Robustness of Compressed Decentralized SGD with Gossip communication

Moulik Choraria (moulik.choraria@epfl.ch)
Harshvardhan (harshvardhan.harshvardhan@epfl.ch)
Aditya Vardhan Varre (aditya.varre@epfl.ch)

***Abstract*—We analyze the ChocoSGD algorithm in a non-convex setting. We observe the performance characteristics for a wide class of optimizers. We implement and analyze the efficacy of various attack and protection schemes for the ChocoSGD algorithms, for two popular network topologies.**

I. Introduction

Over the last few years, the size of training datasets has increased rapidly and the ability to learn in a distributed fashion has become increasingly important. One such setting is Federated learning, where multiple decentralized devices or node hold local data samples and learns an algorithm, without exchanging their data samples and with limited communication. In these settings, fault tolerance (may be adversarial) is an important requirement. Compressed SGD methods like signSGD, in a single aggregate server setting, have been shown to be fault-tolerant in presence of byzantine nodes in [1]. Ghosh et al. [2] also came up with a communication efficient learning algorithm with error feedback which is robust to the presence of byzantine workers for the same.

Recently, the authors of [3] proposed a gossip algorithm based decentralized stochastic optimization scheme, ChocoSGD, and proved its convergence for strongly convex objectives. Further, compressed communication methods like QSGD and signSGD were also shown to converge in the decentralized networks in the absence of a single aggregation server using gossip algorithms. The robustness of this method against byzantine attacks remains to be investigated.

In this report, we consider the problem of convergence of ChocoSGD for non-convex functions for various compression schemes. We also experiment to determine the robustness of ChocoSGD against adversarial attack by introducing byzantine nodes into the network. We try to merge these two settings to analyze fault-tolerance in gossip-style gradient communication for compressed SGD schemes. We experiment how known fault-tolerant methods ([1], [4]) can be incorporated in ChocoSGD. In the following sections, we describe the theory and intuition behind our experiments and we present our experiment results and conclusions.

II. Theory

A. Convergence

We considered the following optimizers for our study:

- signSGD with Error Feedback[5]
- Quantized SGD with lossy quantization[6]
- Error Compensated Quantized SGD [7]

Algorithm 1 Base Gossip Algorithm

for $t = 0$ to $T - 1$ **do**
  for each node ComputeNodeGradient()
  for each adversarial node ComputeAttackGradient()
  Communicate updates to neighbors
  Aggregate updates from neighbors
  Update node weights from aggregation
end for

B. Design of the Attack

For the byzantine nodes, we implement two adversarial attacks which are prevalent in literature:

- Full Reversal
- Random Reversal

In the Full Reversal attack, the adversarial node flips the sign of the gradient from the model, broadcasts the flipped gradient and **trains on the basis of the aggregation of the flipped gradient and the gradients of its neighbors**. In the random reversal attack, the adversarial node randoms flips the sign of each coordinate of the gradient as per the Bernoulli distribution with probability p and broadcasts it. Here too, **the node trains using the aggregate gradients of its local flipped value and the gradients of its neighbors**.

C. Fault tolerant methods

Variable network connectivity due to the gossip-style algorithms should in principle, make the corruption more localized near the byzantine nodes. Further, compressed communication along with the byzantine assumption leaves very little scope for incorporating faults, as can be seen by the robustness of signSGD under majority vote. Following this idea, we believe, quantizedSGD, being a compressed gradient scheme, should also be able to tolerate adversarial corruptness under a suitable protection scheme, along the lines of majority vote. We consider various methods which essentially try to remove outlier gradients. These are described in detail below:

- Majority Vote [1]: This method is specific to signSGD. We return the majority of signs of neighbors(including its own, this is same for the remaining methods too).
- Median [4] : For every element of the update vector, we return the median among all its neighbors.
- Trimmed Mean:[4] For every element of the update vector, we remove the first and last β fraction of elements,

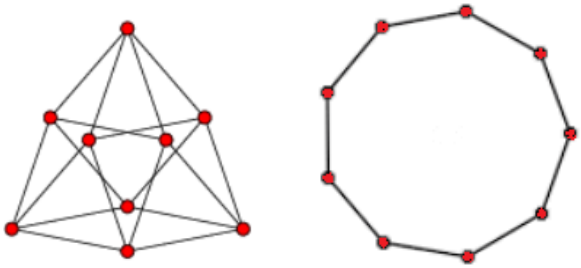
when ordered element-wise in terms of value and return the mean of the remaining elements.

- Fractional Mean:[2] We return the mean of the gradients of the first $(1-\beta)$ fraction of neighbors, when ordered by the L_2 norm of the gradients in ascending order.

III. Experiment Setup and Results

In this experimental setup, we considered a distributed system consisting of 9 worker nodes, with the training set of MNIST dataset equitably and randomly distributed across the nodes. The choice of MNIST, being one of the simplest datasets for image processing learning tasks, allows us to run extensive experiments for multiple cases in a reasonable amount of time, and with a reasonable amount of computational resources. For the model architecture on each node, we choose a simple variant of LeNet [8].

We run the experiments for two network topologies, with the nodes being connected according to either the ring or torus topology. This choice offers us the opportunity to study two interesting cases, since the ring is relatively sparsely connected where each node has two neighbours whereas a torus is more densely connected with each node having four neighbours each.



(a) Torus (b) Ring

The plots for training and testing loss and accuracies are recorded for each node. For obtaining the test accuracy of the system, we use a consensus for all nodes. For reasons of brevity, we include the plots of one healthy worker node per experiment. However, we mention the interesting aspects of training as well as elaborate on effects of the attack on different nodes, in the subsection where we describe common trends.

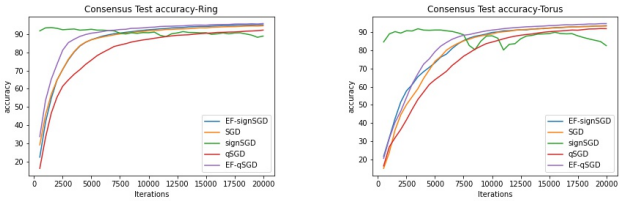
For adversarial attacks, we consider either attacking two or three nodes at a time. For any attack, we consider byzantine attacker nodes to be placed such that certain uniformity conditions are satisfied. For the ring topology, we ensure that each healthy node has at-most one infected node in its neighbourhood, while for the torus topology, we ensure that each healthy node has at-most two infected nodes in its neighbourhood. Because of this, when we use fractional mean method for defense, we choose the value of β accordingly. For ring, we chose the infected nodes such that atleast one neighbour is not infected hence β in this case would be $\frac{1}{3}$, similarly in torus atmost 2 of the 4 neighbours are infected hence we run it with $\beta = \frac{2}{5}$.

Note that we only include plots for the optimizers and their respective attack and defense schemes for which ChocoSGD is able to ensure that the system trains successfully. For the cases where the system fails to train, we briefly mention them in the Conclusions.

Finally, while we implemented the random reversal attack, we believe that it is just a weaker subset of the full reversal scheme and hence, we do not run the experiments to evaluate its effectiveness in the cases where we are able to defend against full reversal.

A. Base Case

The ChocoSGD algorithm gives convergence proofs for strongly convex functions. However, the task of training neural networks is inherently non-convex. Therefore, we first establish a baseline for various optimizers when the system is allowed to train without any adversarial attacks. As we can see from the plots below, we are able to attain substantial test accuracy for both these architectures for all the optimizers.



(a) Torus (b) Ring

B. Common Trends & Expectations

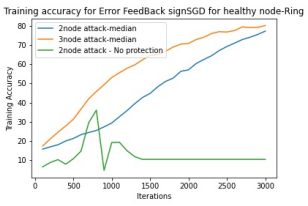
- We observe that the main difference between the torus and ring topologies is the number of faulty nodes that a single healthy node is exposed to. Since ring has lower exposure for the same number of faulty nodes, we find that it is easier for ring architectures to handle robustness. This is expressed in lower number of iterations for the ring architecture to obtain the test accuracy.
- We observe that the fault has the worst effect on the model belonging to the faulty node. Thus, even though the healthy nodes show convergence(around 80% accuracy), the faulty nodes have drastically smaller accuracy(around 10%). Further, the neighbors of the faulty nodes perform the worst among all healthy nodes, but not by much if there is overall convergence. The best performance among healthiest nodes is obtained for the node which is at the largest distance from all faulty nodes.
- Increasing the number of faulty nodes should worsen the overall performance of the network
- The node degree comparison for torus and ring does not have as straightforward implications when we include faulty nodes. While higher node degree implies higher exposure to faults, it also means higher exposure to healthy nodes to improve performance. We find that these two effects have different implications for different

settings. When the model converges, the differences due to higher fault exposure are more likely to appear. Thus, when we obtain convergence, we should find that rings perform better than torus. However, in the cases when we see no convergence, the advantage of having more correct nodes in the neighborhood should improve our performance and torus should work better.

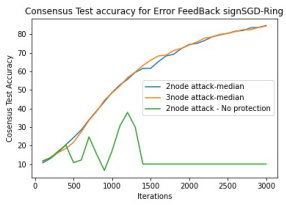
C. EF-signSGD

We consider Error-Feedback signSGD across both configurations. For this optimizer, we were able to obtain convergence even in the presence of adversarial attacks for the protection scheme of median. [1] is able to establish robustness for signSGD based on only majority protection scheme. Median is the equivalent protection scheme for majority when the gradients are real numbers instead of signs. As is evident, simple averaging of neighborhood gradients does not converge in the presence of 2 adversarial nodes spaced equally apart. Matching our prior intuition, the ring architectures are more robust and require fewer iterations to reach the same accuracy. The training accuracy for the healthy node for the ring architecture is much larger for the 3 node case than the 2 node case, which is a counter-intuitive but nevertheless an interesting observation. However, despite this huge difference in training accuracy, we find that the test accuracies for these two cases are very close to each other for the ring topology. For torus, higher fault levels perform worse during both training and consensus test.

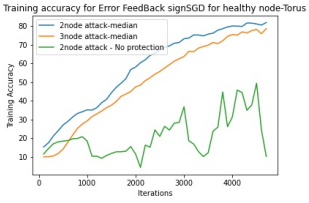
Additionally, when we observe no system convergence, the torus case has higher fluctuations, which highlights the advantage of having a higher degree and consequently, a higher number of healthy neighbors.



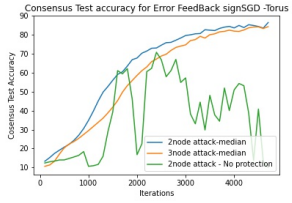
(a) Ring- healthy node training



(b) Ring - Consensus Test



(c) Torus- healthy node training



(d) Torus - Consensus Test

D. qSGD

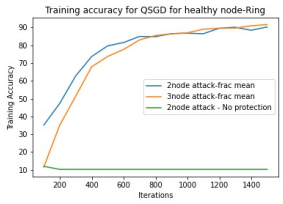
We found that qSGD was not able to converge using median protection scheme. The problem we faced was that the adversary nodes often drifted towards regions with high gradient norms, which influenced the neighbours and as a result in some cases, we were unable to control the norm of gradients even for healthy nodes.

We chose the norm-based frac-mean scheme to tackle this problem.

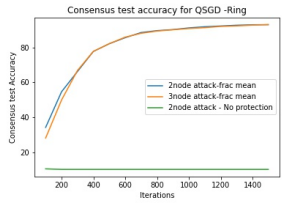
Inspite of this exploding gradient problem, we find that qSGD is able to attain similar performances to its EF-signSGD counterparts in fewer iterations with an appropriate protection scheme. This may be attributed to the greater degree of compression in EF-signSGD as compared to qSGD, which consequently allows qSGD to retain the gradient information more faithfully.

Firstly, since EF-signSGD augments signSGD, we feel that the gradient updates for it should have a lower variance than in case of qSGD and the lossy quantizer. This high variance may cause median protection scheme to pick updates with large gradient norms from faulty nodes in case of qSGD.

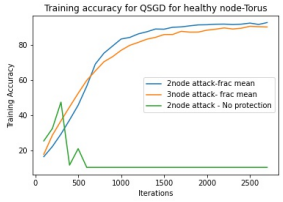
In contrast to the failing median method, the gradient norm based schemes of fractional mean, picks gradient updates with smaller norms and thus, even though it might obtain a smaller update per iteration than it could have, its updates are not faulty. This scheme also saves us from very high gradient updates from faulty nodes. The results here are similar to the common trends we described above.



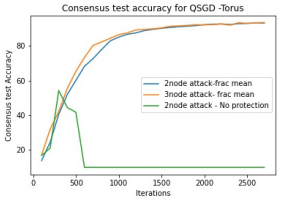
(a) Ring - healthy node training



(b) Ring - Consensus Test



(c) Torus - healthy node training



(d) Torus - Consensus Test

IV. Conclusions

ChocoSGD appears to be inherently robust to certain attacks for certain protection schemes. In the random reversal attack with probabilty $1/2$, we found that ChocoSGD with sign compression and Error-Feedback was robust against it. In ChocoSGD, the aggregation happens in place, where as for the fault tolerant methods we used, we finally do some sorting to eliminate the outliers, as a result of which, the training happens at a slower pace. Due to the space constraints, we do not elaborate further into the experiments in which methods were not robust and we also do not include the results of experiment involving qSGD with Error-Feedback. While we did observe robustness for ChocoSGD, we still need to carefully select our faulty nodes and correction schemes to achieve this robustness.

We believe that our experiments shed some light on how the sparsity in the graph, positioning of nodes and compression quality of the optimizer are integral in ensuring robustness.

REFERENCES

[1] J. Bernstein, J. Zhao, K. Azizzadenesheli, and A. Anandkumar, "signsgd with majority vote is communication efficient and fault tolerant," 2018.

[2] A. Ghosh, R. K. Maity, S. Kadhe, A. Mazumdar, and K. Ramchandran, "Communication-efficient and byzantine-robust distributed learning," 2019.

[3] A. Koloskova, S. U. Stich, and M. Jaggi, "Decentralized stochastic optimization and gossip algorithms with compressed communication," 2019.

[4] D. Yin, Y. Chen, K. Ramchandran, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," 2018.

[5] S. P. Karimireddy, Q. Rebjock, S. U. Stich, and M. Jaggi, "Error feedback fixes signsgd and other gradient compression schemes," 2019.

[6] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "Qsgd: Communication-efficient sgd via gradient quantization and encoding," in *Advances in Neural Information Processing Systems*, 2017, pp. 1709–1720.

[7] J. Wu, W. Huang, J. Huang, and T. Zhang, "Error compensated quantized sgd and its applications to large-scale distributed optimization," 2018.

[8] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, 1998, pp. 2278–2324.