Scalable Methods for 8-bit Training of Neural Networks

, Ron Banner*, Itay Hubara*, Elad Hoffer*, Daniel Soudry
ron.banner@intel.com, {elad.hoffer, itayhubara, daniel.soudry}@gmail.com

TECHNION Israel Institute of Technology

intel AI

Background

It has been shown that 16-bit is sufficient precision for most network training but further quantization (i.e., 8-bit) results with severe degradation. Our work is the first to almost exclusively train at 8-bit without harming classification accuracy. This is addressed by overcoming two main obstacles known to hamper numerical stability: batch normalization and gradient computations:

- Batch norm: The traditional batch normalization implementation requires the computation of the sum of squares, square-root and reciprocal operations; these require high precision (to avoid zero variance) and a large dynamic range.
- Gradient computations: Our analysis indicates that the statistics of the neuron gradient g_l violates the assumptions at the crux of common quantization schemes. As such, quantizing these gradients constitutes the main cause of degradation in performance through training.

Quantized Back-Propagation

In the back-propagation algorithm we recursively calculate the gradients of the loss function $\mathcal{L}$ with respect to I_ℓ, the input of the ℓ neural layer,

$$g_\ell = \frac{\partial \mathcal{L}}{\partial I_\ell}, \tag{1}$$

starting from the last layer. Each layer needs to derive two sets of gradients to perform the recursive update. The layer activation gradients:

$$g_{\ell-1} = g_\ell W_\ell^T, \tag{2}$$

served for the Back-Propagation (BP) phase thus passed to the next layer,and the weights gradients

$$g_{W_\ell} = g_\ell I_{\ell-1}^T, \tag{3}$$

used to updated the weights in layer ℓ.

Since g_ℓ, the gradients streaming from layer ℓ, are required to compute $g_{\ell-1}$, it is important to expedite the matrix multiplication described in Eq.2. The second set of gradient derive in Eq.3 is not required for this sequential process and thus we choose to keep this matrix multiplication in full precision.

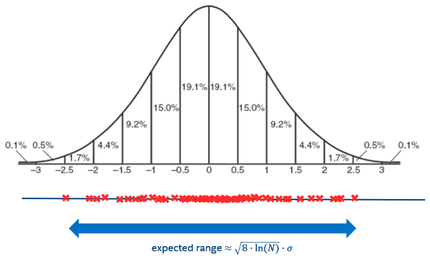
Range Batch-Normalization

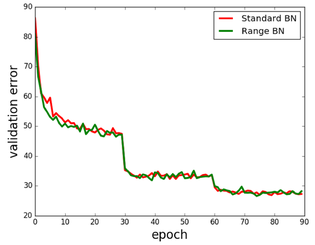
For a layer with $n \times d-$dimensional input $x = (x^{(1)}, x^{(2)}, ..., x^{(d)})$, traditional batch norm normalizes each dimension $\hat{x}^{(d)} = \frac{x^{(d)} - \mu^d}{\sigma^{(d)}}$, where μ^d is the expectation over $x^{(d)}$, n is the batch size and $\sigma^{(d)} = \sqrt{\frac{1}{n}||x^{(d)} - \mu^d||_2^2}$.

$$\sqrt{\frac{1}{n}||x^{(d)} - \mu^d||_2^2}$$

$$\downarrow$$

$$\text{range} = \max(x) - \min(x)$$

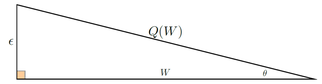




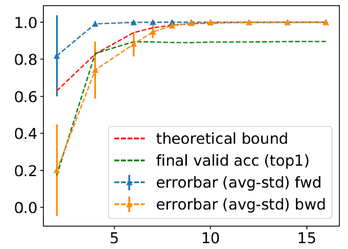
Res50 runs on ImageNet

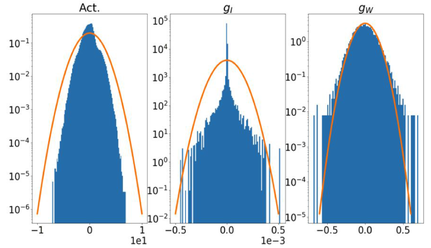
When is quantization of neural networks possible?

Given a vector of weights $W = (w_0, w_1, ..., w_{N-1})$, where the weights follow a Gaussian distribution $W \sim \mathcal{N}(0, \sigma^2)$, we prove that the cosine similarity (i.e., cosine of the angle) between W and its quantized version $Q(W)$ satisfies the following:



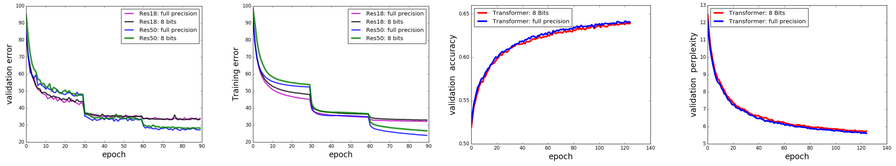
$$\cos(\theta) \geq \frac{2^M}{2^M + \sqrt{\ln N}/\sqrt{6}}$$





Putting it all together: RangeBN + Quantized BackProp

We conducted experiments using RangeBN together with Quantized Back-Propagation. To validate this low precision scheme, we were quantizing the vast majority of operations to 8-bit. The only operations left at higher precising were the updates (float32) needed to accumulate small changes from stochastic gradient descent, and a copy of the layer gradients at 16 bits needed to compute g_W.



Conclusions

- Directionality is well preserved during quantization when tensor is Gaussian.
- Layer gradients are non-gaussian, they fail to preserve direction and constitute the primary accuracy bottleneck in quantization.
- To maintain similar performance level, layer gradients should be kept at 16 bit precision for certain operations.
- Range BN is shown to be a viable normalization alternative for low precision.
- We show for the first time that 8-bit training on large data-sets is possible with very small (if any) accuracy degradation.

Rough relative costs in 45nm 0.9V from Sze et al. (2017).

Operation	Energy(pJ)		Area(μm²)	
	MUL	ADD	MUL	ADD
8-bit INT	0.2 pJ	0.03 pJ	282	36
16-bit FP	1.1 pJ	0.40 pJ	1640	1360
32-bit FP	3.7 pJ	0.90 pJ	7700	4184