Qsparse-local-SGD: Distributed SGD with Quantization, Sparsification, and Local Computations
Debraj Basu, Deepesh Data, Can Karakus, Suhas Diggavi

**Motivation**
Communication bottlenecks in exchange of stochastic gradients for distributed training of high dimensional models over bandlimited networks

**State of the Art:**
1) Quantization (Q): Stochastic such as QSGD, TemGrad, etc. or deterministic such as signSGD
2) Sparsification (Comp_k): Selecting TopK or Random k elements
3) Increased Local Computations: Mini batch and local iterations

**Composition of Quantizer and Sparsifier**
- Step 1. Select the Top k or Random k elements of the d dimensional vector v to be transmitted
- Step 2. Quantize the sparse vector using aforementioned quantizers

Ex. 1: \[
\frac{Q_{\text{SGD}}(\text{Topk}(v))}{1 + p_{\text{k},k}} \text{ has a compression factor of } \gamma = \frac{k}{d(1 + p_{\text{k},k})} \in (0,1]
\]
Ex. 2: \[
\frac{||\text{Topk}(v)||_{1}}{k} \text{ has comp. factor } \max \{\frac{1}{d}, \frac{k}{\sqrt{d}||\text{Topk}(v)||_{2}}\}
\]

**Qsparse-SGD**
At time t on worker r,
- Use a composed operator \(Q_{\text{Comp}}\)
- Compress the previous error to be compensated and the new mini batch update together

\[
g^{(t)}_r \leftarrow Q_{\text{Comp}} \left( m^{(t)}_r + \eta t \nabla f(r)(x_t) \right)
\]
Send \(g^{(t)}_r\) to the master
- Master will compute \(x_{t+1} = x_t - \frac{1}{R} \sum_{r=1}^{R} g^{(t)}_r\) and broadcast to the workers.
- Store the compression error in memory\(l^{(t)}\)

\[
m^{(t)}_{t+1} \leftarrow m^{(t)}_r + \eta t \nabla f(r)(x_t) - g^{(t)}_r
\]

**Synchronous Operation**
Worker Node 1
Worker Node 2
Worker Node k

**Qsparse-SGD:** Sparse Quantized updates together with error feedback at each worker locally. Synchronization schedule same as distributed vanilla SGD

**Asynchronous Operation**

**Theorem 3** (Non-convex). \(H \text{ must be } O(\sqrt{\gamma T}/(bR)^{1/4})\) for convergence at a rate of \(O(1/\sqrt{(bR)})\)

**Theorem 4** (Strongly-convex). \(H \text{ must be } O(1/(bR)^{1/4})\) for convergence at a rate of \(O(1/bR)\)

**Experiments**

Multiclass logistic classifier trained on MNIST:
- Training loss vs epochs
- Plots of Training Loss.
- Training multiclass logistic classifier on MNIST dataset, \(b=8, R=15, d=7850, k=40\)
- Training ResNet-50 on ImageNet dataset, \(b=256, R=8, d=25,610,216, k=99,400\)
- We recover convergence at rates matching distributed vanilla SGD for non-convex and convex objectives with 15-20x savings in communicated bits over state-of-the-art and 1000x savings over full precision SGD

**Publications**