Privacy-Preserving Data Quality Evaluation in Federated Learning Using Influence Approximation

Ljubomir Rokvic
Artificial Intelligence Laboratory
École Polytechnique Fédérale de Lausanne (EPFL)
ljubomir.rokvic@epfl.ch

Panayiotis Danassis
Harvard University
pdanassis@seas.harvard.edu

Boi Faltings
Artificial Intelligence Laboratory
École Polytechnique Fédérale de Lausanne (EPFL)
boi.faltings@epfl.ch

Abstract

In Federated Learning, it is crucial to handle low-quality, corrupted, or malicious data, but traditional data valuation methods are not suitable due to privacy concerns. To address this, we propose a simple yet effective approach that utilizes a new influence approximation called "lazy influence" to filter and score data while preserving privacy. To do this, each participant uses their own data to estimate the influence of another participant's batch and sends a differentially private obfuscated score to the Center of the federation. Our method has been shown to successfully filter out corrupted data in various applications, achieving a recall rate of over $> 90\%$ (sometimes up to 100%) while maintaining strong differential privacy guarantees with epsilon values of less than or equal to one.

1 Introduction

The success of Machine Learning (ML) depends to a large extent on the availability of high-quality data. This is a critical issue in Federated Learning (FL) since the model is trained without access to raw training data. Instead, a single *Center* uses data from independent and sometimes self-interested *data holders* to train a model jointly. Having the ability to *score* and *filter* irrelevant, noisy, or malicious data can (i) significantly improve model accuracy, (ii) speed up training, and even (iii) reduce costs for the Center when it pays for data.

Federated Learning [33, 25, 42] differs from traditional centralized ML approaches. Challenges such as scalability, communication efficiency, and privacy can no longer be treated as an afterthought; instead, they are *inherent constraints* of the setting. For example, data holders often operate resource-constrained edge devices and include businesses and medical institutions that must protect the privacy of their data due to confidentiality or legal constraints.

A clean way of quantifying the effect of data point(s) on the accuracy of a model is via the notion of *influence* [26, 6]. Intuitively, influence quantifies the marginal contribution of a data point (or batch of points) on a model's accuracy. One can compute this by comparing the difference in the model's empirical risk when trained with and without the point in question. While the influence metric can be highly informative, it is impractical to compute: re-training a model is time-consuming, costly, and often impossible, as participants do not have access to the entire dataset. We propose a simple and practical approximation of the *sign* of the exact influence (*lazy influence*), which is based on an estimate of the direction of the model after a small number of local training epochs with the new data.

Preprint. Under review.

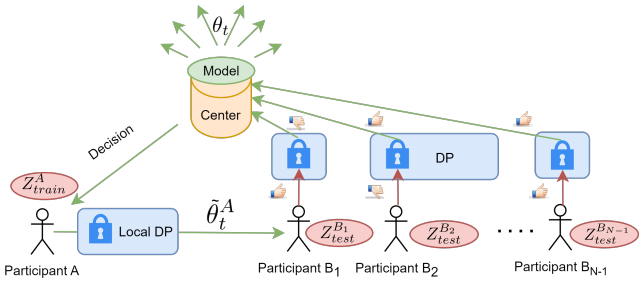


Figure 1: Data filtering procedure. A Center heads a federation of participants $A, B_1, .., B_{N-1}$ that holds private data relevant to the joint model. A sends an obfuscated *lazy* (i.e., partial/approximate) model update to B_i, who submit a vote based on their own testing data. The differentially private aggregated votes are used to decide whether to incorporate A's data. See Section 1.2.

Another challenge is to approximate the influence while preserving the privacy of the data. Many approaches to Federated Learning (e.g., [35, 40]) remedy this by combining FL with Differential Privacy (DP) [11, 12, 13, 14], a data anonymization technique that many researchers view as the gold standard [39]. We show how the sign of influence can be approximated in an FL setting while maintaining strong differential privacy guarantees. Specifically, there are two sets of participants' data that we need to protect: the training and the test data (see also Section 1.2). We clip and add noise to the gradients for the evaluated training data according to [34], which achieves a *local* differential privacy guarantee. To ensure the privacy of the test data and the influence approximation itself, we employ a differentially private defense mechanism based on the idea of randomized response [43] (inspired by [16]). Together the two mechanisms ensure strong, *worst-case privacy* guarantees while allowing for accurate data filtering.

The proposed approach can be used as a 'right of passage' every time a participant joins the federation, periodically during communication rounds (most resource intensive, but would provide the best results), or even as a diagnostic tool. A quality score is helpful for various purposes beyond filtering poor data, such as rewarding the data provider, incentivizing users in a crowdsourcing application, assessing a data provider's reputation, etc.

1.1 Our Contributions

We address two major challenges in this work: (i) efficiently estimating the quality of a batch of training data and (ii) keeping both the training and test data used for this estimate private. For the former, we develop a novel metric called *lazy influence*, while for the latter, we add noise to the gradients and propose a differentially private voting scheme. More specifically:

(1) We present a novel technique (*lazy influence approximation*) for scoring and filtering data in Federated Learning.

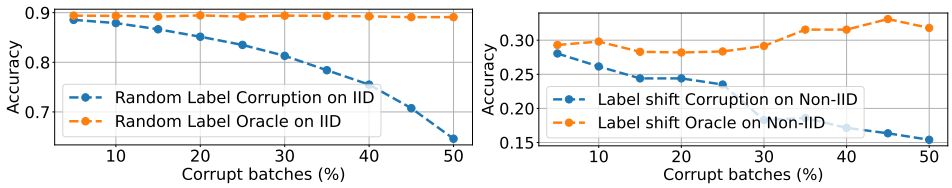
(2) We show that our proposed distributed influence aggregation scheme allows for robust scoring, even under rigorous, worst-case differential privacy guarantees (privacy cost $\varepsilon < 1$). This is the recommended value in DP literature and much smaller than many other AI or ML applications.[1].

(3) We evaluate our approach on two well-established datasets (CIFAR10 and CIFAR100) and demonstrate that **filtering using our scheme can eliminate the adverse effects of inaccurate data**.

1.2 High-Level Description of Our Setting

A center C coordinates a set of participants to train a single model (Figure 1). C has a small set of 'warm-up' data, which are used to train an initial model M_0 that captures the desired input/output relation. We assume that each data holder has a set of training points that will be used to improve the model and a set of test points that will be used to evaluate other participants' contributions. It must be kept private to prohibit participants from tailoring their contributions to the test data. For

[1]AI or ML applications often assume ε as large as 10 [40] (see, e.g., [38]). For specific attacks, $\varepsilon = 10$ means that an adversary can theoretically reach an accuracy of 99.99% [40]



(a) Random Label IID

(b) Label Shift Non-IID ($\alpha = 0.01$)

Figure 2: Model accuracy relative to different mislabel rates (5% - 50%). These models have been trained over 25 communication rounds and 100 participants. We compare a centralized model with no filtering of mislabeled data (blue) to an FL model under perfect (oracle) filtering (orange). Note that the lower accuracy on the Non-IID setting is due to the fact that we are considering the most extreme non-IID case. This is where the majority of the participants have access to at most 1 class.

each federated learning round t (model M_t), each data holder participant will assume two roles: the role of the contributor (A) and the role of the tester (B). As a contributor, a participant performs a small number of local epochs to M_t – enough to get an estimate of the gradient[2] – using a batch of his training data $z_{A,t}$. Subsequently, A sends the updated partial model $M_{t,A}$, with specifically crafted noise to ensure local DP, to every other participant (which assumes the role of a tester). The applied noise protects the updated gradient while still retaining information on the usefulness of data. Each tester B uses its test dataset to approximate the empirical risk of A's training batch (i.e., the approximate influence). This is done by evaluating each test point and comparing the loss. In an FL setting, we can not re-train the model to compute the exact influence; instead, B performs only a small number of training epochs, enough to estimate the direction of the model (lazy influence approximation). As such, we look at the sign of the approximate influence (and not the magnitude). Each tester aggregates the signs of the influence for each test point, applies controlled noise to ensure DP, and sends this information to the center. Finally, the center accepts A's training batch if most of the Bs report positive influence and reject otherwise.

2 Related Work and Discussion

Federated Learning Federated Learning (FL) [33, 25, 42, 29] has emerged as an alternative method to train ML models on data obtained by many different agents. In FL, a center coordinates agents who acquire data and provide model updates. FL has been receiving increasing attention in both academia [30, 46, 20, 2] and industry [19, 4], with a plethora of real-world applications (e.g., training models from smartphone data, IoT devices, sensors, etc.).

Influence functions Influence functions are a standard method from robust statistics [6] (see also Section 3), which were recently used as a method of explaining the predictions of black-box models [26]. They have also been used in the context of fast cross-validation in kernel methods and model robustness [32, 5]. While a powerful tool, computing the influence involves too much computation and communication, and it requires access to the training and testing data (see [26] and Section 3).

Data Filtering A common but computationally expensive approach for filtering in ML is to use the Shapley Value of the Influence to evaluate the quality of data [24, 17, 23, 45, 18]. Other work includes, for example, rule-based filtering of least influential points [36], or constructing weighted data subsets (corsets) [9]. Because of the privacy requirements in FL, contributed data is not directly accessible for assessing its quality. [41] propose a decentralized filtering process specific to federated learning, yet they do not provide any formal privacy guarantees.

While data filtering might not always pose a significant problem in traditional ML, in an FL setting, it is more important because even a small percentage of mislabeled data can result in a significant drop in the combined model's accuracy. Consider Figure 2 as a motivating example. In this scenario, we have participants with corrupted data. Even a very robust model (ViT) loses performance when corruption is involved. This can also be observed in the work of [28]. Filtering those corrupted participants (orange line) restores the model's performance.

[2]The number of local epochs is a hyperparameter. We do not need to train the model fully. See Section 3.2.

Client Selection and Attack Detection Our setting can also be interpreted as potentially adversarial, but it should not be confused with Byzantine robustness. We do not consider threat scenarios as described in [3] and [37], where participants carefully craft malicious updates. Instead we assume that the data used for those updates might be corrupt. For completeness and in lack of more relevant baselines, we compare our work to two Byzantine robust methods: KRUM [1] and Trimmed-mean [47] (along to an oracle filter). These methods, though, require gradients to be transmitted as is, i.e., lacking any formal privacy guarantee to the participants' training data. Furthermore, both of these techniques require the center to know the number of malicious participants a priori. Another important drawback is that they completely eliminate "minority" distributions due to their large distance relative to other model updates.

Differential Privacy Differential Privacy (DP) [11, 12, 13, 14] has emerged as the de facto standard for protecting the privacy of individuals. Informally, DP captures the increased risk to an individual's privacy incurred by participating in the learning process. Consider a participant being surveyed on a sensitive topic as a simplified, intuitive example. To achieve differential privacy, one needs a source of randomness; thus, the participant decides to flip a coin. Depending on the result (heads or tails), the participant can reply truthfully or randomly. An attacker can not know if the decision was taken based on the participant's preference or due to the coin toss. Of course, to get meaningful results, we need to bias the coin toward the actual data. In this simple example, the logarithm of the ratio $Pr[\text{heads}]/Pr[\text{tails}]$ represents the privacy cost (also referred to as the privacy budget), denoted traditionally by ε. Yet, one must be careful in designing a DP mechanism, as it is often hard to practically achieve a meaningful privacy guarantee (i.e., avoid adding a lot of noise and maintain high accuracy) [40, 8]. A variation of DP, instrumental in our context, given the decentralized nature of federated learning, is Local Differential Privacy (LDP) [15]. LDP is a generalization of DP that provides a bound on the outcome probabilities for any pair of individual participants rather than populations differing on a single participant. Intuitively, it means that one cannot hide in the crowd. Another strength of LDP is that it does not use a centralized model to add noise–participants sanitize their data themselves– providing privacy protection against a malicious data curator. For a more comprehensive overview of DP, we refer the reader to [39, 15]. We assume that the participants and the Center are *honest but curious*, i.e., they don't actively attempt to corrupt the protocol but will try to learn about each other's data.

3 Methodology

We aim to address two challenges: (i) approximating the influence of a (batch of) data point(s) without having to re-train the entire model from scratch and (ii) doing so while protecting the privacy of training and testing data. The latter is essential not only to protect users' sensitive information but also to ensure that malicious participants can not tailor their contributions to the test data. We first introduce the notion of *influence* [6] (for detailed definitions please see the supplement) and our proposed lazy approximation. Second, we describe a differentially private reporting scheme for crowdsourcing the approximate influence values.

Setting We consider a classification problem from some input space $\mathcal{X}$ (e.g., features, images, etc.) to an output space $\mathcal{Y}$ (e.g., labels). In a FL setting, there is a center C that wants to learn a model $M(\theta)$ parameterized by $\theta \in \Theta$, with a non-negative loss function $L(z, \theta)$ on a sample $z = (\bar{x}, y) \in \mathcal{X} \times \mathcal{Y}$. Let $R(Z, \theta) = \frac{1}{n}\sum_{i=1}^{n} L(z_i, \theta)$ denote the empirical risk, given a set of data $Z = \{z_i\}_{i=1}^{n}$. We assume that the empirical risk is differentiable in θ.The training data are supplied by a set of data holders.

3.1 Shortcomings of the Exact and Approximate Influence in a FL Setting

Definitions In simple terms, influence measures the marginal contribution of a data point on a model's accuracy. A positive influence value indicates that a data point improves model accuracy, and vice-versa. More specifically, let $Z = \{z_i\}_{i=1}^{n}$, $Z_{+j} = Z \cup z_j$ where $z_j \notin Z$, and let $\hat{R} = \min_\theta R(Z, \theta)$ and $\hat{R}_{+j} = \min_\theta R(Z_{+j}, \theta)$, where $\hat{R}$ and $\hat{R}_{+j}$ denote the minimum empirical risk of their respective set of data. The *influence* of datapoint z_j on Z is defined as $\mathcal{I}(z_j, Z) \triangleq \hat{R} - \hat{R}_{+j}$

Algorithm 1: Filtering Poor Data Using Lazy Influence Approximation in Federated Learning

Data: θ_0, Z_i, Z_{test}, Z_{init}
Result: θ_T

C: The center (C) initializes the model $M_0(\theta_0)$
for $t \in T$ *rounds of Federated Learning* **do**
  C: Broadcasts θ_t
  for P_i *in* $Participants$ **do**
    P_i: Acts as a contributor (A). Performs k local epochs with $Z_{A,t}$ on the partially-frozen model $\tilde{\theta}_t^A$.
    P_i: Applies DP noise to $\tilde{\theta}_t^A$.
    P_i: sends last layer of $\tilde{\theta}_t^A$ to $Participants_{-i}$.
    for P_j *in* $Participants_{-i}$ **do**
      P_j: Acts as a tester (B). Evaluates the loss of Z_{test}^B on θ_t
      P_j: Evaluates the loss of Z_{test}^B on $\tilde{\theta}_t^A$
      P_j: Calculates vote v (sign of influence), according to Equation 1
      P_j: Applies noise to v according to his privacy parameter p to get v' (Equation 2)
      P_j: Sends v' to C
    C: Filters out P_i's data based on the votes from $Participants_{-i}$ (i.e., if $\sum_{\forall B} I_{proposed}(Z_{test}^B) < T$).
  C: Updates θ_t using data from unfiltered $Participants$;

Despite being highly informative, influence functions have not achieved widespread use in FL (or ML in general). This is mainly due to the computational cost. The exact influence requires complete retraining of the model, which is time-consuming and very costly, especially for state-of-the-art, large ML models (specifically for our setting, we do not have direct access to the training data). Recently, the first-order Taylor approximation of influence [26] (based on [7]) has been proposed as a practical method to understanding the effects of training points on the predictions of a *centralized* ML model. While it can be computed without having to re-train the model, according to the following equation $\mathcal{I}_{appr}(z_j, z_{test}) \triangleq -\nabla_\theta L(z_{test}, \hat{\theta}) H_{\hat{\theta}}^{-1} \nabla_\theta L(z_j, \hat{\theta})$, it is still ill-matched for FL models for several key reasons, as explained in the following paragraph.

Challenges To begin with, computing the influence approximation of [26] requires *forming and inverting* the Hessian of the empirical risk. With n training points and $\theta \in \mathbb{R}^m$, this requires $O(nm^2 + m^3)$ operations [26], which is *impractical* for modern-day deep neural networks with millions of parameters. To overcome these challenges, [26] used implicit Hessian-vector products (HVPs) to more efficiently approximate $\nabla_\theta L(z_{test}, \hat{\theta}) H_{\hat{\theta}}^{-1}$, which typically requires $O(p)$ [26]. While this is a somewhat more efficient computation, it is *communication-intensive*, as it requires *transferring all of the (either training or test) data* at each FL round. Most importantly, it *can not provide any privacy* to the users' data, an important, inherent requirement/constraint in FL. Finally, the loss function has to be strictly convex and twice differentiable (which is not always the case in modern ML applications). The proposed solution is to swap out non-differentiable components for smoothed approximations, but there is no quality guarantee of the influence calculated in this way.

3.2 Lazy Influence: A Practical Influence Metric for Filtering Data in FL Applications

The key idea is that *we do not need to approximate the influence value* to filter data; we only need an accurate estimate of its *sign* (in expectation). Recall that a positive influence value indicates a data point improves model accuracy. Thus, we only need to approximate the sign of loss and use that information to filter out data whose influence falls below a certain threshold.

Recall that each data holder participant assumes two roles: the role of the contributor (A) and the role of the tester (B). Our proposed approach works as follows (Algorithm 1):

(i) For each federated learning round t (model $M_t(\theta_t)$), the contributor participant A performs a small number k of local epochs to M_t using a batch of his training data $Z_{A,t}$, resulting in $\tilde{\theta}_t^A$. k is a

hyperparameter. $\tilde{\theta}_t^A$ is the partially trained model of participant A, where most of the layers, except the last one, have been frozen. The model should not be fully trained for two key reasons: efficiency and avoiding over-fitting (e.g., in our simulations, we only performed 1-9 epochs). Furthermore, A adds noise to $\tilde{\theta}_t^A$ (see Section 3.2.2) to ensure strong, worst-case local differential privacy. Finally, A sends only the last layer (to reduce communication cost) of $\tilde{\theta}_t^A$ to every other participant.

(ii) Each tester B uses his test dataset Z_{test}^B to estimate the sign of the influence using Equation 1. Next, the tester applies noise to $I_{proposed}(Z_{test}^B)$, as described in Section 3.2.3, to ensure strong, worst-case differential privacy guarantees (i.e., keep his test dataset private).

$$\mathcal{I}_{proposed}(Z_{test}^B) \triangleq \text{sign}\left(\sum_{z_{test}\in Z_{test}^B} L(z_{test},\theta_t) - L(z_{test},\theta_t^A)\right) \tag{1}$$

(iii) Finally, the center C aggregates the obfuscated votes $I_{proposed}(Z_{test}^B)$ from all testers and filters out data with cumulative score *below a threshold* ($\sum_{\forall B} I_{proposed}(Z_{test}^B) < T$). Specifically, we cluster the votes into two clusters (using k-means) and use the arithmetic mean of the cluster centers as the filtration threshold.

3.2.1 Advantages of the proposed lazy influence

Depending on the application, the designer may select any optimizer to perform the model updates. We do not require the loss function to be twice differentiable and convex; only once differentiable. It is significantly more *computation and communication efficient*; an essential prerequisite for any FL application. This is because participant A only needs to send (a *small part* of) the model parameters θ, not his training data. Moreover, computing a few model updates (using, e.g., SGD or any other optimizer) is significantly faster than computing either the exact influence or an approximation due to the numerous challenges (please refer to the supplementary materials for a detailed description). Finally, and importantly, we ensure the *privacy* of both the train and test dataset of every participant.

3.2.2 Sharing the Partially Updated Joint Model: Privacy and Communication Cost

Each contributor A shares a partially trained model $\tilde{\theta}_t^A$ (see step (i) of Section 3.2). It is important to stress that A only sends the last layer of the model. This has two significant benefits: it *reduces the communication overhead* (in our simulations, *we only send* 0.009% *of the model's weights*),[3] and minimize the impact of the differential privacy noise. We follow [34] to ensure strong local differential privacy guarantees by (i) imposing a bound on the gradient (using a clipping threshold Δ), and (ii) adding carefully crafted Gaussian noise (parameterized by σ). For more details, see [34].

3.2.3 Differentially Private Reporting of the Influence

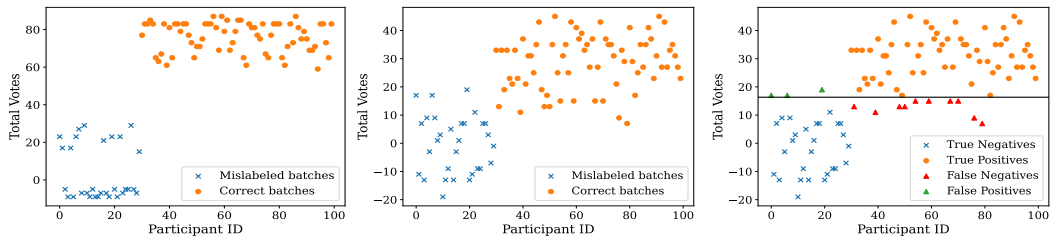
Along with the training data, we also need to ensure the privacy of the test data used to calculate the influence. Protecting the test data in an FL setting is critical since (i) it is an important constraint of the FL setting, (ii) participants want to keep their sensitive information (and potential means of income, e.g., in a crowdsourcing application) private, and (iii) the center wants to ensure that malicious participants can not tailor their contributions to the test set.

We obfuscate the influence reports using RAPPOR [16], which results in an ε-differential privacy guarantee [14]. The obfuscation (permanent randomized response [43]) takes as input the participant's true influence value v (binary) and privacy parameter p, and creates an obfuscated (noisy) reporting value v', according to Equation 2. p is a *user-tunable* parameter that allows the participants themselves to *choose their desired level of privacy*, while maintaining reliable filtering. The worst-case privacy guarantee can be computed by each participant *a priori*, using Equation 3 [16].

$$v' = \begin{cases} +1, & \text{with probability } \frac{1}{2}p \\ -1, & \text{with probability } \frac{1}{2}p \\ v, & \text{with probability } 1-p \end{cases} \tag{2}$$

$$\varepsilon = 2\ln\left(\frac{1-\frac{1}{2}p}{\frac{1}{2}p}\right) \tag{3}$$

[3] Moreover, as explained in the Introduction, this communication cost will be incurred as little as *one* time, when we use our approach as a 'right of passage' every time a participant joins the federation.



(a) (b) (c)

Figure 3: Visualization of the voting scheme. The x-axis represents a contributor participant A. The y-axis shows the sum of all votes from all the testers, i.e., $\sum_{\forall B} I_{proposed}(Z^B_{test})$. Figure 3a corresponds to the sum of true votes (no privacy) for the test data of each contributor on the x-axis, while Figure 3b depicts the sum of differentially private votes ($\epsilon = 1$), according to randomized reporting algorithm (see the supplementary materials for a detailed description). Finally, Figure 3c shows the filtration threshold, corresponding to the arithmetic mean of the two cluster centers (computed using k-means).

It is important to note that in a Federated Learning application, the center C aggregates the influence sign from *a large number of participants*. This means that even under *really strict* privacy guarantees, *the aggregated influence signs (which is exactly what we use for filtering) will match the true value* in expectation. This results in *high-quality filtering*, as we will demonstrate in Section 4.

To demonstrate the effect of Equation 2, we visualize the obfuscation process in Figure 3. Figure 3a shows the sum of true votes (y-axis) for the test data of each contributor (x-axis). Here we can see a clear distinction in votes between corrupted and correct batches. Most of the corrupted batches (corrupted contributor participants) take negative values, meaning that the majority of the testers voted against them. In contrast, the correct batches are close to the upper bound. Figure 3b demonstrates the effect of applying DP noise ($\epsilon = 1$) to the votes: differentiating between the two groups becomes more challenging. To find an effective decision threshold, we use k-means to cluster the votes into two clusters and use the arithmetic mean of the cluster centers as the filtration threshold (Figure 3c).

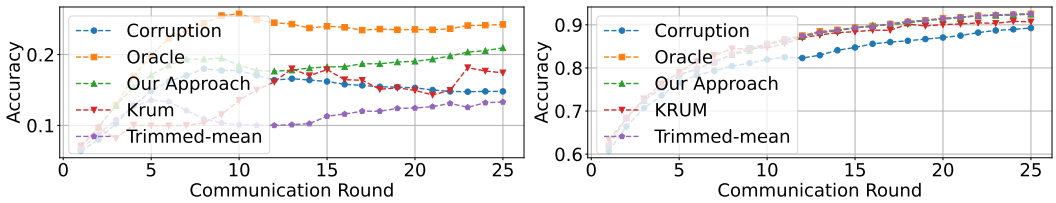
4 Evaluation Results

We evaluated the proposed approach on two well-established datasets: **CIFAR10** [27], and **CIFAR100** [27]. Furthermore, we consider two corruption methods:

1. **Random label**: A random label is sampled for every training point. Used for the IID setting (as it does not make sense to assign a random label to a highly skewed Non-IID setting).
2. **Label shift**: Every correct label is mapped to a different label and this new mapping is applied to the whole training dataset. Used in both IID and non-IID settings.

Setup Our evaluation involves a single round of Federated Learning. A small portion of every dataset (around 1%) is selected as the 'warm-up' data used by the center C to train the initial model M_0. Each participant has two datasets: a training batch (Z_A, see Section 3.2, step (i)), which the participant uses to update the model when acting as the contributor participant, and a test dataset (Z^B_{test}, see Section 3.2, step (ii)), which the participant uses to estimate the sign of the influence when acting as a tester participant. The ratio of these datasets is $2:1$. The training batch size is 100 (i.e., the training dataset includes 100 points, and the test dataset consists of 50 points). This means that, e.g., for a simulation with 100 participants, each training batch is evaluated on $50 \times (100-1)$ test points, and that for each training batch (contributor participant A), the center collected $(100-1)$ estimates of the influence sign (Equation 1). We corrupted 30% of the total batches (i.e., participants). For each corrupted batch, we corrupted 90% of the data points. Each simulation was run 8 times. We report average values and standard deviations. Please see the supplement for detailed results.

Implementation The proposed approach is model-agnostic and can be used with *any* gradient-descent-based machine learning method. For our simulations, we used a Vision Transformer (ViT), as it exhibits state-of-the-art performance [10] (specifically, HuggingFace's implementation [44]).



(a) Label shift Non-IID ($\alpha = 0.01$)

(b) Random Label IID

Figure 4: Model accuracy over 25 communication rounds with a 30% mislabel rate on CIFAR-10. We compare a centralized model with no filtering (blue) to an FL model under perfect (oracle) filtering (orange), KRUM (red), Trimmed-mean (purple), and our approach (green). Note that the jagged line for KRUM is because only a single gradient is selected instead of performing FedAvg.

Non-IID Setting The main hurdle for FL is that not all data is IID. Heterogeneous data distributions are all but uncommon in the real world. To simulate non-IID data, we used the Dirichlet distribution to split the training dataset as in related literature [22, 31, 21, 48]. This distribution is parameterized by α, which controls the concentration of different classes. See the supplement for a visualization. In this work, we use $\alpha \to 0.1$ for a non-IID distribution, as in related literature (e.g., [48]).

Baselines We compare against four baselines: **(i) Corrupted model**: this shows us the training performance of a model which any technique has not sanitized. **(ii) Oracle filtration**: this represents the ideal scenario where we know which participants contribute bad data. **(iii) KRUM**: byzantine robustness technique [1] that selects the best gradient out of the update based on a pair-wise distance metric. **(iv) Trimmed-mean**: another byzantine robustness technique [47] that takes the average of gradients instead of just selecting one, also based on a pair-wise distance metric (see also Section 2).

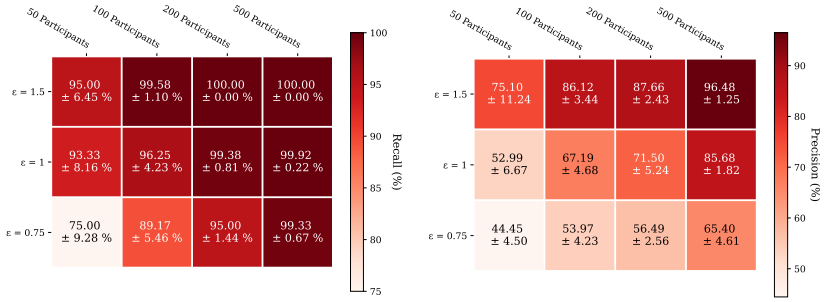
4.1 Model Accuracy

The proposed approach achieves *high model accuracy, close to the perfect (oracle) filtering* (13.6% worse in the non-IID setting, and 0.1% in the IID setting). Focusing on the non-IID setting (Figure 4a), which is the more challenging and relevant for FL, our approach achieves a 20.3% *improvement over KRUM*, and a 57.5% *improvement over the Trimmed-mean baseline*, after 25 communication rounds. Finally, in the IID setting, all methods perform similarly (Figure 4b), though recall that the baselines do not provide privacy guarantees (see Section 2).

4.2 Recall, Precision, and F1 Score of Filtration

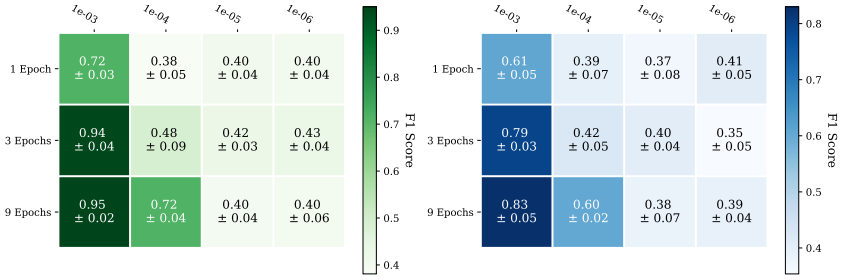
Recall is the most informative metric to evaluate the efficiency of our filtering approach. Recall refers to the ratio of detected mislabeled batches over all of the mislabeled batches. *Including a mislabeled batch can harm a model's performance significantly more compared to removing an unaltered batch.* Thus, achieving *high recall* is of paramount importance. Meanwhile, precision represents the ratio of correctly identified mislabeled batches over all batches identified as mislabeled. An additional benefit of using the proposed lazy influence metric for scoring data is that it also allows us to identify correctly labeled data, which nevertheless do not provide a significant contribution to the model.

The proposed approach achieves both high recall and precision (Figure 5), despite the *high degree of non-IID* (low concentration of classes per participant). Notably, the metrics improve significantly as we increase the number of participants (horizontal axis). In simple terms, more testers mean more samples of the different distributions. Thus, 'honest' participants get over the filtering threshold, even in highly non-IID settings. Recall reaches 100%, and precision 96.48% by increasing the number of participants to just 500, in the non-IID setting and under really strict worst-case privacy guarantees. Results for the IID setting are significantly better (please see the supplement). Finally, Figure 6 depicts the effects of different training parameters (for partially training the model by the contributor participant A, see step (i) of Section 3.2) to the F1 score (harmonic mean of the precision and recall). Our proposed approach requires *only 3-9 epochs* to achieve high-quality filtration instead of a complete model re-training for the exact influence.



(a) Recall (b) Precision

Figure 5: Recall (left), and Precision (right) on CIFAR 10, non-IID, for increasing problem size (number of participants), and varying privacy guarantees (ε – lower ε provides stronger privacy).



(a) IID (b) Non-IID

Figure 6: F1 score on Cifar10, IID (Left) and Non-IID (Right), $\varepsilon = 1$, with 100 participants. We vary the training parameters (training epochs in the vertical axis, learning rate in the horizontal) used for partially training a model by the contributor participant A (see step (i) of Section 3.2).

4.3 Privacy

As expected, there is a trade-off between privacy and filtration quality (see Figure 5, vertical axis, where ε refers to the privacy guarantee for both the training and test data/participant votes). Nevertheless, Figure 5 demonstrates that our approach can provide *reliable filtration*, even under *really strict, worst-case privacy requirements* ($\varepsilon = 1$, which is the recommended value in the DP literature [39]). Importantly, our decentralized framework allows each participant to *compute* and *tune* his *own* worst-case privacy guarantee *a priori* (see Section 3.2.3).

The *privacy trade-off can be mitigated*, and the quality of the filtration can be significantly improved by increasing the number of participants (Figure 5, horizontal axis). The higher the number of participants, the better the filtration (given a fixed number of corrupted participants). This is because as the number of participants increases, the aggregated influence signs (precisely what we use for filtering) will match the actual value in expectation. For 500 participants, we achieve high-quality filtration even for $\varepsilon = 0.75$. This is important given that in most real-world FL applications, we *expect a large number of participants*.

5 Conclusion

Privacy protection is a core element of Federated Learning. However, this privacy also means that it is significantly more difficult to ensure that the training data actually improves the model. Mislabeled, corrupted, or even malicious data can result in a strong degradation of the performance of the model – as we also demonstrated empirically – and privacy protection makes it significantly more challenging to identify the cause. In this work, we propose the *"lazy influence"*, a *practical* influence approximation that characterizes the quality of training data and allows for effective filtering (recall of $> 90\%$, and even up to 100% as we increase the number of participants), while providing *strict, worst-case* ε-differential privacy guarantees ($\varepsilon < 1$) for both the training and test data. The proposed

approach can be used to filter bad data, recognize good and bad data providers, and pay data holders according to the quality of their contributions.

References

[1] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[2] Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

[3] Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong. Fltrust: Byzantine-robust federated learning via trust bootstrapping. *arXiv preprint arXiv:2012.13995*, 2020.

[4] Mingqing Chen, Rajiv Mathews, Tom Ouyang, and Françoise Beaufays. Federated learning of out-of-vocabulary words. *arXiv preprint arXiv:1903.10635*, 2019.

[5] Andreas Christmann and Ingo Steinwart. On robustness properties of convex risk minimization methods for pattern recognition. *JMLR*, 2004.

[6] R Dennis Cook and Sanford Weisberg. Characterizations of an empirical influence function for detecting influential cases in regression. *Technometrics*, 1980.

[7] R Dennis Cook and Sanford Weisberg. *Residuals and influence in regression*. New York: Chapman and Hall, 1982.

[8] Panayiotis Danassis, Aleksei Triastcyn, and Boi Faltings. A distributed differentially private algorithm for resource allocation in unboundedly large settings. In *Proceedings of the 21th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS-22*. International Foundation for Autonomous Agents and Multiagent Systems, 2022.

[9] Anirban Dasgupta, Petros Drineas, Boulos Harb, Ravi Kumar, and Michael W Mahoney. Sampling algorithms and coresets for \ell_p regression. *SIAM Journal on Computing*, 2009.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020.

[11] Cynthia Dwork. Differential privacy. In *33rd International Colloquium on Automata, Languages and Programming, part II (ICALP 2006)*, volume 4052, pages 1–12, Venice, Italy, July 2006. Springer Verlag.

[12] Cynthia Dwork. Differential privacy. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Automata, Languages and Programming*, pages 1–12, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[13] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 486–503. Springer, 2006.

[14] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, 2006.

[15] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

[16] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, pages 1054–1067, 2014.

[17] Amirata Ghorbani and James Zou. Data shapley: Equitable valuation of data for machine learning. In *International Conference on Machine Learning*, pages 2242–2251. PMLR, 2019.

[18] Amirata Ghorbani and James Zou. Data shapley: Equitable valuation of data for machine learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2242–2251. PMLR, 09–15 Jun 2019.

[19] 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.

[20] Chaoyang He, Songze Li, Jinhyun So, Xiao Zeng, Mi Zhang, Hongyi Wang, Xiaoyang Wang, Praneeth Vepakomma, Abhishek Singh, Hang Qiu, et al. Fedml: A research library and benchmark for federated machine learning. *arXiv preprint arXiv:2007.13518*, 2020.

[21] Haley Hoech, Roman Rischke, Karsten Müller, and Wojciech Samek. Fedauxfdp: Differentially private one-shot federated distillation, 2022.

[22] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

[23] Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nezihe Merve Gurel, Bo Li, Ce Zhang, Costas J Spanos, and Dawn Song. Efficient task-specific data valuation for nearest neighbor algorithms. *arXiv preprint arXiv:1908.08619*, 2019.

[24] Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Gurel, Bo Li, Ce Zhang, Dawn Song, and Costas Spanos. Towards efficient data valuation based on the shapley value. In *AISTATS*, 2019.

[25] 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.

[26] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR, 2017.

[27] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Learning Multiple Layers of Features from Tiny Images*, 2009.

[28] Anran Li, Lan Zhang, Juntao Tan, Yaxuan Qin, Junhao Wang, and Xiang-Yang Li. Sample-level data selection for federated learning. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*, pages 1–10. IEEE, 2021.

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

[30] Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 22(3):2031–2063, 2020.

[31] Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. *Advances in Neural Information Processing Systems*, 33:2351–2363, 2020.

[32] Yong Liu, Shali Jiang, and Shizhong Liao. Efficient approximation of cross-validation for kernel methods using bouligand influence function. In *ICML*, 2014.

[33] 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.

[34] H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. *arXiv preprint arXiv:1710.06963*, 2017.

[35] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations*, 2018.

[36] Kohei Ogawa, Yoshiki Suzuki, and Ichiro Takeuchi. Safe screening of non-support vectors in pathwise svm computation. In *ICML*, 2013.

[37] Jinhyun So, Başak Güler, and A Salman Avestimehr. Byzantine-resilient secure federated learning. *IEEE Journal on Selected Areas in Communications*, 39(7):2168–2181, 2020.

[38] Jun Tang, Aleksandra Korolova, Xiaolong Bai, Xueqiang Wang, and Xiaofeng Wang. Privacy loss in apple's implementation of differential privacy on macos 10.12. *arXiv preprint arXiv:1709.02753*, 2017.

[39] Aleksei Triastcyn. *Data-Aware Privacy-Preserving Machine Learning*. PhD thesis, EPFL, Lausanne, 2020.

[40] Aleksei Triastcyn and Boi Faltings. Federated learning with bayesian differential privacy. In *IEEE International Conference on Big Data (Big Data)*. IEEE, 2019.

[41] Tiffany Tuor, Shiqiang Wang, Bong Jun Ko, Changchang Liu, and Kin K Leung. Overcoming noisy and irrelevant data in federated learning. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 5020–5027. IEEE, 2021.

[42] Jianyu Wang, Zachary Charles, Zheng Xu, Gauri Joshi, H Brendan McMahan, Maruan Al-Shedivat, Galen Andrew, Salman Avestimehr, Katharine Daly, Deepesh Data, et al. A field guide to federated optimization. *arXiv preprint arXiv:2107.06917*, 2021.

[43] Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

[44] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics.

[45] Tom Yan, Christian Kroer, and Alexander Peysakhovich. Evaluating and rewarding teamwork using cooperative game abstractions. *Advances in Neural Information Processing Systems*, 33:6925–6935, 2020.

[46] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.

[47] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5650–5659. PMLR, 10–15 Jul 2018.

[48] Yaodong Yu, Alexander Wei, Sai Praneeth Karimireddy, Yi Ma, and Michael Jordan. TCT: Convexifying federated learning using bootstrapped neural tangent kernels. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.