Coordinate-Wise Median in Byzantine Federated Learning

Mélanie Cambus
Aalto University
Espoo, Finland

Darya Melnyk
TU Berlin
Berlin, Germany

Tijana Milentijević
TU Berlin
Berlin, Germany

Stefan Schmid
TU Berlin
Berlin, Germany

Abstract

Federated learning enables clients to train local models on private data while exchanging model updates only. A key step in this process is gradient aggregation. We investigate the coordinate-wise median as an aggregation rule in both centralized and decentralized federated learning under Byzantine failures. In order to lower the communication cost in the decentralized setting, we allow clients to agree approximately on model parameters, which is referred to as approximate agreement problem. We propose two aggregation algorithms for centralized coordinate-wise median aggregation: the Minimum Diameter (MD) algorithm and the Hyperbox algorithm. We prove that both satisfy the box validity condition and can tolerate up to $\frac{n}{3}$ and $\frac{n}{2}$ Byzantine clients, respectively. We further show that only the Hyperbox algorithm can be generalized to the decentralized setting. Through empirical evaluation, we demonstrate that the MD algorithm with coordinate-wise median aggregation is more resilient to sign-flip attacks than its mean-based counterpart, highlighting the robustness of median-based aggregation in adversarial environments.

CCS Concepts

• **Computing methodologies** → *Machine learning algorithms*; **Distributed algorithms**.

Keywords

federated learning, collaborative learning, Byzantine failures, validity conditions, coordinate-wise median

ACM Reference Format:
Mélanie Cambus, Darya Melnyk, Tijana Milentijević, and Stefan Schmid. 2018. Coordinate-Wise Median in Byzantine Federated Learning. In . ACM, New York, NY, USA, 6 pages. https://doi.org/XXXXXXX.XXXXXXX

Research supported by the German Research Foundation (DFG), Schwerpunktprogramm: Resilienz in Vernetzten Welten (SPP 2378), project ReNO, 2023-2027.

Conference'17, Washington, DC, USA

ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Federated learning allows training a machine learning model in a distributed manner, where clients can keep their data locally. It is usually implemented using a central server that aggregates model parameters and shares the result with the clients. Recently, decentralized solutions have been considered, where clients aggregate the model parameters in a peer-to-peer manner. One particularly interesting solution is collaborative learning [8]: the idea is to allow clients to agree on approximately the same model parameters, thus reducing the communication cost compared to exact agreement.

One of the main challenges in federated learning is dealing with adversarial client inputs. Such input can be unpredictable, e.g., due to failures in the training process or malicious behavior, and can thus influence the training process in an arbitrary or worst-case manner. These failures are called Byzantine. To make non-faulty clients agree on a reasonable output, validity conditions are used. In layman's terms, these conditions define where an agreement vector can lie based on the input distribution.

In this work, we investigate the coordinate-wise median aggregation rule for centralized and decentralized federated learning under Byzantine failures. The coordinate-wise median (CWM) can be efficiently computed and is scalable. Unlike the mean, it resists outliers and Byzantine failures and ensures robust aggregation [12]. In the theoretical part, we consider approximate instead of averaging agreement, that has a stronger assumption on the agreement output. We propose two algorithms that solve the multidimensional approximate agreement problem and satisfy validity conditions. In the centralized setting, we show that both, the Minimum Diameter (MD) algorithm and the Hyperbox algorithm satisfy the box validity condition and can tolerate up to $\frac{n}{3}$ and $\frac{n}{2}$ Byzantine failures, respectively. We then show that only the Hyperbox algorithm can be generalized in the decentralized setting. We evaluate the MD and the Hyperbox algorithms practically and compare them to MD and Hyperbox algorithms that use mean for aggregation. Our experiments consider homogeneous and heterogeneous data distributions with Byzantine attacks. In the decentralized federated learning setting, we show that MD CWM approach can tolerate sign flip attack better than the known mean-based approaches.

2 Related work

Federated learning was introduced by McMahan et al. [21, 23] for supervised learning, receiving a lot of attention in follow-up work [19, 20, 22]. The original work however did not tolerate malicious attacks. Malicious attacks have been investigated by the distributed computing community, where several Byzantine-tolerant

federated learning algorithms have been introduced [3, 6, 18, 27, 28]. El-Mhamdi et al. [9] propose a decentralized solution, where multiple clients replicate the server, appearing as a single trusted entity to the user. Their fully decentralized Byzantine-tolerant federated learning model [8] introduces two optimal averaging agreement algorithms: a minimum-diameter-based algorithm, optimal for correctness when most clients are non-faulty, and a trimmed mean algorithm, resilient to $< \frac{n}{3}$ Byzantine failures [15].

Jere et al. [17] provide a survey of practical malicious attacks considered in federated learning. They divide the attacks into model poisoning, comprising of label flipping and backdoor attacks [1]; and data poisoning attacks, including gradient manipulation [3, 11] and training rule manipulation [2]. In the practical part, our paper focuses on a data poisoning attack—the sign flip attack [8, 25].

Besides using mean as an aggregation function, many other aggregation rules have been considered in the literature [16]. Yin et al. [28] propose a distributed gradient descent algorithm using coordinate-wise median for aggregation. They present a simple and robust algorithm that aggregates local solutions using coordinate-wise median in one round, thus improving communication efficiency. To our best knowledge, this is the first work studying coordinate-wise median as an aggregation function in centralized federated learning.

Our decentralized federated learning algorithm makes use of multidimensional approximate agreement. The first algorithm proposed for multidimensional approximate agreement requires all clients to converge inside the convex hull of all honest input vectors. In [24], the authors show that convergence is possible if $t < n/(\max\{3, d+1\})$, where t is an upper bound on the number of Byzantine failures. The strong guarantee this algorithm gives on its output makes it attractive. However, the algorithm cannot be used in the presence of faulty clients when $n \leq d$ and is thus not applicable to our work. The Hyperbox algorithm presented in this work is an adaptation of the Hyperbox algorithm for the mean aggregation rule presented in [5].

3 Preliminaries

3.1 Distributed machine learning

We consider a system with n clients, each possessing an input vector $v_1, \ldots, v_n \in \mathbb{R}^d$. Each client i has access to its own dataset, sampled from an unknown distribution $\mathcal{D}_i$. We make the standard assumptions on local loss functions of honest clients [8]. We further assume that communication between clients is synchronous and reliable [4, 26]. We define t to be the maximum number of Byzantine vectors the system can tolerate, and f as the actual number of Byzantine vectors in the system. We call the $n-f$ non-faulty clients honest. Note that the honest clients do not know f.

Centralized federated learning model. In the centralized federated learning framework, a single server orchestrates the learning process. The dataset is distributed across clients, who retain their data locally. At the start of each round, each client's local model is initialized with the weights of the global model. Clients compute a stochastic estimate $g_\tau^{(i)}$ of the gradient $\nabla Q_i(\theta_\tau^{(i)})$ for all local models $\theta_\tau^{(i)}$ in iteration τ. The gradient estimate $g_\tau^{(i)}$ is computed by drawing a data point v or a sample from the local data generating distribution $\mathcal{D}_i$:

$$g_\tau^{(i)} = \nabla q(\theta_\tau^{(i)}, v) \qquad \text{with} \quad v \sim \mathcal{D}_i. \tag{1}$$

The gradient estimate $g_\tau^{(i)}$ equals the true gradient $\nabla Q_i(\theta_\tau^{(i)})$ in expectation. The central server then collects stochastic gradients g_τ from all clients and aggregates them into $\widehat{g_\tau}$. As a result, the global model's parameter θ_τ is updated to $\theta_{\tau+1}$ according to: $\theta_{\tau+1} = \theta_\tau - \gamma_\tau \cdot \widehat{g_\tau}$, where γ_τ is the learning rate. At the start of each new round, local models reset their weights to match those of the updated global model, and the process repeats. The total number of iterations, T, is predefined before training begins. After each round, the global model's performance is evaluated, and its accuracy is recorded.

Decentralized federated learning model. In the decentralized federated learning model, there is no central entity overseeing the process. Similar to the centralized approach, data is distributed among clients and remains local. Each client initializes a local model at the beginning, which is then stored and updated across iterations. Each client i computes a stochastic gradient $g_\tau^{(i)}$ of its local loss function's gradient $\nabla Q_i(\theta_\tau^{(i)})$ as in Equation (1) in the centralized collaborative learning model. However, in the decentralized model, clients broadcast their gradients $g_\tau^{(i)}$ to all other clients. Each client then gathers the received gradients and aggregates them using a predefined aggregation function.

Since there is no central entity maintaining a global model, clients may not agree on the same aggregated gradient, especially in the presence of communication faults. Solving agreement exactly would require at least $t+1$ rounds of communication [13], we thus only focus on (faster) approximate solutions. To ensure agreement close to the aggregated gradient of honest clients, we use agreement algorithms that operate in multiple sub-rounds. In each sub-round, clients transmit their vectors to all others and apply an aggregation rule to the collected vectors. The output from one sub-round serves as input for the next. The number of sub-rounds is predefined, usually set to $\log \tau$ [8], where τ represents the "big" iteration. In the final sub-round of iteration τ, clients update their models and proceed to iteration $\tau+1$, repeating the process until the stopping criteria are met.

Aggregation rules. In the following, we define mean and coordinate-wise median aggregation rules.

Definition 3.1 (Mean). The mean of a finite set of n vectors $v_i, i \in [n]$ is $\frac{1}{n}\sum_{i=1}^{n} v_i$.

Definition 3.2 (Coordinate-wise median (CMW)). For vectors $v_i \in \mathbb{R}^d$, where $i \in [n]$, the coordinate-wise median $CWM \in \mathbb{R}^d$ is defined as a vector, where in each coordinate a median on the input vectors is computed. If the number of values is even, the coordinate-wise median takes the midpoint of two central values.

In other words, coordinate-wise median minimizes the sum of Manhattan distances to all points. Additionally, we use CWM^* to denote the coordinate-wise median computed with only honest input vectors, and refer to it as the true coordinate-wise median.

3.2 Multidimensional approximate agreement

To be able to aggregate local gradients in the presence of faulty clients, we extend two algorithms from the literature presented

for mean aggregation: the Minimum Diameter Averaging (MDA) algorithm [8] and the Hyperbox algorithm [5]. In the following, we adapt the definitions used in the literature to coordinate-wise median aggregation.

3.2.1 Byzantine-tolerant aggregation. We start with the Minimum Diameter (MD) strategy that is based on the MDA algorithm [8]. Given a set of input vectors $S = v_1, \ldots, v_n$, this strategy first defines a subset containing $n - t$ clients of minimal diameter. Based on this subset, the coordinate-wise median is computed:

$$\mathrm{MD}(S, n-t) \in \arg\min_{S \subset [n], |S|=n-t} \max_{j,k \in S} ||v_j - v_k||_2$$

We refer to the algorithm in [5], which is based on picking a vector in the intersection of hyperboxes, as the **Hyperbox algorithm**. The computed output vector of each client is guaranteed to be inside a so-called trusted hyperbox, which is defined as follows.

Definition 3.3 (Trusted hyperbox). Let $f \leq t$ be the number of Byzantine clients and let v_i^*, $i \in [n-f]$ denote the true vectors. Let $v_i^*[k]$ denote the k^{th} coordinates of these vectors. The *trusted hyperbox* TH is the Cartesian product of $\mathrm{TH}[k] \coloneqq \left[\min_{i \in [n-f]} v_i^*[k], \max_{i \in [n-f]} v_i^*[k]\right]$, for all $k \in [d]$.

The trusted hyperbox cannot be computed locally, since neither the central server, nor the clients can identify Byzantine behavior. Therefore, the algorithm is based on computing local hyperboxes that are guaranteed to lie inside TH.

Definition 3.4 (Locally trusted hyperbox). Let $v_1, \ldots, v_{m_p}$ denote the vectors received by some participant p (server or client), where m_p denotes the number of messages p received. The number of Byzantine values for each coordinate is at most $m_p - (n-t)$. Denote $\phi_p : [m_p] \to [m_p]$ a bijection s.t. $v_{\phi_p(j_1)}[k] \leq v_{\phi_p(j_2)}[k], \forall j_1, j_2 \in [m_p]$. The locally trusted hyperbox of p is the Cartesian product of $\mathrm{TH}_p[k] \coloneqq \left[v_{\phi_p(m_p-(n-t)+1)}[k], v_{\phi_p(n-t)}[k]\right]$ for all $k \in [d]$.

In the decentralized setting, the aggregation algorithm additionally has to satisfy **multidimensional approximate agreement** [5]. That is, the algorithm must fulfill the following conditions: *ϵ-agreement*: every honest client decides on a vector, s.t. any two vectors are at most ϵ distance from each other; *termination*: all honest clients must terminate; *strong validity*: if all honest clients start with the same initial vector, then they must agree on that vector.

3.2.2 Validity conditions. Per definition [5, 24], multidimensional approximate agreement satisfies **strong validity** (all-same-validity). This can be replaced by **weak validity** [7], which requires the output to be identical to the input vector if only correct clients and no Byzantine parties are present in the system. An additional validity condition is **box validity**, which is stronger than the strong validity condition and requires agreement vector to be inside the trusted hyperbox TH of the input vectors.

4 Coordinate-wise median: algorithms and analysis

The coordinate-wise median (CWM) is efficient to compute, as a median is computed independently in each dimension. It is therefore more practical for large-scale systems and extensive learning processes. Unlike the mean, which is sensitive to outliers and Byzantine failures, the coordinate-wise median provides a more robust aggregation rule [12]. By selecting the median value in each coordinate, this method reduces the influence of extreme adversarial values, making it more robust than the mean.

Algorithm 1 Centralized MD algorithm

1: Aggregate up to n messages $M = \{v_j, j \in [n]\}$
2: Compute $\mathrm{MD}(M, n-t)$
3: **return** $CWM(\mathrm{MD}(M, n-t))$

Algorithm 2 Centralized Hyperbox algorithm

1: Aggregate up to n messages $M = \{v_j, j \in [n]\}$
2: **return** $Midpoint(TH \cap CWMBox)$

Firstly, we need to introduce local sets of possible coordinate-wise medians.

Definition 4.1 (Local set of possible coordinate-wise medians). Local set of possible medians of client i includes all coordinate-wise medians of all subsets of $n - t$ clients, denoted $S_{CWM}(i)$.

We next define the hyperbox around the set of possible coordinate-wise medians as follows:

Definition 4.2. The coordinate-wise median hyperbox $CWMBox$ is the smallest coordinate-parallel hyperbox containing S_{CWM} and the local coordinate-wise median hyperbox of client i denoted by $CWMBox(i)$ is the smallest hyperbox containing $S_{CWM}(i)$.

4.1 Centralized algorithms

We start by adapting the MDA algorithm [10] to the coordinate-wise median and present it in Algorithm 1. In each round, the central server finds the minimum diameter set that contains a subset of size $n-t$ of the received vectors, and computes the coordinate-wise median of this set.

Theorem 4.3. *Algorithm 1 satisfies box validity with $t < n/3$.*

Proof. We prove this statement for each coordinate $k \in d$. We need to show that $CWM(\mathrm{MD})[k]$ is in $TH[k]$. $CWM(\mathrm{MD})[k]$ is computed on the subset of at least $n - t$ vectors with the smallest diameter. Since $n - t \geq 2t + 1$, there are at least t smaller and t larger values than $CWM(\mathrm{MD})[k]$ in coordinate k, also when all input vectors are considered. Thus $CWM(\mathrm{MD})[k]$ must be in $TH[k]$. □

Next, we present a centralized algorithm based on hyperboxes, the pseudocode is given in Algorithm 2. In each round, the server computes the coordinate-wise median hyperbox and trusted hyperbox and outputs the midpoint of their intersection. The $Midpoint()$ function in Algorithm 2 on a multidimensional interval returns a one-dimensional midpoint calculated in each dimension of the interval.

Theorem 4.4. *Algorithm 2 satisfies box validity with $t < n/2$.*

Proof. We start by showing that $CWMBox \cap TH \neq \emptyset$. We will show this for a coordinate k, where $k \in [n]$. Consider the median of $TH[k]$, denoted $Median(TH[k])$. We argue that $Median(TH[k]) \in CWMBox$. Observe that the $CWMBox[k]$ contains a median that is

computed without the largest $|M|-(n-t)$ values, denoted med_l and a median computed without the smallest $|M|-(n-t)$ values, denoted med_r. Median med_l is computed with the values in $TH[k]$ and additionally $|M|-(n-t)$ smaller values, thus $med_l \leq Median(TH[k])$. Analogously, $Median(TH[k]) \leq med_r$. Since med_l and med_r are contained in $CWMBox[k]$, so is $Median(TH[k])$. Since the output is chosen inside TB, the algorithm satisfies box validity. □

For $t < n/3$, we show a stronger connection between $CWMBox$ and TH, namely:

Lemma 4.5. *If $t < n/3$, the coordinate-wise median hyperbox is inside the trusted hyperbox: $CWMBox \subseteq TH$.*

Proof. The proof of Theorem 4.3 shows that $CWM(\mathrm{MD})[k]$ is inside $TH[k]$. Note that we did not use the property of MD that the vectors inside MD have the smallest diameter among all subsets of $n-t$ vectors. The proof can be analogously derived for any subset of $n-t$ vectors. Thus, the coordinate-wise median of any subset of $n-t$ vectors is in TH. Finally, the hyperbox $CWMBox$ is in TH. □

4.2 Decentralized algorithms

In the decentralized setting, it is appealing to use the MD algorithm and let each client update its local vector to the median of the received vectors in each round. This strategy however does not work in the presence of Byzantine clients, unless signatures are used. There may be only two input vectors, and due to the uncertainty in communication, the clients may repeatedly choose either of the vectors as their new median.

Consider the following one dimensional case and round r of the MD algorithm. Suppose there are t Byzantine clients in the system, $\frac{n-t}{2}$ true clients start round r with vector v_1, and the other $\frac{n-t}{2}$ true clients with vector v_2. Then, $\frac{t}{2}$ Byzantine clients could send vector v_1 to half of the true clients and $\frac{t}{2}$ Byzantine clients could send vector v_2 to the other half of true clients. A true client would hence receive exactly $\frac{n-t}{2}+\frac{t}{2}$ vectors, and all of them would be contained in MD. For half of the true clients, MD would contain $\frac{n-t}{4}+\frac{t}{2}$ times v_1 i.e. more than half of the vectors in MD would be equal to v_1, and for the other half of true clients, more than half of the vectors in their set MD would be equal to v_2. Since a true client's new vector is the median of vectors in its MD set, the system would then start round $r+1$ in exactly the same configuration as round r, i.e. the algorithm doesn't converge. Despite its theoretical limitations, MD performs competitively under certain attack scenarios, which is shown in Section 5. For further theoretical analysis, we therefore only focus on the generalization of the centralized Hyperbox algorithm.

Lemma 4.6. *True coordinate-wise median CWM^* is inside the local hyperbox of all possible medians $CWMBox_i$, for all honest clients i.*

Proof. Since we assume synchronous communication, by definition, all honest vectors will be received by all honest clients in every round. $CWMBox_i$ contains all medians computed on subsets of $n-t$ received vectors. One of these vectors must be CWM^*. □

We can now generalize Algorithm 2 to the distributed setting. Note that this algorithm is similar to the Hyperbox algorithm that was introduced to approximate the centroid [5]. Algorithm 3 presents this algorithm in pseudocode.

Algorithm 3 Distributed Hyperbox algorithm for $t < n/3$

```
for each round r = 1, 2, ... do
    for each client i with input v_i: do
        broadcast v_i reliably to all clients
        receive up to n messages M_i = {v_j, j ∈ [n]}
        compute S_CWM(i) from all v_j ∈ M_i
        compute TH_i from M_i by excluding |M_i| − (n−t) values on each side
        compute CWMBox_i on subsets of (n − t) clients
        v_i = Midpoint(CWMBox_i)
    end for
end for
```

Theorem 4.7. *Algorithm 3 converges and satisfies box validity with resilience $t < n/3$.*

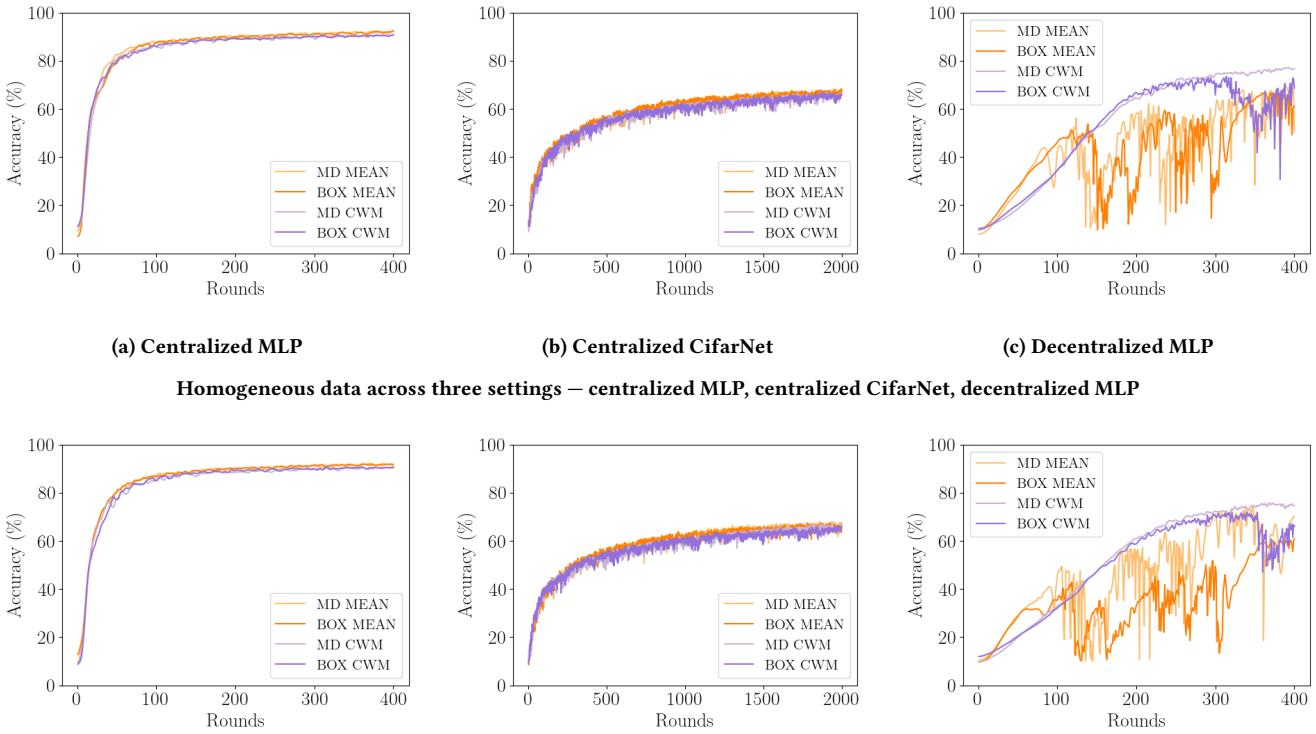
The bound $t < n/3$ is due to the impossibility result for Byzantine agreement [14]. In order to prove that the Algorithm 3 satisfies box validity, we will first note that the local coordinate-wise median hyperbox is inside the local trusted hyperbox in every round of the algorithm. This statement follows from Lemma 4.5. Observe further that the trusted hyperbox of every following round is contained in the trusted hyperbox of the previous round, and thus also in the trusted hyperbox of the input vectors, denoted TH_i^0.

Corollary 4.8. *For each correct client i, the local coordinate-wise median hyperbox is inside the local trusted hyperbox: $CWMBox_i \subseteq TH_i \subseteq \mathrm{TH}^0$.*

Proof of Theorem 4.7. We show convergence for each coordinate $k \in [n]$. Let $H^r[k]$ denote the set of honest values and $D^r[k]$ denote the interval between the largest and the smallest honest value in coordinate k at the beginning of round r. Our goal is to show that $|D^{r+1}[k]| < |D^r[k]/2|$, where $|D^r[k]|$ denotes the length of interval $D^r[k]$. Due to synchronous communication, all honest clients will receive all honest values. By using reliable broadcast, we also ensure that every honest client either accepts the same value from a faulty client or has no value at all.

Let $CWMBox^r$ denote the coordinate-wise median hyperbox computed in round r of the algorithm. From Lemma 4.6 we know that the true coordinate-wise median of the inputs of round r, denoted CMW^r, is inside $CWMBox_i^r$. Therefore, $\bigcap_{i\in H_r[k]} CWMBox_i^r \neq \emptyset$. Observe that $CWMBox^r[k] \subseteq D^r[k]$, as each median is inside the trusted hyperbox of all honest vectors. Let d_i^r denote the maximum distance between $CMW^r[k]$ and one of the endpoints of the interval $CWMBox_i^r[k]$ in coordinate k. Since each client chooses the midpoint of $CWMBox_i[k]$ as its new input for the next round, the distance between CMW^r and $v_i^{r+1}[k]$ is at most half of d_i^r. This holds for each client i. Furthermore, there must have been true input values on each side outside the open interval $CWMBox_i^r[k]$ for each i. Thus, a reduction of the interval of all honest values by $1/2$ w.r.t. $CWMBox^r[k]$ by a factor of $1/2$ also means a reduction of $D^r[k]$ by at least $1/2$. □

Observe that Algorithm 3 does not have a stopping condition. With the result in Theorem 4.7, we can introduce a stopping condition after $O(\log(D/\varepsilon))$ communication rounds, where D denotes the maximum coordinate-wise diameter of the honest input vectors.



(a) Centralized MLP **(b) Centralized CifarNet** **(c) Decentralized MLP**

Homogeneous data across three settings — centralized MLP, centralized CifarNet, decentralized MLP

(d) Centralized MLP **(e) Centralized CifarNet** **(f) Decentralized MLP**

Mildly heterogeneous data across the same three settings

Figure 1: Performance comparisons of centralized and decentralized federated learning

Lemma 4.9. *Algorithm 3 with the stopping condition satisfies approximate agreement.*

Proof. After $O(\log(D/\varepsilon))$ rounds, the maximum diameter of the honest input vectors in each coordinate is ε. Additionally, Algorithm 3 satisfies the strong validity condition, as the agreement is located within the trusted hyperbox. As a result, it successfully solves the multidimensional approximate agreement problem. □

5 Experimental Results

In order to understand how the convergence of algorithms influences the convergence of the machine learning model, we empirically evaluate our algorithms. Although decentralized MD coordinate-wise median algorithm lacks theoretical convergence guarantees, we include it in our experiments to assess its performance.

Centralized and decentralized federated learning models are evaluated on the MNIST and CIFAR10 dataset. We study uniform and heterogeneous data distributions to reflect a range of federated learning scenarios, from idealized (uniform) to more realistic and challenging (heterogeneous). The first heterogeneous scenario studies mild heterogeneity, where each class from the train dataset is split into 10 parts, where 8 parts contain 10% of the class, one part 5% and one part 15% of the class. The second scenario studies extreme heterogeneity, also known as 2-class heterogeneity [29]. The dataset is sorted and split into 20 pieces. Each client gets randomly 2 parts of the data, so that each client holds up to 2 classes of data.

The underlying neural network for solving the image classification task on MNIST dataset is a MultiLayer Perceptron (MLP) with 3 layers. For the CIFAR10 dataset we implemented CifarNet, a medium-sized convolutional network with thousands of trainable parameters and the ability to capture spatial relationships in colored images. To study a standard Byzantine scenario, we set the number of clients to $n = 10$ and number of Byzantine clients to $f = 1$. We consider the sign flip attack [25]. Instead of sending their true gradients, f Byzantine clients invert their sign before sharing.

How is the performance in centralized MLP and CifarNet architectures? Figure 1 illustrates the performance of MD and Hyperbox algorithms with mean and coordinate-wise median as aggregation rules. Figures 1a and 1d show all methods converge in homogeneous and mild heterogeneous case under MNIST dataset, and achieve 91% and 90% accuracy, respectively. When the same setting is evaluated on CIFAR10 dataset and CifarNet architecture, the accuracy of all methods drops, as shown in Figures 1b and 1e. Mean is showing better accuracy (67%) over coordinate-wise median methods (65%). The overall accuracy achieved by the CifarNet is lower than that of the MLP architecture, which is explained by the increased architectural complexity of CifarNet and the greater visual and structural complexity of the colored CIFAR dataset.

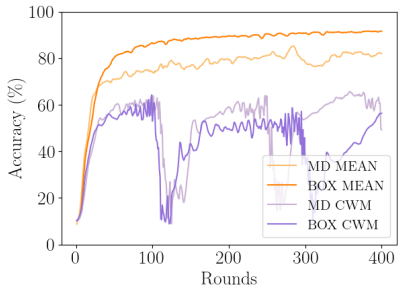


Figure 2: Centralized federated learning on MLP architecture under extreme heterogeneous data

How is the performance in decentralized MLP architecture? Figures 1c and 1f illustrate decentralized federated learning model with homogeneous and heterogeneous data distribution on MLP architecture. Here, MD mean and Box mean fail to produce a model under the sign flip attack. Box algorithm with coordinate-wise median is not stable and does not seem to converge. However, MD with coordinate-wise median converges with accuracy 77% for homogeneous and 74% for mild heterogeneous case. This result indicates that the convergence of the agreement algorithm does not influence the convergence of the machine learning model. As shown in Section 4.2, MD approach for coordinate-wise median does not converge. However, this did not prevent convergence of the machine learning model using MD coordinate-wise median as an aggregation rule. This example also highlights the advantage of coordinate-wise median based approach over mean.

How is the convergence behavior under extreme heterogeneity? An additional experiment in Figure 2 shows extreme heterogeneous scenario in a centralized setting. Coordinate-wise median approaches fail to converge, since the data is extremely heterogeneous and each client has up to two classes of data. Hyperbox algorithm with mean achieves around 90% accuracy, while MD mean reaches 80% and is more unstable than the Hyperbox approach. Under extremely heterogeneous data distribution in a decentralized setting, all aggregation rules fail, suggesting that a different approach should be considered in this case.

References

[1] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. 2020. How To Backdoor Federated Learning. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 108)*. PMLR, 2938–2948.

[2] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. 2019. Analyzing Federated Learning through an Adversarial Lens. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*. PMLR, 634–643.

[3] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. 2017. Machine learning with adversaries: Byzantine tolerant gradient descent. *Advances in neural information processing systems* 30 (2017).

[4] Gabriel Bracha. 1987. Asynchronous Byzantine agreement protocols. *Information and Computation* 75, 2 (1987), 130–143. doi:10.1016/0890-5401(87)90054-X

[5] Melanie Cambus and Darya Melnyk. 2023. Improved Solutions for Multidimensional Approximate Agreement via Centroid Computation. arXiv:2306.12741 [cs.DC]

[6] Yudong Chen, Lili Su, and Jiaming Xu. 2017. Distributed Statistical Machine Learning in Adversarial Settings: Byzantine Gradient Descent. *Proc. ACM Meas. Anal. Comput. Syst.* (2017).

[7] Pierre Civit, Seth Gilbert, Rachid Guerraoui, Jovan Komatovic, and Manuel Vidigueira. 2023. On the Validity of Consensus. In *Proceedings of the 2023 ACM Symposium on Principles of Distributed Computing* (Orlando, FL, USA) *(PODC '23)*. 332–343.

[8] El-Mahdi El-Mhamdi, Sadegh Farhadkhani, Rachid Guerraoui, Arsany Guirguis, Lê-Nguyên Hoang, and Sébastien Rouault. 2021. Collaborative learning in the jungle (decentralized, byzantine, heterogeneous, asynchronous and nonconvex learning) *(NIPS '21)*.

[9] El-Mahdi El-Mhamdi, Rachid Guerraoui, Arsany Guirguis, Lê Nguyên Hoang, and Sébastien Rouault. 2020. Genuinely Distributed Byzantine Machine Learning. In *Proceedings of the 39th Symposium on Principles of Distributed Computing* (Virtual Event, Italy) *(PODC '20)*.

[10] El Mahdi El Mhamdi, Rachid Guerraoui, and Sébastien Rouault. 2018. The Hidden Vulnerability of Distributed Learning in Byzantium. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*. PMLR, 3521–3530.

[11] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. 2020. Local Model Poisoning Attacks to Byzantine-Robust Federated Learning. In *29th USENIX Security Symposium (USENIX Security 20)*. USENIX Association.

[12] P Filzmoser, S Serneels, R Maronna, PJ Van Espen, B Walczak, RT Ferre, and S Brown. 2009. Comprehensive Chemometrics.

[13] Michael J. Fischer and Nancy A. Lynch. 1982. A Lower Bound for the Time to Assure Interactive Consistency. *Information Processing Letters* 14, 4 (1982).

[14] Michael J. Fischer, Nancy A. Lynch, and Michael Merritt. 1990. Easy impossibility proofs for distributed consensus problems. In *Fault-Tolerant Distributed Computing*, Barbara Simons and Alfred Spector (Eds.). Springer New York, New York, NY, 147–170.

[15] Rachid Guerraoui, Arsany Guirguis, Jérémy Plassmann, Anton Ragot, and Sébastien Rouault. 2021. GARFIELD: System Support for Byzantine Machine Learning (Regular Paper). In *2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*.

[16] Rachid Guerraoui, Nirupam Gupta, and Rafael Pinot. 2024. Byzantine Machine Learning: A Primer. *ACM Comput. Surv.* 56, 7, Article 169 (April 2024), 39 pages.

[17] Malhar S. Jere, Tyler Farnan, and Farinaz Koushanfar. 2021. A Taxonomy of Attacks on Federated Learning. *IEEE Security & Privacy* 19, 2 (2021), 20–28.

[18] Sai Praneeth Karimireddy, Lie He, and Martin Jaggi. 2021. Learning from History for Byzantine Robust Optimization. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 139)*. PMLR, 5311–5319.

[19] Jakub Konečný, Brendan McMahan, and Daniel Ramage. 2015. Federated Optimization:Distributed Optimization Beyond the Datacenter. arXiv:1511.03575

[20] Jakub Konečný, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. 2017. Federated Learning: Strategies for Improving Communication Efficiency. arXiv:1610.05492

[21] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 54)*, Aarti Singh and Jerry Zhu (Eds.). PMLR, 1273–1282.

[22] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 54)*. PMLR, 1273–1282.

[23] H Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas. 2016. Federated learning of deep networks using model averaging. *arXiv preprint arXiv:1602.05629* (2016).

[24] Hammurabi Mendes, Maurice Herlihy, Nitin Vaidya, and Vijay K. Garg. 2015. Multidimensional agreement in Byzantine systems. *Distrib. Comput.* 28, 6 (dec 2015), 423–441. doi:10.1007/s00446-014-0240-5

[25] Chanho Park and Namyoon Lee. 2024. SignSGD with Federated Defense: Harnessing Adversarial Attacks through Gradient Sign Decoding. In *Proceedings of the 41st International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 235)*. PMLR, 39762–39780.

[26] T.K. Srikanth and Sam Toueg. 1987. Simulating Authenticated Broadcasts to Derive Simple Fault-Tolerant Algorithms. *Distributed Computing* 2, 2 (June 1987).

[27] Lili Su and Nitin H. Vaidya. 2016. Non-Bayesian Learning in the Presence of Byzantine Agents. In *Distributed Computing*, Cyril Gavoille and David Ilcinkas (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 414–427.

[28] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. 2018. Byzantine-Robust Distributed Learning: Towards Optimal Statistical Rates. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*. PMLR, 5650–5659.

[29] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. 2018. Federated Learning with Non-IID Data. (2018).