the value of the noise added to the computed weights.

Sampling from a Laplace distribution $Lap(0, b)$ with zero mean and scale b in a privacy-preserving manner is achieved by letting the parties execute protocol π_{LAP} to compute a secret-sharing of $x = -b \cdot \text{sgn}(u) \cdot \ln(1 - 2|u|)$, where u is a random variable drawn from a uniform distribution in $[-0.5, 0.5]$.[4] To obtain a secret-sharing of u, on Line 1 in in Prot. 1, the parties execute $\pi_{\mathsf{GR-RANDOM}}$. In $\pi_{\mathsf{GR-RANDOM}}$, each party generates d random bits, where d is the fractional precision of the power 2 ring representation of real numbers, and then the parties define the bitwise XOR of these d bits as the binary representation of the random number jointly generated. The rest of π_{LAP} is fairly straightforward. π_{LN} called on Line 4 is a known MPC protocol for computing the natural logarithm of a secret-shared value (Keller, 2020).

Extending π_{RW} to other reweighing algorithms. Protocol π_{RW} can be easily extended to other reweighing techniques (Kamiran and Calders, 2012) by computing the required statistics and adding noise to them before Line 19, and replacing Lines 11–20 to compute the weights according to the technique. For example, for reweighing techniques that balance based only on the parties would compute $C(y) = \sum_s C(s, y)$ after Line 10 (which is straightforward using addition of secret shares), which would be followed by a for-loop that iterates over y and computes $[\![W(y)]\!] \leftarrow \pi_{\mathsf{DIV}}(N', 2 \cdot [\![C(y)]\!])$.

4.2. Privacy Preserving Post-processing for Bias Mitigation

In `PrivFairFL-Post`, the predicted *outcomes* are debiased by finding optimal classification thresholds for each group. In the centralized paradigm, this is done by constructing ROC-curves to find thresholds that maximize the given objective (Bird et al., 2020; Hardt et al., 2016). This requires labels and predictions over the unified training data which violates privacy in the FL scenario. Below we detail a solution for obtaining ROC-curves and optimal classification thresholds in a privacy-preserving manner in FL. `PrivFairFL-Post` is applied *after* the training phase, so we assume that each client has obtained the last model θ. Next:

1. Each client k generates predictions (probabilities) with θ over its local training dataset. Subsequently each client secret-shares with the MPC servers its sensitive attribute value, the predicted probabilities, and the ground truth labels for its local training examples. None of the MPC servers can derive any of this information from the encrypted shares received.
2. The MPC servers run protocol π_{ROC} (see Prot. 4) to construct secret shares of the ROC curves for the protected and the unprotected group. Furthermore, noise is added inside the MPC protocol so that the ROC curves can be made public with ϵ-DP guarantees.
3. All MPC servers send their secret shares of the noisy ROC curves to one of the MPC servers, who then computes optimal thresholds, and sends these to the clients.

In protocol π_{ROC}, the parties construct ROC curves by computing secret-sharings of FPR and TPR values for a list of predefined candidate thresholds (Line 1 in Prot. 4). On Line 3–19, they compute secret shares of the TPR and FPR at each threshold, for the protected and the unprotected group. On Line 5–7, the parties use the comparison protocol π_{GTE} to determine for each training instance whether it would be classified as positive or negative

4. x has distribution $Lap(0, b)$. This follows from the inverse cumulative distribution function for $Lap(0, b)$.

Protocol 3: π_{CF} for generation of confusion matrix for protected and unprotected group

Input : Total number of samples as used for FL N, secret-shares of vectors Y, $\hat{Y}$ and S of length N
Output : Secret-shares of confusion matrix for each group

1 **for** s in $\{p, u\}$ **do**
2 | Set TP(s), FP(s), TN(s), FN(s) to 0
3 **end**
4 **for** $i \leftarrow 1$ **to** N **do**
5 | $\llbracket \text{tp} \rrbracket \leftarrow \pi_{\mathsf{MUL}}(\llbracket Y[i] \rrbracket, \llbracket \hat{Y}[i] \rrbracket)$
6 | $\llbracket \text{ys} \rrbracket \leftarrow \pi_{\mathsf{MUL}}(\llbracket Y[i] \rrbracket, \llbracket S[i] \rrbracket)$
7 | $\llbracket \text{ps} \rrbracket \leftarrow \pi_{\mathsf{MUL}}(\llbracket \hat{Y}[i] \rrbracket, \llbracket S[i] \rrbracket)$
8 | $\llbracket \text{tps} \rrbracket \leftarrow \pi_{\mathsf{MUL}}(\llbracket \text{tp} \rrbracket, \llbracket S[i] \rrbracket)$
9 | $\llbracket \text{TP}(p) \rrbracket \leftarrow \llbracket \text{TP}(p) \rrbracket + \llbracket \text{tps} \rrbracket$
10 | $\llbracket \text{TP}(u) \rrbracket \leftarrow \llbracket \text{TP}(u) \rrbracket + (\llbracket \text{tp} \rrbracket - \llbracket \text{tps} \rrbracket)$
11 | $\llbracket \text{FP}(p) \rrbracket \leftarrow \llbracket \text{FP}(p) \rrbracket + (\llbracket \text{ps} \rrbracket - \llbracket \text{tps} \rrbracket)$
12 | $\llbracket \text{FP}(u) \rrbracket \leftarrow \llbracket \text{FP}(u) \rrbracket + (\llbracket \hat{Y}[i] \rrbracket - \llbracket \text{ps} \rrbracket - \llbracket \text{tp} \rrbracket + \llbracket \text{tps} \rrbracket)$
13 | $\llbracket \text{FN}(p) \rrbracket \leftarrow \llbracket \text{FN}(p) \rrbracket + (\llbracket \text{ys} \rrbracket - \llbracket \text{tps} \rrbracket)$
14 | $\llbracket \text{FN}(u) \rrbracket \leftarrow \llbracket \text{FN}(u) \rrbracket + (\llbracket Y[i] \rrbracket - \llbracket \text{ys} \rrbracket - \llbracket \text{tp} \rrbracket + \llbracket \text{tps} \rrbracket)$
15 | $\llbracket \text{TN}(p) \rrbracket \leftarrow \llbracket \text{TN}(p) \rrbracket + (\llbracket S[i] \rrbracket - \llbracket \text{ys} \rrbracket - \llbracket \text{ps} \rrbracket + \llbracket \text{tps} \rrbracket)$
16 | $\llbracket \text{TN}(u) \rrbracket \leftarrow \llbracket \text{TN}(u) \rrbracket + (1 - \llbracket S[i] \rrbracket - \llbracket Y[i] \rrbracket + \llbracket \text{ys} \rrbracket - \llbracket \hat{Y}[i] \rrbracket + \llbracket \text{ps} \rrbracket + \llbracket \text{tp} \rrbracket - \llbracket \text{tps} \rrbracket)$
17 **end**
18 **return** $(\llbracket \text{TP}(s) \rrbracket, \llbracket \text{TN}(s) \rrbracket, \llbracket \text{FP}(s) \rrbracket, \llbracket \text{FN}(s) \rrbracket \mid s \in \{u, p\})$

Protocol 4: π_{ROC} for privacy-preserving debiasing of predicted outcomes

Input : Total number of samples N as used for FL , vectors of secret-shares of true labels Y, predicted probabilities $\hat{Y}_{prob}$ and S each of length N; privacy budget ϵ allotted for bias mitigation
Output : Secret-shares of ROC curves for protected and unprotected groups

1 $\llbracket \text{thresholds} \rrbracket \leftarrow \llbracket (0.000, 0.001, 0.002, \ldots, 0.999, 1.000) \rrbracket$; Set T to 1001
2 Declare $\llbracket \text{ROC}(p) \rrbracket$ and $\llbracket \text{ROC}(u) \rrbracket$ as secret-shared $3 \times T$ arrays.
3 Declare $\hat{Y}_{th}$ as an N-dimensional vector // *holds predictions at thresholds*
4 **for** $j \leftarrow 1$ **to** T **do**
5 | **for** $i \leftarrow 1$ **to** N **do**
6 | | $\llbracket \hat{Y}_{th}[i] \rrbracket \leftarrow \pi_{\mathsf{GTE}}\ (\llbracket \hat{Y}_{prob}[i] \rrbracket, \llbracket \text{thresholds}[j] \rrbracket)$
7 | **end**
8 | $\llbracket \text{TP}(p) \rrbracket, \llbracket \text{TN}(p) \rrbracket, \llbracket \text{FP}(p) \rrbracket, \llbracket \text{FN}(p) \rrbracket, \llbracket \text{TP}(u) \rrbracket, \llbracket \text{TN}(u) \rrbracket, \llbracket \text{FP}(u) \rrbracket, \llbracket \text{FN}(u) \rrbracket \leftarrow \pi_{\mathsf{CF}}(\llbracket Y \rrbracket, \llbracket \hat{Y}_{th} \rrbracket, \llbracket S \rrbracket)$
9 | **for** s in $\{p, u\}$ **do**
10 | | $\llbracket \text{TP}(s) \rrbracket \leftarrow \llbracket \text{TP}(s) \rrbracket + \pi_{\mathsf{LAP}}\ (1/\epsilon)$; $\llbracket \text{TN}(s) \rrbracket \leftarrow \llbracket \text{TN}(s) \rrbracket + \pi_{\mathsf{LAP}}\ (1/\epsilon)$
11 | | $\llbracket \text{FP}(s) \rrbracket \leftarrow \llbracket \text{FP}(s) \rrbracket + \pi_{\mathsf{LAP}}\ (1/\epsilon)$; $\llbracket \text{FN}(s) \rrbracket \leftarrow \llbracket \text{FN}(s) \rrbracket + \pi_{\mathsf{LAP}}\ (1/\epsilon)$
12 | | $\llbracket \text{TPR}(s) \rrbracket \leftarrow \pi_{\mathsf{DIV}}(\llbracket \text{TP}(s) \rrbracket, \llbracket \text{TP}(s) \rrbracket + \llbracket \text{FN}(s) \rrbracket)$
13 | | $\llbracket \text{FPR}(s) \rrbracket \leftarrow \pi_{\mathsf{DIV}}(\llbracket \text{FP}(s) \rrbracket, \llbracket \text{FP}(s) \rrbracket + \llbracket \text{TN}(s) \rrbracket)$
14 | | $\llbracket \text{ROC}(s)[1, j] \rrbracket \leftarrow \llbracket \text{FPR}(s) \rrbracket$
15 | | $\llbracket \text{ROC}(s)[2, j] \rrbracket \leftarrow \llbracket \text{TPR}(s) \rrbracket$
16 | | $\llbracket \text{ROC}(s)[3, j] \rrbracket \leftarrow \llbracket \text{thresholds}[j] \rrbracket$
17 | **end**
18 **end**
19 **return** $\llbracket \text{ROC}(p) \rrbracket$, $\llbracket \text{ROC}(u) \rrbracket$

based on the jth threshold. This information, along with the ground truth labels Y and the sensitive attribute values S is then passed to a subprotocol π_{CF} that returns secret-shares of TP, TN, FP and FN for each group based on the jth threshold. Code for π_{CF} is provided separately as Prot. 3; it is designed to minimize the amount of multiplications, and to avoid control flow statements (if-then-else) which would make the number of instructions performed dependent on the values of the input. The sensitivity of each of the returned counts is 1, and they are based on disjoint subsets of the data D, so to provide ϵ-DP, on Line 10–11 the parties draw secret-shared noise from $Lap(0, 1/\epsilon)$ with π_{LAP} (cfr. Prot. 1) and add it to the counts. All MPC servers then send their secret shares of the lists of FPRs, TPRs, and corresponding thresholds to one of the MPC servers who finds the optimal classification threshold for each group, following the procedure in Bird et al. (2020).

4.3. Extensions to `PrivFairFL`

Extending `PrivFairFL` to cross-silo setups is straightforward. In `PrivFairFL-Pre`, the clients will now instead secret-share local counts $LC(y, k, s)$ for *each* value of S (the count can be 0 if a certain sensitive attribute value is not present with the client), requiring only a small modification in the code for π_{RW} (Prot. 2). `PrivFairFL-Post` can be used as is even in the cross-silo setup, as it is independent of the data distribution among the clients in FL. A straight-forward technique to extend our defined protocols for multi-class classification or multi-valued sensitive attributes is to adopt a one-vs-rest approach for each class and/or each value of sensitive attribute. This would then require the clients to release statistics for each binary combination of S and Y in a pre-defined sequence. Our approach can also be utilized in dynamic scenarios with client dropout and change in local data distributions, by computing the weights to be assigned to the training samples after a set of FL rounds. This will lead to a technique that is a combination of the pre-processing and in-processing techniques.

5. Results and Conclusion

We evaluate `PrivFairFL` on ADS (Roffo and Vinciarelli, 2016) and ML-1M (Harper and Konstan, 2016). We implemented our MPC protocols for the pre-processing and the post-processing phase in MP-SPDZ (Keller, 2020). For the runtime experiments, every computing party ran on a separate VM instance co-located F48s V2 Azure virtual machines each of which contains 48 cores, 96 GiB of memory, and network bandwidth of up to 21 Gb/s. All computations in Tab. 3 are done with $q = 2^{64}$ (See Appendix B). We implemented the model training phase in Python using Flower (Beutel et al., 2020) for FL. We use FedAvg (McMahan et al., 2017) as the aggregation strategy for the weights and add noise to the model parameters locally to provide DP guarantees during FL. We train some of the FL models with DP-SGD (Abadi et al., 2016) as specified in Table 2. We provide a more detailed overview of the datasets, model architectures and hyperparameters in the Appendix, and make all code publicly available. We investigate the following research questions:

Q1: To what extent do our proposed pre-processing and post-processing approaches in `PrivFairFL` improve fairness while addressing privacy? From the results in Tab. 2, it is seen that the fairness mitigation techniques aid in achieving group fairness in FL, and that the best type of technique depends on the dataset. `PrivFairFL-Pre` benefits the highly imbalanced data in ADS, while `PrivFairFL-Post` successfully finds optimal thresholds for the almost balanced subset of data in ML-1M. We also observe that randomization provided by DP guarantees can either aid in achieving a group-fair model or worsen the bias existing in the datasets. Our MPC+DP approaches not only aid in achieving group fairness while providing privacy guarantees but also improve the utility of the model when compared to the baseline approach using local DP (as done by Abay et al. (2020)). We observe the fairness-utility trade-off in `PrivFairFL-Pre` and `PrivFairFL-Post`.

Q2: How does `PrivFairFL` affect performance on real-world datasets? Based on our evaluation, `PrivFairFL` can lead to less accurate models as opposed to models trained in CL and pure-FL. We think this is due to trade-offs for training fair models. Privacy-preserving techniques like DP can randomly affect the utility of the model, while our MPC+DP based approaches reduce such trade-offs and, as observed, improve utility over both the datasets.

Table 2: Utility and fairness for $\epsilon = 1$ averaged over three runs with different value of seed

	Fairness	Privacy	ADS					ML-1M				
			Acc.	|1-DI|	ΔEOP	ΔEODD	ΔSP	Acc.	|1-DI|	ΔEOP	ΔEODD	ΔSP
CL	–	–	85.81%	1.214	0.070	0.037	0.004	62.15%	0.138	0.091	0.141	0.094
FL	–	–	85.04%	1.654	0.089	0.050	0.018	59.04%	0.096	0.081	0.096	0.091
FL-DP-SGD	–	DP-SGD	85.21%	6.606	0.070	0.043	0.023	58.30%	0.027	0.026	0.027	0.052
FL-DP-SGD	Pre	Local DP	83.52%	0.260	0.036	0.033	0.032	58.47%	0.045	0.042	0.052	0.061
PrivFairFL	Pre	MPC+DP	83.57%	0.122	0.018	0.027	0.036	58.52%	0.045	0.042	0.051	0.063
PrivFairFL	Post	MPC+DP	85.17%	0.572	0.008	0.005	0.001	58.46%	0.006	0.006	0.014	0.045

Q3: How does `PrivFairFL` compare with existing work? As per our evaluation, `PrivFairFL` provides improvement over the baseline local DP approach based on Abay et al. (2020)[5] w.r.t. utility and fairness. `PrivFairFL` unlike the other techniques in Tab. 1, not only reduce bias but also fully protects the data with provable privacy guarantees with little/no cost in performance.

Table 3: Runtimes for different MPC schemes for 2PC (dishonest majority) and 3PC/4PC (honest majority). OTSemi2k is semi-honest adapt. of Cramer et al. (2018), Replicated2k is Araki et al. (2016), SPDZ2k is Cramer et al. (2018); Damgård et al. (2019), SPDZ-wise Replicated2k is Dalskov et al. (2021), Rep4-2k is Dalskov et al. (2021)

			ADS		ML-1M	
		MPC scheme	`Pre-proc.`	`Post-proc.`	`Pre-proc.`	`Post-proc.`
passive	2PC	OTSemi2k	0.10 sec	43093.79 sec	0.10 sec	13088.52 sec
	3PC	Replicated2k	0.02 sec	7993.73 sec	0.02 sec	2427.87 sec
active	2PC	SPDZ2k	3.55 sec	1199060.00 sec	3.54 sec	364180.50 sec
	3PC	SPDZ-wise Replicated2k	0.06 sec	16832.09 sec	0.05 sec	5112.27 sec
	4PC	Rep4-2k	0.02 sec	8673.63 sec	0.02 sec	2634.37 sec

Q4: How does `PrivFairFL` scale? We address the scalability of `PrivFairFL` based on the number of clients and computing parties. Tab. 3 contains an overview of runtimes of our protocols for different MPC schemes with different computing parties and security settings. The runtime results show substantial difference between passive and active security settings and between honest (3PC/4PC) and dishonest (2PC) majority settings; these results align with existing literature (Dalskov et al. (2020, 2021)). Our experiments are evaluated over a set of 75 and 109 clients, demonstrating the scalability of our approach. As could be expected, `PrivFairFL-Post` takes much longer than `PrivFairFL-Pre` to complete. We argue that even the longest runtimes of `PrivFairFL-Post` are an acceptable price to pay for training fair models in a privacy-preserving way.

Conclusion & Future Directions. In this paper, we proposed `PrivFairFL`, an MPC-based framework for training group-fair models in Federated Learning (FL). We proposed pre- and post-processing bias mitigation algorithms in cross-device FL under complete and formal privacy guarantees. We showed that `PrivFairFL` not only efficiently balances the data and

5. Adopted for cross-silo setup where each client adds noise to their local counts and uses randomized response to publish the value of their sensitive attribute.

provides fair predictions, but also preserves the model quality. As a next step, we want to add in-processing bias mitigation techniques to `PrivFairFL`. In addition to the group fairness notions that we have used in this paper, we plan to investigate other fairness notions, such as individual and causal notions of fairness, in our future work.

Acknowledgments

The authors would like to thank Marcel Keller for making the MP-SPDZ framework available. Funding support for project activities has been partially provided by The University of Washington Tacoma's Founders Endowment fund, Canada CIFAR AI Chair, Facebook Research Award for Privacy Enhancing Technologies, and the Google Cloud Research Credits Program. This research was enabled in part by support provided by Calcul Québec and Compute Canada.

References

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, 2016.

Annie Abay, Yi Zhou, Nathalie Baracaldo, Shashank Rajamoni, Ebube Chuba, and Heiko Ludwig. Mitigating bias in federated learning. *arXiv preprint arXiv:2012.02447*, 2020.

Gergely Ács and Claude Castelluccia. I have a dream! (DiffeRentially privatE smArt Metering). In *International Workshop on Information Hiding*, pages 118–132. Springer, 2011.

Alekh Agarwal, Alina Beygelzimer, Miroslav Dudík, John Langford, and Hanna Wallach. A reductions approach to fair classification. In *International Conference on Machine Learning*, pages 60–69. PMLR, 2018.

Muhammad Ali, Piotr Sapiezynski, Miranda Bogen, Aleksandra Korolova, Alan Mislove, and Aaron Rieke. Discrimination through optimization: How Facebook's ad delivery can lead to biased outcomes. In *Proceedings of the ACM on Human-Computer Interaction*, volume 3, pages 1–30, 2019.

Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Machine bias. *ProPublica*, May 23, 2016.

Toshinori Araki, Jun Furukawa, Yehuda Lindell, Ariel Nof, and Kazuma Ohara. High-throughput semi-honest secure three-party computation with an honest majority. In *ACM SIGSAC Conference on Computer and Communications Security*, pages 805–817, 2016.

Daniel J Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Titouan Parcollet, and Nicholas D Lane. Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390*, 2020.

Vincent Bindschaedler, Shantanu Rane, Alejandro E. Brito, Vanishree Rao, and Ersin Uzun. Achieving differential privacy in secure multiparty data aggregation protocols on star networks. CODASPY '17, page 115–125, 2017.

Sarah Bird, Miro Dudík, Richard Edgar, Brandon Horn, Roman Lutz, Vanessa Milan, Mehrnoosh Sameki, Hanna Wallach, and Kathleen Walker. Fairlearn: A toolkit for assessing and improving fairness in AI. Technical Report MSR-TR-2020-32, Microsoft, May 2020. URL https://www.microsoft.com/en-us/research/publication/fairlearn-a-toolkit-for-assessing-and-improving-fairness-in-ai/.

Franziska Boenisch, Adam Dziedzic, Roei Schuster, Ali Shahin Shamsabadi, Ilia Shumailov, and Nicolas Papernot. When the curious abandon honesty: Federated learning is not private. *arXiv preprint arXiv:2112.02918*, 2021.

Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191, 2017.

Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Proceedings of the Conference on Fairness, Accountability and Transparency*, pages 77–91. PMLR, 2018.

Flavio Calmon, Dennis Wei, Bhanukiran Vinzamuri, Karthikeyan Natesan Ramamurthy, and Kush R. Varshney. Optimized pre-processing for discrimination prevention. *Advances in Neural Information Processing Systems*, 30:3995–4004, 2017.

R. Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, 2000.

O. Catrina and S. De Hoogh. Secure multiparty linear programming using fixed-point arithmetic. In *European Symposium on Research in Computer Security*, pages 134–150. Springer, 2010.

L Elisa Celis, Lingxiao Huang, Vijay Keswani, and Nisheeth K Vishnoi. Classification with fairness constraints: A meta-algorithm with provable guarantees. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 319–328, 2019.

Ronald Cramer, Ivan Damgard, and Jesper Nielsen. *Secure Multiparty Computation and Secret Sharing*. Cambridge University Press Print, New York, 2015.

Ronald Cramer, Ivan Damgård, Daniel Escudero, Peter Scholl, and Chaoping Xing. SPD$\mathbb{Z}_{2^k}$: Efficient MPC mod 2^k for dishonest majority. In *Annual International Cryptology Conference*, pages 769–798. Springer, 2018.

Sen Cui, Weishen Pan, Jian Liang, Changshui Zhang, and Fei Wang. Addressing algorithmic disparity and performance inconsistency in federated learning. *Advances in Neural Information Processing Systems*, 34, 2021.

A. Dalskov, D. Escudero, and M. Keller. Secure evaluation of quantized neural networks. *Proceedings on Privacy Enhancing Technologies*, 2020(4):355–375, 2020.

Anders Dalskov, Daniel Escudero, and Marcel Keller. Fantastic four: Honest-majority four-party secure computation with malicious security. In *USENIX 2021*, pages 2183–2200, 2021.

I. Damgård, D. Escudero, T. Frederiksen, M. Keller, P. Scholl, and N. Volgushev. New primitives for actively-secure MPC over rings with applications to private machine learning. In *IEEE Symposium on Security and Privacy (SP)*, pages 1102–1120, 2019.

Michele Donini, Luca Oneto, Shai Ben-David, John S Shawe-Taylor, and Massimiliano Pontil. Empirical risk minimization under fairness constraints. *Advances in Neural Information Processing Systems*, 31:2796–2806, 2018.

Wei Du, Depeng Xu, Xintao Wu, and Hanghang Tong. Fairness-aware agnostic federated learning. In *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*, pages 181–189. SIAM, 2021.

Moming Duan, Duo Liu, Xianzhang Chen, Yujuan Tan, Jinting Ren, Lei Qiao, and Liang Liang. Astraea: Self-balancing federated learning for improving classification accuracy of mobile deep learning applications. In *37th International Conference on Computer Design (ICCD)*, pages 246–254. IEEE, 2019.

Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014.

Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 214–226, 2012.

e-CFR. Adverse impact and four-fifths rule. § 1607.4 Information on impact. https://www.ecfr.gov/current/title-29/subtitle-B/chapter-XIV/part-1607/subject-group-ECFRdb347e844acdea6/section-1607.4, 1981.

D. Evans, V. Kolesnikov, and M. Rosulek. A pragmatic introduction to secure multi-party computation. *Foundations and Trends in Privacy and Security*, 2(2-3):70–246, 2018.

Yahya H. Ezzeldin, Shen Yan, Chaoyang He, Emilio Ferrara, and Salman Avestimehr. FairFed: Enabling group fairness in federated learning. *arXiv preprint arXiv:2110.00857*, 2021.

Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 259–268, 2015.

Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*, 2018.

Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. *Advances in Neural Information Processing Systems*, 29, 2016.

F. Maxwell Harper and Joseph A. Konstan. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4):19:1–19:19, 2016.

Junyuan Hong, Zhuangdi Zhu, Shuyang Yu, Zhangyang Wang, Hiroko H Dodge, and Jiayu Zhou. Federated adversarial debiasing for fair and transferable representations. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 617–627, 2021.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

Faisal Kamiran and Toon Calders. Data preprocessing techniques for classification without discrimination. *Knowledge and Information systems*, 33(1):1–33, 2012.

Faisal Kamiran, Asim Karim, and Xiangliang Zhang. Decision theory for discrimination-aware classification. In *12th International Conference on Data Mining*, pages 924–929. IEEE, 2012.

Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. Fairness-aware classifier with prejudice remover regularizer. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 35–50. Springer, 2012.

Samhita Kanaparthy, Manisha Padala, Sankarshan Damle, and Sujit Gujar. Fair federated learning for heterogeneous face data. *arXiv preprint arXiv:2109.02351*, 2021.

Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. In *International Conference on Machine Learning*, pages 2564–2572. PMLR, 2018.

Marcel Keller. MP-SPDZ: A versatile framework for multi-party computation. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 1575–1590, 2020.

Jon Kleinberg, Sendhil Mullainathan, and Manish Raghavan. Inherent trade-offs in the fair determination of risk scores. *arXiv preprint arXiv:1609.05807*, 2016. URL https://arxiv.org/pdf/1609.05807.pdf.

Jeff Larson, Surya Mattu, Lauren Kirchner, and Julia Angwin. How we analyzed the compas recidivism algorithm. *ProPublica (5 2016)*, 9(1):3–3, 2016.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017.

Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. Agnostic federated learning. In *International Conference on Machine Learning*, pages 4615–4625. PMLR, 2019.

Manisha Padala, Sankarshan Damle, and Sujit Gujar. Federated learning meets fairness and differential privacy. In *International Conference on Neural Information Processing*, pages 692–699. Springer, 2021.

Afroditi Papadaki, Natalia Martinez, Martin Bertran, Guillermo Sapiro, and Miguel Rodrigues. Federating for learning group fair models. *arXiv preprint arXiv:2110.01999*, 2021.

Matthias Paulik, Matt Seigel, Henry Mason, Dominic Telaar, Joris Kluivers, Rogier van Dalen, Chi Wai Lau, Luke Carlson, Filip Granqvist, Chris Vandevelde, et al. Federated evaluation and tuning for on-device personalization: System design & applications. *arXiv preprint arXiv:2102.08503*, 2021.

Dana Pessach and Erez Shmueli. Algorithmic fairness. *arXiv preprint arXiv:2001.09784*, 2020.

Borja Rodríguez-Gálvez, Filip Granqvist, Rogier van Dalen, and Matt Seigel. Enforcing fairness in private federated learning via the modified method of differential multipliers. *arXiv preprint arXiv:2109.08604*, 2021.

Giorgio Roffo and Alessandro Vinciarelli. Personality in computational advertising: A benchmark. In *4 th Workshop on Emotions and Personality in Personalized Systems (EMPIRE) 2016*, page 18, 2016.

Tanushree Shenwai. Google AI implements machine learning model that employs federated learning with differential privacy guarantees. MarkTechPost, 2022.

Till Speicher, Muhammad Ali, Giridhari Venkatadri, Filipe Nunes Ribeiro, George Arvanitakis, Fabrício Benevenuto, Krishna P Gummadi, Patrick Loiseau, and Alan Mislove. Potential for discrimination in online targeted advertising. In *Proceedings of the Conference on Fairness, Accountability and Transparency*, pages 5–19. PMLR, 2018.

Latanya Sweeney. Discrimination in online ad delivery: Google ads, black names and white names, racial discrimination, and click advertising. *Queue*, 11(3):10–29, 2013.

Sahil Verma and Julia Rubin. Fairness definitions explained. In *2018 IEEE/ACM International Workshop on Software Fairness (FairWare)*, pages 1–7. IEEE Computer Society, 2018.

Neil Vigdor. Apple card investigated after gender discrimination complaints. *The New York Times*, Nov 10, 2019.

Sheridan Wall and Hilke Schellmann. LinkedIn's job-matching AI was biased. The company's solution? More AI. *MIT Technology Review*, June 23, 2021.

Shen Yan, Hsien-te Kao, and Emilio Ferrara. Fair class balancing: enhancing model fairness without observing sensitive attributes. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 1715–1724, 2020.

Zhe Yu. Fair balance: Mitigating machine learning bias against multiple protected attributes with data balancing. *arXiv preprint arXiv:2107.08310*, 2021.

Xubo Yue, Maher Nouiehed, and Raed Al Kontar. Gifair-fl: An approach for group and individual fairness in federated learning. *arXiv preprint arXiv:2108.02741*, 2021.

Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. Learning fair representations. In *International Conference on Machine Learning*, pages 325–333. PMLR, 2013.

Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell. Mitigating unwanted biases with adversarial learning. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 335–340, 2018.

Daniel Yue Zhang, Ziyi Kou, and Dong Wang. Fairfl: A fair federated learning approach to reducing demographic bias in privacy-sensitive classification models. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 1051–1060. IEEE, 2020.

Fengda Zhang, Kun Kuang, Yuxuan Liu, Chao Wu, Fei Wu, Jiaxun Lu, Yunfeng Shao, and Jun Xiao. Unified group fairness on federated learning. *arXiv preprint arXiv:2111.04986*, 2021.

Zirui Zhou, Lingyang Chu, Changxin Liu, Lanjun Wang, Jian Pei, and Yong Zhang. Towards fair federated learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 4100–4101, 2021.