Secure Byzantine-Robust Distributed Learning via Clustering

Raj Kiriti Velicheti
Coordinated Sciences Laboratory
University of Illinois at Urbana-Champaign
rkv4@illinois.edu

Derek Xia
Department of Computer Science
University of Illinois at Urbana-Champaign
derekx3@illinois.edu

Oluwasanmi Koyejo
Department of Computer Science
University of Illinois at Urbana-Champaign
sanmi@illinois.edu

Abstract

Federated learning systems that jointly preserve Byzantine robustness and privacy have remained an open problem. Robust aggregation, the standard defense for Byzantine attacks, generally requires server access to individual updates or nonlinear computation – thus is incompatible with privacy-preserving methods such as secure aggregation via multiparty computation. To this end, we propose SHARE (Secure Hierarchical Robust Aggregation), a distributed learning framework designed to cryptographically preserve client update privacy and robustness to Byzantine adversaries simultaneously. The key idea is to incorporate secure averaging among randomly clustered clients before filtering malicious updates through robust aggregation. Experiments show that SHARE has similar robustness guarantees as existing techniques while enhancing privacy.

1 Introduction

An increasing amount of data is being collected in a decentralized manner on devices across institutions[9]. Traditionally, machine learning with such devices require centralized data collection, which increases communication costs while posing a threat to privacy, especially when these devices gather personal user data. Distributed learning frameworks like federated learning attempt to address these issues by sharing model updates from client devices, rather than data, to a centralized server [11, 9, 7, 14].

Among the most popular implementations of federated learning is Federated Averaging [15]. While the central coordinating server follows a designated aggregation protocol, the required communication can pose a privacy threat when the system is compromised by a malicious external agent leaking individual model updates. To this end, Bonawitz et al. [3] proposed a secure averaging oracle that masks individual client updates such that the server learns their average alone. Nevertheless, since the collaboratively learned model update includes the contribution of all participating clients, benign averaging might fall prey to incorrect device updates either due to arbitrary failures or maliciously crafted updates preventing the devices from learning a good model.

In recent years, federated learning robustness to Byzantine failures (i.e., worst-case adversarial coordinated training-time attacks) has gained attention. However, existing robust aggregation techniques require sophisticated nonlinear operations [21, 24, 2], sometimes with server access to individual model updates in the clear – thus leading to privacy loss. These nonlinear operations adversely affect privacy since privacy-preserving methods such as secure Multi-Party Computation (MPC) are inefficient for nonlinear operations [3]. This observation highlights a fundamental tension between existing solutions to the two critical problems of privacy and robustness.

1st NeurIPS Workshop on New Frontiers in Federated Learning (NFFL 2021), Virtual Meeting.

To the best of our knowledge, ours is the first approach that scalably combines Byzantine-robustness with privacy using the common single-server architecture. We prose a novel hierarchical framework that decouples MPC-based privacy and Byzantine robustness protection mechanisms in this work. The basic idea is to implement a secure averaging oracle among randomly clustered clients, then filtering these updates using robust aggregation. This approach reveals only the cluster averaged update to the server, thus can help preserve privacy. Simultaneously, the second level of robust aggregation helps to maintain Byzantine robustness.

Taken together, this manuscript proposes a federated learning architecture that preserves security and privacy jointly, thus addressing this gap in the literature. Due to the hierarchical approach, existing robust distributed learning frameworks [21, 24, 2] and non-robust secure distributed learning frameworks like [3, 4] can be considered special cases of our proposed approach. **Summary of contributions:** We propose SHARE; a robust distributed learning framework which flexibly incorporates any Byzantine-robust defenses while enhancing privacy in a single server systems setting. We extend existing theoretical guarantees of robust aggregation oracles to the SHARE framework. Further, we present empirical evaluation of SHARE on benchmark datasets.

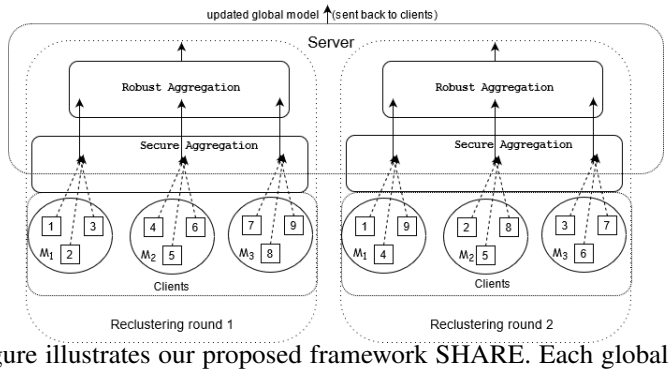


Figure 1: This figure illustrates our proposed framework SHARE. Each global round consists of multiple reclustering rounds, updates from which are averaged to obtain the final model update. In each reclustering round (shown by dotted rectangle), updates from clients (numbered squares) are clustered randomly ($\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3$), then averaged followed by robust aggregation.

2 Related Work

Byzantine robustness and secure aggregation in distributed learning both have a large existing literature – though often from different (somewhat disconnected) communities.

Robust Aggregation. Robustness to Byzantine adversaries is a well-studied problem in distributed and federated learning [13]. Broadly, existing defenses can be categorized into distance-based robust aggregation or validation data-based aggregation. The general idea behind distance-based metrics is to find an update closer to the benign mean in l_2 norm distance. Xie et al. [21] suggest utilizing coordinate-wise trimmed mean. Blanchard et al. [2] choose a model update closest to most other updates.Ghosh et al. [6] suggest optimal statistical rates utilizing median and trimmed mean. All these distance-based defenses require a majority of the clients participating in the protocol to be benign. On the other hand, validation data-based aggregation defenses such as Zeno [22, 24] perform suspicion-based aggregation based on a score evaluated on validation data held at the server. These methods can tolerate arbitrary Byzantine poisoning. All the above techniques require the server to see the local model updates in the clear, posing a privacy threat.

Privacy via Secure Aggregation. In many distributed learning settings, the secure computation boils down to computing a secure average. Bonawitz et al. [3] utilizes a pairwise secret share to achieve the same. Aono et al. [1] follow a slightly different approach and utilize additively homomorphic encryption for secure update computation. While these methods work well with linear aggregation methods, extending secure multiparty computation to non-linear robust aggregation schemes introduces additional computational and communication overhead, quickly becoming impractical for real computational loads.

He et al. [8], Pillutla et al. [16], Wang et al. [20] are the closest to our work in the sense that they are proposed to address the problem of robustness and privacy in distributed learning jointly. Compared to He et al. [8], which requires two non-colluding servers, we achieve this with a single server, which may be a more realistic architecture for practical use cases. Further, He et al. [8] tailor their approach to distance-based robust aggregation. In contrast, our proposed approach is easily combined with most existing Byzantine-robust aggregation schemes, including filtering-based defenses such as Zeno++ [24]. On the other hand, Pillutla et al. [16] reduce the filtering computation of the median into a sequence of linear computations. Unfortunately, this approach requires that the Byzantine device follows the computational protocol over multiple rounds, which is a strong assumption in practice. Thus, while inspired by Byzantine tolerance, Pillutla et al. [16] do not claim Byzantine robustness. After completing this work, we were made aware of a related approach [20] using a hierarchical architecture with a robust mean aggregator. Compared to Wang et al. [20], our approach is a wrapper method that can be combined with any robust aggregator, with analysis and performance depending on the choice of aggregator (e.g., we analyze and compare trimmed mean, krum, Zeno++). Further, we address the problem of signal loss due to clustering via a novel reclustering step.

3 Problem Formulation

We consider the optimization problem $\min_{x\in\mathbb{R}^d} F(x)$ where, $F(x) = \frac{1}{n}\sum_{i\in[n]} \mathbb{E}_{z_i\sim\mathcal{D}_i} f_i(x; z_i)$, hence the goal is to learn a model x which performs well on average using z_i sampled from local data distribution $\mathcal{D}_i, \forall i \in [n]$. The notations used in this paper are summarized in Table 1 (Appendix A).

This problem is solved in a distributed and iterative manner. In each global iteration ($t < T$), sampled clients compute a private model update (Δx_i^K) by running multiple steps (K-steps) of Stochastic Gradient Descent (SGD) on the local data available ($z_i \sim \mathcal{D}_i$). Then server can computes a global model update. For instance, when using simple averaging, the server update is $x^t = x^{t-1} + \eta \sum_{i\in[n]} \Delta x_i^K$, where η is global learning rate. We consider the following privacy and security threats:

- *Privacy threat model*: We consider an honest but curious server. This specification allows the server to interpret the device data from the updates, hence breaching privacy. The assumption of honest server implies that the server still follows the underlying protocol.
- *Robustness threat model*: We consider a fixed (unknown) subset (q) of machines that can co-ordinate and send arbitrary updates to the server hence deviating from the intended distributed learning protocol.

4 Methodology

We propose two-step hierarchical aggregation SHARE (Secure Hierarchical Robust Aggregation) as a defense against the specified robustness and privacy threat models. In particular, our approach allows a decoupling of the security and robustness into two steps (as illustrated in Figure 1). First, in every global epoch, all participating clients are clustered randomly into groups. Clients within each group share pairwise secret keys and utilize them to mask their individual updates such that the server only learns the average within the cluster. This ensures client update privacy. These client cluster updates are then filtered using Byzantine-robust aggregation techniques. Further, we can repeat this process multiple times in a global epoch to aid in reducing variance. The detailed algorithm is outlined in Algorithm 1. Without loss of generality, we assume clusters of uniform size.

4.1 System Components

Secure Aggregation: This is the first step in hierarchical aggregation. We follow an approach similar to [3], using pairwise keys between clients in a cluster. The server in this setup learns just the mean and hence the privacy of individual client updates are protected (Detailed discussion in Appendix C).

Robust Aggregation: This is the second step in every reclustering round. In this step, the secure cluster averages are filtered through robust aggregation. The goal ideally is to eliminate clusters with malicious client updates. Any existing robustness techniques like trimmed mean[21], median[16] or Zeno[24] can be utilized at this stage. We show theoretical guarantees and experiments based on existing methods in the following sections.

Random Reclustering: As specified in Algorithm 1, we repeat the secure aggregation followed by robust aggregation multiple times randomizing client clusters in each global epoch. Note that across

Algorithm 1 SHARE (Secure Hierarchical Robust Aggregation)

Server:
for $t = 0, \ldots, T-1$ **do**
  for $r = 1, \ldots, R$ **do**
    Assign clients to clusters $\mathcal{S} = \mathcal{M}_1 \cup \ldots \mathcal{M}_i \ldots \cup \mathcal{M}_c$ with $|\mathcal{M}_i| = |\mathcal{M}_j| \forall i, j \in [c]$
    Compute secure average $g_j^r \leftarrow \texttt{SecureAggr}(\{\Delta_i\}_{i \in \mathcal{M}_j}) = \sum_{i \in \mathcal{M}_j} u_i, \forall j \in [c]$
    $g^r \leftarrow \texttt{RobustAggr}(\{g_j^r\}_{j \in [c]})$
  end for
  if stopping criteria met **then**
    break
  end if
  Push $x^t = x^{t-1} + \eta \frac{1}{R} \sum_r g^r$ to the clients
end for
Client:
for each client $i \in \mathcal{S}$ (if honest) **in parallel do**
  $x_{i,0}^t \leftarrow x^t$
  for $k = 0, \ldots, K-1$ **do**
    Compute an unbiased estimate $g_{i,k}^t$ of $\nabla f_i(x_{i,k}^t)$
    $x_{i,k+1}^t \leftarrow \texttt{ClientOptimize}(x_{i,k}^t, g_{i,k}^t, \eta_l, k)$
  end for
  $\Delta_i = \frac{n_i}{n}(x_{i,K}^t - x^t)$
  Push Δ_i to the assigned clusters using secure aggregation
end for
return x^T

these reclustering rounds, the same local model update is paired with different clients each time. In addition to malicious updates, benign updates paired with malicious clients might be filtered in the proposed approach. Reclustering helps mitigate this loss of signal and hence reduces variance. In particular, as number of reclustering rounds (R) increase, the probability of this loss in signal decreases (Detailed discussion in Appendix E).

Remark. Although reclustering increases communication cost, we note that in addition to helping decrease the variance, reducing secure aggregation to within clusters, decreases communication cost as pairwise key exchange is now limited to within the cluster. Hence overall, communication cost for each client changes from $\mathcal{O}(n)$ to $\mathcal{O}(\frac{Rn}{m})$. In experiments, we often find that even a single clustering round gives good results (Section 6).

5 Theory

5.1 Exactness

Algorithm 1 can be implemented using any aggregation technique. However, due to clustering, the result is resilient to fewer malicious clients – as (in the worst case) malicious clients are assumed to completely corrupt their assigned cluster. We formalize these ideas next, with proofs in Appendix B.

Lemma 1. *If robust aggregation is replaced by averaging, the output of Algorithm 1 is identical to Federated Averaging[15].*

Lemma 2. *In presence of robust aggregation, Algorithm 1 is robust to $q = \frac{q_0}{m}$ adversaries, where q_0 is the tolerance limit of the robust aggregation oracle followed and m is the cluster size.*

5.2 Convergence Analysis

To highlight the flexibility of the proposed algorithm, we analyze convergence when using both using a distance based robust aggregation strategy or a validation data based aggregation strategy, such as Zeno++. We first define the few terms used to develop convergence analysis.

Definition 5.1 ((G,B)-Bounded Gradient Dissimilarity)**.** There exists constants $G \geq 0, B \geq 1$ such that $\frac{1}{n} \sum_i^n \|\nabla f_i(x)\|^2 \leq G^2 + B^2 \|\nabla F(x)\|^2$

Definition 5.2 (Bounded client updates variance)**.** We define benign mean model update across clients to be $\mu = \sum_i \Delta_i^K$, hence the variance across client updates as $\mathbb{E}[\|\Delta_i^K - \mu\|^2] \leq \sigma_g^2$ for all i across all rounds of training

Definition 5.3 (Bounded variance)**.** For an unbiased stochastic gradient estimator with $g_i(x) = \nabla f_i(x, z_i)$ we define bounded variance as $\mathbb{E}_{z_i}[\|g_i(x) - \nabla f_i(x)\|] \leq \sigma^2$ for any i, x

The difference between Definition 5.2 and 5.3 is that the former bounds the variance between model updates across clients while the latter bounds the variance across gradient estimates within the same client.

5.2.1 Convergence Rates

We now prove that Algorithm 1 converges for various robust aggregation oracles. Firstly, we state a few general assumptions required to prove convergence guarantees standard in papers.

Assumption 5.1. There exists at least one global minima x^* such that $F(x^*) \leq F(x), \forall x$

Assumption 5.2. We assume that $F(x)$ is L-smooth and has μ-lower bounded Taylor approximation (μ weak convexity)

$$\langle \nabla F(x), y - x \rangle + \frac{\mu}{2}\|y - x\|^2 \leq F(y) - F(x) \leq \langle \nabla F(x), y - x \rangle + \frac{L}{2}\|y - x\|^2$$

Note that this Assumption 5.2 covers the case of non-convexity by taking $\mu < 0$. We note that each distance based robust aggregation metric have different bounds from benign mean update. Since the focus of this work is to propose an algorithm that unifies robustness with privacy, we do not concentrate on those bounds and absorb such intricacies into an order constant. Formally,

Assumption 5.3. For any distance based robust aggregation algorithm, *when fraction of faulty inputs is below threshold*, the output of robust aggregation is bounded from benign mean. That is, we assume there exists a V_2 such that for any set of vectors $\{v_i : i \in \mathcal{C}\}$, replaced by faulty vectors $\forall i \notin \mathcal{C}_t \subseteq \mathcal{C}$, $\|\texttt{RobustAggr}(\{v_i\}_{i\in\mathcal{C}}) - \frac{1}{|\mathcal{C}_t|}\sum_{i\in\mathcal{C}_t} v_i\| \leq \mathcal{O}(V_2)$.

We note that Assumptions 5.1,5.2 are standard among existing Federated Learning literature [10, 22, 24]. Additionally Assumption 5.3 is a direct consequence of existing distance based robust aggregation oracles [21, 16, 2]. Finally, for Algorithm 1 with such oracles, we have the following theorem

Theorem 3. *Consider a function F(x) satisfying Assumptions 5.1,5.2 assume a robust aggregation scheme that picks up b updates and satisfies Assumption 5.3, further, assume (G,B)-Bounded gradient dissimilarity, σ_g^2 variance in client updates and σ^2 variance in gradient estimation, there exists η, η_l such that output of Algorithm 1 after T rounds, x^T, satisfies,*

$$\mathbb{E}\left[\|\nabla F(x^T)\|^2\right] \leq \mathcal{O}\left(\frac{LM\sqrt{F}}{\sqrt{TKn}} + \frac{F^{2/3}(LG^2)^{1/3}}{(T+1)^{2/3}} + \frac{B^2LF}{T} + 2L^2V_2 + \frac{\sigma_g^2}{bm}\left(\frac{n-q-bm}{R(n-q)-1}\right)\right)$$

where $M^2 := \sigma^2(1 + \frac{n}{\eta^2})$ and $F := F(x^0) - F(x^*)$

Now we consider Zeno++[24], a defense utilizing server data. Although score based Zeno++ was originally introduced for asynchronous SGD, we generalize it to federated learning setting hence allowing for multiple local epochs. We illustrate this modified algorithm in Appendix B. As in Xie et al. [24], we consider an additional standard assumption

Assumption 5.4. The validation set considered for Zeno++ is close to training set, implying a bounded variance given by $\mathbb{E}[\|\nabla f_s(x) - \nabla F(x)\|^2] \leq V_1, \forall x$

Theorem 4. *Consider L-smooth and potentially non-convex functions $F(x)$ and $f_s(x)$, satisfying Assumption 5.4. Assume $\|f_s(x)\|^2 \leq V_3, \forall x$. Further assuming G-bounded gradient dissimilarity, variance between client updates be σ_g^2 and variance in gradient estimation at each client be σ, with global and local learning rates of $\eta \leq \frac{1}{2L}$ and $\rho \geq \frac{\alpha\sqrt{\eta}}{6K^2\eta_l^2B^2} + \eta$, after T global updates, let $D := F(x^0) - F(x^*)$, Algorithm 1 with Zeno++ as robust aggregation converges at a critical point:*

$$\frac{\mathbb{E}[\sum_{t\in[T]}\|\nabla F(x_{t-1})\|^2]}{T} \leq \frac{\mathbb{E}[D]}{\alpha\sqrt{\eta}T} + \frac{\sqrt{\eta}}{\alpha}\mathcal{O}\left(\frac{\sigma_g^2}{m}\left(\frac{n-q-m}{R(n-q)-1}\right) + G^2 + \sigma^2 + V_1 + V_3 + \epsilon\right)$$

Remark. It can be seen from both Theorem 3,4 that the additional terms, other than standard ones appearing in the convergence rate for federated learning [10], depend on the error caused by the robust aggregation scheme utilized and variance reduction from reclustering. Further, higher number of reclustering rounds R decreases the effect of additional variance. Finally when $R = 1, m = 1, q = 0$, these recover existing results for federated learning with robust aggregation.

5.3 Privacy

Curious server: Since each client masks updates with random vectors as illustrated in Section 4, we note that if we execute the mentioned secure averaging oracle with threshold $t > \frac{m}{2}$, the protocol can deal with $\lceil \frac{m}{2} \rceil - 1$ drop outs while learning nothing more than average. Reclustering introduces additional vulnerability as server can see multiple averages. In particular, the probability that server can decode a model update is $\mathcal{O}(1 - \left(\frac{(m!)^c}{n!}\right)^R)$. Hence as R increases this gets closer to 1 as expected. Further, when all clients are in a single cluster ($m = n$, hence c=1), this is 0 as would be the case with secure averaging without robustness. Further discussion, including comments on privacy in presence of colluding curious clients can be found in Appendix C.

6 Experiments

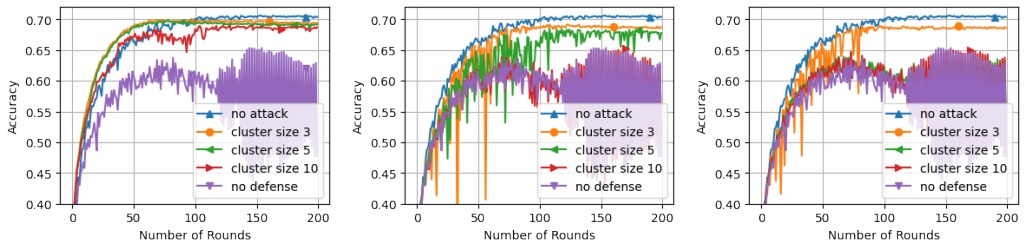
In this section we evaluate the proposed algorithm SHARE with various defenses and corruption models. We conduct experiments on CIFAR-10 [12] (Image classification dataset) and Shakespear (a language modeling dataset from LEAF [5]). We note that we do not propose a new robustness technique but rather we propose a modified federated learning architecture to incorporate any robustness protocol in a privacy preserving manner. Hence we focus our experiments on capturing the effects of cluster sizes and reclustering rounds, hyperparameters introduced by our approach. We defer descriptions of detailed training architecture to Appendix D

6.1 CIFAR-10

We train a CNN with two 5×5 convolutional layers followed by 2 fully connected layers[15] on CIFAR-10 and report top-1 accuracy. We test SHARE incorporating various robust aggregation protocols such as Trimmed mean [21], Krum [2], Zeno++ [24]. For all experiments in this section, trimmed mean removes $2/3$ of the updates before computing the mean. Additionally, we consider two baselines, SHARE with no robust aggregation and SHARE with no attack. We consider homogeneous distribution of data across clients for experiments in this section. Experiments on heterogeneous data distributions can be found in Appendix D.

6.1.1 Impact of cluster size

We first test Byzantine-tolerance for various cluster sizes to mild attacks such as label-flip. In particular, malicious clients train on wrong labels (images whose labels are flipped, i.e., any label $\in \{0, \ldots, 9\}$ is changed to 9-label). We consider 60 total clients of which $q = 12$ being malicious. The result is shown in Figure 2 for various cluster sizes and robust aggregation protocols. It is seen



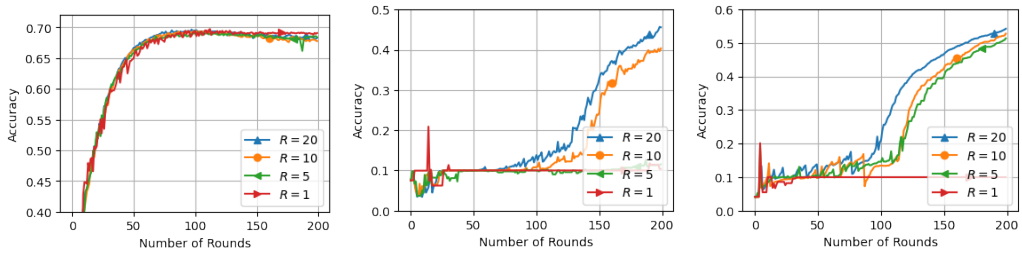
(a) Zeno++, q=12 (b) Krum, q=12 (c) Trimmed mean, q=12

Figure 2: Results of SHARE with various defenses on CIFAR-10, utilizing varying cluster sizes under label flip attack. For Trimmed mean we remove 2/3 of input updates.

that having no defense diverges even with mild attacks as expected. Further Figure (2a) shows that SHARE with a strong defense like Zeno++ converges to benign (no-attack) accuracy for any of the considered cluster sizes. SHARE with trimmed mean and Krum both converge with cluster size 3 but as cluster size increases, accuracy decreases and SHARE begins to diverge. This can be seen directly from Lemma 2, since we set trimmed mean to filter $q_0 = (2/3) * 60 = 40$ of updates, a cluster size of 3 implies the algorithm is robust against $q = 40/3 > 12$ clients being malicious, hence the algorithm converges to benign accuracy, increasing the cluster size decreases this tolerance threshold and hence as shown in Figure (2c) may fail to converge. Further experiments on scaled sign-flip attacks, are included in Appendix D due to space constraints.

6.1.2 Impact of reclustering

Intuitively, increasing the number of reclustering rounds increases the expected number of clusters without a Byzantine client. This hence increases the robustness of SHARE to higher fraction of Byzantine clients with defenses like Zeno++ which can tolerate arbitrary levels of poisoning. We test this hypothesis with several attack and clustering scenarios using sign-flipping attacks. In (a), we



(a) $q = 6$, $m = 10$, -10Δ (b) $q = 30$, $m = 4$, -10Δ (c) $q = 6$, $m = 25$, -100Δ

Figure 3: Results of SHARE with Zeno++and different reclustering rounds R, Byzantine clients q, cluster sizes m on CIFAR-10 with varying attack strengths (Any benign model update Δ is scaled to either -10Δ or -100Δ). In (a),(b) we use $n = 60$ and for (c) we use $n = 100$.

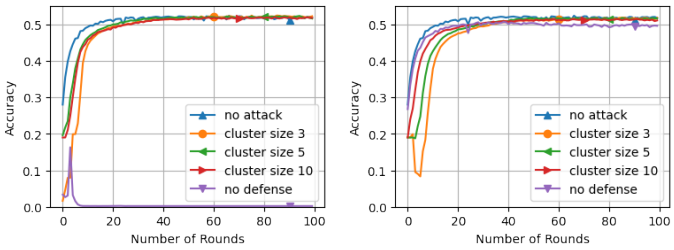
use a relatively small cluster size and a low fraction of Byzantine clients, so 1 round is sufficient. In (b), the fraction of Byzantine clients is high and in (c) the cluster size is large, which increases the probability of a cluster containing a Byzantine client, so $R > 1$ helps converge to higher accuracies.

6.2 Shakespeare

We consider the first 60 speaking roles in the train set as our 60 clients. We train an RNN with 2 LSTM layers followed by 1 fully connected layer[17] and report top-1 accuracy on the testing set.

6.2.1 Empirical Evaluation

In Figure 4 we evaluate Byzantine tolerance of SHARE with Zeno++ under sign-flip attack (malicious clients send an update negative to the benign one $-\Delta$) and scaled sign-flip attack (malicious clients scale the update in addition to flipping the sign and hence send -10Δ). A stronger attack like scaled sign-flip breaks benign averaging and Zeno++ works well with any of the chosen cluster sizes.



(a) Zeno++, -10Δ (b) Zeno++, $-\Delta$

Figure 4: SHARE with Zeno++ defense and sign-flipping attack on Shakespeare.

7 Discussion and Conclusion

We have proposed SHARE, a framework for implementing Byzantine-robustness and privacy. The key idea is hierarchical clustering. Cluster size is an important parameter that controls the trade-off between privacy and robustness. Further, reclustering is an important step and can help decrease variance and increase tolerance to the fraction of malicious clients when the defense can support arbitrary failures like Zeno++. In future, we would like to explore other variations in client clustering, especially in heterogeneous data settings. Further, we plan to work on stronger security guarantees even with multiple reclustering rounds.

Acknowledgments and Disclosure of Funding

Koyejo acknowledges partial funding from a C3.ai Digital Transformation Institute Award and a Jump Arches Award. This work was also funded in part by NSF III 2046795 and IIS 1909577. Additionally, the authors like to acknowledge Microsoft Azure for computational resources. Finally we would like to thank Dakshita Khurana and Nishant Kumar for their insightful discussions.

References

[1] Y. Aono, T. Hayashi, L. Wang, S. Moriai, et al. Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Transactions on Information Forensics and Security*, 13(5): 1333–1345, 2017.

[2] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 118–128, 2017.

[3] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth. Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191, 2017.

[4] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečnỳ, S. Mazzocchi, H. B. McMahan, et al. Towards federated learning at scale: System design. *arXiv preprint arXiv:1902.01046*, 2019.

[5] S. Caldas, S. M. K. Duddu, P. Wu, T. Li, J. Konečnỳ, H. B. McMahan, V. Smith, and A. Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

[6] A. Ghosh, J. Hong, D. Yin, and K. Ramchandran. Robust federated learning in a heterogeneous environment. *arXiv preprint arXiv:1906.06629*, 2019.

[7] A. Hard, K. Rao, R. Mathews, S. Ramaswamy, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*, 2018.

[8] L. He, S. P. Karimireddy, and M. Jaggi. Secure byzantine-robust machine learning. *arXiv preprint arXiv:2006.04747*, 2020.

[9] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

[10] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pages 5132–5143. PMLR, 2020.

[11] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[12] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[13] L. Lamport, R. Shostak, and M. Pease. The byzantine generals problem. In *Concurrency: the Works of Leslie Lamport*, pages 203–226. 2019.

[14] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

[15] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017.

[16] K. Pillutla, S. M. Kakade, and Z. Harchaoui. Robust aggregation for federated learning. *arXiv preprint arXiv:1912.13445*, 2019.

[17] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečnỳ, S. Kumar, and H. B. McMahan. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

[18] J. A. Rice. *Mathematical statistics and data analysis*. Cengage Learning, 2006.

[19] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[20] L. Wang, Q. Pang, S. Wang, and D. Song. Towards bidirectional protection in federated learning, 2021.

[21] C. Xie, O. Koyejo, and I. Gupta. Slsgd: Secure and efficient distributed on-device machine learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 213–228. Springer, 2019.

[22] C. Xie, S. Koyejo, and I. Gupta. Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance. In *International Conference on Machine Learning*, pages 6893–6901. PMLR, 2019.

[23] C. Xie, O. Koyejo, and I. Gupta. Fall of empires: Breaking byzantine-tolerant sgd by inner product manipulation. In R. P. Adams and V. Gogate, editors, *Proceedings of The 35th Uncertainty in Artificial Intelligence Conference*, volume 115 of *Proceedings of Machine Learning Research*, pages 261–270. PMLR, 22–25 Jul 2020. URL `http://proceedings.mlr.press/v115/xie20a.html`.

[24] C. Xie, S. Koyejo, and I. Gupta. Zeno++: Robust fully asynchronous sgd. In *International Conference on Machine Learning*, pages 10495–10503. PMLR, 2020.

Appendix

A Notations

Notation	Description	Notation	Description
n	Total number of clients	c	Number of clusters
q	Number of faulty clients	$\mathcal{S}$	set of all clients
K	Number of local SGD epochs	$[m]$	The set of integers $\{1, \ldots, m\}$
T	Number of global epochs	$\{\mathcal{M}_j\}_{j\in[c]}$	Set of client clusters
R	Number of resampling epochs	n_i	Number of samples on worker i
b	Trim parameter for defense	m	Number of clients in each cluster
η_l, η	Local and global learning rates	$\|\cdot\|$	All norms in this paper are l_2-norms

Table 1: Notations utilized in this paper

B Proofs

In this section, we elaborate on theoretical guarantees of SHARE. We define the following quantity to aid the proofs that follow

Definition B.1 (Clustered Client Update)**.** We define clustered client update as average model updates from all the clients assigned to a particular cluster. Mathematically, the clustered client update in a reclustering round $r \leq R$ is given by $g_i^r = \sum_{j\in\mathcal{M}_k} \Delta_j$ where Δ_j denotes the model update from client j belonging to cluster $\mathcal{M}_k$ after K steps of SGD.

Lemma 5. *If robust aggregation is replaced by averaging, output of Algorithm 1 is identical to Federated Averaging[15].*

Proof. In each re-clustering round, the update with benign averaging becomes $g^r = \sum_{i\in[c]} \sum_{j\in\mathcal{M}_i} \Delta_j^K = \sum_{l\in[n]} \Delta_l^K$ where n is the total number of clients and $c = \frac{n}{m}$ is the total number of clusters, as this update is independent of the random cluster division, the global update at round t becomes $x^t = x^{t-1} + \eta \sum_{l\in[n]} \Delta_l^K$ which is identical to federated averaging. □

Lemma 6. *In presence of robust aggregation, Algorithm 1 is robust to $q = \frac{q_0}{m}$ where q_0 is the tolerance limit of the robust aggregation oracle followed and m is the cluster size.*

Proof. We consider the worst case scenario of each malicious client being in different clusters, hence spreading the attack to the maximum possible number of clients. Although randomization beats this and might offer better clusters in multiple random rounds, there might still exist such attack favorable rounds. Allowing for this worst case sets the threshold to $q = \frac{q_0}{m}$ if the original robustness oracle has a threshold of q_0. □

B.1 Distance based robust aggregation

Theorem 7. *Consider a function F(x) satisfying Assumptions 5.1,5.2 assume a robust aggregation scheme that picks up b updates and satisfies Assumption 5.3, further, assume (G,B)-Bounded gradient dissimilarity, σ_g^2 variance in client updates and σ^2 variance in gradient estimation, there exists η, η_l such that output of Algorithm 1 after T rounds, x^T, satisfies,*

$$\mathbb{E}\left[\|\nabla F(x^T)\|^2\right] \leq \mathcal{O}\left(\frac{LM\sqrt{F}}{\sqrt{TKn}} + \frac{F^{2/3}(LG^2)^{1/3}}{(T+1)^{2/3}} + \frac{B^2LF}{T} + 2L^2V_2 + \frac{\sigma_g^2}{bm}\left(\frac{n-q-bm}{R(n-q)-1}\right)\right)$$

where $M^2 := \sigma^2(1 + \frac{n}{\eta^2})$ and $F := F(x^0) - F(x^*)$

Proof. Firstly, we bound the distance between global model update from Algorithm 1 and expected benign mean model update in each global iteration. In particular, let the expected benign mean model update be denoted by μ_t and global model update in each

iteration is given by $\frac{1}{R}\sum_r \texttt{RobustAggr}(\{g_i^r\}_{i\in[c]})$. We determine an upper bound on $\|\mathbb{E}[\frac{1}{R}\sum_r \texttt{RobustAggr}(\{g_i^r\}_{i\in[c]})] - \mu_t\|$. This is illustrated below

$$\begin{aligned}
\|\mathbb{E}[\frac{1}{R}\sum_r \texttt{RobustAggr}(\{g_i^r\}_{i\in[c]})] - \mu_t\|^2 &\leq \|\mathbb{E}[\frac{1}{R}(\sum_r \texttt{RobustAggr}(\{g_i^r\}_{i\in[m]}) - \sum_{r,i\in B} g_i^r)] \\
&\quad + \mathbb{E}[\frac{1}{R}\sum_{r,i\in B} g_i^r] - \mu_t\|^2 \\
&\leq \mathcal{O}(V_2) + 2\|\mathbb{E}[\frac{1}{R}\sum_{r,i\in B} g_i^r] - \mu_t\|^2 \\
&\leq \mathcal{O}(V_2) + 2\mathbb{E}\|\texttt{Resample}(\{\Delta_i^K\}_{i\in C}) - \mu_t\|^2 \\
&\leq \mathcal{O}(V_2) + \frac{\sigma_g^2}{Rbm}(1 - \frac{R(bm)-1}{R(n-q)-1}) \\
&\leq \mathcal{O}(V_2) + \frac{\sigma_g^2}{bm}(\frac{n-q-bm}{R(n-q)-1})
\end{aligned}$$

Where B denotes indices of benign clusters (clusters with uncorrupted device updates. Mathematically, let $\mathcal{C}_t$ denote set of benign clients among all n clients. $\{B : i \in [c]$ such that $\forall k \in [m], \Delta_k \in \mathcal{M}_i, \Delta_k \in \mathcal{C}_t\}$), $r \leq R$ as mentioned in the text denote reclustering rounds. The second inequality follows from Assumption 5.3. Since each reclustering round randomly groups clients together, the set $\{\Delta_k : \Delta_k \in \mathcal{M}_i, i \in B\}$ is a random resample of bm benign client updates from the available $n-q$, where b is the number of updates available after filteration through robust aggregation. With R resampling rounds, this is equivalent to resampling Rbm updates from $R(n-q)$ benign updates. Following Rice [18](Chapter 7, Theorem B), we obtain the scaled down variance bound.

Using L-smoothness of $F(x)$,

$$\begin{aligned}
\mathbb{E}[\|\nabla F(x^t)\|^2] &\leq 2\mathbb{E}[\|\nabla F(x^t) - \nabla F(\mu_t)\|^2] + 2\mathbb{E}[\|F(\mu_t)\|^2] \\
&\leq 2L^2(\mathcal{O}(V_2) + \frac{\sigma_g^2}{bm}(\frac{n-q-bm}{R(n-q)-1})) + 2\mathbb{E}[\|F(\mu_t)\|^2]
\end{aligned}$$

The rest follows a similar approach as Karimireddy et al. [10] hence we get

$$\mathbb{E}\left[\|\nabla F(x^T)\|^2\right] \leq \mathcal{O}\left(\frac{LM\sqrt{F}}{\sqrt{TKn}} + \frac{F^{2/3}(LG^2)^{1/3}}{(T+1)^{2/3}} + \frac{B^2LF}{T} + 2L^2V_2 + \frac{\sigma_g^2}{bm}\left(\frac{n-q-bm}{R(n-q)-1}\right)\right)$$

where $M^2 := \sigma^2(1 + \frac{n}{\eta^2})$ and $F := F(x^0) - F(x^*)$. □

B.2 Zeno++ as robust aggregation

We first illustrate a modified Zeno++ algorithm and adapt it to Federated Learning setting from its original asynchronous SGD paradigm. Firstly, we define a score that helps filter out updates if they fall below a threshold. Intuitively, the score denotes trustworthiness of a clustered update.

Definition B.2 (Approximated model update score)**.** Denote $f_s(x) = \frac{1}{n_s}\sum_i^{n_s} f(x; z_i)$, where $z_j's$ are drawn independent and identically from $\mathcal{D}_s \neq \mathcal{D}_i, \forall i \in [n]$ and n_s is the batch size of $f_s(\cdot)$, for a clustered client update g, model parameter x, global learning rate η and constant weight $\rho > 0$, we define model update score as

$$Score_{\eta,\rho} \approx -\eta\langle\nabla f_s(x), g\rangle - \rho\|g\|^2$$

where x is the current model available on the server.

Using this approximated model update score, we set hard thresholding parameterized by ϵ to filter client cluster updates. Algorithm 2 illustrates SHARE framework with Zeno++ as robust aggregation. We analyze the convergence of Algorithm 2 in the following theorem.

Theorem 8. *Consider L-smooth and potentially non-convex functions $F(x)$ and $f_s(x)$, Assume validation set is close to training set, implying a bounded variance given by $\mathbb{E}[\|\nabla f_s(x) - \nabla F(x)\|^2] \leq$*

Algorithm 2 SHARE (Secure Hierarchical Robust Aggregation) with Zeno++ defense

Server:
for $t = 0, \ldots, T-1$ **do**
  for $r = 1, \ldots, R$ **do**
    Assign clients to clusters $\mathcal{S} = \mathcal{M}_1 \cup \ldots \mathcal{M}_i \ldots \cup \mathcal{M}_c$ with $|\mathcal{M}_i| = |\mathcal{M}_j| \forall i, j \in [c]$
    Compute secure average $g_j^r \leftarrow \texttt{SecureAggr}(\{\Delta_i\}_{i \in \mathcal{M}_j}) = \sum_{i \in \mathcal{M}_j} u_i, \forall j \in [c]$
    Randomly sample $z_j \sim S, \forall j \in [n_s]$ to compute f_s
    for $i = 1, \ldots, c$ **do**
      if score$(g_i^r, x^{t-1} \geq -\eta\epsilon)$ **then**
        $g^r \leftarrow g^r + g_i^r$,
      end if
    end for
  end for
  if stopping criteria met **then**
    break
  end if
  Push $x^t = x^{t-1} + \eta \frac{1}{R} \sum_r g^r$ to the clients
end for
Client:
for each client $i \in \mathcal{S}$ (if honest) **in parallel do**
  $x_{i,0}^t \leftarrow x^t$
  for $k = 0, \ldots, K-1$ **do**
    Compute an unbiased estimate $g_{i,k}^t$ of $\nabla f_i(x_{i,k}^t)$
    $x_{i,k+1}^t \leftarrow \texttt{ClientOptimize}(x_{i,k}^t, g_{i,k}^t, \eta_l, k)$
  end for
  $\Delta_i = \frac{n_i}{n}(x_{i,K}^t - x^t)$
  Push Δ_i to the assigned clusters using secure aggregation
end for
return x^T

$V_1, \forall x$, Assume $\|f_s(x)\|^2 \leq V_3, \forall x$. Further assuming bounded gradient dissimilarity as stated in 5.1,variance between client updates of σ_g^2 and variance in gradient estimation at each client be σ, with global and local learning rates of $\eta \leq \frac{1}{2L}$ and $\rho \geq \frac{\alpha\sqrt{\eta}}{6K^2\eta_l^2 B^2} + \eta$, after T global updates, let $F := F(x^0) - F(x^*)$, Algorithm 2 with Zeno++ as robust aggregation converges at a critical point:

$$\frac{\mathbb{E}[\sum_{t \in [T]} \|\nabla F(x_{t-1})\|^2]}{T} \leq \frac{\mathbb{E}[F]}{\alpha\sqrt{\eta}T} + \frac{\sqrt{\eta}}{\alpha}\mathcal{O}\left(\frac{\sigma_g^2}{m}\left(\frac{n-q-m}{R(n-q)-1}\right) + G^2 + \sigma^2 + V_1 + V_3 + \epsilon\right)$$

Proof. Since for any cluster update g that passes the test of Zeno++, it follows that

$$-\langle \nabla f_s(x_t), \eta g \rangle - \rho \|g\|^2 \geq -\eta\epsilon.$$

Thus, we have

$$\begin{aligned}
&\langle \nabla F(x_{t-1}), \eta \mathbb{E}[g^r] \rangle \\
&\leq \langle \nabla F(x_{t-1}) - \nabla f_s(x_t), \eta \mathbb{E}[g^r] \rangle - \rho \mathbb{E}[\|g^r\|^2] + \eta\epsilon \\
&\leq \frac{\eta}{2}\|\nabla F(x_{t-1}) - \nabla f_s(x_t)\|^2 + \frac{\eta}{2}\|\mathbb{E}[g^r]\|^2 - \rho \mathbb{E}[\|g^r\|^2] + \eta\epsilon \\
&\leq \frac{\eta}{2}\|\nabla F(x_{t-1}) - \nabla f_s(x_t)\|^2 + (\frac{\eta}{2} - \rho)\mathbb{E}[\|g^r\|^2] + \eta\epsilon \\
&\leq \frac{\eta}{2}\|\nabla F(x_{t-1}) - \nabla F(x_t) + \nabla F(x_t) - \nabla f_s(x_t)\|^2 + (\frac{\eta}{2} - \rho)\mathbb{E}[\|g^r\|^2] + \eta\epsilon \\
&\leq \eta\|\nabla F(x_{t-1}) - \nabla F(x_t)\|^2 + \eta\|\nabla F(x_t) - \nabla f_s(x_t)\|^2 + (\frac{\eta}{2} - \rho)\mathbb{E}[\|g^r\|^2] + \eta\epsilon \\
&\leq \eta\|\nabla F(x_{t-1}) - \nabla F(x_t)\|^2 + \eta V_1 + (\frac{\eta}{2} - \rho)\mathbb{E}[\|g^r\|^2] + \eta\epsilon \\
&\leq \eta^3 L^2 \mathbb{E}[\|g^r\|^2] + \eta V_1 + (\frac{\eta}{2} - \rho)\mathbb{E}[\|g^r\|^2] + \eta\epsilon
\end{aligned}$$

Where g^r is the model update in reclustering round $r \leq R$. From L smoothness, we have

$$\|\nabla F(x_{t-1}) - \nabla F(x_t)\|^2 \leq L^2 \|x_{t-1} - x_t\|^2 \leq L^2 \eta^2 \mathbb{E}[\|g^r\|^2]$$

Using smoothness again, considering a global step size as $\eta L \leq \frac{1}{2}$we get

$$\mathbb{E}[F(x_t)] \tag{1}$$

$$\leq F(x_{t-1}) + \langle \nabla F(x_{t-1}), \eta \mathbb{E}[g^r] \rangle + \frac{L\eta^2}{2} \mathbb{E}[\|g^r\|^2] \tag{2}$$

$$\leq F(x_{t-1}) + (\eta^3 L^2 + \frac{\eta}{2} - \rho + \frac{L\eta^2}{2}) \mathbb{E}[\|g^r\|^2] + \eta V_1 + \eta \epsilon \tag{3}$$

$$\leq F(x_{t-1}) + (\eta - \rho) \mathbb{E}[\|g^r\|^2] + \eta V_1 + \eta \epsilon \tag{4}$$

Now we will bound the term $\mathbb{E}[\|g^r\|^2]$. Further, $\mathbb{E}[\|g^r\|^2] \leq 2(\frac{V_3}{2} + \eta\epsilon) + \mathbb{E}\|\tilde{g}^r\|^2$ where $\|\nabla f_s(x)\|^2 \leq V_3$ and $\tilde{g}^r$ is benign average obtained through sampling of benign clients

$$\mathbb{E}\|\tilde{g}^r\|^2 \leq \mathbb{E}\|x_i^K - x_{t-1}\|^2 + \frac{\sigma_g^2}{m} (\frac{n-q-m}{R(n-q)-1}) \tag{5}$$

Where x_i^K corresponds to model parameters after K rounds of SGD on ith device, σ_g^2 corresponds to variance between device updates. Since sampling successive $g_i's$ can be seen as sampling with replacement, and at least one cluster is selected each time, this has a maximum variance of single cluster selection case. (m is the cluster size, n is the total number of devices). The mean is equal to client drift, which can be bounded as shown below (for notational brevity, present global model x_{t-1} is denoted as x). Let us assume gradients at each data point $g_i(x_i^{k-1}) = \nabla f_i(x_i^{k-1}) +$ error, where error has mean 0 and σ standard deviation as stated in Assumption 5.3. For $k \leq K$ steps of local SGD, we get

$$\begin{aligned}
\mathbb{E}\|x_i^k - x\|^2 &= \mathbb{E}\|x_i^{k-1} - x - \eta_l g_i(x_i^{k-1})\|^2 \\
&\leq \mathbb{E}\|x_i^{k-1} - x - \eta_l \nabla f_i(x_i^{k-1})\|^2 + \eta_l^2 \sigma^2 \\
&\leq (1 + \frac{1}{K-1}) \mathbb{E}\|x_i^{k-1} - x\|^2 + K\eta_l^2 \|\nabla f_i(x_i^{k-1})\|^2 + \eta_l^2 \sigma^2 \\
&\leq (1 + \frac{1}{K-1}) \mathbb{E}\|x_i^{k-1} - x\|^2 + 2K\eta_l^2 \|\nabla f_i(x_i^{k-1}) - \nabla f_i(x)\|^2 + 2K\eta_l^2 \|\nabla f_i(x)\|^2 + \eta_l^2 \sigma^2 \\
&\leq (1 + \frac{1}{K-1} + 2K\eta_l^2 L^2) \mathbb{E}\|x_i^{k-1} - x\|^2 + 2K\eta_l^2 \|\nabla f_i(x)\|^2 + \eta_l^2 \sigma^2
\end{aligned}$$

Where the first inequality uses mean and variance in gradient estimation, the second one follows from relaxed triangle inequality as stated in Karimireddy et al. [10](Lemma 3). Taking appropriate local step size $\eta_l^2 \leq \frac{1}{2L^2 K(K-1)}$ and telescoping the sum, we get

$$\begin{aligned}
\mathbb{E}\|x_i^k - x\|^2 &\leq \sum_{\tau=1}^{k-1} (2K\eta_l^2 \|\nabla f_i(x)\|^2 + \eta_l^2 \sigma^2)(1 + \frac{2}{K-1})^\tau \\
&\leq (2K\eta_l^2 \|\nabla f_i(x)\|^2 + \eta_l^2 \sigma^2) \sum_\tau (1 + \frac{2}{K-1})^\tau \\
&\leq (2K\eta_l^2 \|\nabla f_i(x)\|^2 + \eta_l^2 \sigma^2) 3K
\end{aligned}$$

The last inequality follows from the fact that $\tau < K$ and $(1 + x/n)^n < \exp(x)$. Substituting this back into (5) and averaging over all $i's$ (client devices), we get

$$\begin{aligned}
\mathbb{E}\|g^r\|^2 &\leq \frac{1}{N} 6K^2 \eta_l^2 \sum_i \|\nabla f_i(x)\|^2 + 3K\eta_l^2 \sigma^2 + \frac{\sigma_g^2}{m} (\frac{n-q-m}{R(n-q)-1}) + V_3 + 2\eta\epsilon \\
&\leq 6K^2 \eta_l^2 G^2 + 6K^2 \eta_l^2 B^2 \|\nabla F(x)\|^2 + 3K\eta_l^2 \sigma^2 + \frac{\sigma_g^2}{m} (\frac{n-q-m}{R(n-q)-1}) + V_3 + 2\epsilon\eta
\end{aligned}$$

Where $\frac{1}{N}\sum_i \|\nabla f_i(x)\|^2 \leq G^2 + B^2\|\nabla F(x)\|^2$ follows from bounded gradient assumption. Combining this with (4), we get

$$\mathbb{E}[F(x_t)] \leq F(x_{t-1})$$
$$+ (\eta - \rho)(6K^2\eta_l^2 G^2 + 6K^2\eta_l^2 B^2\|\nabla F(x)\|^2 + 3K\eta_l^2\sigma^2 + \frac{\sigma_g^2}{m}(\frac{n-q-m}{R(n-q)-1})) + \eta V_1 + V_3 + 3\eta\epsilon$$

Taking $\rho \geq \frac{\alpha\sqrt{\eta}}{6K^2\eta_l^2 B^2} + \eta$, we have

$$\|\nabla F(x_{t-1})\|^2 \leq \frac{\mathbb{E}(F(x_{t-1}) - F(x_t)))}{\alpha\sqrt{\eta}} + \frac{\sqrt{\eta}}{\alpha}\mathcal{O}\left(\frac{\sigma_g^2}{m}\left(\frac{n-q-m}{R(n-q)-1}\right) + G^2 + \sigma^2 + V_1 + V_3 + \epsilon\right)$$

Telescoping and using expectation after T global epochs, we get

$$\frac{\mathbb{E}[\sum_{t\in[T]}\|\nabla F(x_{t-1})\|^2]}{T} \leq \frac{\mathbb{E}[F]}{\alpha\sqrt{\eta}T} + \frac{\sqrt{\eta}}{\alpha}\mathcal{O}\left(\frac{\sigma_g^2}{m}\left(\frac{n-q-m}{R(n-q)-1}\right) + G^2 + \sigma^2 + V_1 + V_3 + \epsilon\right)$$

□

C Security

To summarize, the security protocol operates in multiple rounds as is the case with any secure aggregation oracle in distributed learning. Firstly, keys are shared among every pair of clients in a cluster, this is followed by collection of masked inputs among each cluster by the server, which are then averaged within the cluster after a consistency check to make sure enough participants have participated in the round. Since model parameters are 32-bit floating points we convert them to integers and perform the masking modulo 2^{32}.

In particular, each client masks its private update with random vectors such that the server, even if curious, does not learn anything more than the sum of updates from a client cluster. For a given cluster $\mathcal{M}_k, k \in [c]$ assume that $\Delta_i, \sum_{i\in\mathcal{M}_k}\Delta_j \in \mathbb{Z}_P$, for some P. Consider an order on all the clients within a cluster and each pair of users $i, j(i < j)$ agree on a random vector $r_{i,j}$. If i adds this to its updates (Δ_i) and j subtracts it from its update (Δ_j), adding them would cancel and server would learn just the average but not individual updates. Hence, each client $i \in \mathcal{M}_k$ would compute $u_i = \Delta_i + \sum_{j\in\mathcal{M}_k, i<j} r_{i,j} - \sum_{j\in\mathcal{M}_k, i>j} r_{i,j}$ (mod P). If no clients drop in the computation round, it can be seen that $\sum_{i\in\mathcal{M}_k} u_i = \sum_{i\in\mathcal{M}_k}\Delta_i$ (mod P). Further, this can be made communication efficient by coordinating a common agreement on seeds for pseudorandom generator.

C.1 Communication efficiency

Notice that sharing a whole mask has a communication cost that increases linearly with the model size and hence prevents dealing with large models. This can be circumvented by clients agreeing on common seeds for a pseudorandom generator (PRG). PRG takes in a random seed as input and generates uniformly random numbers in $[0, P)^d$ where d is the model dimension. Engaging in a key agreement after broadcasting Diffie–Hellman public keys, as stated in [3] is a way to compute these shared seeds. Hence, each client belonging to a cluster $i \in \mathcal{M}_k$ would compute $u_i = \Delta_i + \sum_{j\in\mathcal{M}_k, i<j}\text{PRG}(s_{i,j}) - \sum_{j\in\mathcal{M}_k, i>j}\text{PRG}(s_{i,j})_{i,j}$ (mod P), where $s_{i,j}$ is the shared seed between clients i, j.

C.2 Handling dropped users

Since masks are shared between clients in a cluster, dropping of a client in a round causes incorrect computation of average as the masks do not exactly cancel each other. This problem is resolved by utilizing Samir's t out of n Secret Sharing [19] to share each clients' Diffie–Hellman secret with others and hence server can retrieve masks for the dropped client. Optionally, double masking as noted in Bonawitz et al. [3] can be used to enhance security.

C.3 Privacy at Server

As noted in Section 5, server learns nothing more than the average of the clustered client updates. However, multiple reclustering rounds poses an additional privacy threat since multiple averages among same clients appear in clear to the server. We note that the server requires at least $R \geq \frac{n}{c}$ to

identify all the updates, this can be used to tune R, c. Further, $R \geq \frac{n}{c}$ does not guarantee that server learns all updates since clusterings can overlap resulting in linearly dependent equations with infinite solutions.

C.4 Privacy from curious clients

As mentioned in Section 5, at least $m-1$ malicious clients are required in a cluster to infer the update of the remaining client. In each reclustering round this probability is upper bounded by $\mathcal{O}(\frac{m(n-m)!}{(q-1-m)!})$ and hence the rest being constant, as cluster size increases, it gets harder to break a client's privacy. Further, we note that just as $n-1$ colluding curious clients can break privacy in traditional Federated Learning, $m-1$ colluding curious clients can in our approach.

C.5 Communication costs

Server: The server communication cost is $\mathcal{O}(Rnm + Rdn)$, where R, m, n, d indicate number of reclustering rounds, number of clients per cluster, total number of clients and model size respectively. Here $\mathcal{O}(Rmn)$ is associated with mediation of pairwise communication between clients in each cluster and $\mathcal{O}(Rdn)$ is for receiving masked data vectors from each user. Although reclustering increases communication costs, clustering helps reduce pairwise communications.

Client: Client communication cost is $\mathcal{O}(Rm + Rd)$. Here $\mathcal{O}(Rm)$ is associated with pairwise key exchange within a cluster over all reclustering rounds and $\mathcal{O}(Rd)$ is for communicating its model to server in every reclustering round.

Note that these are passive adversaries hence while can be curious, they honestly follow the protocol for security. Compared to the two server approach suggested in He et al. [8], our approach can handle attacks on the server, because if an adversary attacks the server(s) or can see communication channels, model updates still remain private.

D Additional Experiments

We use a global learning rate $\eta = 1$, local learning rate $\eta_l = 0.01$, local momentum 0.9, and mini-batch size 64. We run each experiment for 200 global rounds with 2 local rounds each and report top-1 accuracy on the testing set. For all the experiments, unless specified, we use $R = 10$ reclustering rounds.

For Zeno++, we randomly sample 5% of the training data across clients with the same number of samples of each label to use as the server-side validation set as in Xie et al. [24]. We consider batch size of 128, $\rho = 0.0001, \epsilon = 0.2$ as Zeno++ parameters.

In experiments on Shakespear, we use $\eta = 1$, $\eta_l = 1$, local momentum 0.9, and mini-batch size 256. We run each experiment for 100 global rounds with 2 local rounds each. For all the experiments, unless specified, we use $R = 10$ reclustering rounds. Codes for the same would be made available soon.

D.1 Impact of cluster size

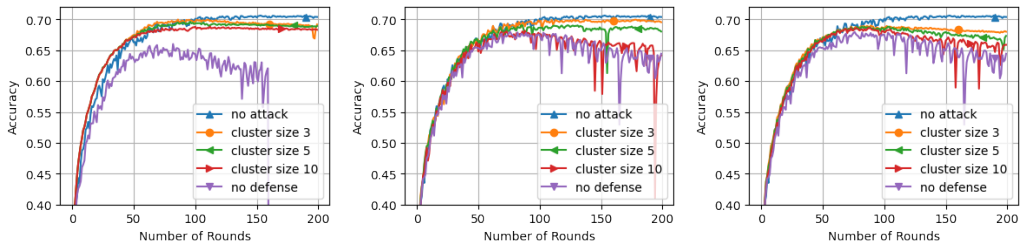
D.1.1 Fall of Empires Attack

We now test the sensitivity of cluster size on Byzantine-tolerance to a stronger attack with colluding adversaries, we utilize a modified version of Fall of Empires (FoE) [23]. In particular, each malicious client sends a negatively scaled averaged model update across all malicious clients. We test scaling these updates by $\beta = -1, -10$. Since Zeno++ can tolerate a greater fraction of clients being Byzantine, we set $q = 6$ while for trimmed mean and Krum, we set $q = 3$. (Parameters for defenses remain the same as in Section 6 unless specified.)

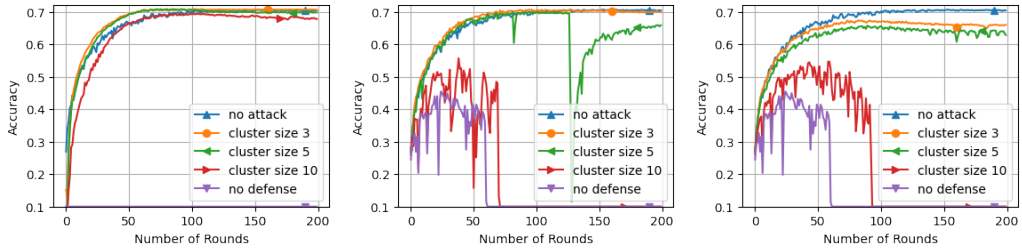
The results are plotted in Figure 5. A similar effect of cluster sizes as discussed in Section 6 can be seen here. Further, Zeno++ ,as expected, is stable even at higher levels of corruption.

D.2 Effect of Reclustering with Non-IID data

Empirically, we test the effect of reclustering on a heterogeneous data distribution. In particular, we divide CIFAR-10 among clients such that each one gets data from a few classes. In all experiments we use cluster size of 3.



(a) Zeno++, $\beta = -1, q = 6$ (b) Krum, $\beta = -1, q = 3$ (c) Trmean, $\beta = -1, q = 3$

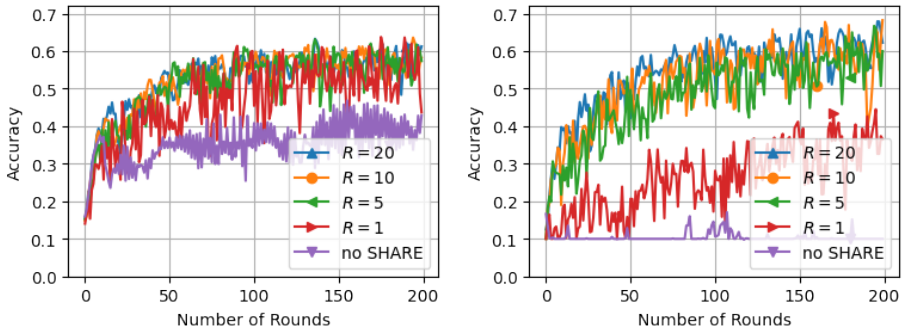


(d) Zeno++, $\beta = -10, q = 6$ (e) Krum, $\beta = -10, q = 3$ (f) Trmean, $\beta = -10, q = 3$

Figure 5: Results of SHARE on CIFAR-10, with varying cluster size across various defenses under FoE attack and q malicious clients out of 60 total clients. Malicious clients send their average update scaled by β as indicated in the subfigures. For trimmed mean (Trmean) we remove 2/3 of input updates and use batch size of 128, $\rho = 0.00001, \epsilon = 0.2$ as Zeno++ parameters.

D.2.1 No Attack

To create heterogeneous data effect, we split CIFAR-10 data across 60 clients such that each client gets only data consisting of 2 labels. As can be seen in Figure 6, robust defenses such as coordinate wise median[16] and Krum[2] fail in this setting, while SHARE with these defenses performs better as the number of reclustering rounds increases. Further, variance reduction is observed as we increase reclustering rounds (R).

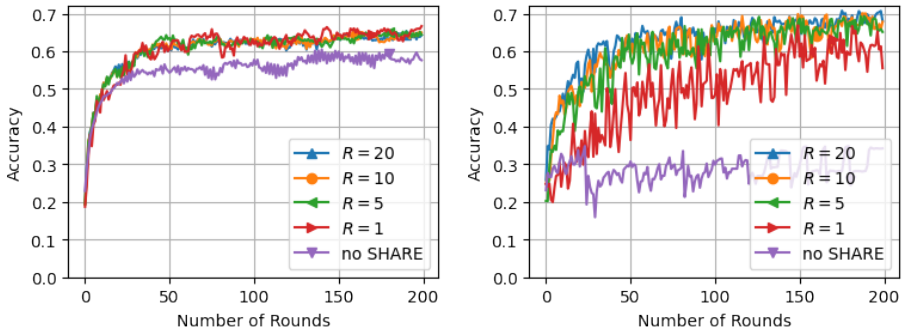


(a) Median (b) Krum

Figure 6: Results of SHARE on non-IID CIFAR-10, where each client has data from 2 labels, across various defenses without attack with 50 clients. We vary the number of reclustering rounds (R). No SHARE indicates the baseline defense without the SHARE framework.

D.2.2 Fall of Empires Attack

We consider a lower level of heterogeneity but with client corruption. In particular, each client gets data consisting of 5 labels and $q = 3$ malicious clients which collude to send their average model update scaled by $\beta = -10$. The results are shown in Figure 7 for various number of reclustering rounds (R).

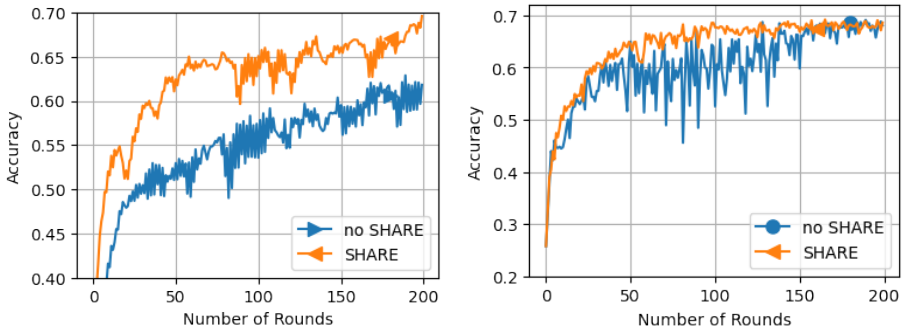


(a) Median (b) Krum

Figure 7: Results of SHARE on non-IID CIFAR-10, where each client has data from 5 labels, across various defenses with FoE attack ($\beta = -10$) and $q = 3$ malicious clients out of 50 total. We vary the number of reclustering rounds (R). No SHARE indicates the baseline defense without the SHARE framework.

D.2.3 Label Flip Attack

We test the effect of SHARE on the label-flip attack with a heterogeneous data split of CIFAR-10. Each client gets data from 5 labels. Malicious clients train on flipped labels, i.e. any label $\in \{0, \ldots, 9\}$ is changed to $9-$label. We test trimmed mean (filtering $2/3$ of input updates) and Zeno++ with batch size of 128, $\rho = 0.00001, \epsilon = 1$ with SHARE and use $\epsilon = 5$ without SHARE framework. These parameters are tuned to achieve good performances within their respective frameworks. We consider $R = 10$ reclustering rounds and $q = 12$ malicious clients. Results as shown in Figure 8 demonstrate the efficacy of SHARE.

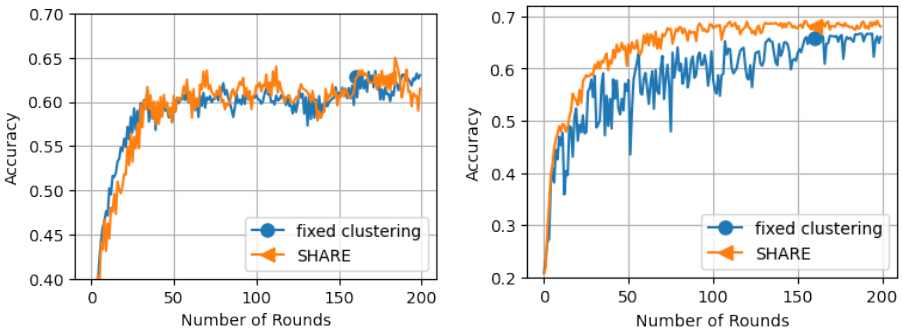


(a) Trimmed mean (b) Zeno++

Figure 8: Results of SHARE on non-IID CIFAR-10, where each client has data from 5 labels, across various defenses with label-flip attack, $q = 12$ malicious clients out of 60 total, and $R = 10$ reclustering rounds. No SHARE indicates the baseline defense without the SHARE framework.

D.2.4 Fixed clustering vs reclustering

Figure 9 compares results between fixed clustering and SHARE. In the former, we fix the client clusters before the learning process starts while the latter allows for random reclustering in every round. CIFAR-10 data is split heterogeneously such that each client receives 5 class labels alone. We use Fall of Empires attack with $\beta = -10$ scaling of the average gradients from the malicious clients. Since Zeno++ can tolerate higher levels of corruption, we consider $q = 6$ malicious clients, while for trimmed mean, we consider $q = 3$. We use $R = 10$ reclustering rounds and 60 clients in total with a cluster size of 3. We test trimmed mean (filtering $2/3$ of input updates) and Zeno++ with batch size of 128, $\rho = 0.00001, \epsilon = 1$ with SHARE and use $\epsilon = 5$ for fixed clustering case. These parameters are tuned to achieve reasonable performances within their respective frameworks.



(a) Trimmed mean, $q = 3$

(b) Zeno++, $q = 6$

Figure 9: Results of SHARE on non-IID CIFAR-10, where each client has data from 5 labels, across various defenses with FoE attack ($\beta = -10$) and q malicious clients out of 60 total. Fixed clustering indicates the baseline defense with fixed clusters of size 5.

E Random Reclustering

In the worst case, a benign client's signal is lost if it is clustered with a malicious client across all reclustering rounds. In particular, the probability that a benign client is effected in a global round is $\left(1 - \frac{\binom{n-q}{\frac{n}{m}-1}}{\binom{n}{\frac{n}{m}-1}}\right)^R$. Hence this probability decreases as R (number of reclustering rounds increase).