How to Initialize your Network? Robust Initialization for WeightNorm & ResNets

Devansh Arpit*
Salesforce Research

Víctor Campos*
Barcelona Supercomputing Center

Yoshua Bengio
MILA

Introduction

Contribution: novel initialization strategy for weight normalized networks and ResNets, with ReLU activation, that prevents information flow from exploding/vanishing in forward and backward pass.

Benefits of the proposed initialization scheme:

- Increased robustness to network depth, choice of hyperparameters and seed.
- When combined with learning rate warmup, it enables the usage of larger learning rates which reduce the generalization gap with batch normalization.
- Further analysis reveals that our proposal initializes networks in regions of the parameter space that have low curvature, thus allowing the use of large learning rates which are known to correlate with better generalization.

Initialization Schemes for Feedforward Networks

Setup: we consider weight normalized ReLU networks of the form

$$\mathbf{h}^l := ReLU(\mathbf{a}^l)$$
$$\mathbf{a}^l := \mathbf{g}^l \odot \hat{\mathbf{W}}^l \mathbf{h}^{l-1} + \mathbf{b}^l \quad l \in \{1, 2, \cdots L\} \qquad \text{where} \qquad \hat{\mathbf{W}}_i^l = \frac{\mathbf{W}_i^l}{\|\mathbf{W}_i^l\|_2} \quad \forall i$$

We propose to initialize weight matrices to be orthogonal and biases to zero. Depending on the direction in which we seek to preserve the norm of the information flow, we derive a different initialization for the gain factor. Our main results can be summarized as follows:

Direction	Theoretical result	Initialization
Forward pass	$\mathbb{E}\left[\|\|ReLU(\mathbf{g}^l \odot \hat{\mathbf{W}}^l \mathbf{h}^{l-1} + \mathbf{b}^l)\|\|\right] \approx \|\|\mathbf{h}^{l-1}\|\|$	$\mathbf{g}^l = \sqrt{2n_{l-1}/n_l}$
Backward pass	$\mathbb{E}\left[\|\|\frac{\partial L}{\partial \mathbf{a}^l}\|\|\right] = \mathbb{E}\left[\|\|\frac{\partial L}{\partial \mathbf{a}^{l+1}}\|\|\right]$	$\mathbf{g}^l = \sqrt{2} \cdot \mathbf{1}$

where L is the loss function with respect to the output of the network, and n_{l-1} and n_l represent the fan-in and fan-out of the l-th layer, respectively.

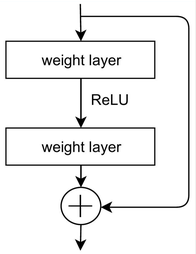
There is a discrepancy between the two schemes. We tested both, as well as combinations of them, and found the scheme for the forward pass to be superior.

Initialization Scheme for Residual Networks

We consider ResNets with K stages. Each stage $f_{\theta_k}(\cdot)$ is characterized by one shortcut connection and B_K residual blocks with hidden states

$$\mathbf{h}^{b+1} := \mathbf{h}^b + \alpha F_b(\mathbf{h}^b) \quad b \in \{0, 1, \ldots, B-1\}$$

We propose to initialize weight matrices to be orthogonal and biases to zero. Our main results can be summarized as follows:



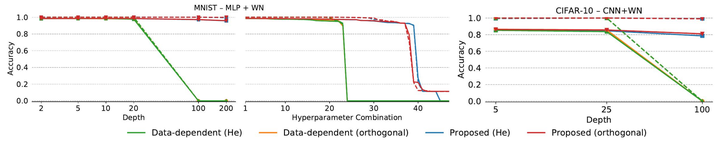
Direction	Theoretical result	Initialization
Forward pass	$\mathbb{E}\left[\|\|f_{\theta_k}(\mathbf{x})\|\|\right] \approx c \cdot \mathbb{E}\left[\|\|\mathbf{x}\|\|\right]$	$\mathbf{g} = \sqrt{\gamma \cdot \text{fan-in}/\text{fan-out}} \cdot \mathbf{1}$
Backward pass	$\mathbb{E}\left[\|\|\frac{\partial L}{\partial \mathbf{h}^1}\|\|\right] \approx c \cdot \mathbb{E}\left[\|\|\frac{\partial L}{\partial \mathbf{h}^B}\|\|\right]$	

where $c \in \left[\sqrt{2}, \sqrt{e}\right]$, L is the loss function wrt $f_{\theta_k}(\cdot)$, and $\mathbf{x}$ is the input. γ is set as follows:

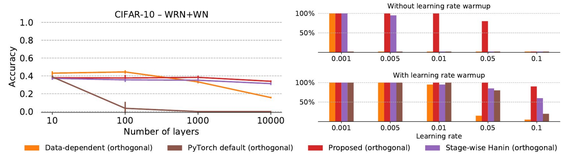
First layer in a ResBlock	Second layer in a ResBlock	Shortcut connections
$\gamma = 2$	$\gamma = 1/B_k$	$\gamma = 1$

Robustness Analysis: Depth, Hyperparameters and Seed

Feedforward networks: the proposed initialization succeeds at training deeper feedforward networks and is more robust to hyperparameter configurations.



ResNets: the proposed initialization achieves similar accuracy rates after 1 epoch of training for varying depths up to 10k layers (left). We also measured the percentage of WRN-40-10 runs that reach epoch 3 without diverging across 20 seeds (right). All schemes benefit from learning rate warmup, but the proposed initialization is the most robust scheme across all configurations.



Comparison with Batch Normalization

- We train state of the art architectures on CIFAR-10 and CIFAR-100.
- When combined with learning rate warmup, the proposed initialization scheme enables the usage of larger learning rates.
- Larger learning rates help reducing the generalization gap with respect to networks with Batch Normalization.

Dataset	Architecture	Method	Test Error (%)
CIFAR-10	ResNet-56	WN w/ datadep init	9.19 ± 0.24
		WN w/ proposed init	7.87 ± 0.14
		WN w/ proposed init + warmup	7.20 ± 0.12
		BN (He et al. [12])	6.97
	ResNet-110	WN w/ datadep init	9.33 ± 0.10
		WN w/ proposed init	7.71 ± 0.14
		WN w/ proposed init + warmup	6.69 ± 0.11
		WN (Shang et al. [29])	7.46
		BN (He et al. [12])	6.61 ± 0.16
	WRN-40-10	WN w/ datadep init + cutout	6.10 ± 0.23
		WN w/ proposed init + cutout	4.74 ± 0.14
		WN w/ proposed init + cutout + warmup	4.75 ± 0.08
		BN w/ orthogonal init + cutout	3.53 ± 0.38
CIFAR-100	ResNet-164	WN w/ datadep init + cutout	30.26 ± 0.51
		WN w/ proposed init + cutout	27.30 ± 0.49
		WN w/ proposed init + cutout + warmup	25.31 ± 0.26
		BN w/ orthogonal init + cutout	25.52 ± 0.17

Initialization Method and Generalization Gap

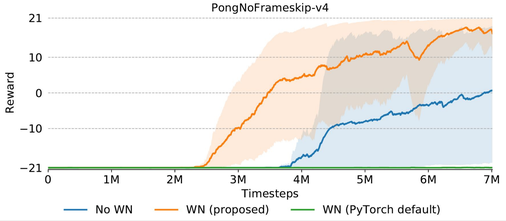
A number of papers have shown that SGD with large learning rates facilitates finding wider local minima, which correlate with better generalization. We compute the log spectral norm of the Hessian at initialization and find that the local curvature is smallest for the proposed scheme. This explains why larger learning rates can be used.

Dataset	Model	PyTorch default	Data-dependent	Stage-wise Hanin	Proposed
CIFAR-10	WRN-40-10	4.68 ± 0.60	3.01 ± 0.02	7.14 ± 0.72	1.31 ± 0.12
CIFAR-100	ResNet-164	9.56 ± 0.54	2.68 ± 0.09	N/A	1.56 ± 0.18

Initial Reinforcement Learning Results

Batch Normalization is seldom used in reinforcement learning (RL), as the online nature of some of the methods and the strong correlation between consecutive batches hinder its performance. We present preliminary RL results using Weigh Normalization with very deep networks.

- **Algorithm:** Asynchronous Advantage Actor-Critic (A3C) with 6 GPU workers
- **Architecture:** 100-layer IMPALA network
- **Number of runs:** 3 random seeds per configuration



*Equal contribution. Work done while Víctor Campos was an intern at Salesforce Research.

Code is available at https://github.com/victorcampos7/weightnorm-init