Robust Distributed Gradient Aggregation Using Projections onto Gradient Manifolds

Kwang In Kim
POSTECH
kimkin@postech.ac.kr

Abstract

We study the distributed gradient aggregation problem where individual clients contribute to learning a central model by sharing parameter gradients constructed from local losses. However, errors in some gradients, caused by low-quality data or adversaries, can degrade the learning process when naively combined. Existing *robust* gradient aggregation approaches assume that local data represent the global data-generating distribution, which may not always apply to heterogeneous (non-i.i.d.) client data. We propose a new algorithm that can robustly aggregate gradients from potentially heterogeneous clients. Our approach leverages the manifold structure inherent in heterogeneous client gradients and evaluates gradient anomaly degrees by projecting them onto this manifold. This algorithm is implemented as a simple and efficient method that accumulates random projections within the subspace defined by the nearest neighbors within a gradient cloud. Our experiments demonstrate consistent performance improvements over state-of-the-art robust aggregation algorithms.

1 Introduction

The effectiveness of deep learning depends on diverse and rich data that accurately represent diverse facets of real-world applications. However, collecting and maintaining such large datasets centrally can be prohibitively expensive. To address this, in distributed collaborative learning, such as federated learning, local *clients* (devices or sites that participate in the learning process) individually manage data. Often, such client datasets are kept undisclosed. For instance, medical or healthcare organizations would want to contribute to building a neural network model while ensuring the privacy of patients and experimental participants.

Training a model in such privacy-preserving environments is coordinated by a centralized server that stores only the model parameters. During each training step, the server distributes the parameters to participating clients who then use their respective local data to calculate their updates, i.e., the loss gradients with respect to the parameters. These local updates are then sent back to the server to be combined and update the global model.

Existing collaborative learning approaches typically assume that all clients possess *clean* and high-quality data. However, in real-world applications, data quality can vary significantly. For example, some clients have highly noisy labels due to the limited availability of experienced annotators, while other clients may be compromised by adversaries. Additionally, some clients might attempt to free-ride, i.e., they wish to use the trained model without contributing adequate data (Fang and Ye 2022). Such *affected* clients can transfer arbitrarily erroneous or nonsensical gradients, which can severely degrade the final model's performance if combined straightforwardly.

Robustness against noisy examples and adversarial attacks has been extensively studied in traditional centralized learning environments. However, existing techniques typically require access to the training data, e.g., to build a noise model or design noise-robust losses; therefore, they are not directly applicable to privacy-preserving distributed learning (see Sec. 2 for examples). For collaborative learning, (Turan et al. 2022) presented a robust gradient aggregation scheme assuming a known number of affected clients, while (Fang and Ye 2022) proposed to store a separate dataset in the server to identify outliers. Recently, (Kim 2022) formulated the gradient aggregation problem into an iteratively reweighted averaging process. The resulting algorithm does not require separate datasets or information on affected clients but is limited to cases where client data are roughly homogeneous (Sec. 3.2).

In this paper, we present a novel gradient aggregation algorithm that can handle situations where there is no known information on the client data or the types of noise or attacks, and it can be applied to both heterogeneous and homogeneous clients. Our algorithm leverages the manifold structure of client gradients and assesses the *degrees of anomaly* of the gradients by projecting them onto this manifold. This is achieved through an intuitive and computationally efficient algorithm that accumulates random projections onto the nearest neighbors in the point cloud of sample gradients. In the experiments with six benchmarks, our algorithm consistently outperformed both the existing robust gradient aggregation algorithm and a baseline uniform gradient averaging approach.

2 Related Work

Robust (Centralized) Learning. State-of-the-art approaches include the use of robust losses (Xu et al. 2019; van Rooyen, Menon, and Williamson 2015; Brooks 2011), regularizers (Liu et al. 2020), filtering out noisy examples (Huang

et al. 2019), and learning noise models (Li et al. 2022b; Yao et al. 2020; van Rooyen and Williamson 2018). For example, (van Rooyen, Menon, and Williamson 2015) eliminated the negative bound of the hinge loss to achieve robustness against symmetric label noise. (Brooks 2011) designed robust losses and training algorithms for support vector machines: The *ramp loss* bounds the upper limit of individual loss values while *hard margin loss* counts the number of mistakes and the points that lie on the decision margin. (Xu et al. 2019) generalized these results for neural networks and proved that their *determinant based mutual information* loss is robust to instance-independent label noise. Other robust losses include mean absolute error (Ghosh, Kumar, and Sastry 2017) and weighted mean absolute error (Wang et al. 2019).

A simple approach to *filter out* noisy examples is to identify those with significant losses (Shen and Sanghavi 2019). However, this strategy may not be effective for large-scale neural networks, as have sufficient capacity to memorize all data, resulting in small loss values even for noisy examples. (Huang et al. 2019) addressed this by scheduling the learning rate to cycle between two modes of overfitting and underfitting, preventing the model from memorizing the noise. Alternatively, (Zhu, Dong, and Liu 2022) proposed a method that evaluates the training losses and the consistency of spatially adjacent labels to detect noisy examples.

(Yao et al. 2020) proposed a method to learn a noise transition matrix that represents the probability of changing a ground-truth label to a noisy observation. They demonstrated that adjusting the loss function based on the learned transition matrix and accordingly training on noisy examples has a similar effect to training on clean examples with the original loss. To achieve this, they employed a Dirichlet prior on the transition matrix and formulated the problem of jointly learning the model parameters and transition matrix into Bayesian inference. (Li et al. 2022b) built on this idea by using the estimated transition matrix to select data instances to examine the correctness of their labels.

Robust Gradient Aggregation. In privacy-preserving collaborative learning, the learning algorithm has access only to the client gradients, and thus, existing centralized robust learning algorithms that require direct access to training data are not applicable. Gradient clipping was initially proposed as an acceleration method for stochastic gradient descent (Chen, Wu, and Hong 2020). Recent studies in distributed learning have observed that the magnitudes of gradients from affected clients tend to be significant. In such cases, gradient clipping can help identify or suppress the effect of these gradients. For example, (Sun et al. 2019) proposed a federated learning scheme that employs a predetermined threshold to clip the client gradients explicitly. Similarly, (Hu et al. 2020) designed an algorithm that normalizes all gradients to reduce the influence of the affected clients. However, our preliminary experiments indicated that the affected gradients' magnitudes are often smaller than those of the clean ones, making it challenging to determine an appropriate threshold value. Our experiments further demonstrate that gradient clipping is only comparable to the baseline uniform gradient aggregation when the clients are heterogeneous (Sec. 4).

The task of aggregating gradients can be viewed as a robust averaging problem, where the objective is to estimate the mean of a set of observations in the presence of outliers. Various robust mean estimators have been proposed as alternatives to the conventional uniform average. For example, one can use the median, which is less sensitive to outliers, as the mean estimator (Pillutla, Kakade, and Harchaoui 2019). (Turan et al. 2022) introduced a method that identifies clean gradients by selecting the nearest neighbors of the median gradient. (Kim 2022)'s algorithm performs iterative weighted averaging of the local client gradients, gradually suppressing outliers. However, in our experiments, we demonstrate that these algorithms are competitive only when clients are homogeneous, and they can perform significantly worse than uniform averaging when clients are heterogeneous (with non-i.i.d. data) in general.

The strategy of filtering examples based on their training losses, originally used in centralized learning (Zhu, Dong, and Liu 2022; Huang et al. 2019; Shen and Sanghavi 2019), can also be applied in collaborative learning by detecting affected clients based on the model's losses averaged on their respective client data, assuming that each client faithfully communicates their average loss (which may not always be the case). (Fang and Ye 2022) proposed a framework that learns multiple local models for heterogeneous clients, where their client re-weighting algorithm monitors the average losses and their temporal differences to weigh the local learning objectives. We performed experiments with an adaptation of this algorithm for our problem setting, showing that our algorithm provides a more robust alternative.

Several existing algorithms rely on certain assumptions, such as additional clean datasets (Regatti, Chen, and Gupta 2021; Pan et al. 2020; Han and Zhang 2019) or access to local client datasets from the server (Cao and Lai 2018), or the knowledge of the number of affected clients (Mhamdi, Guerraoui, and Rouault 2018; Blanchard et al. 2017). Since these assumptions do not align with our problem, these algorithms are not directly applicable in our context

3 Robust Gradient Aggregation Algorithm

3.1 Distributed Gradient Aggregation Scheme

The primary objective of training neural networks is to minimize the sum of losses $\mathcal{L}$ with respect to the model parameters $\mathbf{w}$ on a dataset $\mathcal{P} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{L}$: $\mathcal{F}(\mathbf{w}) = \frac{1}{L}\sum_{i=1}^{L} \mathcal{L}(\mathbf{x}_i, \mathbf{y}_i; \mathbf{w})$. Following (McMahan et al. 2017)'s learning framework, we assume that $\mathcal{P}$ is distributed across K clients. In this case, $\mathcal{F}$ can be rewritten as

$$\mathcal{F}(\mathbf{w}) = \sum_{k=1}^{K} \frac{L_k}{L} \mathcal{F}_k(\mathbf{w}), \quad \text{where} \tag{1}$$

$$\mathcal{F}_k(\mathbf{w}) = \frac{1}{L_k} \sum_{(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{P}_k} \mathcal{L}(\mathbf{x}_i, \mathbf{y}_i; \mathbf{w}),$$

$\mathcal{P}_k$ is the subset of $\mathcal{P}$ corresponding to client k, and $L_k = |\mathcal{P}_k|$. Since the client data $\{\mathcal{P}_k\}$ may be privacy-sensitive, they are kept hidden in individual clients. Generally, the local datasets $\{\mathcal{P}_k\}$ can be non-i.i.d. and they can differ substantially from one another (McMahan et al. 2017). Training

in this scenario can be facilitated by exploiting the additive property of the gradient operator: A single step of gradient descent-based minimization of $\mathcal{F}$ can be stated as

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \sum_{k=1}^{K} \frac{L_k}{L} \mathbf{g}_k^t, \text{ where } \mathbf{g}_k^t = \nabla_{\mathbf{w}} \mathcal{F}_k(\mathbf{w}^t) \quad (2)$$

and η is the learning rate. A server can coordinate these updates by only maintaining the model parameters $\mathbf{w}$: At step t, it broadcasts $\mathbf{w}^t$ to individual clients, and collects and aggregates the resulting gradients $\{\mathbf{g}_k^t\}$ to compute $\mathbf{w}^{t+1}$.

Collaborative learning can enhance privacy and reduce security risks by limiting access to the entire dataset (McMahan et al. 2017). However, this approach can also increase the risk of attacks or data corruption at local clients, especially for those with limited security resources. Such attacks could alter the data or gradients of affected clients, leading to misleading updates if these gradients are naively aggregated in Eq. 2. In practice, the identities and numbers of affected clients may not be known in advance, making it challenging to detect and mitigate such attacks.

3.2 Estimating Gradient Aggregation Weights

Inspired by recent progress in robust collaborative learning (Fang and Ye 2022; Turan et al. 2022; Kim 2022), we introduce a method to regulate the contribution of individual gradients using convex combination weights $\{\alpha_k^t\}_{k=1}^K$:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta K \overline{\mathbf{g}}^t, \text{ where} \quad (3)$$
$$\overline{\mathbf{g}}^t = \sum_{k=1}^{K} \alpha_k^t \frac{L_k}{L} \mathbf{g}_k^t, \quad \alpha_k^t \geq 0, \quad \text{and} \quad \sum_{k=1}^{K} \alpha_k^t = 1.$$

When $\{\alpha_k^t\}$ are identical (i.e. $\alpha_k^t = \frac{1}{K}$), Eq. 3 simplifies to the standard uniform update (Eq. 2) while in the ideal case, $\alpha_k^t = 0$ for the affected clients k's. Between these two extremes, various robust learning algorithms are created by specifying how $\{\alpha_k^t\}$ are determined. For instance, (Turan et al. 2022) proposed constructing the median $\widetilde{\mathbf{g}}^t$ of $\{\mathbf{g}_k^t\}$ as a robust alternative to the average. Subsequently, clients that do not belong to the nearest neighbors of $\widetilde{\mathbf{g}}^t$ are assigned zero α-values. On the other hand, (Kim 2022) proposed an algorithm involving iteratively reweighted averaging of $\{\mathbf{g}_k^t\}$: Initially (at $t = 1$), $\{\alpha_k^t\}$ are uniform. From step $t = 2$ onwards, α_k^{t+1} is inversely proportional to the distance between $\mathbf{g}_k^t$ and $\overline{\mathbf{g}}^t$.

These algorithms have demonstrated state-of-the-art performance when the distributions of local data $\{\mathcal{P}_k\}$ are homogeneous (i.e., i.i.d.). In this case, if a client k is not affected, $\mathcal{P}_k$ provides a good approximation of $\mathcal{P}$, and the corresponding client gradient $\mathbf{g}_k^t$ should be similar to the (weighted) average $\overline{\mathbf{g}}^t$ or median $\widetilde{\mathbf{g}}^t$ calculated based on $\mathcal{P}$. However, when the client datasets are sampled from heterogeneous distributions, even clean gradients may exhibit significant differences from each other. Therefore, the affected gradients may not be adequately characterized by their weak similarities to their average or median, as illustrated in Fig. 1.

Our algorithm leverages the manifold structure M of client gradients and evaluates the (degree of) anomaly of a gradient based on its projection distance onto M.

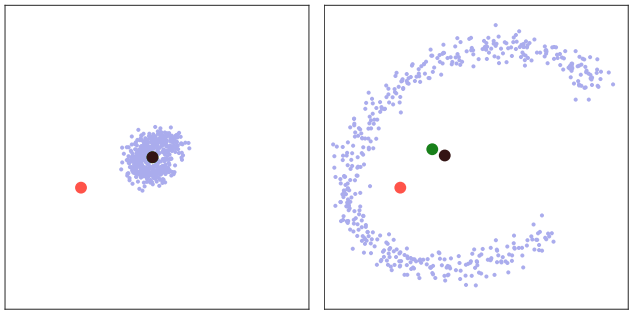

Figure 1: Two approaches to assessing client anomaly. Each data point in the plots represents a client gradient. In the case of relatively i.i.d. client data (Left), the corresponding client gradients tend to be closely clustered around their weighted average (black points) or median (green points). Here, detecting outlier clients (orange points) can be achieved by measuring their distance from the average or median. Alternatively, when clients are heterogeneous in nature, a client may be similar to only a small subset of other clients while significantly differing from the remaining ones (Right). In this scenario, even outlier clients may be closer to the average or median, and therefore client distances to their projections onto the underlying manifold could provide more reliable estimates of anomalies.

Projection Onto Gradient Manifold M. Let M be a smooth Riemannian manifold that is embedded in an ambient inner-product space X by the mapping $\mathcal{I}$. A tangent space $T_{\mathbf{p}}M$ at a point $\mathbf{p} \in M$ is the best linear approximation of M around $\mathbf{p}$.[1] We will use the distance between a point $\mathbf{z} \in X$ and its *orthogonal projection* onto M to indicate its anomaly. To construct this projection, we first identify a point $\mathbf{p} \in M$ with $\mathbf{x} = \mathcal{I}(\mathbf{p})$ which lie in X, using the embedding $\mathcal{I}$. Based on the corresponding *differential* $d\mathcal{I}$, we can also identify $\mathbf{v} \in T_{\mathbf{p}}M$ with $d\mathcal{I}(\mathbf{v}) \in T_{\mathbf{x}}X = X$ where due to the vector space structure of X, the tangent space $T_{\mathbf{x}}X$ can be naturally identified with X (Jost 2011). An orthogonal projection $P[\mathbf{z}]$ of $\mathbf{z}$ onto M is defined by using the inner-product $\langle \cdot, \cdot \rangle_X$ of X such that

$$\langle P[\mathbf{z}] - \mathbf{z}, \mathbf{v} \rangle_X = 0, \quad \forall \mathbf{v} \in d\mathcal{I}(T_{\mathcal{I}^{-1}(P[\mathbf{z}])}M). \quad (4)$$

This implies that the vector $P[\mathbf{z}] - \mathbf{z} \in TX$ lies in the *normal bundle* NM of M (Jost 2011). See (Jost 2011) for a rigorous and general construction of (local) orthogonal projections via normal bundles. In general, P cannot be uniquely defined: There can be multiple $\mathbf{p} \in M$ that satisfy the orthogonality condition in Eq. 4. For example, when M is a sphere centered at the origin of the Euclidean space X and $\mathbf{z}$ is at the origin, all points in M satisfy the orthogonality condition. Since our goal is to evaluate the projection *distance*, these points can be considered equivalent: We take an arbitrary point $\mathbf{p} \in M$ that minimizes the projection distance $\|P[\mathbf{z}] - \mathbf{z}\|$ as $\mathcal{I}^{-1}(P[\mathbf{z}])$. If we assume that the clean gradient vectors form a manifold M, we can assess the anomaly of a given

[1] More generally, $T_{\mathbf{p}}M$ is a vector space of derivative operators on smooth functions around $\mathbf{p} \in M$ (Jost 2011).

gradient vector $\mathbf{z} \in X$ by checking whether $\mathbf{z} \in d\mathcal{I}(T_{\mathbf{p}}M)$ for some $\mathbf{p} \in M$, or equivalently, whether $|P[\mathbf{z}] - \mathbf{z}| \approx 0$.

Projection Onto a Gradient Cloud G as a Sample of M. We consider normalized gradient vectors $G = \{\mathbf{g}_k\}_{k=1}^K$ that are not directly given as elements of an underlying manifold M but rather presented as a potentially noisy sample in the ambient space $X \subset \mathbb{R}^R$: $\|\mathbf{g}_k\| = 1$ and $\sum_j (\mathbf{g}_k)^j = 0$ where $(\mathbf{g}_k)^j$ is the j-th component of $\mathbf{g}_k$. Since we do not have direct access to M, we estimate the projection P from G: Given a gradient vector $\mathbf{z} \in X$, we first find its nearest neighbors $N(\mathbf{z})$ in G. Arranging the elements of $N(\mathbf{z})$ column-wise in a matrix, we obtain a local design matrix $\mathbf{G}_{\mathbf{z}} = [\mathbf{g}_{q(1)}, \ldots, \mathbf{g}_{q(H)}]$ where q assigns the global indices of G from the local indices of $N(\mathbf{z})$. The span of vectors in $\mathbf{G}_{\mathbf{z}}$ can be considered as an estimate of the tangent space $d\mathcal{I}(T_{\mathbf{p}}M)$ at the point $\mathcal{I}(\mathbf{p})$ closest to $\mathbf{z}$. Our *empirical* projection is then defined as the orthogonal projection onto this $d\mathcal{I}(T_{\mathbf{p}}M)$-estimate as a proxy geometry of M around $\mathbf{p}$:

$$P_{N(\mathbf{z})}[\mathbf{z}] = \mathbf{G}_{\mathbf{z}}(\mathbf{G}_{\mathbf{z}}^\top \mathbf{G}_{\mathbf{z}})^{-1}\mathbf{G}_{\mathbf{z}}^\top \mathbf{z}. \tag{5}$$

While $P_{N(\mathbf{z})}[\mathbf{z}]$ is usually well-defined, in the unlikely case that $\mathbf{G}_{\mathbf{z}}$ is not full rank, we can perform column selection using the F-*statistic* (Mood, Graybill, and Boes 1974) in $\mathbf{G}_{\mathbf{z}}$ before constructing $P_{N(\mathbf{z})}$. Once $P_{N(\mathbf{z})}[\mathbf{z}]$ is calculated, the anomaly of $\mathbf{z}$ is obtained as $A(\mathbf{z}) = \|P_{N(\mathbf{z})}[\mathbf{z}] - \mathbf{z}\|^2$.

Estimating Gradient Aggregation Weights. Our algorithm generates aggregation weights $\{\alpha_k\}_{k=1}^K$ by first sampling random network parameters $\mathbf{w}^s$ at the server and sharing them with each client. The corresponding client gradients $\{\mathbf{g}_k^s\}_{k=1}^K$ form a data cloud G^s. For each gradient $\mathbf{g}_k^s \in G^s$, we evaluate its anomaly $A(\mathbf{g}_k^s)$ by projecting it to $\mathbf{G}_k^s$, formed by the nearest neighbors N_k^s of $\mathbf{g}_k^s$ in $G^s \setminus \{\mathbf{g}_k^s\}$. The gradient aggregation weights at s are then determined as

$$\alpha_k^s = \frac{\overline{\alpha}_k^s}{\sum_{k=1}^K \overline{\alpha}_k^s}, \quad \text{where } \overline{\alpha}_k^s = \exp\left(-\frac{A(\mathbf{g}_k^s)}{\sigma_\alpha^s}\right). \tag{6}$$

Here, the scale parameter σ_α^s is set to the mean of $\{A(\mathbf{g}_k^s)\}_{k=1}^K$. We repeat this process $S = 20$ times with different random parameters $\mathbf{w}^s$, and determine the final aggregation weights $\{\alpha_k\}$ as the average of their corresponding counterparts $\{\alpha_k^s\}$.

The time complexity of each step (out of S steps) in our algorithm is $O(RKH + RK^2 + H^3)$, where R is the size of the gradient vector, K is the number of clients, and H is the local neighborhood size. The calculation of the projection in our algorithm, which involves solving the linear system in Eq. (5), results in negligible computational costs due to the small size of the system ($H \times H$ with $H = 10$; see Sec. 4). The primary bottleneck is twofold: 1) evaluating client gradients from each set of random parameters $\mathbf{w}^s$ as our algorithm requires gradient evaluations before the main network training occurs and 2) finding the neighborhood N_k of each gradient k. For a large number K of clients, we can improve the complexity by employing an approximate nearest neighbor search. For *CIFAR100* dataset (see Sec. 4), the initial gradient evaluation took around one minute. Our algorithm incurs $O(RK)$ memory complexity to store the client gradients. The resulting robust collaborative learning process is summarized in Algorithm 1.

Algorithm 1: Robust collaborative learning algorithm. Training data is distributed across K clients. An unknown number out of K clients will provide erroneous gradients.

Input: Gradient neighborhood size $H = 10$ and the number of random projection steps $S = 20$.
for $s = 1, \ldots, S$ **do**
  Randomly construct $\mathbf{w}^s$ and send it to clients.
  Fetch gradients $G^s = \{\mathbf{g}_k^s\}_{k=1}^K$ from clients.
  for $k = 1, \ldots, K$ **do**
    Build global indices $\{q\}$ for the neighborhood N_k^s of $\mathbf{g}_k^s$ and $\mathbf{G}_k^s = [\mathbf{g}_{q(1)}^s, \ldots, \mathbf{g}_{q(H)}^s]$.
    Calculate $\mathbf{g}_k^s$-projection: $P_{N_k^s}[\mathbf{g}_k^s] = \mathbf{G}_k^s(\mathbf{G}_k^{s\top}\mathbf{G}_k^s)^{-1}\mathbf{G}_k^s\mathbf{g}_k^s$.
    Calculate anomaly degree $A(\mathbf{g}_k^s) = \|P_{N_k^s}[\mathbf{g}_k^s] - \mathbf{g}_k^s\|^2$.
  end for
  $\sigma_\alpha^s = \sum_{k=1}^K A(\mathbf{g}_k^s)/K$.
  for $k = 1, \ldots, K$ **do**
    $\overline{\alpha}_k^s = \exp(-A(\mathbf{g}_k^s)/\sigma_\alpha^s)$; $\alpha_k^s = \overline{\alpha}_k^s / \sum_{k=1}^K \overline{\alpha}_k^s$.
  end for
end for
for $k = 1, \ldots, K$ **do**
  $\alpha_k = \sum_{s=1}^S \alpha_k^s / S$; $\alpha_k = \alpha_k / \sum_{l=1}^K \alpha_l$.
end for
Perform network training steps in Eq. 3 using $\{\alpha_k\}_{k=1}^K$.
Output: Model parameters $\mathbf{w}$ and aggregation weights $\{\alpha_k\}_{k=1}^K$.

3.3 Robustness of the Anomaly Estimates

Our method for estimating the tangent space $T_{\mathbf{p}}M$ differs from common approaches: Typically, principal component analysis (PCA) is first performed on N_k to reduce the dimensionality to that of $T_{\mathbf{p}}M$ (Singer and Wu 2016). This provides a rigorous convergence guarantee of the resulting tangent space estimate as $|G| \to \infty$ even under noise. However, this method requires a known manifold dimensionality, which can be challenging to estimate from a sampled point cloud. Another significant limitation of this local PCA-based approach applied to our gradient aggregation setting is that $G \setminus \{\mathbf{g}_k\}$ may contain outliers as more than one client can be affected in G. Since PCA is susceptible to outliers, the resulting tangent space estimates can be erroneous. The supplemental document shows that our method offers more reliable anomaly estimates compared to the PCA-based alternative.

A deeper insight into the robustness of our approach against outliers in $G \setminus \mathbf{g}_k$ can be gained by noting that all gradients in G are normalized. In this case, the anomaly degree can be restated as

$$A(\mathbf{g}_k) = \sum_{j=1}^R \left(h_k((\mathbf{G}_k)^{j,*}) - (\mathbf{g}_k)^j\right)^2, \quad \text{where} \tag{7}$$
$$h_k(\mathbf{s}) = \mathbf{a}_k^\top \mathbf{s}, \ \mathbf{a}_k = (\mathbf{G}_k^\top \mathbf{G}_k)^{-1}\mathbf{G}_k^\top \mathbf{g}_k,$$

and $(\mathbf{G}_k)^{j,*}$ is the j-th row of $\mathbf{G}_k$. This suggests that the quantity $A(\mathbf{g}_k)$ can be interpreted as the error incurred when attempting to *reconstruct* the elements of $\mathbf{g}_k$ using linear

least-squares estimator h. In this method, the rows of $\mathbf{G}_k$ are used as training inputs, and the resulting reconstruction is a weighted combination of the columns of $\mathbf{G}_k$ as *features*, each corresponding to a client gradient in N_k. Outlier gradients that do not contribute to constructing $\mathbf{g}_k$ can be ignored by assigning low-magnitude weight values $\{(\mathbf{a}_k)^j\}$ to them. For instance, on the *CIFAR10* dataset (see Sec. 4), the average magnitudes of the elements of $\mathbf{a}_k$ corresponding to clean and affected clients were 0.147 and 0.009, respectively.

The above highlights two main factors that contribute to the robustness of the projection operator P_N: (1) the assumed manifold structure of clean gradients and (2) considerably smaller neighborhood size $H = |N_k|$ than the size R of the gradient vector. When a clean gradient $\mathbf{g}_k$ is present, both $\mathbf{g}_k$ and its spatial neighborhood N_k lie within a low-dimensional subspace (tangent space $T_{\mathbf{p}}M$) of $X = \mathbb{R}^R$ (1); therefore combinations of columns in $\mathbf{G}_k$ can effectively reconstruct $\mathbf{g}_k$ (Eq. 7). When N_k contains outliers, if $H \ll R$ (2), these gradients are unlikely to be coincidentally strongly correlated with $\mathbf{g}_k$ (and hence contribute to reconstructing $\mathbf{g}_k$), resulting in small $\mathbf{a}_k$ weights in the estimator h_k. On the other hand, if $\mathbf{g}_k$ is an outlier lying outside $T_{\mathbf{p}}M$ and $H \ll R$ (2), both clean and outlier gradients in N_k are likely to deviate significantly from $\mathbf{g}_k$. As a result, $\mathbf{G}_k$ is likely to fail to reproduce $\mathbf{g}_k$.

The robustness of the projection operator P_N is specific to the gradient aggregation setting and should not be considered a general indicator of the robustness of least-squares estimation. It also points out a simple failure case where the number of features, H, becomes equal to the number of data instances, R, in $\mathbf{G}_k$ and those features are in general position. In this case, $\mathbf{G}_k$ becomes invertible, leading to constant zero estimation error (Eq. 7). The abnormalities of outliers $\mathbf{g}_k$ were substantially reduced when N_k was augmented with 10,000 random vectors. This suggests that as the value of K increases, it is advisable to keep the neighborhood size H small. Accordingly, in all experiments conducted, a fixed neighborhood size of 10 was used (see our supplemental document). While optimizing this parameter for each problem and dataset may enhance performance, it may not be feasible in certain scenarios as it would necessitate separate clean validation sets at the server.

Explicit Robustification. Considering the least-square estimation interpretation of P_N and its empirical robustness against outlier features,[2] it is enticing to explore the possibility of further enhancing its robustness through feature selection approaches. We conducted preliminary experiments with two robust feature selection methods. We conducted preliminary experiments with LASSO as an implicit feature selector. This replaces A by calculating the coefficients $\mathbf{a}_k$ of h_k (Eq. 7) as the minimizer of

$$A'(\mathbf{g}_k) = \sum_{j=1}^{R} \left(\mathbf{a}_k^\top (\mathbf{G}_k)^{j,*} - (\mathbf{g}_k)^j\right)^2 + \gamma \|\mathbf{a}_k\|_1. \qquad (8)$$

[2]This should not be confused with the well-known high-sensitivity of least-squares estimation to outlier *output instances* (rows of $\mathbf{g}_k$; see Eq. 7).

We consistently observed that the best gradient aggregation performance was achieved when $\gamma = 0$, resulting in A' being equal to A. LASSO promotes sparsity in $\mathbf{a}$, helping suppress the contribution of outlier gradients. However, this has an adversarial effect of suppressing potentially beneficial correlated clean gradients since, in this way, it can enhance $\mathbf{a}$'s sparsity while only moderately sacrificing the reconstruction error (the first term in Eq. 8). Consequently, this can lead to increased abnormalities A' for clean clients (see supplemental document for an example).

4 Experiments

To evaluate the effectiveness of our robust gradient aggregation algorithm, we conducted experiments with six benchmark datasets.

Datasets. The *CIFAR10* and *CIFAR100* datasets consist of 60,000 color images from 10 and 100 classes, respectively (Krizhevsky 2009). For each dataset, 50,000 images were used for training, and the remaining 10,000 images were reserved for testing. *TinyImageNet* is a subset of the ImageNet 2017 benchmark, consisting of 100,000 training and 10,000 testing images evenly covering 200 object categories (Le and Yang 2015). The *Kuzushiji49* dataset provides 223,365 training and 38,547 testing images of 49 Japanese characters (Clanuwat et al. 2018). The Fashion-MNIST (*FMNIST*) and extended MNIST letters (*EMNISTL*) datasets provide 70,000 images of 10 cloth categories (Xiao, Rasul, and Vollgraf 2017) and 124,800 letter images (LeCun et al. 1998), respectively.

Data Allocation and Implementation Details. We used $K = 100$ clients for all datasets. To distribute each dataset to these clients, we extended (McMahan et al. 2017)'s approach (to more than ten classes), which is commonly used to prepare learning environments with heterogeneous clients (Li et al. 2020, 2022a). Firstly, we partitioned the dataset into $0.2 \times C \times K$ *shards*, where C is the number of classes. Each shard contained only a single class, and all shards were of equal size. Then, we randomly assigned $0.2 \times C$ shards to each client, ensuring that each client received a subset of up to 20% of all classes. Our supplementary material presents the results obtained when client data allocation was performed by sampling class labels from Dirichlet distributions.

To simulate the affected clients, we employed the class-symmetric flip model (van Rooyen, Menon, and Williamson 2015; Patrini et al. 2017; Han et al. 2018). For an affected client k, we generated a transition matrix $\mathbf{T}_k \in \mathbb{R}^{C \times C}$ by randomly sampling each row from the uniform distribution on $[0, 1]^C$ and normalizing the results. Then, the label of each data point with class j in $\mathcal{P}_k$ was reassigned by sampling from $(\mathbf{T}_k)^{j,*}$. For each number of affected clients in $\{10, 20, 30, 40, 50, 60, 70\}$ (out of $K = 100$), we repeat experiments 10 times and report the average results.

We used convolutional neural networks with two convolution layers and two fully-connected layers for *Kuzushiji49*, *FMNIST*, and *EMNISTL*, following the setup of (LeCun et al. 1998). The convolution layers comprised 10 and 20 filters of size 5×5, followed by 2×2 max pooling. The fully-connected

Affected clients		10	20	30	40	50	60	70
Dataset	Method	Mean accuracy ± standard deviation × 100						
CIFAR10	*Uniform*	89.58±0.15	89.08±0.26	88.79±0.28	87.80±0.50	86.98±0.47	85.51±0.92	82.19±1.17
	(Kim 2022)	89.50±0.21	88.98±0.33	87.86±0.82	85.83±0.81	84.79±1.03	83.06±1.44	78.32±1.16
	(Turan et al. 2022)	89.48±0.15	88.46±0.40	87.68±0.44	83.92±1.18	79.52±1.44	73.24±2.21	69.23±1.95
	(Sun et al. 2019)	***90.00±0.21***	**89.61±0.23**	**88.92±0.26**	87.61±0.39	86.82±0.62	85.10±1.07	81.00±1.37
	(Fang and Ye 2022)	89.59±0.17	89.09±0.26	88.72±0.29	87.64±0.39	84.87±7.04	85.56±0.99	82.16±1.23
	Denoising	89.29±0.20	88.88±0.31	88.47±0.32	87.62±0.41	86.61±0.53	85.41±1.08	82.12±1.23
	Ours	**89.91±0.16**	***89.62±0.41***	***89.71±0.30***	***89.40±0.32***	***89.00±0.71***	***88.33±1.42***	***85.64±2.85***
CIFAR100	*Uniform*	66.64±0.36	65.68±0.24	64.41±0.38	62.82±0.40	60.35±0.53	56.23±0.69	50.55±0.82
	(Kim 2022)	*66.90±0.25*	65.84±0.36	64.60±0.39	62.45±0.26	59.48±0.40	55.27±0.82	49.16±1.15
	(Turan et al. 2022)	66.74±0.27	65.34±0.35	63.24±0.43	59.39±0.36	45.54±0.89	42.82±0.39	39.28±1.03
	(Sun et al. 2019)	66.73±0.26	65.74±0.23	64.41±0.22	62.83±0.32	60.13±0.45	55.96±0.83	49.83±0.80
	(Fang and Ye 2022)	66.76±0.34	65.77±0.36	64.44±0.29	62.81±0.44	60.20±0.60	56.22±0.76	50.62±1.06
	Denoising	66.17±0.24	65.19±0.27	63.60±0.51	61.93±0.32	59.14±0.53	55.01±0.94	49.27±0.81
	Ours	66.87±0.35	***66.50±0.46***	***65.83±0.46***	***64.39±0.32***	***63.47±0.42***	***62.67±0.49***	***58.38±1.27***
TinyImageNet	*Uniform*	65.66±0.30	64.89±0.27	63.58±0.29	61.89±0.25	59.33±0.52	55.07±0.85	49.00±0.76
	(Kim 2022)	65.62±0.26	64.91±0.26	**63.83±0.19**	61.84±0.19	59.10±0.57	54.13±0.94	47.87±1.13
	(Turan et al. 2022)	65.43±0.25	64.42±0.31	62.70±0.26	58.26±0.26	44.88±0.74	41.66±0.59	37.74±0.74
	(Sun et al. 2019)	65.55±0.19	64.84±0.21	63.63±0.38	61.96±0.26	59.49±0.66	54.83±0.93	48.75±0.96
	(Fang and Ye 2022)	65.55±0.19	64.79±0.28	63.70±0.47	62.03±0.33	59.50±0.48	55.36±0.64	49.07±0.86
	Denoising	65.27±0.25	64.19±0.17	62.99±0.34	61.10±0.21	58.51±0.50	53.88±0.82	47.77±0.88
	Ours	*65.77±0.23*	***65.25±0.19***	***64.59±0.38***	***63.48±0.33***	***62.69±0.34***	***61.73±0.55***	***58.52±0.80***
FMNIST	*Uniform*	77.69±0.48	77.11±0.83	76.29±0.57	75.46±0.62	73.25±1.14	72.14±2.46	69.91±1.75
	(Kim 2022)	76.34±1.03	75.83±1.07	75.79±1.13	75.12±0.98	72.25±1.05	70.61±2.55	68.73±1.59
	(Turan et al. 2022)	77.06±0.59	75.83±1.11	74.36±0.63	72.22±1.13	65.48±1.78	62.23±3.76	62.51±2.81
	(Sun et al. 2019)	*77.98±0.52*	76.76±0.76	76.10±0.98	75.42±0.53	72.93±1.24	70.93±1.92	69.12±2.14
	(Fang and Ye 2022)	77.65±0.51	77.19±0.64	76.53±0.75	75.69±0.93	72.91±1.17	71.87±2.33	69.89±2.00
	Denoising	77.62±0.38	77.07±0.62	76.23±0.82	75.54±0.86	73.55±1.05	71.80±2.19	70.04±1.56
	Ours	77.87±0.72	***77.66±0.67***	***77.40±0.97***	***77.31±0.90***	***76.36±0.99***	***74.96±1.36***	***72.31±1.67***
Kuzushiji49	*Uniform*	48.45±0.78	47.75±0.60	46.71±0.73	44.07±0.87	41.22±1.11	37.97±0.68	31.28±1.93
	(Kim 2022)	48.08±0.58	47.14±0.79	46.50±0.93	44.24±0.49	40.62±0.52	37.07±0.94	30.27±2.12
	(Turan et al. 2022)	48.40±0.54	47.60±0.46	45.41±0.79	40.81±1.05	25.33±2.04	23.39±1.69	20.70±1.54
	(Sun et al. 2019)	48.42±0.78	47.35±0.63	46.35±0.58	44.32±0.96	40.97±1.33	37.66±0.96	30.89±2.43
	(Fang and Ye 2022)	48.51±0.62	47.59±0.61	46.52±0.64	44.18±0.98	41.20±1.14	38.11±1.12	32.00±2.53
	Denoising	*48.52±0.67*	47.06±0.33	45.98±0.73	44.04±0.88	40.70±0.79	37.83±0.80	31.26±1.77
	Ours	48.50±0.59	*47.85±0.72*	*46.76±0.71*	***45.50±0.90***	***43.51±1.40***	***41.73±1.04***	***37.62±2.08***
EMNISTL	*Uniform*	67.91±0.56	66.68±0.88	64.60±0.84	61.68±0.94	57.81±1.30	49.89±2.33	40.51±2.66
	(Kim 2022)	67.41±0.69	65.48±0.91	63.62±1.23	61.01±1.36	55.84±1.57	47.55±1.84	39.36±3.55
	(Turan et al. 2022)	67.46±0.90	66.03±0.47	62.17±0.77	55.78±1.24	37.08±1.13	29.96±2.94	26.06±2.80
	(Sun et al. 2019)	67.55±0.68	66.38±0.90	63.99±0.82	61.33±0.89	56.43±1.50	49.61±2.11	37.31±2.19
	(Fang and Ye 2022)	67.76±0.90	66.57±0.74	64.42±1.10	62.24±1.29	57.60±1.81	51.12±1.55	41.70±3.99
	Denoising	67.21±0.84	66.73±0.70	64.22±0.99	62.19±1.69	56.37±2.05	49.87±2.09	41.90±3.49
	Ours	***68.68±0.78***	***67.70±0.84***	***66.81±1.28***	***65.74±1.28***	***64.11±2.24***	***59.74±1.87***	***49.81±2.36***

Table 1: Performance of different gradient aggregation methods when client data are allocated using an extension of McMahan et al.'s approach (McMahan et al. 2017). Our supplementary material presents the corresponding results achieved when client data allocation was conducted by sampling class labels from Dirichlet distributions. The best results are highlighted in *italic*. The results of the statistical significance tests for differences from baseline *Uniform* are represented with bold (significantly better) and underline (significantly worse) fonts using a t-test with $\alpha = 0.95$. With the exceptions of (Sun et al. 2019) on *CIFAR10* with $\{10,20,30\}$ affected clients and (Kim 2022) on *TinyImageNet* with 30 affected clients, all existing approaches were statistically equivalent or significantly worse than *Uniform*. In contrast, our algorithm was significantly better than *Uniform* in 36 out of the total 42 cases and was never significantly worse.

layers were of 50 and 10. For the remaining datasets, we combined a ResNet50 pre-trained on ImageNet dataset with three fully connected layers of size 300 each, following (Kim 2022). All experiments were conducted on a machine with two Intel Xeon Silver 4210R CPUs and two NVIDIA RTX3090 GPUs.

Baselines. For comparison, we also conducted experiments with 1) baseline uniform aggregation (i.e. $\alpha_k = \frac{1}{K}, \forall 1 \leq k \leq K$; *Uniform*), 2) (Kim 2022)'s iterative reweighting algorithm, 3) (Turan et al. 2022)'s median-based gradient selection approach, 4) (Sun et al. 2019)'s gradient clipping algorithm, and 5) (Fang and Ye 2022)'s robust federated learning algorithm, which was originally designed for learning personalized models for heterogeneous clients. We adapted their *client confidence re-weighting* scheme to centralized gradient aggregation by introducing a new client maintaining the combined gradient $K\overline{\mathbf{g}}^t$. Turan et al.'s algorithm determines the gradients distinct from the median as outliers and

requires a known number of outlier clients: We used the ground-truth numbers of affected clients in our experiments while in practical applications, one would need to determine them as a hyperparameter. 6) We also performed experiments with an algorithm that aims to denoise the gradient matrix $\mathbf{G}^s = [\mathbf{g}_1^s, \ldots, \mathbf{g}_K^s]$ (*Denoising*) by considering the affected gradients as noisy observations of underlying ground truths. This constructs a new matrix $(\mathbf{G}_*^s)$ by minimizing

$$\mathcal{Q}(\mathbf{G}) = \frac{1}{2}\|\mathbf{G} - \mathbf{G}^s\|_F^2 + \lambda_N \|\mathbf{G}\|_*, \tag{9}$$

where $\|\mathbf{A}\|_F$ and $\|\mathbf{A}\|_*$ are the Frobenius and nuclear norms of $\mathbf{A}$, respectively. The energy $\mathcal{Q}$ is convex, and its minimum can be determined through singular value shrinkage (Bach et al. 2012). Once $\mathbf{G}_*^s$ is obtained, the corresponding gradients are uniformly aggregated.

Analysis. The results of our experiments are summarized in Table 1. Existing gradient combination algorithms demonstrated comparable or inferior performance to uniform averaging across all datasets: (Fang and Ye 2022)'s algorithm was on par with *Uniform*. This algorithm determines the aggregation weights $\{\alpha_k^t\}_{k=1}^K$ based on the average client losses and their temporal variations. However, in our setting, these loss values did not vary notably across the clean and affected clients, resulting in almost uniform weights. (Sun et al. 2019)'s algorithm assumes that the outlier gradients exhibit larger magnitudes than clean ones. This assumption is similar to that made in (Fang and Ye 2022), as the gradient magnitudes are directly proportional to the loss scales, and it also attained an equivalent level of performance as *Uniform*. (Kim 2022)'s algorithm performed significantly worse than *Uniform*, which can be attributed to the fact that when clients are *heterogeneous*, outlier gradients can be closer to the averages (Fig. 1), and therefore contribute materially to the final aggregation. (Turan et al. 2022)'s algorithm also achieved considerably worse results as it filtered out a large portion of clean gradients, assuming that the affected gradients were often close to the median.

We set λ_N to 10^{-2} for *Denoising* since smaller values led to nearly identical results as *Uniform*. Nonetheless, the resulting accuracies were slightly worse than *Uniform*. In cases where the affected gradients stem from incorrect training labels, such as those arising from adversarial attacks, they can substantially deviate from the ground-truth gradients. In such scenarios, attempting to denoise $\mathbf{G}^s$ by lowering its nuclear norm (second term in $\mathcal{Q}$; Eq. 9) can inadvertently alter the clean gradients, as they are typically more structured: Modifying them, rather than the affected ones, can more readily reduce the nuclear norm while retaining low reconstruction error (first term in $\mathcal{Q}$).[3]

Our findings suggest that existing robust gradient aggregation algorithms, which have demonstrated competitive performance on homogeneous clients, may not be effective in the presence of heterogeneous clients. Specifically, our experiments indicate that the aggregation weights, calculated based on the gradient distances to the weighted averages or median, can be misleading in such scenarios. Moreover, the assumptions typically imposed in prior work, such as the uniformity of loss values and gradient magnitudes (Fang and Ye 2022; Sun et al. 2019), may not hold in heterogeneous settings, rendering those methods ineffective.

Our algorithm consistently outperformed *Uniform* and state-of-the-art gradient aggregation algorithms by effectively leveraging the nonlinear structures of heterogeneous gradients through the assessment of manifold projection distances. The performance gains were more noticeable in scenarios with 1) a large number of affected clients and 2) datasets with a large number of classes. Specifically, for *TinyImageNet* with 200 classes, when 70 clients were affected, ours achieved a 19.43% accuracy improvement from *Uniform*, while the advantage almost disappeared (0.17%; statistically insignificant) when 10 clients were affected. For *CIFAR10* with only ten classes, our algorithm achieved a 4.2% performance margin for 70 affected clients. When all clients were clean, *Uniform* and ours were statistically equivalent, with accuracy values of 90.01 ± 0.12 and 89.94 ± 0.18, respectively.

In the supplementary material, we present additional experiments on homogeneous clients. Despite our algorithm not being explicitly designed for this scenario, our results demonstrate that it is still competitive with or even superior to existing algorithms, highlighting its general applicability.

5 Conclusions

We have considered the challenge of robustly aggregating heterogeneous client gradients in scenarios where no information about the affected clients, such as their number and types of attacks, is available. Our thesis posits that the client gradients lie on a manifold, and we exploit this structure by assessing the anomalies of given gradients through their projections onto this manifold. To this end, we have devised an efficient algorithm that iteratively generates random model parameters and projects the resulting gradients onto the subspace spanned by their spatial neighborhoods. This instantiation of our thesis has produced promising results, outperforming existing state-of-the-art methods on heterogeneous clients while delivering comparable outcomes in homogeneous ones.

Further research should focus on developing theoretical foundations to provide a robust underpinning for these findings. This would entail the development of new convergence analysis techniques, as individual client gradients may not provide unbiased estimators of the clean aggregated gradients in environments that involve heterogeneous clients, unlike the settings in which existing robust distributed learning techniques were designed. All robust gradient aggregation algorithms considered in this paper may fail when client attacks are *coordinated*. As a simple example, when local clients transfer multiple identical erroneous gradients, these algorithms will mistake them for clean ones. While simple heuristics can easily detect this particular case, future work should systematically investigate extending our algorithm to respond to such coordinated attacks.

[3] Based on this observation, we performed preliminary experiments with a new client anomaly degree evaluating how the columns of the minimizer $\mathbf{G}_*^s$ deviate from $\mathbf{G}^s$ (with $(\mathbf{G}^s)^{*,k}$ being the k-th column of $\mathbf{G}^s$): $A''(\mathbf{g}_k) = \|(\mathbf{G}^s)^{*,k} - (\mathbf{G}_*^s)^{*,k}\|^2$. While this approach led to measurable improvements over the *Uniform* approach, our final design consistently and significantly outperformed it.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant (No. 2021R1A2C2012195) and the Institute of Information & communications Technology Planning & Evaluation (IITP) grant (No.2019-0-01906, Artificial Intelligence Graduate School Program (POSTECH)), funded by the Korea government (MSIT).

References

Bach, F.; Jenatton, R.; Mairal, J.; and Obozinski, G. 2012. Optimization with Sparsity-Inducing Penalties. *Foundations and Trends in Machine Learning*, 4(1): 1–106.

Blanchard, P.; Mhamdi, E. M. E.; Guerraoui, R.; and Stainer, J. 2017. Machine learning with adversaries: Byzantine tolerant gradient descent. In *NIPS*.

Brooks, J. P. 2011. Support vector machines with the ramp loss and the hard margin loss. *Operations Research*, 59: 467–479.

Cao, X.; and Lai, L. 2018. Robust distributed gradient descent with arbitrary number of Byzantine attackers. In *ICASSP*.

Chen, X.; Wu, S. Z.; and Hong, M. 2020. Understanding gradient clipping in private SGD: a geometric perspective. In *NeurIPS*.

Clanuwat, T.; Bober-Irizar, M.; Kitamoto, A.; Lamb, A.; Yamamoto, K.; and Ha, D. 2018. Deep learning for classical Japanese literature. In *arXiv:1812.01718*.

Fang, X.; and Ye, M. 2022. Robust federated learning with noisy and heterogeneous clients. In *CVPR*, 10072–10081.

Ghosh, A.; Kumar, H.; and Sastry, P. S. 2017. Robust loss functions under label noise for deep neural networks. In *AAAI*.

Han, B.; Yao, Q.; Yu, X.; Niu, G.; Xu, M.; Hu, W.; Tsang, I. W.; and Sugiyama, M. 2018. Co-teaching: robust training of deep neural networks with extremely noisy labels. In *NeurIPS*.

Han, Y.; and Zhang, X. 2019. Robust federated learning via collaborative machine teaching. In *AAAI*.

Hu, Z.; Shaloudegi, K.; Zhang, G.; and Yu, Y. 2020. FedMGDA+: federated learning meets multi-objective optimization. In *arXiv:2006.11489*.

Huang, J.; Qu, L.; Jia, R.; and Zhao, B. 2019. O2U-Net: a simple noisy label detection approach for deep neural networks. In *ICCV*, 3326–3334.

Jost, J. 2011. *Riemannian Geometry and Geometric Analysis*. New York: Springer.

Kim, K. I. 2022. Robust combination of distributed gradients under adversarial perturbations. In *CVPR*.

Krizhevsky, A. 2009. Learning multiple layers of features from tiny images. Technical report, University of Toronto.

Le, Y.; and Yang, X. 2015. Tiny ImageNet visual recognition challenge. Technical Report CS231N Course, Stanford University.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11): 2278–2324.

Li, Q.; Diao, Y.; Chen, Q.; and He, B. 2022a. Federated learning on non-IID data silos: an experimental study. In *ICDE*.

Li, S.-Y.; Shi, Y.; Huang, S.-J.; and Chen, S. 2022b. Improving deep label noise learning with dual active label correction. *Machine Learning*.

Li, T.; Sahu, A. K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; and Smith, V. 2020. Federated optimization in heterogeneous networks. In *MLSys*, 429–450.

Liu, S.; Niles-Weed, J.; Razavian, N.; and Fernandez-Granda, C. 2020. Early-learning regularization prevents memorization of noisy labels. In *NeurIPS*.

McMahan, H. B.; Moore, E.; Ramage, D.; Hampson, S.; and y Arcas, B. A. 2017. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*.

Mhamdi, E. M. E.; Guerraoui, R.; and Rouault, S. 2018. The hidden vulnerability of distributed learning in Byzantium. In *ICML*.

Mood, A. M.; Graybill, F. A.; and Boes, D. C. 1974. *Introduction to the Theory of Statistics*. McGraw Hill, 3rd edition.

Pan, X.; Zhang, M.; Wu, D.; Xiao, Q.; Ji, S.; and Yang, M. 2020. Justinian's GAAvernor: Robust distributed learning with gradient aggregation agent. In *USENIX Security Symposium*.

Patrini, G.; Rozza, A.; Menon, A. K.; Nock, R.; and Qu, L. 2017. Making deep neural networks robust to label noise: a loss correction approach. In *CVPR*.

Pillutla, K.; Kakade, S. M.; and Harchaoui, Z. 2019. Robust aggregation for federated learning. In *arXiv:1912.13445*.

Regatti, J. R.; Chen, H.; and Gupta, A. 2021. ByGARS: Byzantine SGD with arbitrary number of attackers using reputation scores. In *ICML Workshop on Federated Learning for User Privacy and Data Confidentiality*.

Shen, Y.; and Sanghavi, S. 2019. Learning with bad training data via iterative trimmed loss minimization. In *ICML*.

Singer, A.; and Wu, H.-T. 2016. Spectral convergence of the connection Laplacian from random samples. *Information and Inference: A Journal of the IMA*, 6: 58–123.

Sun, Z.; Kairouz, P.; Suresh, A. T.; and McMahan, H. B. 2019. Can you really backdoor federated learning? In *arXiv:1911.07963*.

Turan, B.; Uribe, C. A.; Wai, H.-T.; and Alizadeh, M. 2022. Robust distributed optimization with randomly corrupted gradients. *IEEE TSP*, 70: 3484–3498.

van Rooyen, B.; Menon, A. K.; and Williamson, R. C. 2015. Learning with symmetric label noise: the importance of being unhinged. In *NIPS*.

van Rooyen, B.; and Williamson, R. C. 2018. A theory of learning with corrupted labels. *JMLR*, 18: 1–50.

Wang, X.; Hua, Y.; Kodirov, E.; and Robertson, N. M. 2019. IMAE for noise-robust learning: mean absolute error does not treat examples equally and gradient magnitude's variance matters. In *arXiv:1903.12141*.

Xiao, H.; Rasul, K.; and Vollgraf, R. 2017. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. In *arXiv:1708.07747*.

Xu, Y.; Cao, P.; Kong, Y.; and Wang, Y. 2019. $\mathcal{L}_{\text{DMI}}$: A Novel Information-theoretic Loss Function for Training Deep Nets Robust to Label Noise. In *NeurIPS*.

Yao, Y.; Liu, T.; Han, B.; Gong, M.; Deng, J.; Niu, G.; and Sugiyama, M. 2020. Dual T: reducing estimation error for transition matrix in label-noise learning. In *NeurIPS*.

Zhu, Z.; Dong, Z.; and Liu, Y. 2022. Detecting corrupted labels without training a model to predict. In *ICML*.