Online Normalization for Training Neural Networks

Vitaliy Chiley Ilya Sharapov Atli Kosson Urs Köster
Ryan Reece Sofía Samaniego de la Fuente Vishal Subbiah Michael James

Motivation and Background

- Normalization accelerates learning
- Normalization transforms a neural network from a function to a statistical operator that depends on its input distribution.
- Existing methods either use batches to approximate the input distribution [1,2] or restrict normalization's domain to a single sample [3,4,5].
- We propose an online normalization process that:
  - Eliminates batch dependency without restricting the domain
  - Decreases memory usage
  - Computes unbiased gradients
  - Provides train/inference symmetry
  - Integrates into auto differentiation frameworks

Memory Usage (GB)

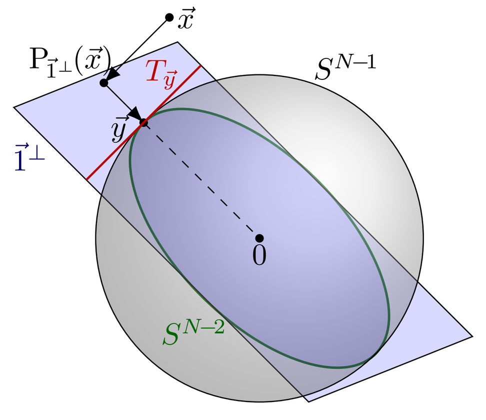
Network	Online Norm	Batch Norm 32	Batch Norm 128
ResNet-50, ImageNet, theory	1	2	4
ResNet-50, ImageNet, measured[a]	2	5	15
3D U-Net, $150 \times 150 \times 150$ voxels, theory	1	29	115
3D U-Net, $250 \times 250 \times 250$ voxels, theory	6	195	785
2D U-Net, 1024×1024 pixels, theory	2	31	123
2D U-Net, 2048×2048 pixels, theory	5	137	546

[a] PyTorch implementation stores multiple copies of activations for improved performance.

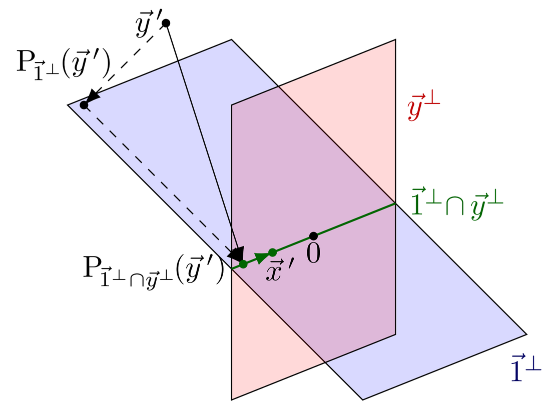
Principles of Normalization

- Normalization and its derivative are statistical operators
  - Notation: $(\cdot)' = \nabla_{(\cdot)} L$

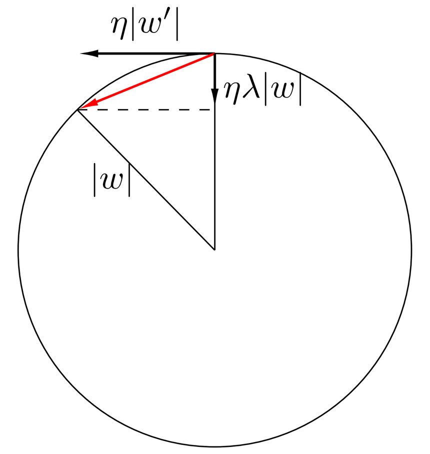
$$y = f_{\mathbb{X}}[x] \equiv (x - \mu[x]) / \sigma[x] \quad \text{and} \quad x' = (\nabla_x f_{\mathbb{X}}[x])\, y', \quad x \sim \mathbb{X}$$



FWD: Projection and rescaling s.t. normalized output lies in the zero-centered unit sphere: $\mu[y] = 0$ and $\mu[y^2] = 1$



BWD: Two projections st. gradient satisfies the orthogonality conditions that follow from the backward eqs: $\mu[x'] = 0$ and $\mu[x'y] = 0$



Normalized networks are invariant to gradient scale

Weight decay is required to prevent magnitude growth

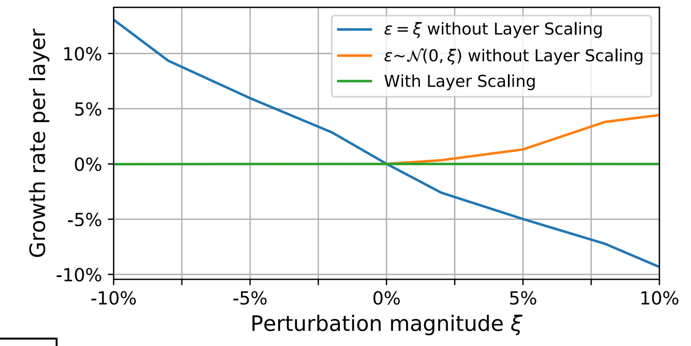
Online Normalization

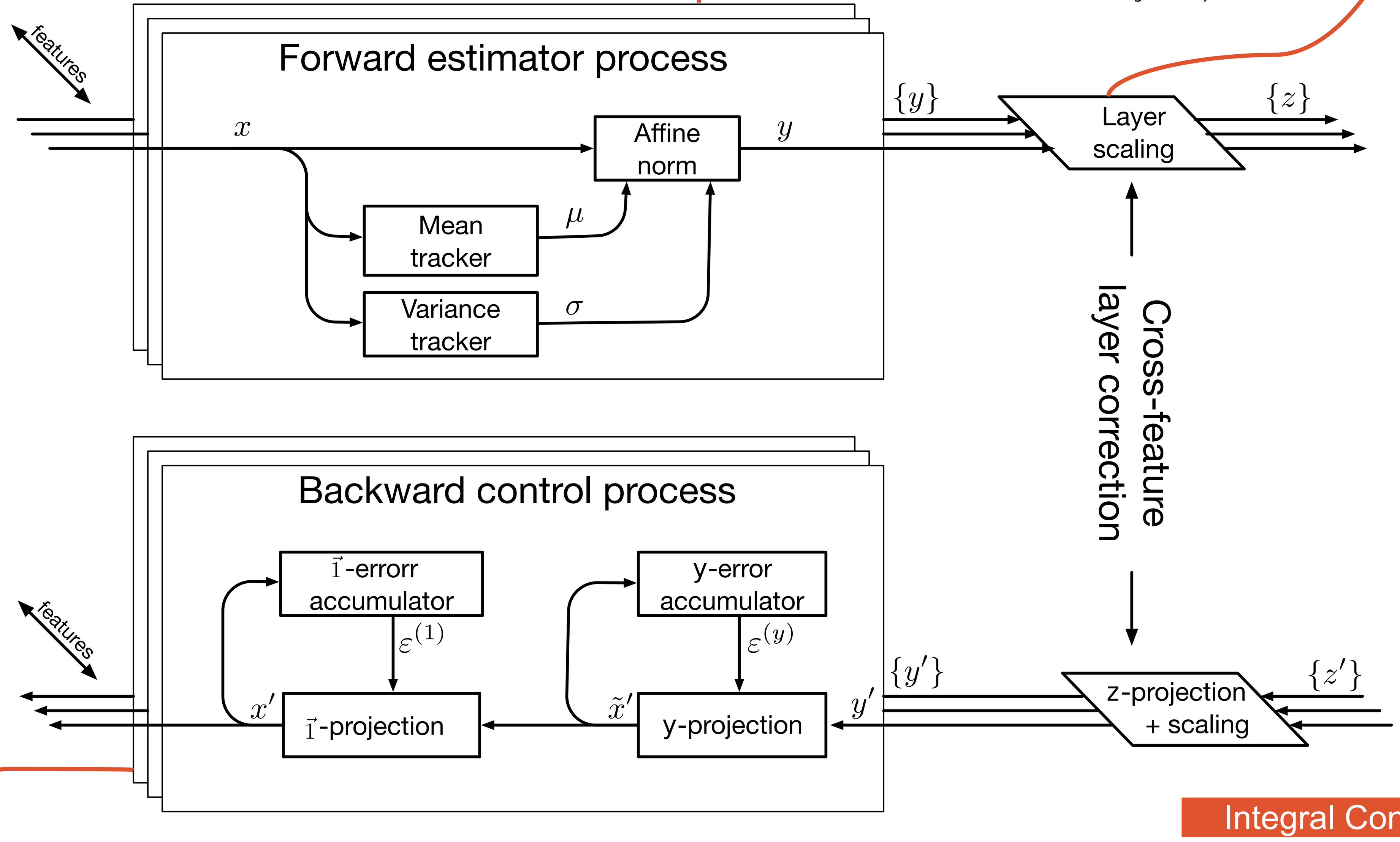
Forward Pass : Tracking process

- Exponential moving average
- Maintains zero mean, unit variance in expectation

Layer-wide correction

- Cross-Feature second moment correction
- Avoids exponential growth of activations





Backward Pass: Control Process

- Integral controller to enforce backward pass orthogonality constraints:

$$\langle x', 1 \rangle = 0 \quad \text{and} \quad \langle x', y \rangle = 0$$

- Leads to uniformly bounded error in gradient calculation

Integral Controller

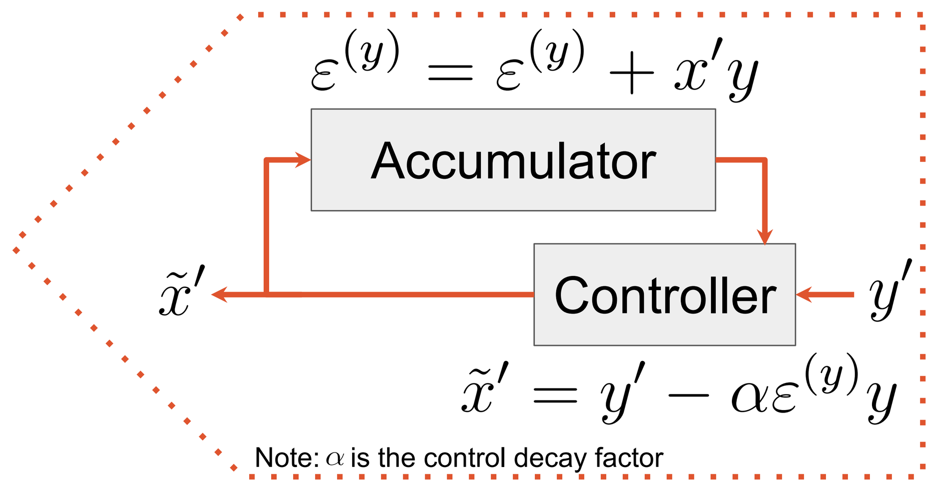
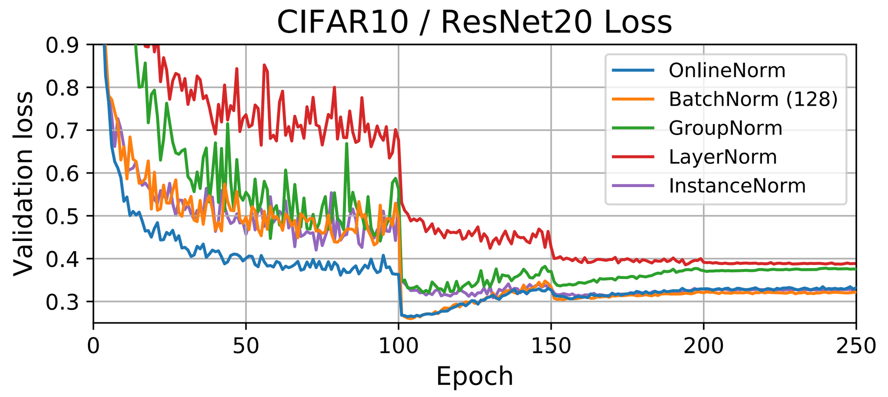
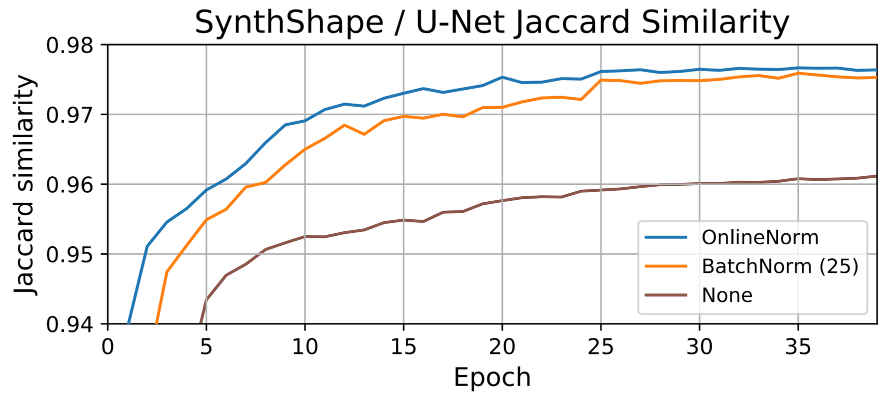
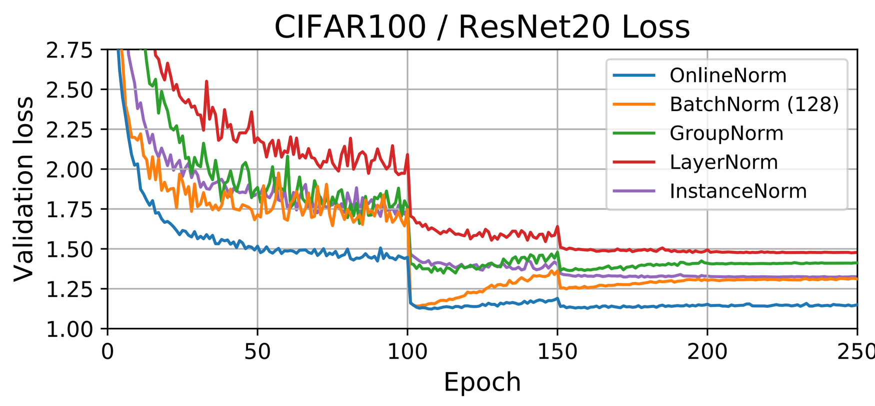
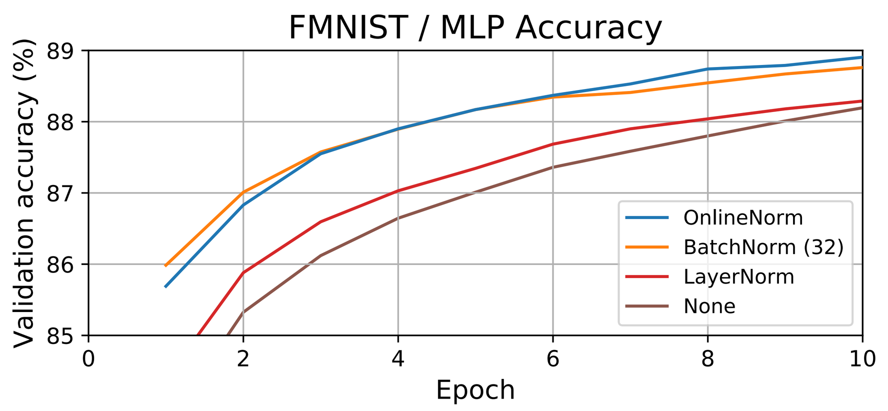


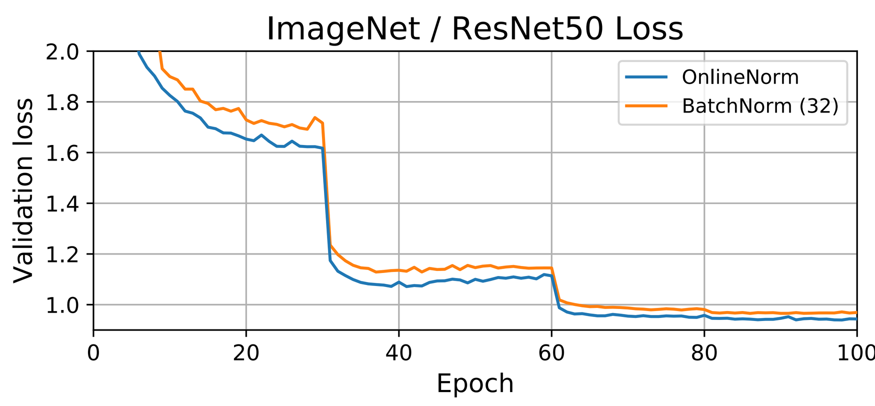
Image / Language / Generative Models

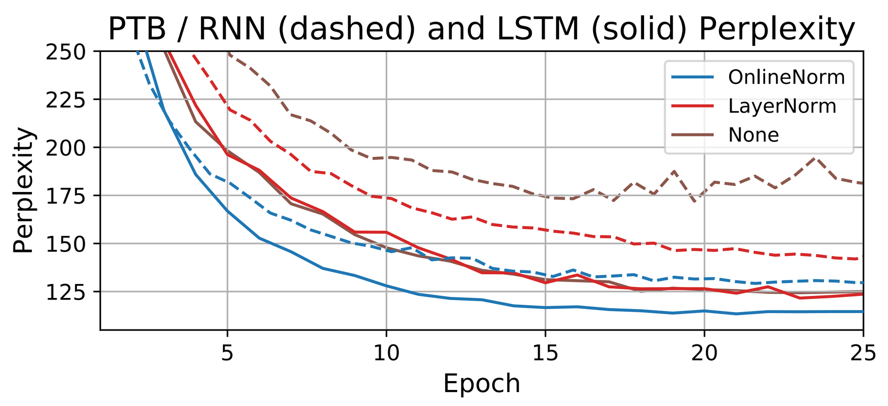








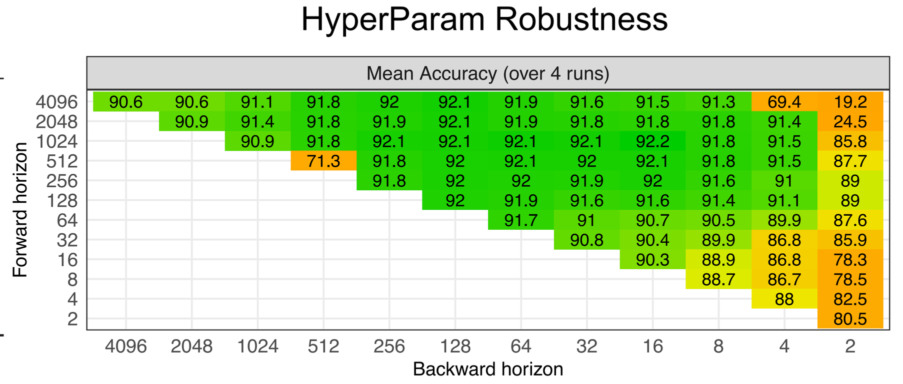




Normalizer	CIFAR-10 ResNet-20		CIFAR-100 ResNet-20		ImageNet ResNet-50	
Online (ON)	**0.26**	(**92.3%**)	**1.12**	(**68.6%**)	**0.94**	(76.3%)
Batch[a] (BN)	**0.26**	(92.2%)	1.14	(**68.6%**)	0.97	(**76.4%**)
Group (GN)	0.32	(90.3%)	1.35	(63.3%)		(75.9%)[b]
Instance (IN)	0.31	(90.4%)	1.32	(63.1%)		(71.6%)[b]
Layer (LN)	0.39	(87.4%)	1.47	(59.2%)		(74.7%)[b]
Weight		-		-		(67 %)[b]
Propagation		-		-		(71.9%)[b]

[a] Batch size 128 for CIFAR and 32 for ImageNet.
[b] Data from [5,6,7].

HyperParam Robustness



References

[1] Sergey Ioffe and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift."
[2] Sergey Ioffe. "Batch renormalization: Towards reducing minibatch dependence in batch-normalized models."
[3] Yuxin Wu and Kaiming He. "Group normalization."
[4] Lei Jimmy Ba, Ryan Kiros, and Geoffrey E. Hinton. "Layer Normalization."
[5] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. "Instance normalization: The missing ingredient for fast stylization"
[6] Igor Gitman and Boris Ginsburg. "Comparison of batch normalization and weight normalization algorithms for the large-scale image classification."
[7] Wenling Shang, Justin Chiu, and Kihyuk Sohn. "Exploring normalization in deep residual networks with concatenated rectified linear units."

https://github.com/Cerebras/online-normalization, arXiv https://arxiv.org/abs/1905.05894, https://www.cerebras.net