Distributed Asynchronous Optimization of Convolutional Neural Networks

William Chan and Ian Lane
{williamchan,lane}@cmu.edu

Background and Motivation
- Deep Convolutional Neural Networks (CNNs) provide state-of-the-art Acoustic Models
- CNNs model the two dimensional structure of acoustic signals in time and frequency
- CNNs are expensive to train
- Large corpora means long training times (e.g., days to weeks)

Distributed Asynchronous Optimization
- Distribute the gradient workload computation across multiple GPUs
- Independent GPU workers responsible for an independent minibatch of the dataset
- Single master GPU parameter server to asynchronously accumulate the gradients
- Convergence and scaling problem in optimization

Sparse Gradients
- Sparse gradients will lead to less fighting between independent workers
- Apply ReLU neurons, dropout and max pooling to achieve sparsity
- Cosine angle between sparse gradients are small
- \(\approx 30\%\) sparsity in gradients

Master Momentum
- Momentum is inexpensive and stabilizes the optimization procedure
- Momentum is applied at the master parameter server

\[
\begin{align*}
v_{t+1} &= \mu v_t - \eta \nabla f(\theta) \\
\theta_{t+1} &= \theta_t + v_{t+1}
\end{align*}
\]

where \(\nabla f(\theta)\) is the gradient, \(\eta > 0\) is the learning rate and \(\mu \in [0, 1]\) is the momentum coefficient.

Gradient Decay
- Stale gradients hurt our (non-convex) optimization
- Penalize the stale gradients

\[
\alpha = \beta^{j-i}
\]

where \(j - i \geq 0\) and \(\beta\) is our exponential decay parameter.

Experimental Results

![Figure 3: Speedup comparison with distributed asynchronous stochastic gradient descent. Baseline is standard SGD using 1x GPU.](image)

<table>
<thead>
<tr>
<th>Workers</th>
<th>40% FA</th>
<th>43% FA</th>
<th>44% FA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5:50 (100%)</td>
<td>14:36 (100%)</td>
<td>19:29 (100%)</td>
</tr>
<tr>
<td>2</td>
<td>3:36 (81.0%)</td>
<td>8:59 (81.3%)</td>
<td>11:58 (81.4%)</td>
</tr>
<tr>
<td>3</td>
<td>2:48 (69.4%)</td>
<td>5:59 (81.3%)</td>
<td>7:58 (81.5%)</td>
</tr>
<tr>
<td>4</td>
<td>2:05 (70.0%)</td>
<td>4:28 (81.7%)</td>
<td>6:32 (74.6%)</td>
</tr>
<tr>
<td>5</td>
<td>1:40 (70.0%)</td>
<td>3:49 (76.5%)</td>
<td>5:43 (68.2%)</td>
</tr>
</tbody>
</table>

Table 1: Time (hours) and scaling efficiency (in brackets) comparison for convergence to 40%, 43% and 44% Frame Accuracy (FA).

Conclusion
- Stable Convergence
- Efficiency Scaling
- Combine asynchronous optimization with Sparse Gradients, Master Momentum and Gradient Decay to speed up your deep learning problem