HSE Optimization course project report: Comparison of Local SGD and Minibatch SGD algorithms

Evgeny Gurov

1 Introduction

Local SGD and Minibatch SGD are two standard optimization algorithms widely used in machine learning and other practical tasks requiring parallelism. However, developing a theoretical understanding of their limitations and properties remains a complex challenge, even in smooth convex cases. This project is a study of the paper "Is Local SGD Better than Minibatch SGD?" by Woodworth et al., where the authors, in 2020, sought to bridge the gap in the theoretical understanding of the Local SGD algorithm. They highlighted its advantages and drawbacks compared to the well-studied Minibatch SGD algorithm and identified the regimes in which each method performs better. To explore their findings and experiment with parameters, several numerical experiments were conducted based on the results presented by the authors.

2 Preliminaries

In this section, I a concise overview of the key concepts underlying the problem being addressed will be provided, as well as the algorithms employed to solve it.

2.1 Problem

The following stochastic convex optimization problem is considered:

$$\min_{x \in \mathbb{R}^d} F(x) := \mathbb{E}_{z \sim D}\left[f(x; z)\right]. \tag{1}$$

The focus is on objectives F that are H-smooth, either (general) convex or λ-strongly convex, with a minimizer $x^* \in \arg\min_x F(x)$ satisfying $\|x^*\| \leq B$. ∇f is assumed to have uniformly bounded variance, i.e.,

$$\sup_x \mathbb{E}_{z \sim D} \|\nabla f(x; z) - \nabla F(x)\|^2 \leq \sigma^2. \tag{2}$$

$\mathcal{F}(H, \lambda, B, \sigma^2)$ is used to refer to the set of all pairs (f, D) which satisfy these properties. [1]

2.2 Local SGD algorithm

Local SGD employs M parallel workers that, during each of R rounds, independently perform K steps of SGD starting from a shared initial iterate. After completing these steps, the workers communicate and average their iterates to produce a new common iterate, which serves as the starting point for the next round. Each worker computes $T = KR$ stochastic gradients and performs KR SGD steps locally. Across all workers, this results in a total of $N = KRM$ stochastic gradients computed (equivalent to $N = KRM$ data samples processed), with communication occurring once every K steps of computation.

[1] An important instance of (1) is a supervised learning problem where $f(x; z) = \ell(\langle x, \phi(z)\rangle, \text{label}(z))$ is the loss on a single sample. When $|\ell'(x)|, |\ell''(x)| \leq 1$ (referring to derivatives with respect to the first argument), we have $H \leq |\ell''(x)| \|\phi(z)\|^2 \leq \|\phi(z)\|^2$ and also $\sigma^2 \leq \|\nabla f\|^2 \leq |\ell'(x)|^2 \|\phi(z)\|^2 \leq \|\phi(z)\|^2$. Thus, assuming that the upper bounds on $\ell'(x)$ and $\ell''(x)$ are comparable, the relative scaling of parameters which is considered as most "natural" is $H \approx \sigma^2$.

Algorithm 1 Local SGD Algorithm

```
1: Input: Number of workers $M$, number of communication rounds $R$, local steps $K$, learning rate $\eta$
2: Initialize parameters $\mathbf{x}_0$ (shared across all workers)
3: for $r = 0$ to $R - 1$ do
4:     for each worker $m = 1$ to $M$ in parallel do
5:         Set local parameters: $\mathbf{x}_r^m \leftarrow \mathbf{x}_r$
6:         for $k = 1$ to $K$ do
7:             Compute stochastic gradient $\mathbf{g}_k^m = \nabla f(\mathbf{x}_{k-1}^m; z_k^m)$ using local data $z_k^m$
8:             Update local parameters:
                   $\mathbf{x}_k^m = \mathbf{x}_{k-1}^m - \eta \mathbf{g}_k^m$
9:         end for
10:    end for
11:    Average the local parameters across all workers:
           $\mathbf{x}_{r+1} = \frac{1}{M} \sum_{m=1}^{M} \mathbf{x}_K^{(m)}$
12: end for
13: Output: Final parameters $\mathbf{x}_R$
```

2.3 Minibatch SGD algorithm

Minibatch SGD can be implemented within the same computational and communication framework as follows: In each round, compute K stochastic gradient estimates at the current iterate on each machine, then average all KM estimates to obtain a single gradient estimate. This implementation results in minibatch SGD taking R stochastic gradient steps, with each step using a minibatch of size KM.

Algorithm 2 Minibatch SGD

```
1: Input: Number of workers $M$, number of rounds $R$, local steps $K$, minibatch size $KM$, learning rate $\eta$
2: Initialize parameters $\mathbf{x}_0$ (shared across all workers)
3: for $r = 0$ to $R - 1$ do
4:     For each worker $m = 1$ to $M$ in parallel:
5:     for $k = 1$ to $K$ do
6:         Compute the stochastic gradient:
               $\mathbf{g}_k^{(m)} = \nabla f(\mathbf{x}_{k-1}^{(m)}; z_k^{(m)})$
7:     end for
8:     Average all $KM$ gradient estimates:
           $\mathbf{g}_r = \frac{1}{KM} \sum_{m=1}^{M} \sum_{k=1}^{K} \mathbf{g}_k^{(m)}$
9:     Update parameters:
           $\mathbf{x}_{r+1} = \mathbf{x}_r - \eta \mathbf{g}_r$
10: end for
11: Output: Final parameters $\mathbf{x}_R$
```

2.4 Comparison method

Comparison of the algorithms is performed based on worst-case performance with respect to $F(H, \lambda, B, \sigma^2)$:

$$\varepsilon_A = \max_{(f,D)\in F(H,\lambda,B,\sigma^2)} (F(\hat{x}_A) - F(x^*))$$

The good news is that worst-case performance of Minibatch SGD for general convex objectives is tightly understood:

$$\varepsilon_{\text{MB-SGD}} = \Theta\left(\frac{HB^2}{R} + \frac{\sigma B}{\sqrt{MKR}}\right),$$

and the authors usually compare LocalSGD performance bounds with it.

Authors present comparison of known upper bounds on worst-case performance of Local SGD adapted for their case of $F(H, \lambda, B, \sigma^2)$. One interesting thing here worth noticing is that bounds mainly differ in the first term (optimization term) and are similar in the last one (statisctical term) which cannot be improved by any first-order algorithm.[2]

Table 1: Comparison of existing analyses of Local SGD for general convex functions, with constant factors and low-order terms (in the natural scaling $H \approx \sigma^2$) omitted.

Algorithm	**Performance Bound**
Minibatch SGD	$\frac{HB^2}{R} + \frac{\sigma B}{\sqrt{MKR}}$
Thumb-twiddling SGD	$\frac{HB^2}{R} + \frac{\sigma B}{\sqrt{MR}}$
Stich	$\frac{HB^2}{R^{2/3}} + \frac{HB^2}{(KR)^{3/5}} + \frac{\sigma B}{\sqrt{MKR}}$
Stich and Karimireddy	$\frac{HB^2M}{R} + \frac{\sigma B}{\sqrt{MKR}}$
Khaled et al.	$\frac{\sigma^2 M}{HR} + \frac{H^2B^2+\sigma^2}{H\sqrt{MKR}}$

3 Quadratic objectives

For quadratic objectives authors improved upper bound for LocalSGD significantly exploiting quite general theory for the class of linear update algorithms.

Definition 1 (Linear update algorithm) *We say that a first-order optimization algorithm is a linear update algorithm if, for fixed linear functions $L_1^{(t)}$, $L_2^{(t)}$, the algorithm generates its $t+1$-th iterate according to*

$$x_{t+1} = L_2^{(t)}\big(x_1, \ldots, x_t, \nabla f\big(L_1^{(t)}(x_1, \ldots, x_t); z_t\big)\big).$$

This family captures many standard first-order methods including SGD, which corresponds to the linear mappings $L_1^{(t)}(x_1, \ldots, x_t) = x_t$ and $x_{t+1} = x_t - \eta_t \nabla f(x_t; z_t)$.

Theorem 1 *Let A be a linear update algorithm which, when executed for T iterations on any quadratic $(f, D) \in F(H, \lambda, B, \sigma^2)$, guarantees $\mathbb{E}F(x_T) - F^* \le \varepsilon(T, \sigma^2)$. Then, local-$A$'s averaged final iterate $\bar{x}_{KR} = \frac{1}{M}\sum_{m=1}^{M} x_{KR}^m$ will satisfy $\mathbb{E}F(\bar{x}_{KR}) - F^* \le \varepsilon(KR, \frac{\sigma^2}{M})$.*

To rephrase Theorem 1, on quadratic objectives, local-A is in some sense equivalent to KR iterations of A with the gradient variance reduced by a factor of M. This gives quite satisfying

Corollary 1 *For any quadratic $(f, D) \in F(H, \lambda = 0, B, \sigma^2)$, there is a constant c such that local-SGD returns a point $\hat{x}$ such that*

$$\mathbb{E}F(\hat{x}) - F^* \le c\left(\frac{HB^2}{KR} + \frac{\sigma B}{\sqrt{MKR}}\right).$$

One should notice that the denominator of the first term is KR versus R for the Minbatch SGD which is a decent improvement.

4 General convex objectives

Theorem 2 *Let $(f, D) \in F(H, \lambda, B, \sigma^2)$. When $\lambda = 0$, an appropriate average of the iterates of Local SGD with an optimally tuned constant step size satisfies for a universal constant c:*

$$\mathbb{E}[F(\hat{x}) - F(x^*)] \le c \cdot \min\left(\frac{HB^2}{KR} + \frac{\sigma B}{\sqrt{MKR}} + \frac{(H\sigma^2 B^4)^{\frac{1}{3}}}{K^{\frac{1}{3}}R^{\frac{2}{3}}}, \frac{HB^2}{KR} + \frac{\sigma B}{\sqrt{KR}}\right).$$

Comparison of obtained upper bound with Minibatch SGD performance becomes quite depicting with natural scaling $H = B = \sigma^2 = 1$. In this setting, the worst-case error of Minibatch SGD is:

$$\epsilon_{\text{MB-SGD}} = \Theta\left(\frac{1}{R} + \frac{1}{\sqrt{MKR}}\right) \tag{3}$$

The guarantee for local SGD from Theorem 2 reduces to:

$$\epsilon_{\text{L-SGD}} \le O\left(\frac{1}{K^{\frac{1}{3}}R^{\frac{2}{3}}} + \frac{1}{\sqrt{MKR}}\right)$$

One should notice that the denominator for Local SGD bound $K^{\frac{1}{3}}R^{\frac{2}{3}}$ dominates the denominator of Minibatch SGD guarantee R when K is sufficiently big compared to R. This trend will be shown further in practical part. It is important to note that this happens when optimization term is not dominated by statistical one, i.e. M is sufficiently big compared to K and R. But in regimes where Local SGD bound is inferior it is not guaranteed that the actual performance will be worse than for Minibatch SGD since we deal with the upper bound. To clarify this proceed to the next section.

5 Minibatch SGD Can Outperform Local SGD

Theorem 3 *For $0 \le \lambda \le \frac{H}{16}$, there exists $(f, D) \in F(H, \lambda, B, \sigma^2)$ such that for any $K \ge 2$ and $M, R \ge 1$, local SGD initialized at 0 with any fixed stepsize will output a point $\hat{x}$ such that for a universal constant c,*

$$E[F(\hat{x}) - \min_x F(x)] \ge c \cdot \min\left(\frac{H^{1/3}\sigma^{2/3}B^{4/3}}{K^{2/3}R^{2/3}}, \frac{H\sigma^2}{\lambda^2 K^2 R^2}, HB^2\right) + c \cdot \min\left(\frac{\sigma B}{\sqrt{MKR}}, \frac{\sigma^2}{\lambda MKR}\right).$$

To compare this lower bound with Theorem 2 and Minibatch SGD, the general convex case is considered again where $H = B = \sigma^2 = 1$. Under these conditions, the lower bound simplifies to

$$K^{-\frac{2}{3}}R^{-\frac{2}{3}} + (MKR)^{-\frac{1}{2}}.$$

When comparing this with Theorem 2, one could observe that the upper bound is tight except for a factor of $K^{-\frac{1}{3}}$ in the optimization term. Additionally, when comparing this with the worst-case error of Minibatch SGD 3, we find that local SGD performs worse than Minibatch SGD in the worst case when K is sufficiently small relative to R. This confirmed the concerns expressed in the previous section, in general convex setting in the worst case, Minibatch SGD is indeed superior to LocalSGD when number of local steps is small.

6 Experiments

6.1 Implementation details

Local SGD and Minibatch SGD algorithms were implemented with Pytorch. Parallel execution was replaced by sequential (imitating parallel) due to big number of workers M which should be 50 or more for representative results. For each M, K, and algorithm, the constant step size was tuned (via gridsearch on a logarithmic grid) to minimize the loss after r rounds of communication, individually for each $1 \leq r \leq R$. For more details please refer to the Github repo.

6.2 Synthetic data

We generated a dataset of 50,000 points in $\mathbb{R}^{25}$, where the i-th coordinate of each point was independently sampled from a Gaussian distribution $\mathcal{N}(0, 10/i^2)$. The labels were generated based on the probability

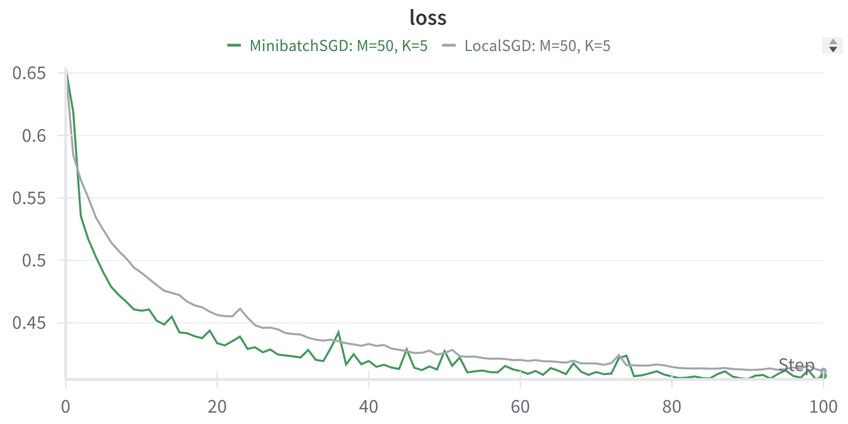
$$P[y = 1 \mid x] = \sigma\left(\min\{\langle w_1^*, x\rangle + b_1^*, \langle w_2^*, x\rangle + b_2^*\}\right),$$

where $w_1^*, w_2^* \sim \mathcal{N}(0, I_{25\times 25})$ and $b_1^*, b_2^* \sim \mathcal{N}(0, 1)$. Here, $\sigma(a) = 1/(1 + e^{-a})$ is the sigmoid function. The labels correspond to the intersection of two half-spaces, with label noise increasing near the decision boundary.

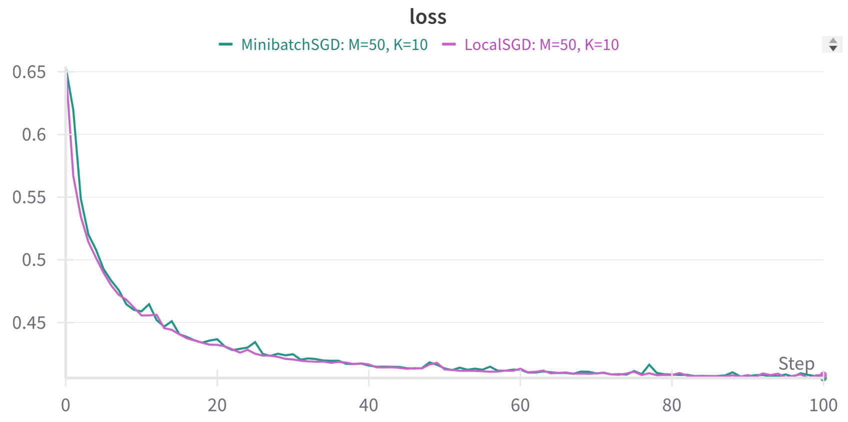
Each algorithm was used to train a linear model with a bias term by minimizing the binary cross-entropy loss over the dataset of 50,000 points, where f denotes the binary cross-entropy loss for one sample and $\mathcal{D}$ is the empirical distribution over all samples. Graphics can be found in 2. It can be clearely seen that Local SGD outperforms Minibatch SGD with bigger K (local steps) and loses when the K is relatively small. This is exactly what we have seen in the theory for convex functions in the section 4 and 5.

6.3 MNIST

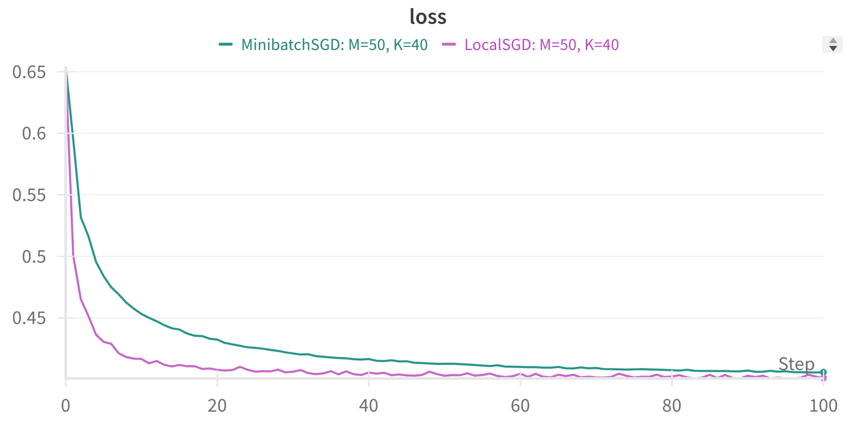
For this experiment, the simple MNIST dataset is used. Although it is relatively basic, it is still more complex than the synthetic dataset used previously. Each sample is a 28×28 image, and the classification task is now multi-class. A slightly more sophisticated model is employed here as well: a two-layer perceptron with a ReLU activation function. It is interesting to compare Local SGD and Minibatch SGD in more non-trivial scenarios, where the objective is far from being convex. Will the trend of "the larger K, the better" still hold?



(a) M = 50, K = 5: Minibatch SGD performs better.

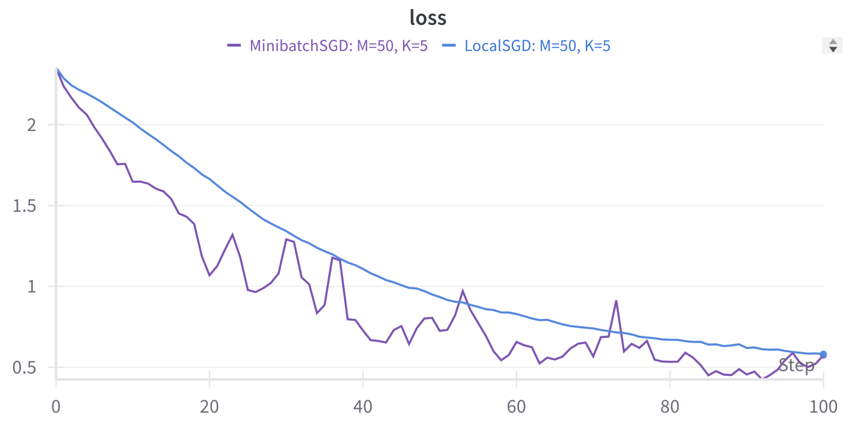


(b) M = 50, K = 10: Identical performance.

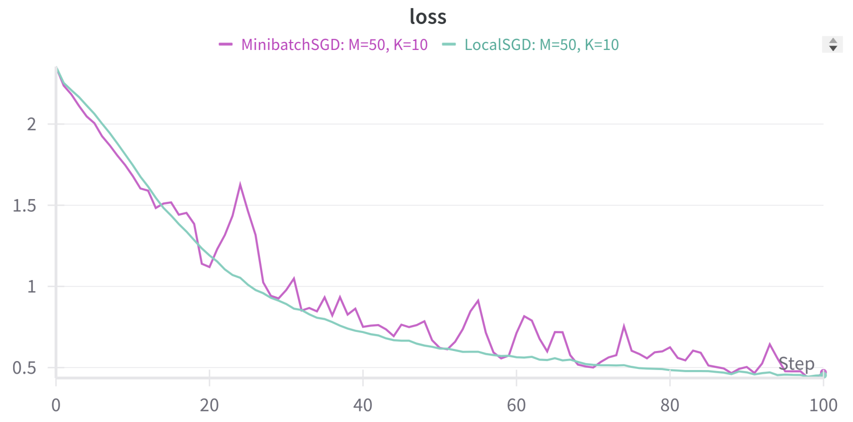


(c) M = 50, K = 40: Local SGD is superior.

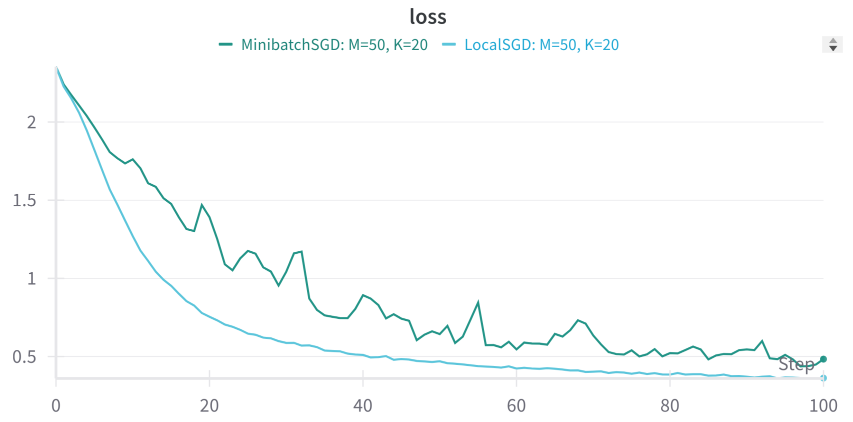
Figure 1: Local SGD and Minibatch SGD performance comparison for synthetic data. The plotted lines correspond to $\min_\eta F(x_{A,r,\eta}) = \min_\eta \mathbb{E}_{z\sim\mathcal{D}}[f(x_{A,r,\eta}, z)]$.



(a) M=50, K=5: Minibatch does better but it is very noisy due to big learning rate.



(b) M=50, K=10: Local SGD is nearly the same as Minibatch but way less noisy.



(c) M=50, K=20: Local SGD performs significantly better than Minibatch SGD.

Figure 2: Local SGD and Minibatch SGD performance comparison for MNIST data. The plotted lines correspond to $\min_\eta F(x_{A,r,\eta}) = \min_\eta \mathbb{E}_{z\sim\mathcal{D}}[f(x_{A,r,\eta}, z)]$.

References

[1] Woodworth et al., *"Is Local SGD Better than Minibatch SGD?"*, 2020

[2] Nemirovsky, Yudin, *Problem complexity and method efficiency in optimization*, 1983.