A Double Residual Compression Algorithm for Efficient Distributed Learning

Xiaorui Liu
Joint work with Yao Li, Jiliang Tang, Ming Yan

Michigan State University

June 15th, AISTATS 2020
Distributed Learning

Problem

\[ \minimize_{x \in \mathbb{R}^d} f(x) + R(x) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_{\xi \sim D_i} [\ell(x, \xi)] + R(x) \]

\[ := f_i(x) \]

Data is partitioned at different worker machines
Parallel SGD

- Gradient reduce & model (or averaged gradient) broadcasting
- Widely supported and used in PyTorch/TensorFlow/MXNET ...
Parallel SGD

Hopefully

Minibatch 1
Compute gradient | Exchange gradient | Update params
Minibatch 2
Minibatch 3

D. Alistarh’s Tutorial at PODC 2018
Parallel SGD

Big model

D. Alistarh’s Tutorial at PODC 2018
Parallel SGD

Big network

Time to Train Model

- Communication
- Computation

Days

Number of GPU Nodes

2 4 8 16 32 64

9.6 days 5 days 3.2 days 2.4 days 2.5 days 3.1 days

D. Alistarh’s Tutorial at PODC 2018
Parallel SGD

Gradient compression

1Bit SGD / QSGD / Terngrad / ECQ-SGD / DIANA / MEM-SGD …

Server

Communication <50%

$\begin{align*}
\text{Worker 1} & : f_1(x) \\
\text{Worker 2} & : f_2(x) \\
\vdots & \\
\text{Worker n-1} & : f_{n-1}(x) \\
\text{Worker n} & : f_n(x)
\end{align*}$

Most algorithms either

(1) directly broadcast the full-precision model or
(2) allgather the compressed gradients
DOuble REsidual compression (DORE)

Reduce gradient & Broadcast model

Worker 1  Worker 2  Worker n-1  Worker n

Worker side: gradient residual compression + running average
Server side: model residual compression + error compensation
Algorithm DORE

**DORE with R(X)**

**Input:** Step size $\alpha, \beta, \gamma, \eta$, initialize $h^0 = h_i^0 = 0^d$, $\hat{x}^0_i = \hat{x}^0$, $\forall i \in \{1, \ldots, n\}$.

**for** $k = 1, 2, \cdots, K - 1$ **do**

**For each worker** $\{i = 1, 2, \cdots, n\}$:

- Sample $g_i^k$ such that $E[g_i^k | \hat{x}_i^k] = \nabla f_i(\hat{x}_i^k)$

- Gradient residual: $\Delta_i^k = g_i^k - h_i^k$

- Compression: $\hat{\Delta}_i^k = Q(\Delta_i^k)$

- $h_i^{k+1} = h_i^k + \alpha \hat{\Delta}_i^k$

- $\{ \hat{g}_i^k = h_i^k + \hat{\Delta}_i^k \}$

- Sent $\hat{\Delta}_i^k$ to the master

- Receive $\hat{q}_i^k$ from the master

- $\hat{x}_i^{k+1} = \hat{x}_i^k + \beta \hat{q}_i^k$

**For the master**:

- Receive $\{\hat{\Delta}_i^k\}$ from workers

- $\hat{\Delta}_i^k = 1/n \sum_{i}^n \hat{\Delta}_i^k$

- $\hat{g}_i^k = h_i^k + \hat{\Delta}_i^k \{ = \frac{1}{n} \sum_{i}^n \hat{g}_i^k \}$

- $x^{k+1} = \text{prox}_{\gamma R}(\hat{x}_i^k + \gamma \hat{g}_i^k)$

- $h_i^{k+1} = h_i^k + \alpha \hat{\Delta}_i^k$

- Model residual: $q_i^k = x_i^{k+1} - \hat{x}_i^k + \eta e_i^k$

- Compression: $\hat{q}_i^k = Q(q_i^k)$

- $e_i^{k+1} = q_i^k - \hat{q}_i^k$

- $\hat{x}_i^{k+1} = \hat{x}_i^k + \beta \hat{q}_i^k$

- Broadcast $\hat{q}_i^k$ to workers

**Output:** $\hat{x}^K$ or any $\hat{x}_i^K$
Worker side: gradient residual compression + running average

Intuition 1: issue of simple gradient compression

\[ x = x^* - \frac{\gamma}{n} \sum_{i=1}^{n} Q(\nabla f_i(x^*)) \]

\[ = x^* - \frac{\gamma}{n} \sum_{i=1}^{n} \nabla f_i(x^*) + \frac{\gamma}{n} \sum_{i=1}^{n} (\nabla f_i(x^*) - Q(\nabla f_i(x^*))) \]

- The convergence requires either
  (1) diminishing stepsize \( \gamma \) or (2) diminishing compression error

- Error compensation on the worker side doesn’t solve this issue
Worker side: gradient residual compression + running average

Intuition 2: Gradient for smooth function changes smoothly

- Keep a state \( h \) to track the local gradient
- Residual between current gradient and \( h \) vanishes
- Recover the estimated gradient on server side

\[
x = x^* - \frac{\gamma}{n} \sum_{i=1}^{n} \left( h_i + Q(\nabla f_i(x^*) - h_i) \right)
\]

\[
= x^* - \frac{\gamma}{n} \sum_{i=1}^{n} \nabla f_i(x^*) + \frac{\gamma}{n} \sum_{i=1}^{n} (\nabla f_i(x^*) - h_i - Q(\nabla f_i(x^*) - h_i))
\]

\[
\text{C-contraction compressor: } \mathbb{E}\|v - Q(v)\|^2 \leq C\|v\|^2, \quad \forall v \in \mathbb{R}^d
\]

Mishchenko et al. 19'}
Worker side: gradient residual compression + running average

Intuition 3: Achieve vanishing residual by running average

\[
h_{i+1}^k = (1 - \alpha)h_i^k + \alpha(h_i^k + Q(\nabla f_i(x^k) - h_i^k))
= h_i^k + \alpha Q(\nabla f_i(x^k) - h_i^k)
\]

With unbiasedness compression \( \mathbb{E}Q(v) = v \), we have

\[
\mathbb{E}Qh_{i+1}^k = (1 - \alpha)h_i^k + \alpha \nabla f_i(x^k)
\]

such that \( h_i^k \to f_i(x^*) \) once \( x^k \to x^* \)

Mishchenko et al. 19’
Algorithm DORE

- **Server side**: model residual compression + error compensation

Intuition 1: model changes slowly when approaching optima
  - Model residual compression will only incur diminishing error

Intuition 2: compensate the compression error to next iteration
  - Consider the error as delay and maintain it for faster convergence

**Remark:**
To prove the convergence, most works using error compensation require the bounded gradient assumption, but DORE doesn’t.
Convergence analysis

Assumption on the compression

Assumption

The stochastic compression operator $Q : \mathbb{R}^d \rightarrow \mathbb{R}^d$ is unbiased, i.e., $\mathbb{E}Q(x) = x$ and satisfies

$$\mathbb{E}\|Q(x) - x\|^2 \leq C\|x\|^2,$$

for a nonnegative constant $C$ that is independent of $x$.

- Random Quantization
- Random Sparsification
- $P$-norm Quantization
- …
Convergence analysis

Assumption

Each worker node samples an unbiased estimator of the gradient stochastically with bounded variance, i.e., for \( i = 1, 2, \cdots, n \) and \( \forall x \in \mathbb{R}^d \),

\[
E[g_i|x] = \nabla f_i(x), \quad E\|g_i - \nabla f_i(x)\|^2 \leq \sigma_i^2,
\]

where \( g_i \) is the estimator of \( \nabla f_i \) at \( x \). In addition, we define \( \sigma^2 = \frac{1}{n} \sum_{i=1}^{n} \sigma_i^2 \).

Assumption

Each \( f_i \) is \( L \)-Lipschitz differentiable, i.e., for \( i = 1, 2, \cdots, n \) and \( \forall x, y \in \mathbb{R}^d \),

\[
f_i(x) \leq f_i(y) + \langle \nabla f_i(y), x - y \rangle + \frac{L}{2} \|x - y\|^2.
\]
Convergence analysis

<table>
<thead>
<tr>
<th>Theorem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Choose parameters such that</td>
</tr>
<tr>
<td>[ \beta = \frac{1}{C + 1}, \quad \alpha = \frac{1}{2(C + 1)}, ]</td>
</tr>
<tr>
<td>[ \gamma = \eta = \frac{1}{12L(1 + 2C/n)(1 + \sqrt{K/n})} ]</td>
</tr>
<tr>
<td>Then we have</td>
</tr>
<tr>
<td>[ \frac{1}{K} \sum_{k=1}^{K} \mathbb{E} | \nabla f(\hat{x}^k) |^2 \leq \frac{1}{K} + \frac{1}{\sqrt{Kn}}. ]</td>
</tr>
</tbody>
</table>

- Sublinear convergence to stationary points for non-convex cases
- Linear speedup w.r.t. number of workers
Convergence analysis

Assumption

Each $f_i$ is $\mu$-strongly convex ($\mu \geq 0$), i.e., for $i = 1, 2, \cdots, n$ and $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d$,

$$f_i(\mathbf{x}) \geq f_i(\mathbf{y}) + \langle \nabla f_i(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle + \frac{\mu}{2} \| \mathbf{x} - \mathbf{y} \|^2.$$  

Theorem

Choose parameters such that

$$0 < \beta \leq \frac{1}{C+1},$$

$$\frac{1-\sqrt{1-\delta}}{2(C+1)} \leq \alpha \leq \frac{1+\sqrt{1-\delta}}{2(C+1)},$$

$$\eta < \min \left( \frac{\sqrt{C^2 + 4(1-(C+1)\beta) - C}}{2C}, \frac{4\mu L}{(\mu + L)^2 \left( 1 + \frac{4C(C+1)}{n\delta} \alpha \right) - 4\mu L} \right),$$

$$\frac{\eta(\mu + L)}{2(1+\eta)\mu L} \leq \gamma \leq \frac{2}{\left( 1 + \frac{4C(C+1)}{n\delta} \alpha \right) (\mu + L)}.$$  

Then we have

$$\mathbf{V}^{k+1} \leq \rho^k \mathbf{V}^1 + \frac{(1+\eta) \left( 1 + n \frac{4C(C+1)}{n\delta} \alpha \right)}{n(1-\rho)} \beta \gamma^2 \sigma^2,$$

for some $\rho < 1$, and $\mathbf{V}^k$ measures the convergence of $q^k \rightarrow 0$, $\mathbf{x}^k \rightarrow \mathbf{x}^*$, and $h_i^k \rightarrow \nabla f_i(\mathbf{x}^*)$.  

Data Science and Engineering Lab
Convergence analysis

Theoretical comparison with related works

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Compression</th>
<th>Compress. Model</th>
<th>Linear</th>
<th>Nonconvex Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>SGD</td>
<td>No</td>
<td>No</td>
<td>✓</td>
<td>$\frac{1}{\sqrt{Kn}} + \frac{1}{K}$</td>
</tr>
<tr>
<td>QSGD</td>
<td>Grad</td>
<td>2-norm</td>
<td>N/A</td>
<td>$\frac{1}{K} + B$</td>
</tr>
<tr>
<td>MEM-SGD</td>
<td>Grad</td>
<td>k-contraction</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>DIANA</td>
<td>Grad</td>
<td>$p$-norm</td>
<td>✓</td>
<td>$\frac{1}{\sqrt{Kn}} + \frac{1}{K}$</td>
</tr>
<tr>
<td>DoubleSqueeze</td>
<td>Grad + Model</td>
<td>Bdd Variance</td>
<td>N/A</td>
<td>$\frac{1}{\sqrt{Kn}} + \frac{1}{K^{2/3}} + \frac{1}{K}$</td>
</tr>
<tr>
<td>DORE</td>
<td>Grad + Model</td>
<td>Assum. 1</td>
<td>✓</td>
<td>$\frac{1}{\sqrt{Kn}} + \frac{1}{K}$</td>
</tr>
</tbody>
</table>

Most algorithms, except DIANA and DORE, requires bounded gradient assumption and incur extra error.
Numerical experiment

Regularized Least square problem

\[ \|x_k - x^\ast\|^2 \leq \rho^k \|v^1\| \quad + \quad \frac{(1+\eta)^2(1+n\frac{4C(C+1)}{n\delta})\alpha}{n(1-\rho)} \beta \gamma^2 \sigma^2 \]

full-gradient where \( \sigma = 0 \)
Numerical experiment

Regularized Least square problem

Distance to optimum vs communication bits
Numerical experiment

Compression error

Worker side

Server side
Numerical experiment

LeNet trained on MNIST

ResNet18 trained on CIFAR10

Train loss vs communication bits
Numerical experiment

Time cost per iteration vs network bandwidth
Conclusion

• DORE reduces over 95% of the communication cost through the double residual compression;

• Provide a sublinear convergence rate in the nonconvex case and achieve linear speedup;

• Provide a linear convergence analysis to the neighborhood of the optimum for smooth and strongly convex functions;

• DORE achieves state-of-art performance both theoretically and empirically;

• We hope to see more applications or extensions of DORE in bandwidth limited settings such as federated learning.