Norm matters: efficient and accurate normalization schemes in deep networks

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

TECHNION Israel Institute of Technology

intel AI

Background

Deep neural networks are known to benefit from normalization between consecutive layers. This was made noticeable with the introduction of Batch-Normalization (BN) which normalizes the output of each layer to have zero mean and unit variance for each channel across the training batch.
However, there are several issues with current normalization methods:

- Interplay with other regularization mechanisms - it is not clear how weight decay interacts with BN, or if weight decay is even really necessary given that batch norm already constrains the output norms.
- Task-specific limitations - a key assumption in BN is the independence between samples appearing in each batch.
- Computational costs - BN is also not easily parallelized, as it is usually memory-bound on currently employed hardware.
- Numerical precision - the current normalization methods are notably not suited for low-precision and quantized operations.

Scale Invariance of Batch-Normalization

When BN is applied after a linear layer, it is well known that the output is invariant to the channel weight vector norm. Specifically, denoting a channel weight vector with w and $\hat{w} = w/\|w\|_2$, channel input as x and BN for batch-norm, we have

$$BN(\|w\|_2 \hat{w}x) = BN(\hat{w}x). \tag{1}$$

This invariance to the weight vector norm means that a BN applied after a layer renders its norm irrelevant to the inputs of consecutive layers. The same can be easily shown for the per-channel weights of a convolutional layer. The gradient in such case is scaled by $1/\|w\|_2$:

$$\frac{\partial BN(\|w\|_2 \hat{w}x)}{\partial(\|w\|_2 \hat{w})} = \frac{1}{\|w\|_2} \frac{\partial BN(\hat{w}x)}{\partial(\hat{w})}. \tag{2}$$

When a layer is rescaling invariant, the key feature of the weight vector is its direction.
During training, the weights are typically incremented through some variant of stochastic gradient descent, according to the gradient of the loss at mini-batch t, with learning rate η

$$w_{t+1} = w_t - \eta \nabla L_t(w_t) \ . \tag{3}$$

Thus, during training, the weight direction $\hat{w}_t = w_t/\|w_t\|_2$ is updated according to

$$\hat{w}_{t+1} = \hat{w}_t - \eta \|w_t\|^{-2} \left(I - \hat{w}_t \hat{w}_t^\top\right) \nabla L(\hat{w}_t) + O(\eta^2)$$

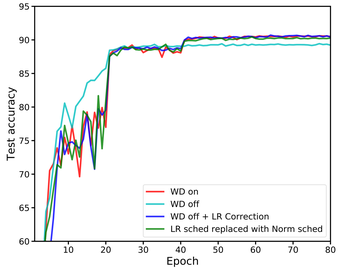
Connection between weight-decay, learning rate and normalization

We claim that when using batch-norm (BN), weight decay (WD) improves optimization only by fixing the norm to a small range of values, leading to a more stable step size for the weight direction ("effective step size"). Fixing the norm allows better control over the effective step size through the learning rate η.

We show empirically that the accuracy gained by using WD can be achieved without it, only by adjusting the learning rate. Given statistics on norms of each channel from a training with WD and BN, similar results can be achieved without WD by mimicking the effective step size using the following correction on the learning rate:

$$\hat{\eta}_{Correction} = \eta \frac{\|w\|_2^2}{\|w_{[WD]}\|_2^2} \tag{4}$$

where w is the weights' vector of a single channel, and $w_{[WD]}$ is the weights' vector of the corresponding channel in a training with WD.



The accuracy achieved is similar to the training with WD throughout the learning process, suggesting that WD affects the training process only indirectly, by modulating the learning rate.

Alternative L^p metrics for batch norm

As the main function of BN is to neutralize the effect of the preceding layer's weights, other operations might replace it, as long as they remain similarly scale invariant.

L^1 batch-norm

For a layer with $d-$dimensional input $x = (x^{(1)}, x^{(2)}, ..., x^{(d)})$, L^1 batch normalization normalize each dimension. μ^k is the expectation over $x^{(k)}$, n is the batch size and $C_{L_1} = \sqrt{\pi/2}$ is a normalization term.

$$\hat{x}^{(k)} = \frac{x^{(k)} - \mu^k}{C_{L_1} \cdot ||x^{(k)} - \mu^k||_1/n} \tag{5}$$

To calculate the constant C_{L_1}, we assume the input $x^{(k)}$ follows Gaussian distribution $N(\mu^k, \sigma^2)$.
In this case, $\hat{x}^{(k)} = (x^{(k)} - \mu^k)$ follows the distribution $N(0, \sigma^2)$. Therefore, $|\hat{x}_i^{(k)}|$ follows a half-normal distribution with expectation $E(|\hat{x}_i^{(k)}|) = \sigma \cdot \sqrt{2/\pi}$. Accordingly, the expected L^1 variability measure follows:

$$E\left[\frac{C_{L_1}}{n} \cdot ||x^{(k)} - \mu^k||_1\right] = \frac{\sqrt{\pi/2}}{n} \cdot \sum_{i=1}^{n} E[|\hat{x}_i^{(k)}|] = \sigma.$$

L^1 batch-norm has two major advantages

- Eliminates the computational efforts required for the square and sqrt operations.
- Absence of these ops makes it much more suitable for low-precision training.

L^∞ batch norm

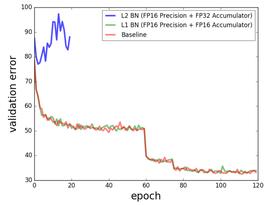
Another alternative measure that avoids the limitations of L^2 batch norm is the *maximum absolute deviation*.

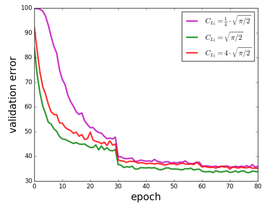
$$\hat{x}^{(k)} = \frac{x^{(k)} - \mu^k}{C_{L_\infty}(n) \cdot ||x^{(k)} - \mu^k||_\infty}, \tag{6}$$

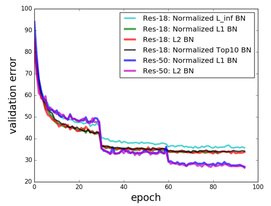
We also suggest Top(k) BN, which generalizes the notion of L^1 and L^∞ metrics, and is more robust to outliers:

$$\text{Top}(k) = \frac{1}{k} \sum_{n=1}^{k} |s_n|$$

where s_n are the n-th largest values in S







Improving weight normalization

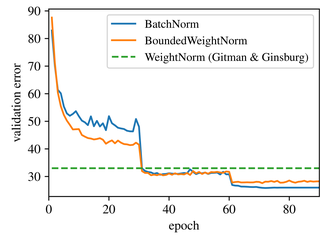
Weight-normalization is a known batch-norm alternative that normalizes weights instead of activations. Although very appealing due its low amount of computations, it was shown to fail for large-scale networks.

We wish to make the weight's norm completely disjoint from its values by keeping the norm fixed as follows:

$$w_i = \rho \frac{v_i}{\|v_i\|_2},$$

where ρ is a fixed scalar for each layer that is determined by its size (number of input and output channels). A simple choice for ρ is the initial norm, thus employing the various successful heuristics used to initialize networks.

This allowed bounded-weight-norm to reach a competitive accuracy on ResNet50 models trained on ImageNet.



References

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.