Privacy-preserving post-processing for multicalibration

Leor Fishman **Vaikkunth Mugunthan** **Garrett Tanzer**

Abstract

It is crucial that machine-learned classifiers are both *private* and *fair*, in order to protect individuals in the training set and individuals affected by the classifier's decisions respectively. In this work, we present two algorithms for post-processing an arbitrary classifier to satisfy *multicalibration*—a particular fairness definition—in a privacy-preserving way. We characterize their behavior with theoretical bounds and empirically evaluate their performance on a real-world census income dataset. We find in both approaches that multicalibration improves test accuracy, with little negative effect from differential privacy.

1 Introduction

Machine learning algorithms are increasingly used to train classifiers on *sensitive datasets* in order to make *sensitive decisions*. For example, models may be trained on consumer financial data and used to decide creditworthiness, or trained on patient medical data and used to make diagnostic decisions. It is important that these training algorithms and their resulting models are both *private*, meaning that they don't reveal sensitive information about the individuals in the training set, and *fair*, in the sense that their decisions are not discriminatory.

There is a rich and rapidly growing body of research on each of these properties in isolation. Dwork et al.'s notion of *differential privacy* [9] is the dominant mathematical privacy framework: roughly, it guarantees that an individual wouldn't be harmed by choosing to be included in a dataset, because an adversary cannot tell whether the individual's data was in fact used. This definition prevents a multitude of privacy attacks [14, 15, 17], like reconstruction and membership inference attacks, on models trained using naive machine learning algorithms; a variety of works [2, 3, 12] modify these training algorithms to be differentially private, by adding noise to certain parts of the training procedure.

One line of research in fairness, starting with Dwork et al.'s "Fairness Through Awareness" [7], aims to prevent discrimination by paying more attention—rather than less—to sensitive characteristics and using this to define exactly what fairness means for the task at hand. Later works [10, 13] recast fairness as something closer to "faithfulness to the data", with *multifairness* criteria. They ensure that the classifier's predictions are fair with respect to a collection of groups defined computationally, in the following sense: each group's membership function is a predicate on the input features that can be computed within some complexity class. In both of these cases, it is advantageous to include sensitive data in an individual's representation—the more data, the more dimensions along which we can prevent discrimination. These criteria are sometimes achieved through *post-processing* rather than during ordinary training, so that we avoid having to dissect legacy models or sophisticated, black-box training algorithms. In the post-processing scenario, we take some pre-existing model that may or may not be multifair *and* may or may not be accurate, and process it in a way such that it becomes both accurate and multifair. This still requires the use of a training set, where again sensitive attributes are more important than ever—and therefore so is the privacy of individuals in the training set.

Despite this increased motivation, there is comparatively little work on the intersection of privacy and fairness [4, 11]—i.e., on modifying fair algorithms to be simultaneously private. Our contributions are as follows:

- We synthesize prior techniques from the privacy and fairness literatures into two algorithms, which post-process classifiers for *multicalibration*—one multifairness criterion—in a differentially private way.
- We provide theoretical bounds that characterize the privacy, fairness, and runtime of these approaches.
- We empirically evaluate these approaches on a real-world census income dataset [16].

The remainder of the paper is organized as follows: in Section 2, we discuss relevant prior works and definitions. In Section 3, we describe our two algorithms and provide theoretical bounds. In Section 4, we discuss experimental results. And in Section 5, we conclude.

2 Background and Related Work

There are two relevant dimensions of prior work—privacy and fairness—as well as the body of work at their intersection.

2.1 Privacy

In a seminal work, Dwork et al. [9] propose *differential privacy*, a definition of the privacy loss associated with any sort of data release drawn from a statistical database.

Definition 1. (ϵ, δ)*-Differential Privacy***.** A randomized mechanism $\mathcal{M}$ satisfies (ϵ, δ)-differential privacy when there exists $\epsilon > 0$, $\delta > 0$, such that

$$\text{Pr}\left[\mathcal{M}(D_1) \in S\right] \leq e^{\epsilon}\text{Pr}\left[\mathcal{M}(D_2) \in S\right] + \delta$$

holds for every $S \subseteq \text{Range}(\mathcal{M})$ and for all datasets D_1 and D_2 differing on at most one element.

When $\delta = 0$, we say that the randomized mechanism $\mathcal{M}$ satisfies ϵ-differential privacy.

Definition 2. *Global Sensitivity***.** For any real-valued query function $t : \mathcal{D} \to \mathbb{R}$, where $\mathcal{D}$ denotes the set of all possible datasets, the global sensitivity Δ of t is defined as

$$\Delta = \max_{\mathcal{D}_1 \sim \mathcal{D}_2} |t(\mathcal{D}_1) - t(\mathcal{D}_2)|,$$

for all $\mathcal{D}_1 \in \mathcal{D}$ and $\mathcal{D}_2 \in \mathcal{D}$.

2.1.1 Learning differentially fair classifiers

Differentially private empirical risk minimization (DP-ERM) is the task of learning a hypothesis that minimizes loss on a training set, in a way that is differentially private with respect to the training set. Chaudhuri et al. [3] proposes two techniques for DP-ERM: *objective perturbation* and *output perturbation*—these correspond to adding noise to the loss function and learned weights respectively. Bassily et al. [2] instead uses *gradient perturbation*, where differential privacy is ensured by adding noise to the gradient during optimization, provided that the loss function is G-Lipschitz. Kang et al. [12] develop an input perturbation technique, where noise is added to the dataset itself, showing that we get similar accuracy guarantees to that of other perturbation techniques. Our two algorithms borrow ideas from gradient perturbation and input perturbation, analogized to the context of post-processing for fairness rather than training from scratch.

2.2 Fairness

In the classical setting, a classifier is well-calibrated if for each predicted score in $[0, 1]$, the average outcome (each in $\{0, 1\}$) for an input with that score is the score itself. For example, if the weather forecast is well-calibrated, it will rain on 30% of the days when a 30% chance of rain is predicted. Hèbert-Johnson et al. [10] define an approximate notion of calibration as follows:

Definition 3. *Calibration.* For any $v \in [0, 1]$, $S \subseteq X$, and predictor p, let $S_v = \{i : x_i = v\}$. For $\alpha \in [0, 1]$, p is α-calibrated with respect to S if there exists some $S' \subseteq S$with $|S'| \geq (1 - \alpha)|S|$, such that for all $v \in [0, 1]$

$$\left| \mathop{\mathbb{E}}_{i \sim S_c \cap S'} [x_i - p_i^*] \right| \leq \alpha, \text{where } S \text{ is a subset of individuals in the overall population } X$$

For a collection of subsets C generated as circuits in some natural complexity class, we say a predictor is α-multicalibrated on C if it is α-calibrated for all S; in some sense C is the set of all computationally identifiable subsets, meaning that we prevent discrimination along all the dimensions that we could have efficiently picked out. This definition has the additional nice property that classifiers become strictly more fair as they are given access to more features, trained on more data, and given access to more computational power.

In the same paper, Hèbert-Johnson et al. [10] provide a *post-processing* meta-algorithm that turns an arbitrary classifier into a multicalibrated classifier using a *guess-and-check oracle*. At a high level, the meta-algorithm loops over every subset and their possible output values, and uses the oracle to check whether the classifier's predictions are calibrated. If not, the oracle returns an update to add to predictions for that subset, bringing the classifier closer to calibration.

2.3 Privacy $\cap$ Fairness

We are aware of two recent works that develop simultaneously private and fair machine learning algorithms. Jagielski et al. [11] achieve differential privacy and *equalized odds* fairness, while Cummings et al. [4] focus on differential privacy and *equal opportunity* fairness.

In their multicalibration construction, Hèbert-Johnson et al. [10] use a differentially private guess-and-check oracle in order to instantiate generalization guarantees from prior works on statistical validity and adaptive data analysis [5, 6]. However, we do not believe that this guarantee is strong enough from a privacy perspective, elaborated in Section 3.1. In this sense, to the best of our knowledge, our work is the first to explore multicalibration post-processing from a privacy perspective.

3 Our Approach

We use the same template for multicalibration post-processing as Hèbert-Johnson et al. [10], where a meta-algorithm uses an oracle with access to the underlying dataset in order to make incremental improvements to calibration. However, we refactor the computation in order to isolate differentially private access to the dataset within the oracle. See Algorithm 1 for the multicalibration meta-algorithm. C is the set of groups, q is the oracle, p is the predictor/classifier, and k is the number of buckets on which to calibrate predicted scores (e.g. intervals(3) returns [0, $\frac{1}{3}$), [$\frac{1}{3}$, $\frac{2}{3}$), and [$\frac{2}{3}$, 1]). See

Algorithm 1 — Multicalibration meta-algorithm

```
multicalibrate($C, q, p, k$):
    repeat:
        updated := false
        for ($c$ in $C$) $\times$ ($i$ in intervals($k$)):
            $r$ := $q(p, c, i)$
            if $r$ = ✓:
                continue
            $p$ := ($\lambda x$. if $x \in c \wedge p(x) \in i$ then $p(x) + r$ else $p(x)$)
            updated := true
        if not updated:
            break
    return $p$
```

Algorithm 2 for the multicalibration oracle, which takes a predictor p, a set c, and an interval i and returns an incremental update to make the elements of c whose scores are predicted to be in i more calibrated, if necessary. The dataset consists of two variables: X is a vector of n representations of individuals, each consisting of d features, and y is a vector of the true outcomes $\in [0, 1]$ for those n individuals. Several variables in this definition are free; these are instantiated in Figure 1 to define our two differentially private post-processing algorithms (input perturbation and numeric sparse).

Algorithm 2 — Oracle

```
q$(p, c, i)$:
    if q.count >= j:
        return ✓

    $S := X \cap c$
    $S_i := S \cap \{x : p(x) \in i\}$
    if $|S_i|$ + Lap$(\psi_1) < m$:
        return ✓

    $y_i := S_i \cap \{y : y \in i\}$
    $\hat{y_i} := S_i \cap \{p(x) : p(x) \in i\}$
    $r := \bar{y_i} - \bar{\hat{y_i}}$
    if $|r||S_i|$ + Lap$(\psi_2) < 2\omega N +$ q.$\psi$:
        return ✓

    q.count += 1
    q.$\psi$ := Lap$(\psi_3)$
    return $r$ + Lap$(\psi_4)/|S_i|$
```

3.1 Numeric Sparse Postprocessing:

Our first strategy for privacy-preserving multicalibration post-processing will be based on the conventional (ϵ, δ)-differentially private Numeric Sparse mechanism[8]. At a high level, we reveal the (noisy) value of a query only if it (noisily) exceeds a (noisy) threshold; otherwise, we simply output an acknowledgement. In this way, the privacy loss scales with the number of successful queries above the threshold, rather than the total number of queries.

Our approach to deriving theoretical convergence bounds and privacy leakage for multicalibration using the numeric sparse oracle will be based on a blend of the (α, β) accuracy bounds on numeric-sparse vectors and the convergence proof/potential argument from the multicalibration paper[10].

Theorem 1. *There exists an $(\epsilon, 0)$ DP algorithm that takes*

$$O\left(\frac{|C|}{\gamma}\sqrt{\frac{m\epsilon}{\ln\frac{|C|}{\beta} + \ln|C|}}\right)$$

time to train an

$$\left(O\left(\sqrt{\frac{\ln\frac{|C|}{\beta} + \ln|C|}{m\epsilon}}\right), \gamma\right)$$

multicalibrated predictor over a collection of sets C with minimum size m, with probability $1 - \beta$, with the probability space over the randomness of the DP algorithm.

Proof. First, note that our queries for normalized error are at most sensitivity $\frac{1}{m}$, where m is the size of the smallest S_i that we attempt to calibrate.

Now, following the proof in NumericSparse, setting ϵ equal to $m\epsilon$ (since our query sensitivity is shifted down), we have accuracy bounds for k queries with at most c above threshold, for all β, of $\alpha = \frac{9c\left(\ln k + \ln \frac{4c}{\beta}\right)}{m\epsilon}$ (see page 64 of [8]).

Now, use this as the basis for a probabilistic guess and check algorithm as in the multicalibration paper (see page 23 of [10]) Specifically, we will set our window size equal to $\alpha = \frac{9c\left(\ln k + \ln \frac{4c}{\beta}\right)}{2m\epsilon}$ (i.e. we return above-threshold if we are 2α away from calibration noisily).

Our update size is then at least α by construction and the definition of (α, β) accuracy for numeric sparse, and, noting that our total update is at most 1, this leads us to solving the following equation: $c = 1/\frac{9c\left(\ln k + \ln \frac{4c}{\beta}\right)}{m\epsilon}$

Note that this is bounded by:

$$c = \frac{\sqrt{m\epsilon}}{3\sqrt{\ln \frac{k}{\beta} + \ln k}}$$
$$= O\left(\frac{\sqrt{m\epsilon}}{\sqrt{\ln \frac{k}{\beta} + \ln k}}\right)$$

updating queries, at most k/γ times that queries in total, with final calibration of:

$$O\left(\frac{\frac{3\sqrt{m\epsilon}}{\sqrt{\ln \frac{k}{\beta} + \ln k}}\left(\ln k + \ln \frac{4\sqrt{2\epsilon}}{3\beta}\right)}{2m\epsilon}\right)$$
$$\approx O\left(\sqrt{\frac{\ln k}{m\epsilon}}\right)$$

for arbitrary exponentially small beta, with privacy leakage of $\epsilon, 0$. □

Note that we have a similar accuracy/time tradeoff as we found in the original multicalibration paper, albeit here based on the size of the collection of sets and the minimum set intersection size which then determines alpha. In other words, having a small number of large sets in our collection will calibrate well but take longer to finish (in terms of updating queries), but large numbers of small sets in our collection will have fewer updating queries but have a larger window size and thus be less well calibrated.

Now, say we want some specific α calibration for our final model – given the above, our best lever for getting that calibration is shifting our minimum group size. We will now prove a simple result about that group size constraint.

Theorem 2. *Our minimum group size for the the above algorithm to yield α multicalibration is bounded by $O\left(\frac{\ln k}{\alpha^2 \epsilon}\right)$.*

Proof. $O\left(\sqrt{\frac{\ln k}{m\epsilon}}\right) = O(\alpha)$, which gives us $m > O\left(\frac{\ln k}{\alpha^2 \epsilon}\right)$. □

Now, for small k, these bounds perform significantly better than the bounds derived in [10], depending as they do on α and $\frac{1}{\alpha^2}$ rather than on $\frac{1}{\alpha^2 \gamma \lambda}$. Note that for k polynomial in n, our minimum size sets can be significantly smaller than they could in the original case. Also note that if we're willing to incur privacy loss equal to $1/\alpha$, our minimum set size compares favorably to that of the cited algorithm even with C sized exponentially in n—however, in that case, we require significantly more updates to reach α calibration.

Our final modification to the algorithm involves adding some noise to the size checker—Lap(ψ_1)—with that noise scaled by $\sqrt{j}$ due to the advanced composition theorem. This modification corrects what we believe to be an oversight in Hèbert-Johnson et al.'s argument. They claim that their post-processing algorithm only interacts with the dataset in the step where it checks whether the query is

variable	**input perturbation**	**numeric sparse**
q.count	0	0
m	$\alpha\|S_i\|/k$	$\ln\|C\|/\alpha^2\epsilon$
j	T	$\sqrt{m\epsilon}/\sqrt{\ln(\|C\|/\beta)+\ln\|C\|}$
ω	$\alpha\|S_i\|/(4n)$	$\sqrt{\ln\|C\|}/\sqrt{m\epsilon}$
N	$\|X\|$	$\|S_i\|$
ψ_1	0	$\epsilon\sqrt{j}$
ψ_2	0	$9j/2\epsilon$
ψ_3	0	$9j/4\epsilon$
ψ_4	0	$9j/\epsilon$

Figure 1: Initialization of free variables for multicalibration postprocessing oracles.

above the threshold, but this is only true in the sense that it's the step where the true outcomes for individuals in the training set (y in our terminology) are accessed—*unlabelled* representations of individuals are still accessed elsewhere.

Consider a dataset such that no computationally identifiable subset is above the size threshold, but the removal of a single individual from the training set would bump one above the threshold. In the former case, the algorithm is entirely deterministic, while in the latter, it is randomized. Because the supports of the output distributions are not equal, this trivially violates differential privacy. This may be sufficient if we wish to avoid overfitting with respect to y, but it fails if privacy is our end rather than generalization (e.g. if X includes SSNs, medical history, criminal record, etc., keeping the supervised labels private is not enough). It is also possible that the algorithm may overfit with respect to X alone, e.g. by not calibrating certain subsets because they happened to be of insufficient size on the training set.

3.2 Input Perturbation

Kang et al. [12] propose an input perturbation technique for DP-ERM to make the gradient and model differentially private at the same time. They perturb the input features by adding differentially private noise drawn from the Gaussian distribution with variance:

$$\sigma^2 = c\frac{G^2 T\log(\frac{1}{\delta})}{n(n-1)\epsilon^2}$$

where n is the number of rows in the dataset, loss function is G-Lipschitz, ϵ is the privacy-loss parameter, T is the number of local iterations, and c is a constant.

Our input perturbation method has the same time and sample bounds for calibration as the original multicalibration paper, since we are essentially just multicalibrating with respect to a different dataset.

4 Evaluation

Our Python implementation and experimental apparatus are publicly available as a Colab notebook. Our evaluation was inspired by the implementation at [1], but we did not reuse any of its code. In particular, it follows the presentation of Hèbert-Johnson et al. [10] by fixing the universe of individuals ahead of time, and performing multicalibration on an explicit vector representation of the classifier and collection of sets C. In contrast, we represent the sets in C implicitly as their membership functions, and construct the post-processed classifier using first-class functions.

We empirically evaluated our two approaches using the Adult dataset [16]: given an individual's demographic data from the 1994 US Census, we must predict whether the person makes over $50K a year. We preprocessed and standardized all of the features, including one-hot encoding all the categorical variables. From the sensitive attributes of race and sex (expanded into several one-hot features), we construct the collection of sets C—each membership predicate is a conjunction of binary sensitive attributes. We used only race and sex because the size of C scales exponentially with the number of features; it was prohibitively time-consuming to evaluate across a grid.

Our logistic regression classifiers were trained with learning rate set to 0.1 and number of iterations set to 1500. For input perturbation, we generated noise according to Section 3.2 with the following

Calibration	Parameters	Train loss	Train acc	Test loss	Test acc
Uncalibrated		44.4	0.807	44.8	0.812
No Privacy	$\alpha = 0.1$	37.8	0.839	37.7	0.842
	$\alpha = 0.5$	44.4	0.808	44.4	0.812
Input Perturbation	$\epsilon = 1, \alpha = 0.1$	37.7	0.839	37.6	0.837
	$\epsilon = 1, \alpha = 0.5$	44.4	0.807	44.6	0.812
Numeric Sparse	$\epsilon = 1, \alpha = 0.1$	37.6	0.837	37.6	0.840
	$\epsilon = 1, \alpha = 0.5$	44.6	0.808	44.4	0.812

Figure 2: Results from multicalibrating our logistic regression classifier for the Adult dataset. We report four types of multicalibration: no calibration (skipping post-processing entirely), no privacy (performing post-processing without any noise), input perturbation, and numeric sparse. We report binary cross-entropy loss and accuracy on the training and test sets.

parameters: $\epsilon = 1$, $\delta = 0.001$, $G = 1$, and $c = 8$. In Figure 2, we report binary cross-entropy loss and accuracy across the training and test sets. We omit parameter settings that cause no change in behavior. We do not perform statistical tests due to time constraints.

Several results were notable to us. First, varying ϵ (from $\epsilon = 0.1$ to $\epsilon = 10$) had little—if any—effect on accuracy, except at the very extremes; second, the classifiers—even the uncalibrated one—performed better on the test set than the training set; third, multicalibration substantially improved accuracy; and fourth, differential privacy did not appreciably affect generalization. All of these could explained or confounded if the model is inundated with data and is nowhere close to overfitting (e.g. improved accuracy from multicalibration could just be from increased expressivity). Our training set was relatively large ($\sim$30,000 data points), compared to a logistic regression model with 104 parameters, meaning that this may well have been the case. Due to time constraints, we were unable to evaluate our methods with a different data set or model where a naive approach is more prone to overfitting.

5 Conclusion

In this work, we explored two differentially private methods to make an arbitrary classifier *multicalibrated*, drawing from prior work on input perturbation and gradient perturbation. We characterized the behavior of these algorithms with theoretical bounds, and found in an empirical evaluation that multicalibration improved test accuracy, with little negative effect from differential privacy. We caution that these results may result from an abundance of training data compared to model complexity, and hope to experiment with more datasets in the future. Nevertheless, these results are encouraging, as they suggest that we can make sensitive classifiers more private and fair while simultaneously increasing accuracy.

6 Acknowledgments

We would like to thank Professor Cynthia Dwork, as well as Elbert Du and Yonadav Shavit, for their help and advice in preparing this paper.

References

[1] Fairness and calibration. `https://github.com/sanatonek/fairness-and-callibration`.

[2] Raef Bassily, Adam Smith, and Abhradeep Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 464–473. IEEE, 2014.

[3] Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(Mar):1069–1109, 2011.

[4] Rachel Cummings, Varun Gupta, Dhamma Kimpara, and Jamie Morgenstern. On the compatibility of privacy and fairness. In *Adjunct Publication of the 27th Conference on User Modeling, Adaptation*

and Personalization, UMAP'19 Adjunct, page 309–315, New York, NY, USA, 2019. Association for Computing Machinery.

[5] Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Roth. Preserving statistical validity in adaptive data analysis. *CoRR*, abs/1411.2664, 2014.

[6] Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Roth. Generalization in adaptive data analysis and holdout reuse. *CoRR*, abs/1506.02629, 2015.

[7] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard S. Zemel. Fairness through awareness. *CoRR*, abs/1104.3913, 2011.

[8] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. In *Foundations and Trends in Theoretical Computer Science*, pages 65–70, University of Pennsylvania, USA, 2014. University of Pennsylvania.

[9] Cynthia Dwork and Adam Smith. Differential privacy for statistics: What we know and what we want to learn. *Journal of Privacy and Confidentiality*, 1(2):2, 2010.

[10] Úrsula Hébert-Johnson, Michael P. Kim, Omer Reingold, and Guy N. Rothblum. Calibration for the (computationally-identifiable) masses. *CoRR*, abs/1711.08513, 2017.

[11] Matthew Jagielski, Michael J. Kearns, Jieming Mao, Alina Oprea, Aaron Roth, Saeed Sharifi-Malvajerdi, and Jonathan Ullman. Differentially private fair learning. *CoRR*, abs/1812.02696, 2018.

[12] Yilin Kang, Yong Liu, Ben Niu, Xinyi Tong, Likun Zhang, and Weiping Wang. Input perturbation: A new paradigm between central and local differential privacy. *arXiv preprint arXiv:2002.08570*, 2020.

[13] Michael P. Kim, Omer Reingold, and Guy N. Rothblum. Fairness through computationally-bounded awareness. *CoRR*, abs/1803.03239, 2018.

[14] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 691–706. IEEE, 2019.

[15] Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 739–753. IEEE, 2019.

[16] DJ Newman, S Hettich, CL Blake, and CJ Merz. Uci repository of machine learning databases. dept. information and computer sciences, univ. california, irvine, 1998.

[17] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE, 2017.