Efficient Mini-batch Training for Stochastic Optimization

Mu Li
muli@cs.cmu.edu

Joint work with Tong Zhang, Yuqiang Chen, and Alex Smola
Big data

- A lot of “big” data
  - texts
  - images
  - voices
  - videos

- Most of them are user activities
  - can be modeled as supervised learning
Convex optimization:

\[
\min_{w \in \Omega} \frac{1}{n} \sum_{i=1}^{n} \phi_i(w)
\]

For example: Risk minimization

\[
\phi_i(w) = \ell(x_i, y_i, w) + \lambda c(w)
\]

✓ \((x_i, y_i)\) are example pairs
Stochastic gradient descent (SGD)

- Process an example each time

\[
\text{for } t = 1, 2, \ldots, T \\
\text{draw a random example } i_t \\
\text{update } w_t = w_{t-1} - \eta_t \nabla \phi_{i_t}(w_{t-1})
\]

- Convergence rate \( O(1/\sqrt{T}) \)

- Sequential, hard to parallelize
Minibatch SGD

✦ One example ➾ several examples

for \( t = 1, 2, \ldots, T \)
draw a random minibatch

\[ I_t \subset \{1, \ldots, n\} \]

update

\[ w_t = w_{t-1} - \eta_t \nabla \phi_{I_t}(w_{t-1}) \]

\[ \phi_{I_t}(w) = \frac{1}{|I_t|} \sum_{i \in I_t} \phi_i(w) \]

✦ 👍 Efficient parallel/distributed implementation within a minibatch
Effects of the minibatch size

- Fix #examples $bT$:
  - minibatch size $b \uparrow$, then #iteration $T \downarrow$
  - 👍 System performance $\uparrow$
    - ✓ #synchronization $\downarrow$
    - ✓ network communication $\downarrow$
  - 👎 Convergence rate $\downarrow$
    - ✓ it is $O(1/\sqrt{bT} + 1/T)$
    - ✓ $O(1/T) \uparrow$
Our goal

Key idea:
When minibatch size ↑, sample variance ↓. Solve a more “accurate” optimization problem over each minibatch.
Observation

- Rewrite the update rule of minibatch SGD:

\[ w_t = \arg\min_{w \in \Omega} \left[ \phi_{I_t}(w_{t-1}) + \langle \nabla \phi_{I_t}(w_{t-1}), w - w_{t-1} \rangle + \frac{1}{2\eta_t} \|w - w_{t-1}\|^2 \right] \]

- 👍 Fast to solve
- 👎 Data utilization is low
  - ✓ large switching cost to the next minibatch
The proposed solution: EMSO

- Solve a conservative subproblem:

\[ w_t = \arg\min_{w \in \Omega} \left[ \phi_I(w) + \frac{\gamma_t}{2} \| w - w_{t-1} \|_2^2 \right]. \]

- 👍 achieve a full utilization of the minibatch
- 👍 avoid over utilization

exact objective over minibatch

a conservative penalty
Convergence rate

- **Minibatch SGD:** $\mathcal{O}(1/\sqrt{bT} + 1/T)$
- **EMSO:** $\mathcal{O}(1/\sqrt{bT})$

✓ only depends on the #examples

- Can be further improved when the objective is $\lambda$-strongly convex

$$
\mathcal{O}(\log T/(\lambda bT) + \lambda/(\sqrt{bT}))
$$
How to solve the subproblem

\[
    w_t = \arg\min_{w \in \Omega} \left[ \phi_I(w) + \frac{\gamma_t}{2} \| w - w_{t-1} \|^2 \right].
\]

✦ The conservative subproblem can be solve by standard technologies:
  ✓ EMSO-GD: by gradient descent
  ✓ EMSO-CD: by coordinate descent
✦ No need to solve it exactly
✦ Early stopping:
  ✓ fix the #iterations be a small constant
Convergence does not slow down with minibatch size

✦ Fix #iterations
✦ Logistic regression
✦ Dataset KDD04: 146K examples, 76 features

minibatch SGD

EMSO-GD: pass the minibatch 5 times

EMSO-CD: pass the minibatch 2 times
EMSO-CD outperforms other algorithms

- Dataset URL: 2.4M #examples, 3.2M #features
Distributed model averaging

assume $d$ machines
for each minibatch $l_t$:

1. partition $l_t$ into $\{l_{t,1}, \ldots, l_{t,d}\}$
2. machine $i$ solve the conservative subproblem on its own data partition $l_{t,i}$
3. average model via communication

$$w_t = \frac{1}{d} \sum_{i=1}^{d} w_t^{(i)}$$
EMSO-CD outperforms other algorithms

- Dataset CTR: 142M #examples, 28M #features, raw text data size 300GB
- Distributed over 12 machines
- Synchronization cost ↓ when minibatch size ↑
- Fix run time

![Graphs showing synchronization cost and objective improvement with increasing minibatch size.](image)
Scalability of EMSO-CD

- Dataset CTR
- Fix target objective
- Compare to L-BFGS

<table>
<thead>
<tr>
<th>#machines</th>
<th>time (sec)</th>
<th>speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>2439</td>
<td>1x</td>
</tr>
<tr>
<td>10</td>
<td>1367</td>
<td>1.78x</td>
</tr>
<tr>
<td>20</td>
<td>962</td>
<td>2.54x</td>
</tr>
</tbody>
</table>

![Graph comparing EMSO-CD and L-BFGS](image)
Conclusion

✦ EMSO: solve the conservative subproblem in each minibatch:

\[ w_t = \arg \min_{w \in \Omega} \left[ \phi_{I_t}(w) + \frac{\gamma_t}{2} \| w - w_{t-1} \|_2^2 \right] . \]

✦ Faster convergence rate \( \mathcal{O}(1/\sqrt{bT}) \)

✓ Does not slow down when minibatch size ↑

✦ Improve the effective workload

✓ Demonstrated in experiments with real large scale datasets