SIGNSGD: FAULT-TOLERANCE TO BLIND AND BYZANTINE ADVERSARIES

Jason Akoun, Sébastien Meyer
École Polytechnique, France
{firstname.lastname}@polytechnique.edu

ABSTRACT

Distributed learning has become a necessity for training ever-growing models. In a distributed setting, the task is shared among several devices. Typically, the learning process is monitored by a server. Also, some of the devices can be faulty, deliberately or not, and the usual distributed SGD algorithm cannot defend itself from omniscient adversaries. Therefore, we need to devise a fault-tolerant gradient descent algorithm. We based our article on the SIGNSGD algorithm, which relies on the sharing of gradients signs between the devices and the server. We provide a theoretical upper bound for the convergence rate of SIGNSGD to extend the results of the original paper. Our theoretical results estimate the convergence rate of SIGNSGD against a proportion of general adversaries, such as Byzantine adversaries. We implemented the algorithm along with Byzantine strategies in order to try to crush the learning process. Therefore, we provide empirical observations from our experiments to support our theory. Our code is available on GitHub[1] and our experiments are reproducible by using the provided parameters.

1 Introdution

With the increasing size of datasets and of the diversity of their sources, the need for large-scale distributed systems has never been so important. In the field of distributed learning, there are two types of distributed settings. The first setting is centralized, that is, a server gathers gradients computed locally on the devices and broadcasts back the changes to make to local models. The second one is decentralized, with the information about model parameters having to propagate from device to device. Moreover, the learning process can happen synchronously or asynchronously. Typical examples of distributed centralized settings are the supercomputers that train state-of-the-art deep learning models. Decentralized asynchronous settings usually happen with small and abundant devices that are not switched on at the same time, such as phones. In the case of phones, there are also models that are centralized but fine-tuned locally (think about your phone's autocompletion of words). In this project, we focused on the centralized synchronous setting. Among the "workers" or "processes", there can be adversaries. All types of adversaries are included in this denomination, from the unintentional faulty processes to the coordinated, omniscient adversaries.

The main issue of the learning task is to avoid the propagation of faults onto the workers. Indeed, the classical stochastic gradient descent algorithm is not fault-tolerant, as we will show later on. Therefore, a gradient descent algorithm must provide the same dynamic of convergence as in Bottou 1998[1], that is, the aggregated gradient must fall in the decreasing half-space of the loss function. One of the most successful gradient descent algorithm that as been proposed in the recent years is Krum, and was detailed in 2017 by Blanchard et al.[2]. Nevertheless, the proposed algorithm and theoretical bounds for the convergence rate only work for a proportion of Byzantine adversaries bounded by $\mathcal{O}(\sqrt{d})$ where d is the space dimension and the learning process is more difficult with non convex loss functions. Despite the fact that there has been several follow-ups to this paper, other alternatives have been developed. In their 2018 paper, Bernstein et al.[3] have proposed a new gradient descent algorithm, namely SIGNSGD. In 2019, Bernstein et al.[4] extended SIGNSGD to SIGNUM and proved the theoretical tolerance of both algorithms to blind adversaries.

[1] https://github.com/sebastienmeyer2/signsgd-fault-tolerance

In this article, we recall the most important results from the initial papers and we try to go further by proposing a more general theoretical bound for the convergence rate of SignSGD, as well as experimental results to support our claims.

2 Previous work

In this section, we mainly recall results and propositions from both the initial paper[3] and the extension to fault-tolerance[4]. When looking at a particular algorithm for gradient descent, we want to verify the following properties:

D1. Fast algorithmic convergence

D2. Good generalisation performance

D3. Communication efficiency

D4. Robustness to network faults

Clearly, it will be unreasonable to think that one can devise an algorithm satisfying all four properties with high certainty. The usual stochastic gradient descent algorithm does satisfy the **D1** and **D2** properties, and this explains why it has been so widely used in machine and deep learning. Regarding **D3**, the stochastic gradient descent algorithm needs to communicate full vectors of gradients from workers to servers and the other way around. In addition, **D4** is not verified for several cases. Consider the example of an omniscient adversary. This adversary would just have to send to the server the inverse sum of the gradients values of all the other processes in order to stop the training. Thus, the authors have proposed a new algorithm, namely Signum, based on the communication of gradients signs.

Algorithm 1: Signum with majority vote. All operations are element-wise. Setting $\beta = 0$ yields SignSGD.

```
Input: learning rate $\eta > 0$, momentum $\beta \in [0, 1)$, weight decay $\lambda \geq 0$, batch size $n$, initial point $x$, number of
       workers $M$.
Initialize momentum $v_m \leftarrow 0$ for each worker;
repeat
    foreach worker $m$ do
        $\widetilde{g}_m \leftarrow \frac{1}{n} \sum_{i=1}^{n} F_i(x)$;
        $v_m \leftarrow (1 - \beta)\widetilde{g}_m + \beta v_m$;
        push $\text{sg}(v_m)$ to server;
    for the server do
        $V \leftarrow \sum_{m=1}^{M} \text{sg}(v_m)$;
        push $\text{sg}(V)$ to workers;
    foreach worker $m$ do
        $x \leftarrow x - \eta(\text{sg}(V) + \lambda x)$;
until convergence (or criterion);
```

It appears that the proposed algorithm verifies **D3** by communicating only signs between devices. Also, the **D2** property stems naturally from this simple algorithm. We will now look at both **D1** and **D4** properties.

2.1 Assumptions

The authors proved in their paper a theoretical bound for the convergence rate of SignSGD. They use four assumptions, of which the first three are usual assumptions in papers concerning gradient descent algorithms.

Assumption 1. (Lower bound) *For all x and some constant f^*, we have objective value $f(x) \geq f^*$.*

Assumption 2. (L-Smooth) *Let $g(x)$ denote the gradient of the objective $f(.)$ evaluated at point x. Then, $\forall x, y$ we require that for some non-negative constant $L = (L_1, ..., L_d)$,*

$$|f(y) - [f(x) +^t g(x)(y - x)]| \leq \frac{1}{2} \sum_i L_i (y_i - x_i)^2$$

Assumption 3. (Variance bound) *Upon receiving query $x \in \mathbb{R}^d$, the stochastic gradient oracle gives us an independent, unbiased estimate $\widetilde{g}$ that has coordinate bounded variance:*

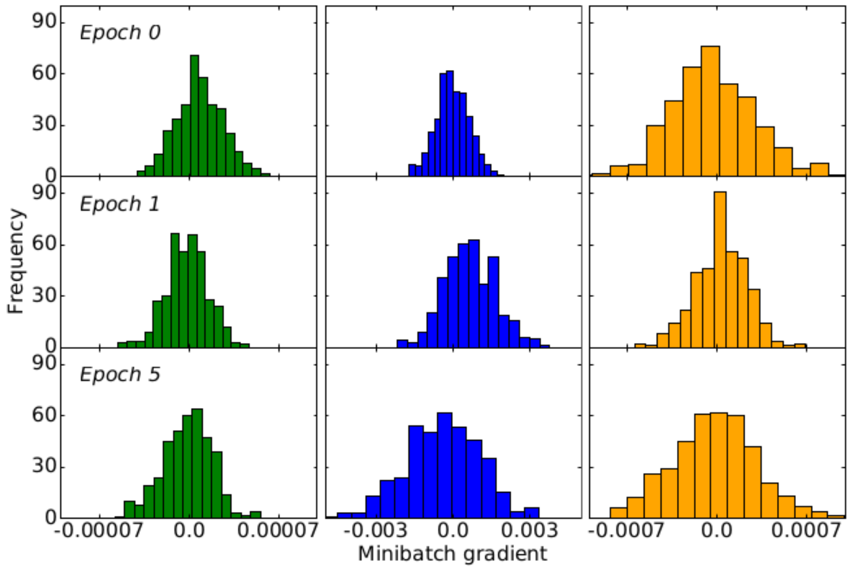

Figure 1: Gradients distributions for ResNet18 on CIFAR-10[4].

$$\mathbb{E}(\widetilde{g}(x)) = g(x) \quad \mathbb{E}((\widetilde{g}(x)_i - g(x)_i)^2) \leq \sigma_i^2$$

for a vector of non-negative constants $\sigma = (\sigma_1, ..., \sigma_d)$*.*

The fourth assumption is less common. The authors assume that the gradients follow unimodal gaussian distributions. This assumption stems from empirical observations, as shown **Figure 1**.

Assumption 4. (Unimodal, symmetric gradient noise) *At any given point* x*, each component of the stochastic gradient vector* $\widetilde{g}(x)$ *has a unimodal distribution that is also symmetric about the mean.*

2.2 Theoretical bound for blind adversaries

In their original paper, the authors have considered blind adversaries, that is, adversaries that do not know about the gradients of other workers. Since SIGNSGD algorithm relies on the communication of gradients signs, all the strategies that a blind adversary can think of come down to the following definition.

Definition 1. (Blind adversaries) *A blind adversary may invert their stochastic gradient estimate* $\widetilde{g}_t$ *at iteration* t*.*

The first result which allows the authors for proving their upper bound on convergence rate relies on **Assumptions 3 and 4**.

Lemma 1. (Bernstein et al., 2018[3]) *Let* $\widetilde{g}_i$ *be an unbiased stochastic approximation to gradient component* g_i*, with variance bounded by* σ_i^2*. Further assume that the noise distribution is unimodal and symmetric. Define signal-to-noise ratio* $S_i = \frac{|g_i|}{\sigma_i}$*. Then we have that*

$$\mathbb{P}(\text{sg}(\widetilde{g}_i) \neq \text{sg}(g_i)) \leq \begin{cases} \frac{2}{9}\frac{1}{S_i^2} & \textit{if } S_i > \frac{2}{\sqrt{3}}, \\ \frac{1}{2} - \frac{S_i}{2\sqrt{3}} & \textit{otherwise} \end{cases}$$

which is in all case less than or equal to $\frac{1}{2}$*.*

The bound gives an estimation of the ability to estimate a good approximation of the gradient component knowing that there is a certain noise. It allows to estimate an upper bound for the convergence rate of SIGNSGD.

Theorem 2. (Non-convex convergence rate of majority vote with adversarial workers, Bernstein et al., 2019[4]) *Run **Algorithm 1** for* K *iterations under **Assumptions 1 to 4**. Switch off momentum and weight decay* ($\beta = \lambda = 0$)*. Set the learning rate,* η*, and mini-batch size,* n*, for each worker as*

$$\eta = \sqrt{\frac{f_0 - f^*}{||L||_1 K}}, \qquad n = K.$$

*Assume that a fraction $\alpha < \frac{1}{2}$ of the M workers behave adversarially according to **Definition 1**. Then majority vote converges at rate:*

$$\left[\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}(||g_k||_1)\right]^2 \leq \frac{4}{\sqrt{N}}\left[\frac{1}{1-2\alpha}\frac{||\sigma||_1}{\sqrt{M}} + \sqrt{||L||_1(f_0 - f^*)}\right]^2$$

where $N = K^2$ is the total number of stochastic gradient calls per worker up to step K.

For further proofs and materials, we link the interested reader to [3] and [4].

3 Our more general theoretical bound

The previous lemma and theorem that we presented are designed to answer to the question of tolerance to blind adversaries. A more general type of adversaries are the Byzantine adversaries.

Definition 2. (Byzantine adversaries) *A Byzantine adversary may send an arbitraty value to the server. It is aware of the gradients values of the other workers and it may collude with other Byzantine adversaries to set up a strategy.*

A more general definition of Byzantine adversaries as well as the concept of (α, f)-Byzantine resilience can be found in Blanchard et al.[2]. Clearly, Byzantine adversaries are much more dangerous than blind adversaries. In the case of basic stochastic gradient descent, a Byzantine adversary can send a gradient of infinite norm and therefore crush the learning process. In this section, we propose a new upper bound for the tolerance of SIGNSGD to any type of adversaries. Moreover, we will only make use of **Assumptions 1 to 3**.

Lemma 1bis. *Let $\widetilde{g}_i$ be an unbiased stochastic approximation to gradient component g_i, with variance bounded by σ_i^2. Define signal-to-noise ratio $S_i = \frac{|g_i|}{\sigma_i}$. Then, we have that*

$$\mathbb{P}(\text{sg}(\widetilde{g}_i) \neq \text{sg}(g_i)) \leq \frac{1}{2S_i^2}$$

Proof. It is a direct application of Bienaymé-Tchebychev's inequality. □

With this new lemma, we are able to prove a new bound for the convergence rate of SIGNSGD.

Theorem 2bis. *Run **Algorithm 1** for K iterations under **Assumptions 1 to 3**. Switch off momentum and weight decay ($\beta = \lambda = 0$). Set the learning rate, η, and mini-batch size, n, for each worker as*

$$\eta = \sqrt{\frac{f_0 - f^*}{||L||_1 K}}, \qquad n = K.$$

*Assume that a fraction $\alpha < 1 - 1/2p$ of the M workers behave adversarially according to **Definition 2**. Then majority vote converges at rate:*

$$\left[\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}(||g_k||_1)\right]^2 \leq \frac{4}{\sqrt{N}}\left[\frac{1}{2\sqrt{2}}\frac{1}{p(1-\alpha)-\frac{1}{2}}\frac{||\sigma||_1}{\sqrt{M}} + \sqrt{||L||_1(f_0 - f^*)}\right]^2$$

where $p = \mathbb{P}(sg(\widetilde{g}_t) = sg(g_t))$ and $N = K^2$ is the total number of stochastic gradient calls per worker up to step K.

Proof. Denote by M the total number of workers, by α the proportion of Byzantine workers, by Z_t the number of correct bits received by the server at iteration t and by Z_t^g the number of bits sent by healthy workers and received by the server at iteration t.

In the worst case, Byzantine adversaries are omniscient and know about the true sign of the gradient. Therefore, they oppose to it. In this case, only healthy workers can help finding the true sign of the gradient. So $\mathbb{P}(Z_t \leq \frac{M}{2}) \leq \mathbb{P}(Z_t^g \leq \frac{M}{2})$.

Now,

$$Z_t^g \hookrightarrow \textit{Binomial}((1-\alpha)M, p)$$

where $p = \mathbb{P}(\text{sg}(\widetilde{g}_t) = \text{sg}(g_t))$.

Thus,

$$\begin{aligned}
\mathbb{P}(Z_t \leq \frac{M}{2}) &\leq \mathbb{P}(Z_t^g \leq \frac{M}{2}) && \text{(Worst case)}\\
&= \mathbb{P}(\mathbb{E}(Z_t^g) - Z_t^g \geq \mathbb{E}(Z_t^g) - \frac{M}{2}) && \mathbb{E}(Z_t^g) > \frac{M}{2}\\
&\leq \frac{1}{1 + \frac{\left(\mathbb{E}(Z_t^g) - \frac{M}{2}\right)^2}{\text{Var}(Z_t^g)}} && \text{(Cantelli's inequality)}\\
&\leq \frac{1}{2}\frac{\sqrt{\text{Var}(Z_t^g)}}{\mathbb{E}(Z_t^g) - \frac{M}{2}} && 1 + x^2 \geq 2x\\
&= \frac{1}{2}\frac{\sqrt{p(1-p)(1-\alpha)}}{p(1-\alpha) - \frac{1}{2}}\frac{1}{\sqrt{M}}\\
&\leq \frac{1}{2}\frac{\sqrt{1-p}}{p(1-\alpha) - \frac{1}{2}}\frac{1}{\sqrt{M}} && p(1-\alpha) \leq 1\\
&\leq \frac{1}{2\sqrt{2}}\frac{1}{p(1-\alpha) - \frac{1}{2}}\frac{1}{S_i\sqrt{M}} && (\textbf{Lemma 1bis})
\end{aligned}$$

The next stage of the proof relies on the same elements as in [4], that is, we compute a telescoping sum over the iterations, and we use our bound to majorize one of the terms. □

Remark 1. The condition $\mathbb{E}(Z_t^g) > \frac{M}{2}$ can be written as $\alpha < 1 - \frac{1}{2p}$ and implies that $\alpha < \frac{1}{2}$ and $p > \frac{1}{2}$.

Remark 2. The probability of failure in estimating the true sign of the gradient decreases as the number of workers M increases, when α is fixed.

Remark 3. If $p = 1$, we do obtain a probability of failure equal to zero. This is coherent with the fact that healthy workers do not make mistakes and are in majority.

Finally, we see that our bound is more general than the one from **Theorem 2**, however we had to introduce a new parameter p. This value measures the ability of estimating the true sign of the gradient and it can depend on many things, such as the dataset.

4 Our implementation

We implemented a basic distributed SGD as well as SIGNUM in Python. We decided to follow the *PyTorch*[5] support and we implemented classes for our datasets, optimizers and neural networks with distributed support[6]. Experiments can be run through command lines for logistic and linear regressions with simple feed-forward networks, MNIST[7] with two different neural networks and ImageNet[8] with ResNet18 or ResNet50[9].

Then, we designed a Byzantine strategy for both the distributed SGD and SIGNUM algorithms. In the case of distributed SGD, one Byzantine worker is enough to stop the learning process. This adversary can invert the sum of the gradients of all the other workers and thus eliminate the gradient. In the case of SIGNUM, the Byzantine adversaries will need to collude. First, they collect the gradients signs of all the other workers. Then, they compute the local sum of these signs to estimate if they can beat the healthy workers. Let f be the number of Byzantine adversaries and s^h the sum of gradients signs for healthy workers. For each coordinate i, if $s_i^h > f$ or $s_i^h < -f$, the Byzantines cannot invert the final sign, therefore they just oppose to the other workers. If $f >= s_i^h >= 0$, $f - s_i^h$ Byzantine workers will send -1, then the other Byzantine adversaries will send -1 and $+1$ one after another, starting with -1, to try to kill the sign. If $0 > s_i^h >= -f$, they do the same starting with $+1$. Clearly, the resulting learning process will depend on the result of the operation $\text{sg}(0)$. In *PyTorch*, the operation results in $\text{sg}(0) = 0$.

In order to optimize the optimizer steps, we used several tricks. We considered that, amongst the Byzantine adversaries, one is selected to be the Byzantine server and it gathers the gradients signs from the healthy workers. Then, in order to limit the number of communications between processes, the Byzantine server sends the whole Byzantine strategy summed to f while the other Byzantine workers send empty tensors. By doing so and by devising operations on *PyTorch* tensors, the computation time of the optimizer steps with and without Byzantine adversaries are similar. This allows for faster training of the models, as we ran our experiments under CPU.

5 Experimental results

The experimental parameters are as follows: $\eta = 10^{-3}$ for distributed SGD and decreases by a factor 10 every 30 steps; $\eta = 10^{-4}$ (10^{-5} for MNIST) for SIGNSGD and decreases by a factor 10 every 30 steps; $\eta = 10^{-4}$ (10^{-5} for MNIST) and $\beta = 0.9$ for SIGNUM and η decreases by a factor 10 every 30 steps. The seed was 8005 across all experiments. We compared the efficiency of the optimizers on basic datasets which are linear and logistic regressions along with simple feed-forward networks. It is still possible to run experiments on more complex datasets such as MNIST, however they will run on CPU and should take longer.

Firstly, **Figure 2** shows the evolution of accuracy and loss for a logistic regression problem, when there are variable numbers of blind adversaries inverting their gradient signs. From this graph, we can deduce that blind adversaries do not prevent the models from learning. The SIGNSGD algorithm allows to maintain a better accuracy overall with the number of blind adversaries increasing, and SIGNUM reduces their effect even more. Still, it is important to keep in mind that our dataset and model are basic, therefore the learning process is globally easy.

Then, **Figure 3** shows the evolution of loss and accuracy when there are variable numbers of Byzantine adversaries. Byzantine adversaries intercept the gradients of the workers and deploy a strategy. Recall that in the case of distributed SGD, a Byzantine can send arbitrary vectors and thus stop the learning process, and in the case of SIGNSGD, Byzantine adversaries are limited to sending signs, therefore they try to bring the aggregation to zero. Here, we see that our Byzantine strategy does not break SIGNSGD. Even more, the SIGNUM version of the algorithm resists to our attacks.

The second experiment that we ran was on MNIST dataset. This dataset is much more complex than a logistic regression problem, as it is an image classification task. In the case of blind adversaries, **Figure 4** shows that distributed SGD can resist to the attacks. However with increasing proportion of blind adversaries such as 30% and 40%, the learning process takes much more time. SignSGD, and more efficiently Signum, allow to reduce the effect of blind adversaries and to achieve good accuracy, although smaller than the accuracy reached with distributed SGD.

Lastly, **Figure 5** shows the evolution of loss and accuracy on MNIST when there are variable numbers of Byzantine adversaries. When there are more than 30% of Byzantine adversaries, it appears that SignSGD is less efficient, however it still allows to learn from the data with decreasing accuracy. Finally, Signum is much more fault-tolerant than SignSGD, as the algorithm allows to achieve an accuracy similar to the one with distributed SGD, even with a proportion of Byzantine adversaries close to 50%.

6 Conclusion

All in all, we have illustrated on simple examples that our new and more general theoretical bound from **Theorem 2bis** is verified in practice. However, more complex models and data might lead to more difficult situations for the SIGNUM algorithm. Therefore, it might be needed to devise other algorithms to counter specific situations. Furthermore, we have observed that the SIGNUM algorithm implies an overfitting more frequently than other optimizers, since the norm of the aggregation made by the server is not proportional to the loss.

Further research has been conducted on SIGNUM. We link the interested reader to two other publications on the subject, namely to Jin et al.[10] where the authors prove a more precise theoretical bound for Byzantine workers than ours when the fourth assumption is not verified, and to Sohn et al.[11] where the authors devise a new algorithm to protect SIGNSGD from Byzantine attacks with intermediary servers and prove an associated theoretical bound more precise and asymptotically similar to ours.

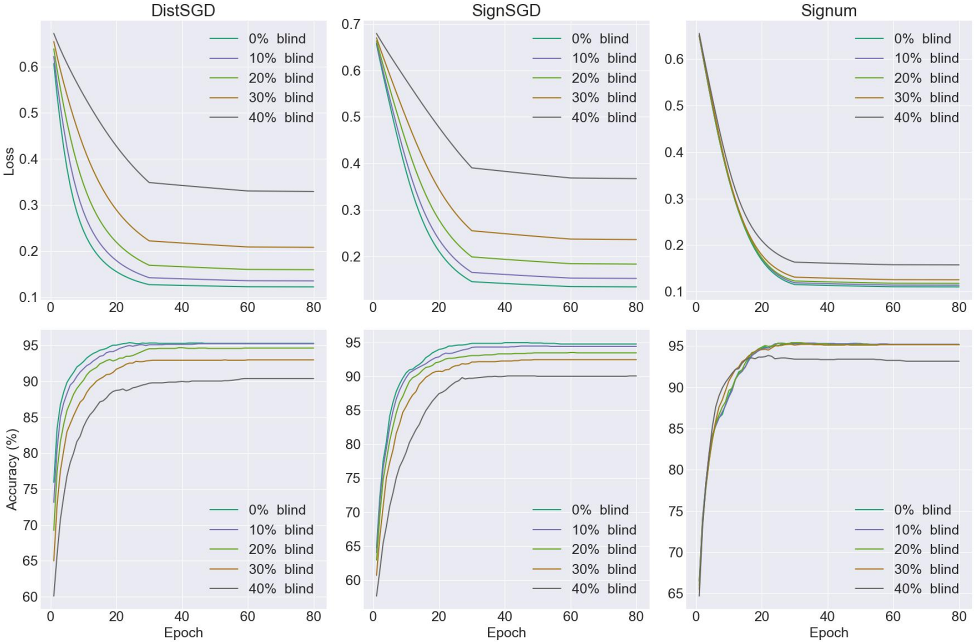

Figure 2: Evolution of loss and accuracy for logistic regression with blind adversaries.

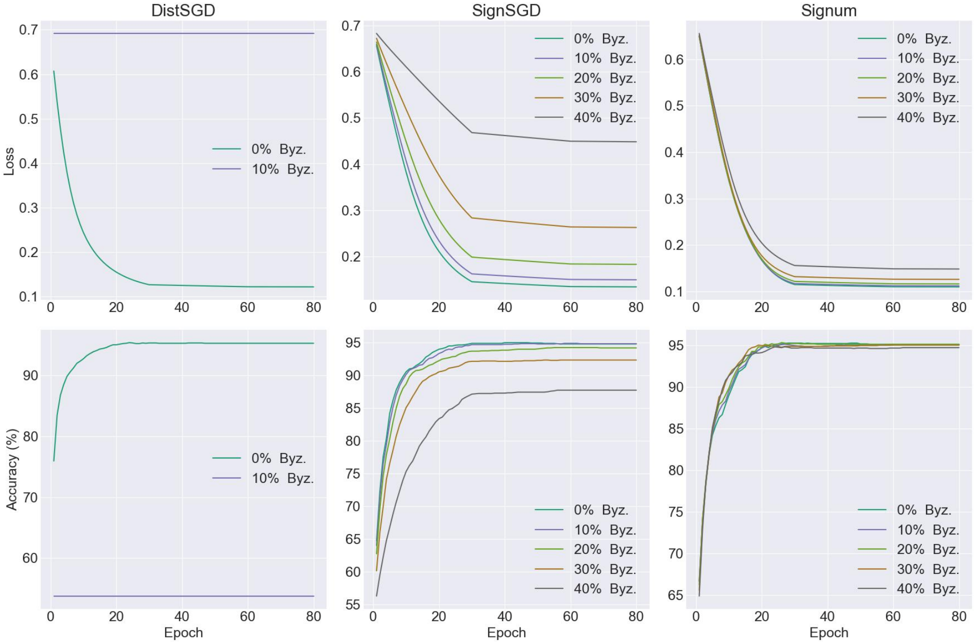

Figure 3: Evolution of loss and accuracy for logistic regression with Byzantine adversaries.

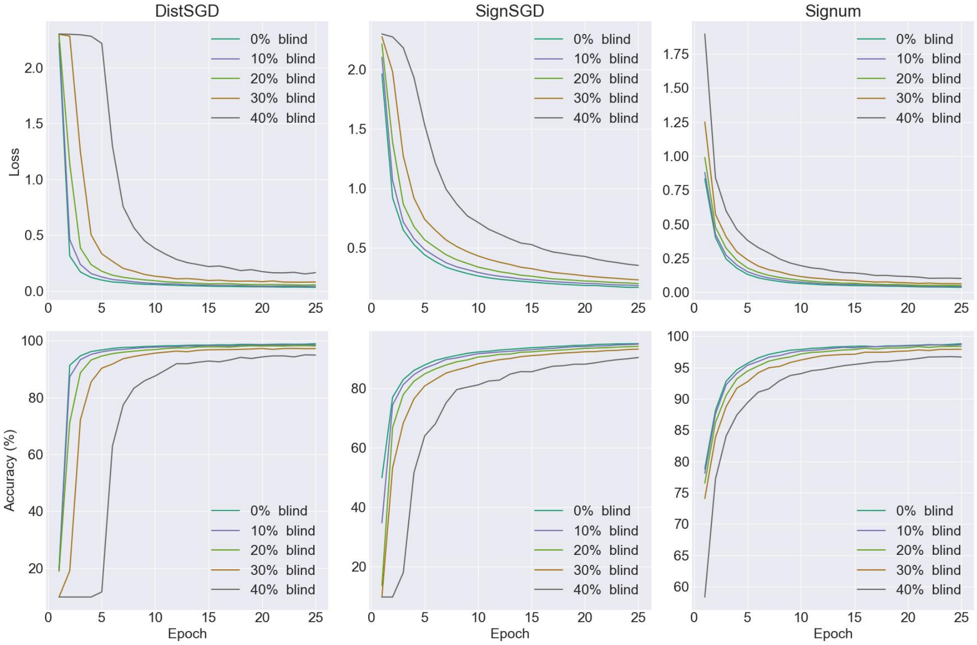

Figure 4: Evolution of loss and accuracy for MNIST dataset with blind adversaries.

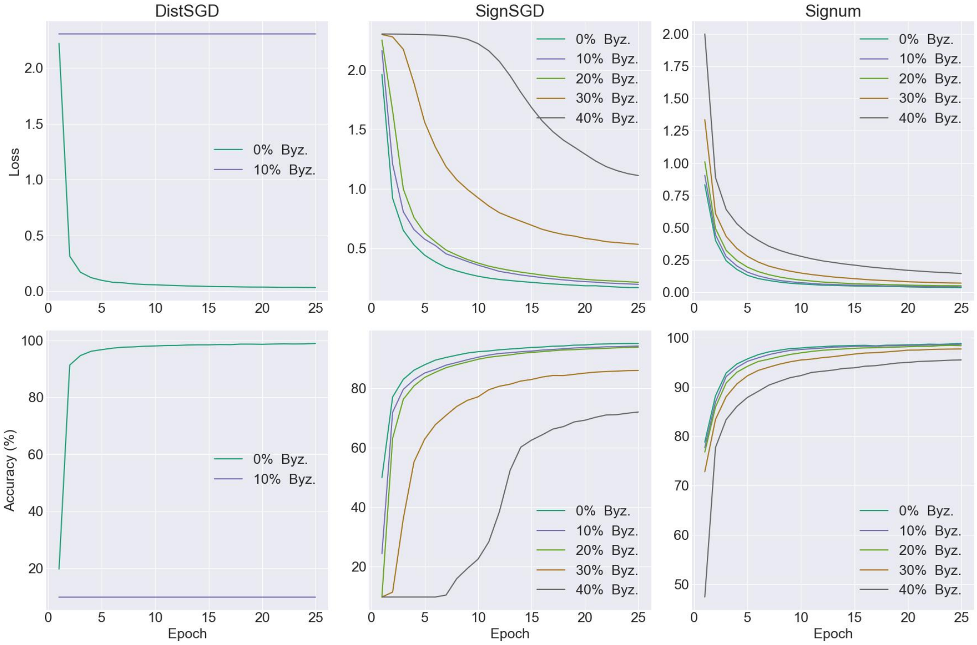

Figure 5: Evolution of loss and accuracy for MNIST dataset with Byzantine adversaries.

References

[1] Léon Bottou. *Online Learning and Stochastic Approximations*. 1998.

[2] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui and Julien Stainer. *Machine Learning with Adversaries: Byzantine Tolerant Graident Descent*. 2017.

[3] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli and Anima Anandkumar. *SignSGD: Compressed Optimisation for Non-Convex Problems.* August 2018.

[4] Jeremy Bernstein, Jiawei Zhao, Kamyar Azizzadenesheli and Anima Anandkumar. *SignSGD with Majority Vote is Communication Efficient and Fault Tolerant.* February 2019.

[5] Adam Paszke et al. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. 2019. Advances in Neural Information Processing Systems, vol. 32, pp. 8024-8035.

[6] Li et al. *PyTorch Distributed: Experiences on Accelerating Data Parallel Training*. 28 June 2020.

[7] Li Deng. *The mnist database of handwritten digit images for machine learning research*. 2012. IEEE Signal Processing Magazine, vol. 29, n°6, pp. 141-142.

[8] Jia Deng, Wei Dong, Richard Socher, Li-jia Li, Kai Li and Li Fei-Fei. *Imagenet: A large-scale hierarchical image database*. 2009. IEEE Conference on computer vision and pattern recognition, pp. 248-255.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. *Deep Residual Learning for Image Recognition*. December 2015.

[10] Richeng Jin, Yufan Huang, Xiaofan He, Huaiyu Dai and Tianfu Wu. *Stochastic-Sign SGD for Federated Learning with Theoretical Guidelines*. September 2021.

[11] Jy-yong Sohn, Don-Jun Han, Beongjun Choi and Jaekyun Moon. *Election Coding for Distributed Learning: Protecting SignSGD against Byzantine Attacks*. October 2020.