This paper talks about using error feedback to fix issues encountered while using SignSGD and other compression techniques. It also provides a glimpse into where SignSGD fails to converge and how to overcome this. We start by understanding why we need to use gradient compression. Gradient Compression reduces the bandwidth of the information which is to be backpropagated making training more scalable and efficient without significant loss either in convergence rate or accuracy. The bigger the model and more the number of fully connected components more the speed up with gradient compression.

Further, we investigate how gradient compression works. Gradient compression uses the approach of delaying the synchronization of weight updates that are small. Although small weight updates might not be sent for that batch, this information is not discarded. Once the weight updates for this location accumulate to become a larger value, they will be propagated. Since there is no information loss, but only delayed updates, it does not lead to a significant loss in accuracy or convergence rate. Methods that perform gradient updates only based on the sign of the gradient at each step have recently gained popularity for training deep learning models. This is commonly known as SignSGD, signed stochastic gradient.

The reason why vanilla SignSGD does not converge is it loses out on two critical inputs the magnitude and the direction of the gradient. SignSGD fails to converge even simple non-convex functions however it has been shown that with optimization it can perform comparably to SGD. After using distributive training majority vote algorithm and the classical optimization theory where local information in gradient tells you about global information about the direction to the minima. And if the problem is nonconvex, the performance is better with SignSGD. They scale the signed vector by the norm of the gradient to ensure the magnitude of the gradient is not forgotten. They locally store the difference between the actual and compressed gradient. Finally, they add it back into the next step so that the correct direction is not forgotten. This is called EF-SIGNSGD.

Algorithm 1 EF-SIGNSGD (SIGNSGD with Error-Feedb.)

Input: learning rate γ, initial iterate $\mathbf{x}_0 \in \mathbb{R}^d$, $\mathbf{e}_0 = \mathbf{0}$
for $t = 0, \ldots, T-1$ **do**
  $\mathbf{g}_t := \text{stochasticGradient}(\mathbf{x}_t)$
  $\mathbf{p}_t := \gamma \mathbf{g}_t + \mathbf{e}_t$ ▷ *error correction*
  $\Delta_t := (\|\mathbf{p}_t\|_1 / d)\, \text{sign}(\mathbf{p}_t)$ ▷ *compression*
  $\mathbf{x}_{t+1} := \mathbf{x}_t - \Delta_t$ ▷ *update iterate*
  $\mathbf{e}_{t+1} := \mathbf{p}_t - \Delta_t$ ▷ *update residual error*
end for

By incorporating error feedback, SIGNSGD always converges for non-convex smooth functions and recovers at the same rate as SGD. Unlike SIGNSGD, EF-SIGNSGD converges to the max-margin solution in over-parameterized least squares. EFSIGNSGD generalizes much better than SIGNSGD.

Experimental results show that EF-SIGNSGD is faster than SGDM on train, EF-SIGNSGD almost matches SGDM on test, SIGNSGD performs poorly for small batch-sizes. The performance of SIGNSGD is always worse than EF-SIGNSGD indicating that scaling is insufficient and that error feedback is crucial for performance. Further, all metrics (train and test, loss and accuracy) increasingly become worse as the batch-size decreases indicating that SIGNSGD is indeed a brittle algorithm.

References: *Karimireddy, Sai Praneeth, et al. Error feedback fixes signsgd and other gradient compression schemes. ICML 2019*