Proxy-Normalizing Activations to Match Batch Normalization while Removing Batch Dependence

Anonymous Author(s)
Affiliation
Address
email

Abstract

We investigate the reasons for the performance degradation incurred with batch-independent normalization. We find that the prototypical techniques of layer normalization and instance normalization both induce the appearance of failure modes in the neural network's pre-activations: (i) layer normalization induces a collapse towards channel-wise constant functions; (ii) instance normalization induces a lack of variability in instance statistics, symptomatic of an alteration of the expressivity. To alleviate failure mode (i) without aggravating failure mode (ii), we introduce the technique "Proxy Normalization" that normalizes post-activations using a proxy distribution. When combined with layer normalization or group normalization, this batch-independent normalization emulates batch normalization's behavior and consistently matches or exceeds its performance.

1 Introduction

Normalization plays a critical role in deep learning as it allows the successful scaling to large and deep models. In vision tasks, the most well-established normalization technique is Batch Normalization (BN) [1]. At every layer in the network, BN normalizes the intermediate activations to have zero mean and unit variance in each channel. While undisputably successful in the regime of large batch size, BN incurs a performance degradation in the regime of small batch size. This performance degradation is commonly attributed to the excessive or simply unwanted regularization stemming from the noise in the approximation of full-batch statistics by mini-batch statistics.

Many techniques have been proposed to avoid this pitall while at the same time retaining BN's benefits. Some techniques mimick BN's principle while decoupling the computing batch from the normalization batch [2, 3, 4]. Other techniques are "batch-independent" in that they operate independently of the batch in various modalities: through an explicit normalization either in activation space [5, 6, 7, 8, 9, 10, 11, 12, 13, 14] or in weight space [15, 16, 17, 18]; through the use of an analytic proxy to track activation statistics [19, 20, 21]; through a change of activation function [22, 11, 13].

In this paper, we push the endeavour to replace BN with a batch-independent alternative a step further. Our main contributions are as follows: (i) we introduce a novel framework to finely characterize the various neural network properties affected by the choice of normalization; (ii) we show via this framework that BN's beneficial properties *are not* retained when solely using the prototypical batch-independent normalization techniques, but *are* retained when combining some of these techniques with the novel and hereby introduced technique Proxy Normalization (PN); (iii) we demonstrate by an extensive set of experiments that reproducing BN's beneficial properties using our novel approach of batch-independent normalization enables us to consistently match or exceed BN's performance.

As starting point of our analysis, we need to gain a better understanding of those beneficial properties of BN that we aim at reproducing.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

2 Batch Normalization's beneficial properties

We consider throughout this paper a convolutional neural network with $d = 2$ spatial axes. This neural network maps an input $\mathbf{x} \in \mathbb{R}^{H \times W \times C_0}$ sampled from the dataset $\mathcal{D}$ to intermediate activations $\mathbf{x}^l \in \mathbb{R}^{H \times W \times C_l}$ at each layer l. The value of $\mathbf{x}^l$ at spatial position $\alpha \in \{1, \dots, H\} \times \{1, \dots, W\}$ and channel $c \in \{1, \dots, C_l\}$ is denoted as $\mathbf{x}^l_{\alpha,c}$, with the dependency on $\mathbf{x}$ kept implicit to avoid overloading notations.

The inclusion of BN at layer l leads in the full-batch setting to adding the following operations $\forall \alpha, c$:

$$\mathbf{y}^l_{\alpha,c} = \left(\frac{\mathbf{x}^l_{\alpha,c} - \mu_c(\mathbf{x}^l)}{\sigma_c(\mathbf{x}^l)} \right), \qquad \tilde{\mathbf{y}}^l_{\alpha,c} = \boldsymbol{\gamma}^l_c \mathbf{y}^l_{\alpha,c} + \boldsymbol{\beta}^l_c. \tag{1}$$

where $\mu_c(\mathbf{x}^l)$, $\sigma_c(\mathbf{x}^l)$ are the full-batch mean and standard deviation of $\mathbf{x}^l$ in channel c, and $\boldsymbol{\gamma}^l_c$, $\boldsymbol{\beta}^l_c$ are the channel-wise scale and shift parameters restoring the degrees of freedom lost in the standardization. In the mini-batch setting, the full-batch statistics $\mu_c(\mathbf{x}^l)$, $\sigma_c(\mathbf{x}^l)$ in Eq. (1) are approximated with mini-batch statistics. Unless specifically referring to BN's regularization stemming from this approximation, we keep however assuming the full-batch setting of Eq. (1).

Table 1 summarizes the beneficial properties that result from including BN in the neural network. Below, we provide additional details on each of these properties and we discuss whether they are reproduced with batch-independent norms.

Scale invariance. When BN is present, the input-output mapping of the neural network is invariant to the scale of the weights preceding any BN layer. With such scale invariance plus weight decay, the scale of weights during training reaches an equilibrium with an effective learning rate depending on both the learning rate and the weight decay strength [23, 24, 25, 3, 26, 27, 28]. Such mechanism of "auto rate-tuning" has been shown to provide optimization benefits [29, 24, 30].

This easily-reproduced property is already obtained with most existing batch-independent norms.

Control of activation scale in residual networks. To be trainable, residual networks require the scale of activations to remain well-behaved at initialization [31, 32, 33, 18, 34]. While this property naturally arises when BN is present on the residual path, it can also be enforced when BN is not present by a proper scaling decaying with depth of the residual path. This "dilution" of the residual path with respect to the skip connection path reduces the effective depth of the network and thereby enables to avoid coarse-grained failures modes [35, 36, 37, 38, 33].

This easily-reproduced property is already obtained with most existing batch-independent norms.

Regularizing noise. Due to the stochasticity of the approximation of $\mu_c(\mathbf{x}^l)$, $\sigma_c(\mathbf{x}^l)$ by mini-batch statistics, training a neural network with BN in the mini-batch setting can be seen equivalently as performing Bayesian inference [39, 40] or as adding a regularizing term to the training of the same network with full-batch statistics [41]. This results in a specific regularization of BN.

This regularization is not easy to reproduce, but we leave it out of the scope of this paper. To avoid biasing our analysis, we "subtract away" this effect by performing all our experiments without and with extra degrees of regularization. This procedure can be seen as disentangling normalization's effects from regularization's effects.

Avoidance of collapse. Unnormalized networks with non-saturating nonlinearities are subject to a phenomenon of "collapse", whereby the data distribution of the intermediate activation vectors $\mathbf{x}^l_{\alpha,:} \in \mathbb{R}^{C_l}$ becomes close to zero- or one-dimensional in deep layers [36, 42, 38, 43, 44, 45, 46].[1] This means that deep in an unnormalized network: (i) layers tend to have their channels imbalanced; (ii) nonlinearities tend to become pseudo-linear w.r.t. the data and not add any effective capacity [36, 47, 48]. Consequently, unnormalized networks can neither effectively use their whole width (imbalanced channels) nor effectively use their whole depth (pseudo linearity).[2]

[1] We use $\mathbf{x}^l_{\alpha,:}$ to denote the vectorial slice of $\mathbf{x}^l$ at spatial position α and we refer as "data distribution" the distribution w.r.t. a random choice of input $\mathbf{x} \in \mathcal{D}$ and spatial position $\alpha \in \{1, \dots, H\} \times \{1, \dots, W\}$.

[2] At first glance, this collapse might seem like another facet of the "internal covariate shift" [1, 49]. There are still notable differences between the two concepts. Firstly, the internal covariate shift, as originally defined [1, 49], only has to do with the rate of change of activation distributions and *not* with "static" failure modes in these distributions. Secondly, the rate of change of activation distributions is not only affected by the choice of normalization but also by confounders such as the learning rate.

Table 1: BN's beneficial properties and their presence (✓) or absence (✗) with various batch-independent norms: Instance Normalization (IN), Layer Normalization (LN), Layer Normalization + Proxy Normalization (LN+PN, cf Section 5). Note that this categorization is still simplifying, with e.g. even BN in the mini-batch setting or LN, LN+PN involving slight alterations of the expressivity.

	scale invariance	control of activation scale	regularizing noise	avoidance of collapse	preservation of the expressivity
BN	✓	✓	✓	✓	✓
IN	✓	✓	✗	✓	✗
LN	✓	✓	✗	✗	✓
LN+PN	✓	✓	✗	✓	✓

Conversely, when BN is used, the standardization at each layer prevents this collapse from happening. Even deep in the network, layers still have balanced channels and nonlinearities remain nonlinear w.r.t. the data. Consequently, networks with BN can effectively use their whole width and depth.

The collapse is, on the other hand, not always avoided with batch-independent norms. Notably, the collapse is *not* avoided with Layer Normalization (LN) [5] or Group Normalization (GN) [10], as observed in previous works [36, 17, 46] and as we show theoretically in Section 4.

We specifically aim at designing a batch-independent normalization that avoids this collapse.

Preservation of the expressivity. We can always express the identity with Eq. (1) by choosing $\boldsymbol{\beta}^l_c = \mu_c(\mathbf{x}^l)$, $\boldsymbol{\gamma}^l_c = \sigma_c(\mathbf{x}^l)$. Conversely, for any choice of $\boldsymbol{\beta}^l_c$, $\boldsymbol{\gamma}^l_c$, we can always "re-absorb" Eq. (1) into a preceding convolution with bias. This means that BN *does not alter the expressivity* compared to an unnormalized network, i.e. it amounts to a plain reparameterization of the hypothesis space.

The expressivity is, on the other hand, not always preserved with batch-independent norms. In activation space, the dependence of batch-independent statistics on the input $\mathbf{x}$ turns the standardization into a nonlinear operation that cannot be "re-absorbed" into a preceding convolution [17]. This phenomenon is most pronounced when statistics get computed over few components. This means e.g. that Instance Normalization (IN) [6] induces a greater change of expressivity than GN, which itself induces a greater change of expressivity than LN.

In weight space, the expressivity can also be altered, namely by the removal of degrees of freedom. This is the case with Weight Standardization (WS) [17, 18] or Centered Weight Normalization [16] that remove degrees of freedom (one per unit) that *cannot* be restored in a succeeding affine transform. This reduction of the expressivity could explain the uneffectiveness of these techniques in the "EfficientNet" class of models [50], as observed in previous works [18] and as we confirm in Section 6.

We specifically aim at designing a batch-independent normalization that preserves the expressivity.

3 Theoretical framework of analysis

We outlined the specific properties that we wish to retain in our design of batch-independent normalization: (i) avoidance of collapse: (ii) preservation of the expressivity. We now introduce a framework to quantify the presence or absence of these properties with various choices of normalization.

Equations of propagation. For simplicity, we assume in our theoretical analysis that any layer l up to depth L consists of the following three steps: (i) convolution step with weights $\boldsymbol{\omega}^l \in \mathbb{R}^{K_l \times K_l \times C_{l-1} \times C_l}$; (ii) normalization step; (iii) activation step sub-decomposed into an affine transform with scale and shift parameters $\boldsymbol{\gamma}^l, \boldsymbol{\beta}^l \in \mathbb{R}^{C_l}$ and an activation function ϕ, assumed positive homogeneous and nonzero (e.g. $\phi = \mathrm{ReLU}$). If we denote $\mathbf{x}^l, \mathbf{y}^l, \mathbf{z}^l \in \mathbb{R}^{H \times W \times C_l}$ the intermediate activations situated just after (i), (ii), (iii) , we may write the propagation through layer l as

$$\mathbf{x}^l = \mathrm{Conv}(\mathbf{z}^{l-1}), \qquad \forall \alpha, c: \quad \mathrm{Conv}(\mathbf{z}^{l-1})_{\alpha,c} = (\boldsymbol{\omega}^l * \mathbf{z}^{l-1})_{\alpha,c}, \tag{2}$$

$$\mathbf{y}^l = \mathrm{Norm}(\mathbf{x}^l), \qquad \forall \alpha, c: \quad \mathrm{Norm}(\mathbf{x}^l)_{\alpha,c} = \left(\frac{\mathbf{x}^l_{\alpha,c} - \mu_{I_{\mathbf{x},c}}(\mathbf{x}^l)}{\sigma_{I_{\mathbf{x},c}}(\mathbf{x}^l)} \right), \tag{3}$$

$$\mathbf{z}^l = \mathrm{Act}(\mathbf{y}^l), \qquad \forall \alpha, c: \quad \mathrm{Act}(\mathbf{y}^l)_{\alpha,c} = \phi\big(\tilde{\mathbf{y}}^l_{\alpha,c}\big) = \phi\big(\boldsymbol{\gamma}^l_c \mathbf{y}^l_{\alpha,c} + \boldsymbol{\beta}^l_c\big), \tag{4}$$

with the convention $\mathbf{z}^0 \equiv \mathbf{x}$ and with $\mu_{I_{\mathbf{x},c}}(\mathbf{x}^l)$, $\sigma_{I_{\mathbf{x},c}}(\mathbf{x}^l)$ the mean and standard deviation of $\mathbf{x}^l$ conditionally on $I_{\mathbf{x},c} \equiv \{c\}, \{\mathbf{x}\}, \{\mathbf{x}, c\}, \{\mathbf{x}, c \bmod G\}$ for the respective cases $\mathrm{Norm} = \mathrm{BN}, \mathrm{LN}, \mathrm{IN}, \mathrm{GN}$ with G groups.[3]

Moments. Extending the previous notations, we use μ, σ, $\mathcal{P}$ indexed with a (possibly empty) subset of variables to denote the operators of conditional mean, standard deviation and power, e.g.

$$\mu_c(\mathbf{y}^l) = \mathbb{E}_{\mathbf{x},\alpha}\left[\mathbf{y}^l_{\alpha,c}\right], \qquad \sigma_c(\mathbf{y}^l) = \sqrt{\mathrm{Var}_{\mathbf{x},\alpha}\left[\mathbf{y}^l_{\alpha,c}\right]}, \qquad \mathcal{P}_c(\mathbf{y}^l) = \underbrace{\mathbb{E}_{\mathbf{x},\alpha}\left[(\mathbf{y}^l_{\alpha,c})^2\right]}_{\sigma_c(\mathbf{y}^l)^2+\mu_c(\mathbf{y}^l)^2},$$

$$\mu_{\mathbf{x},c}(\mathbf{y}^l) = \mathbb{E}_{\alpha}\left[\mathbf{y}^l_{\alpha,c}\right], \qquad \sigma_{\mathbf{x},c}(\mathbf{y}^l) = \sqrt{\mathrm{Var}_{\alpha}\left[\mathbf{y}^l_{\alpha,c}\right]}, \qquad \mathcal{P}_{\mathbf{x},c}(\mathbf{y}^l) = \underbrace{\mathbb{E}_{\alpha}\left[(\mathbf{y}^l_{\alpha,c})^2\right]}_{\sigma_{\mathbf{x},c}(\mathbf{y}^l)^2+\mu_{\mathbf{x},c}(\mathbf{y}^l)^2}.$$

Power decomposition. Using these notations, we may gain important insights by decomposing the power in channel c of $\mathbf{y}^l$, just after the normalization, as

$$\mathcal{P}_c(\mathbf{y}^l) = \underbrace{\mathbb{E}_{\mathbf{x}}\left[\mu_{\mathbf{x},c}(\mathbf{y}^l)\right]^2}_{\mathcal{P}_c^{(1)}(\mathbf{y}^l)} + \underbrace{\mathrm{Var}_{\mathbf{x}}\left[\mu_{\mathbf{x},c}(\mathbf{y}^l)\right]}_{\mathcal{P}_c^{(2)}(\mathbf{y}^l)} + \underbrace{\mathbb{E}_{\mathbf{x}}\left[\sigma_{\mathbf{x},c}(\mathbf{y}^l)\right]^2}_{\mathcal{P}_c^{(3)}(\mathbf{y}^l)} + \underbrace{\mathrm{Var}_{\mathbf{x}}\left[\sigma_{\mathbf{x},c}(\mathbf{y}^l)\right]}_{\mathcal{P}_c^{(4)}(\mathbf{y}^l)}. \tag{5}$$

Since this four-terms *power decomposition* will be at the core of our analysis, we detail two useful views on it. The first view is that of a *hierarchy of scales*: $\mathcal{P}_c^{(1)}(\mathbf{y}^l)$ measures the power of $\mu_c(\mathbf{y}^l)$ at the dataset scale; $\mathcal{P}_c^{(2)}(\mathbf{y}^l)$ measures the power of $\mu_{\mathbf{x},c}(\mathbf{y}^l) - \mu_c(\mathbf{y}^l)$ at the instance scale; the sum of $\mathcal{P}_c^{(3)}(\mathbf{y}^l), \mathcal{P}_c^{(4)}(\mathbf{y}^l)$ measures the power of $\mathbf{y}^l_{\alpha,c} - \mu_{\mathbf{x},c}(\mathbf{y}^l)$ at the pixel scale. A situation where the power would be concentrated at the dataset scale with $\mathcal{P}_c^{(1)}(\mathbf{y}^l)$ equal to $\mathcal{P}_c(\mathbf{y}^l)$ would notably imply that $\mathbf{y}^l$ has its distribution in channel c fully "collapsed", in the sense that $\mathbf{y}^l$ is constant in channel c.

The second view is that of a *two-level binary tree*: on one half of the tree, the sum of $\mathcal{P}_c^{(1)}(\mathbf{y}^l)$ and $\mathcal{P}_c^{(2)}(\mathbf{y}^l)$ measures the power coming from $\mu_{\mathbf{x},c}(\mathbf{y}^l)$, with the relative proportions of $\mathcal{P}_c^{(1)}(\mathbf{y}^l)$ and $\mathcal{P}_c^{(2)}(\mathbf{y}^l)$ functions of the inter-$\mathbf{x}$ similarity and inter-$\mathbf{x}$ variability of $\mu_{\mathbf{x},c}(\mathbf{y}^l)$; on the other half of the tree, the sum of $\mathcal{P}_c^{(3)}(\mathbf{y}^l)$ and $\mathcal{P}_c^{(4)}(\mathbf{y}^l)$ measures the power coming from $\sigma_{\mathbf{x},c}(\mathbf{y}^l)$, with the relative proportions of $\mathcal{P}_c^{(3)}(\mathbf{y}^l)$ and $\mathcal{P}_c^{(4)}(\mathbf{y}^l)$ functions of the inter-$\mathbf{x}$ similarity and inter-$\mathbf{x}$ variability of $\sigma_{\mathbf{x},c}(\mathbf{y}^l)$. A situation where $\mathcal{P}_c^{(2)}(\mathbf{y}^l)$, $\mathcal{P}_c^{(4)}(\mathbf{y}^l)$ would be equal to zero would notably imply that $\mu_{\mathbf{x},c}(\mathbf{y}^l)$, $\sigma_{\mathbf{x},c}(\mathbf{y}^l)$ have zero inter-$\mathbf{x}$ variability, i.e. are constant for all $\mathbf{x}$.

A version of Eq. (5) at the layer level instead of channel level will be easier to work with. To obtain it, we define $\mathcal{P}^{(i)}(\mathbf{y}^l)$ as the averages of $\mathcal{P}_c^{(i)}(\mathbf{y}^l)$ over c for $i \in \{1, 2, 3, 4\}$, such that

$$\mathcal{P}(\mathbf{y}^l) = \mathcal{P}^{(1)}(\mathbf{y}^l) + \mathcal{P}^{(2)}(\mathbf{y}^l) + \mathcal{P}^{(3)}(\mathbf{y}^l) + \mathcal{P}^{(4)}(\mathbf{y}^l).$$

It should be noted that $\mathcal{P}(\mathbf{y}^l) = 1$ for any choice of $\mathrm{Norm} \in \{\mathrm{BN}, \mathrm{LN}, \mathrm{IN}, \mathrm{GN}\}$, as long as the denominator of Eq. (3) is nonzero for all $\mathbf{x}, c$ [C.1]. Consequently, the terms $\mathcal{P}^{(i)}(\mathbf{y}^l)$ sum to one, meaning they can be seen conveniently as the proportion of each term $i \in \{1, 2, 3, 4\}$ into $\mathcal{P}(\mathbf{y}^l)$.

Revisiting BN's avoidance of collapse. When BN is used, $\mathbf{y}^l$ is normalized not only layer-wise but also channel-wise with $\mathcal{P}_c^{(1)}(\mathbf{y}^l) = 0$ and $\mathcal{P}_c(\mathbf{y}^l) = 1$. As first consequence, $\tilde{\mathbf{y}}^l$ (that is only separated from $\mathbf{y}^l$ by an affine transform) is likely to *not have* its channel-wise data distributions collapsed. This means that the nonlinearity ϕ acting on $\tilde{\mathbf{y}}^l$ can be effectively nonlinear w.r.t. $\tilde{\mathbf{y}}^l$'s channel-wise data distributions. As a result, each layer adds capacity and the network effectively uses its whole *depth*. This is opposite to a situation where $\tilde{\mathbf{y}}^l$ has its channel-wise data distributions collapsed with $\mathcal{P}_c(\tilde{\mathbf{y}}^l) - \mathcal{P}_c^{(1)}(\tilde{\mathbf{y}}^l) \ll \mathcal{P}_c(\tilde{\mathbf{y}}^l)$, which results in ϕ being close to linear w.r.t. $\tilde{\mathbf{y}}^l$'s channel-wise data distributions. This is illustrated in Figure 1 and formalized in Appendix C.2.[4]

[3] We assume that the norm's numerical stability constant is zero, with the convention $\mathbf{y}^l_{\alpha,c} = 0$, $\forall \alpha$ if $\sigma_{I_{\mathbf{x},c}}(\mathbf{x}^l) = 0$.

[4] Note that BN only guarantees an intra-task nonlinearity. If there are several modes in the task (e.g. in contexts such as adversarial training [51, 52] or conditional GANs [53]), there is still no guarantee of intra-mode nonlinearity unless the modes are decoupled in BN's computation [54, 55, 56, 57, 58].

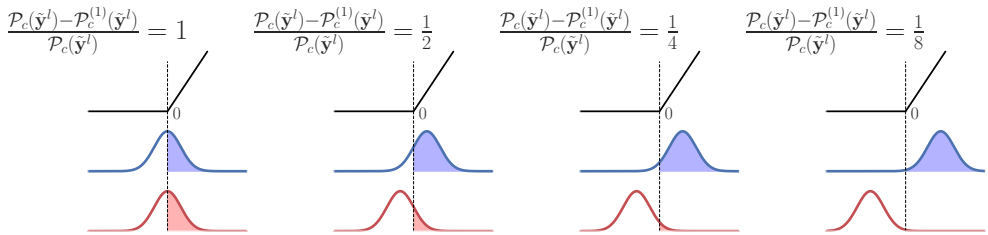

Figure 1: Connection between channel-wise collapse and pseudo-linearity for $\phi = \mathrm{ReLU}$ (black activation function). We show two potential data distributions positioned symmetrically around zero with $\sigma_c(\mathbf{y}^l) = 1$ and different values of $\frac{\mathcal{P}_c(\tilde{\mathbf{y}}^l)-\mathcal{P}_c^{(1)}(\tilde{\mathbf{y}}^l)}{\mathcal{P}_c(\tilde{\mathbf{y}}^l)} = \frac{\sigma_c(\tilde{\mathbf{y}}^l)^2}{\sigma_c(\tilde{\mathbf{y}}^l)^2+\mu_c(\tilde{\mathbf{y}}^l)^2} \in \{1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}\}$. As $\frac{\mathcal{P}_c(\tilde{\mathbf{y}}^l)-\mathcal{P}_c^{(1)}(\tilde{\mathbf{y}}^l)}{\mathcal{P}_c(\tilde{\mathbf{y}}^l)}$ becomes small, the part of the distribution corresponding to active ReLU (shaded region) becomes either overly dominant (blue distribution) or negligible (red distribution). In either case, the data distribution ends up concentrated on only one side of piece-wise linearity of ReLU.

As other consequence, $\mathbf{y}^l$ is guaranteed to have its channels well balanced with equal power $\mathcal{P}_c(\mathbf{y}^l)$ for all c. As a result, the network effectively uses its whole *width*. This is opposite to a situation where a single channel c becomes overly dominant over the others with $\mathcal{P}_{c'}(\mathbf{y}^l) \ll \mathcal{P}_c(\mathbf{y}^l)$ for $c' \neq c$, which results in downstream layers only "seeing" this channel c and the network behaving as if it had a width equal to one at layer l.

Revisiting BN's preservation of the expressivity. When BN is used, $\mathcal{P}_c^{(1)}(\mathbf{y}^l) = 0$ implies for all c that the terms $\mathcal{P}_c^{(2)}(\mathbf{y}^l), \mathcal{P}_c^{(3)}(\mathbf{y}^l), \mathcal{P}_c^{(4)}(\mathbf{y}^l)$ sum to one. Apart from that, BN does not impose any particular constraints on the relative proportions of each term into the sum. This means that the relative proportions of $\mathcal{P}_c^{(i)}(\mathbf{y}^l)$ and $\mathcal{P}^{(i)}(\mathbf{y}^l)$ for $i \in \{2, 3, 4\}$ are free to evolve as naturally dictated by the task and the optimizer during learning.

This absence of constraints seems sensible. Indeed, imposing constraints on these relative proportions would *alter the expressivity*, which would not have any obvious justification in general and could even be detrimental in some cases, as we discuss in Section 4.

4 Failure modes with batch-independent normalization

With our theoretical framework in hand, we now turn to showing that the prototypical batch-independent norms are subject to failures modes at the opposite of BN's beneficial properties.

In the case of LN, the failure mode does not manifest in an absolute sense but rather as a favoring in the hypothesis space. This favoring is quantified by Theorem 1 in the context of networks with random model parameters.

Definition 1 (random net)**.** *We define a "random net" as a neural network having an input* $\mathbf{x}$ *sampled from the dataset* $\mathcal{D}$ *and implementing Eq. (2), (3) and (4) in its first* L *layers with the components of* $\boldsymbol{\omega}^l$,$\boldsymbol{\gamma}^l$,$\boldsymbol{\beta}^l$ *at each layer* $l \in \{1, \dots, L\}$ *sampled i.i.d. from the fixed distributions* $\nu_{\boldsymbol{\omega}}$,$\nu_{\boldsymbol{\gamma}}$,$\nu_{\boldsymbol{\beta}}$ *(up to a fan-in's square root scaling for* $\boldsymbol{\omega}^l$*).*

In such networks, we assume that: (i) $\mathcal{D}$ *is composed of a finite number of elements, none of which being identically zero; (ii)* $\nu_{\boldsymbol{\omega}}$, $\nu_{\boldsymbol{\beta}}$, $\nu_{\boldsymbol{\gamma}}$ *have well-defined moments, with strictly positive associated root mean squares* $\omega, \gamma, \beta > 0$*; (iii)* $\nu_{\boldsymbol{\omega}}$, $\nu_{\boldsymbol{\beta}}$ *are symmetric around zero.*

Theorem 1 (layer-normalized nets collapse (informal))**.** [D.3] *Fix a layer* $l \in \{1, \dots, L\}$ *and* $\nu_{\boldsymbol{\omega}}$, $\nu_{\boldsymbol{\beta}}$, $\nu_{\boldsymbol{\gamma}}$, $\mathcal{D}$ *in Definition 1. Further suppose* $\mathrm{Norm} = \mathrm{LN}$ *and suppose that the convolution of Eq. (2) uses periodic boundary conditions.*

Then for random nets of Definition 1, it holds that

$$\mathcal{P}(\mathbf{y}^l) - \mathcal{P}^{(1)}(\mathbf{y}^l) \lesssim \rho^{l-1}, \qquad \mathcal{P}(\mathbf{y}^l) \simeq 1, \tag{6}$$

where $\rho \equiv \gamma^2/(\gamma^2 + \beta^2) < 1$*, and* $\lesssim$ *and* $\simeq$ *denote inequality and equality up to arbitrarily small constants with probability arbitrarily close to 1 for* $\min_{0 \leq k \leq l} C_k$ *large enough.*

Discussion on LN's failure mode. Theorem 1 implies that, with high probability, $\mathbf{y}^l$ is subject to *channel-wise collapse* in deep layers with $\mathcal{P}(\mathbf{y}^l) - \mathcal{P}^{(1)}(\mathbf{y}^l) \ll \mathcal{P}(\mathbf{y}^l)$ for $l \gg 1$. This means that $\tilde{\mathbf{y}}^l$, that is only separated from $\mathbf{y}^l$ by an affine transform, is likely to have most of its channel-wise distributions equally collapsed with $\mathcal{P}_c(\tilde{\mathbf{y}}^l) - \mathcal{P}_c^{(1)}(\tilde{\mathbf{y}}^l) \ll \mathcal{P}_c(\tilde{\mathbf{y}}^l)$. This implies that the nonlinearity ϕ acting on $\tilde{\mathbf{y}}^l$ becomes close to linear w.r.t. $\tilde{\mathbf{y}}^l$'s channel-wise data distributions [C.2]. Being pseudo-linear in deep layers, layer-normalized networks are unable to effectively use their whole depth.

Since the inequality $\lesssim$ can be replaced by an equality $\simeq$ in the case $\phi = \text{identity}$ of Theorem 1 [D.4], the aggravation at each layer l of the upper bound of Eq. (6) does not stem from the activation function itself but rather by the preceding affine transform. The phenomenon of channel-wise collapse — also referred as "domain collapse" [36] or "elimination singularity" [17] — is therefore not only induced by a "mean-shifting" activation function such as $\phi = \text{ReLU}$ [44, 18], but also by the injection of non-centeredness through the application of the channel-wise shift parameter $\boldsymbol{\beta}^l$ at each layer l. The fact that the general case of positive-homogeneous ϕ is upper-bounded by the case $\phi = \text{identity}$ in Eq. (6) still means that the choice $\phi = \text{ReLU}$ can only be an aggravating factor.

Crucially, in the context of random nets of large width, LN's operation at each layer l does not compensate this "mean shift". This stems from the fact that LN's mean and variance statistics can be approximated by zero and a constant value independent of $\mathbf{x}$, respectively. This means that LN's operation can be approximated by a layer-wise constant scaling independent of $\mathbf{x}$.[5]

The predominance of LN's failure mode in the hypothesis space — implied by its predominance in random nets — is expected to have at least two negative impacts on the actual learning and final performance: (i) being expected to "linger" along the training trajectory and being associated with reduced effective capacity, the failure mode is expected to cause degraded performance on the training loss; (ii) even if avoided to some extent during training, the failure mode is still expected to be present in the vicinity of the training trajectory, implying an ill-conditioning of the loss landscape and a prohibition of large learning rates that could have led otherwise to generalization benefits.

After detailing LN's failure mode, let us now detail IN's failure mode.

Theorem 2 (instance-normalized nets lack variability in instance statistics)**.** [E] *Fix a layer $l \in \{1, \dots, L\}$. Further suppose* $\text{Norm} = \text{IN}$ *and suppose that model parameters are chosen such that the denominator of Eq. (3) at layer l is nonzero for all inputs $\mathbf{x} \in \mathcal{D}$ and channels $c \in \{1, \dots, C_l\}$.*

Then it holds that

- $\mathbf{y}^l$ *is normalized in each channel c with*
$$\mathcal{P}_c^{(1)}(\mathbf{y}^l) = 0, \qquad \mathcal{P}_c(\mathbf{y}^l) = 1;$$
- $\mathbf{y}^l$ *lacks variability in instance statistics in each channel c with*
$$\mathcal{P}_c^{(2)}(\mathbf{y}^l) = 0, \qquad \mathcal{P}_c^{(3)}(\mathbf{y}^l) = 1, \qquad \mathcal{P}_c^{(4)}(\mathbf{y}^l) = 0.$$

Discussion on IN's failure mode. Theorem 2 directly implies two constraints that are not present in unnormalized networks: $\mathcal{P}^{(2)}(\mathbf{y}^l) = 0$ and $\mathcal{P}^{(4)}(\mathbf{y}^l) = 0$. By contrapositive, this necessarily means that the expressivity is altered since it would always be possible to achieve the same behavior of $\mathcal{P}^{(2)}(\mathbf{y}^l)$, $\mathcal{P}^{(4)}(\mathbf{y}^l)$ if the expressivity was preserved.

Another way of recognizing this alteration of the expressivity is by noting that some network mappings that are possible to express without normalization are not possible to express with IN. Let us take the example of a mapping that would provide in channel c through $\tilde{\mathbf{y}}^l_{\alpha,c}$, just before the nonlinearity, a "detector" of a given concept at position α in the input $\mathbf{x}$. With IN, the lack of variability in instance statistics implies that the mean $\mu_{\mathbf{x},c}(\tilde{\mathbf{y}}^l)$ and standard deviation $\sigma_{\mathbf{x},c}(\tilde{\mathbf{y}}^l)$ of the feature map in channel c are necessarily constant for all $\mathbf{x}$, equal to $\boldsymbol{\beta}_c$ and $\boldsymbol{\gamma}_c$, respectively. This does not allow to express for some inputs $\mathbf{x}$ the presence of the concept at some position α: $\mu_{\mathbf{x},c}(\tilde{\mathbf{y}}^l) > 0$, $\sigma_{\mathbf{x},c}(\tilde{\mathbf{y}}^l) > 0$; and for other inputs $\mathbf{x}$ the absence of the concept: $\mu_{\mathbf{x},c}(\tilde{\mathbf{y}}^l) = 0$, $\sigma_{\mathbf{x},c}(\tilde{\mathbf{y}}^l) = 0$.

Such an example is not just anectodal. Indeed, it is accepted that networks trained on high-level conceptual tasks have their initial layers related to low-level features and their deep layers related to

[5] In this view, we would expect layer-normalized networks to be equally subject to a phenomenon whereby channels become increasingly imbalanced with depth. This phenomenon would be driven not only by the succession of affine transforms and activation functions, but also by the succession of convolutions [42, 59, 45].

high-level concepts [60, 61]. This view explains the success of IN on the specific task of style transfer with fixed "style" input, IN being then incorporated inside a generator network that only acts on the low-level features of the "content" input [6, 62, 63, 64]. On high-level conceptual tasks, however, this view hints at a harmful tension between IN's constraints and the requirement of instance variability to express high-level concepts in deep layers. In short, not only is the expressivity altered with IN, but the alteration of the expressivity results in the exclusion of useful network mappings.

Failure modes with GN. Group Normalization is a middle ground between the two extremes of LN ($G = 1$ group) and IN ($G = C_l$ groups at layer l). Networks with GN are consequently affected by both problems of Theorem 1 and Theorem 2, but to a lesser extent than networks with LN and IN. On the one hand, since GN becomes equally equivalent to a constant scaling when group sizes become large, networks with GN are likely to be subject to channel-wise collapse. On the other hand, since GN can be seen as removing a fraction — inversely dependent on the group size — of $\mathcal{P}^{(2)}(\mathbf{y}^l)$, $\mathcal{P}^{(4)}(\mathbf{y}^l)$ in between each layer, networks with GN are likely to lack variability in instance statistics.

The balance stricken by GN between the two failure modes of LN and IN could still be beneficial, which would explain GN's superior performance in practice. It makes sense intuitively that being subject to two failure modes weakly is preferable over being subject to one failure mode strongly.

Experimental validation. Figure 3 shows $\mathcal{P}^{(1)}(\mathbf{y}^l)$, $\mathcal{P}^{(2)}(\mathbf{y}^l)$, $\mathcal{P}^{(3)}(\mathbf{y}^l)$, $\mathcal{P}^{(4)}(\mathbf{y}^l)$ as a function of the depth l during the training of ResNet-50 (v2) [65] with notably BN, LN, GN on ImageNet [66]. With these "power plots", we confirm: (i) the tendency of networks with LN and — to a lesser extent — GN to be subject to channel-wise collapse as depth increases; (ii) the lack of variability in instance statistics, particularly at low depth, in networks with GN compared to networks with BN and LN.

5 Proxy Normalization

With the goal of remedying the failure modes of Section 4, we now introduce our proposed approach of batch-independent normalization. The rationale that supports our approach is simple: if convolutions do not play any role in the channel-wise collapse and if LN or GN can be approximated as constant scalings independent of $\mathbf{x}$, then the channel-wise collapse can be altogether avoided by compensating at each layer l the mean shift associated with the affine transform *and* the activation function ϕ.

To achieve this mean shift compensation, we introduce the technique "Proxy Normalization" (PN). PN amounts for each channel c to normalizing $\phi\big(\boldsymbol{\gamma}^l_c \mathbf{y}^l_{\alpha,c} + \boldsymbol{\beta}^l_c\big)$ by assimilating it with $\phi\big(\boldsymbol{\gamma}^l_c Y^l_c + \boldsymbol{\beta}^l_c\big)$, where Y^l_c is a Gaussian "proxy" variable with mean $\tilde{\boldsymbol{\beta}}^l_c$ and variance $(1 + \tilde{\boldsymbol{\gamma}}^l_c)^2$ derived from the additional parameters $\tilde{\boldsymbol{\beta}}^l, \tilde{\boldsymbol{\gamma}}^l \in \mathbb{R}^{C_l}$.

By letting $\tilde{\boldsymbol{\beta}}^l, \tilde{\boldsymbol{\gamma}}^l$ be subject to weight decay, Y^l is further kept close to channel-wise normalized. This ensures that the mean shift compensation is effective under the iterative assumption that $\mathbf{y}^l$ itself remains close to channel-wise normalized after the norm. Importantly, unlike with IN in activation space or WS in weight space, PN's mean shift compensation does not come at the cost of an alteration of the expressivity.

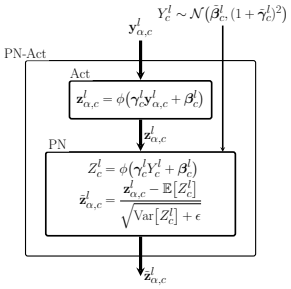

Figure 2: PN is plugged in on the activation step Act to yield the proxy-normalized activation step $\mathrm{PN\text{-}Act}$.

Formally, PN is incorporated in the network by replacing the activation step of Eq. (4) by the following "proxy-normalized" activation step (cf Figure 2 and Appendix B):

$$\tilde{\mathbf{z}}^l = \mathrm{PN\text{-}Act}(\mathbf{y}^l), \qquad \forall \alpha, c: \quad \mathrm{PN\text{-}Act}(\mathbf{y}^l)_{\alpha,c} = \left(\frac{\phi\big(\boldsymbol{\gamma}^l_c \mathbf{y}^l_{\alpha,c} + \boldsymbol{\beta}^l_c\big) - \mathbb{E}_{Y^l_c}\big[\phi\big(\boldsymbol{\gamma}^l_c Y^l_c + \boldsymbol{\beta}^l_c\big)\big]}{\sqrt{\mathrm{Var}_{Y^l_c}\big[\phi\big(\boldsymbol{\gamma}^l_c Y^l_c + \boldsymbol{\beta}^l_c\big)\big] + \epsilon}} \right)$$

where $Y^l_c \sim \mathcal{N}\big(\tilde{\boldsymbol{\beta}}^l_c, (1 + \tilde{\boldsymbol{\gamma}}^l_c)^2\big)$ and $\epsilon \geq 0$ is the numerical stability constant of PN.

PN is a generic technique that can be "plugged in" on any activation function ϕ. Unlike previous approaches based on the tracking of activation statistics [19, 20, 21], PN requires a local tracking of activation statistics only locally around ϕ. This makes it "cheap" and tractable even in residual networks. The only required constraint for PN to work optimally is that each activation step should be immediately preceded by a normalization step.

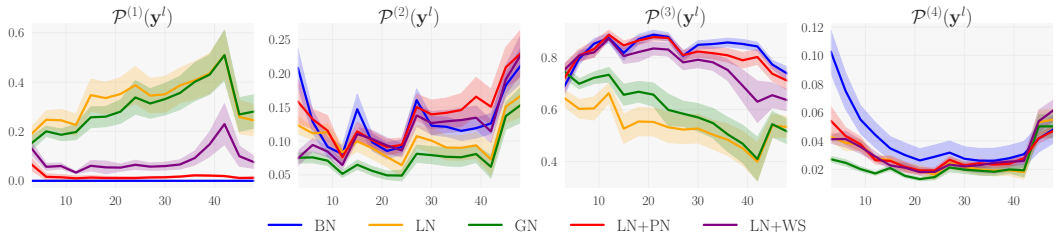

Figure 3: $\mathcal{P}^{(1)}(\mathbf{y}^l)$, $\mathcal{P}^{(2)}(\mathbf{y}^l)$, $\mathcal{P}^{(3)}(\mathbf{y}^l)$, $\mathcal{P}^{(4)}(\mathbf{y}^l)$ as a function of the depth l in ResNet-50 (v2) with different choices of norm. We show the means (solid lines) as well as 1σ intervals (shaded regions) over 100 epochs of training on ImageNet. Further experimental details are reported in Appendix A.2.

Experimental validation. As visible in Figure 3, the power plots of networks with LN+PN resemble the power plots of networks with BN. This confirms the view that PN remedies LN's failure mode without incurring IN's failure mode. Importantly, *this also means that BN's behavior can be emulated in a fully batch-independent manner with LN+PN or GN+PN*.

Figure 3 also outlines the fact that networks with LN+WS are less effective in avoiding the channel-wise collapse. This stems from the implicit reliance of WS on the assumption that different channels have the same channel-wise means after ϕ. While this assumption is valid at initialization, it becomes less valid as the affine transform starts deviating from the identity during training [A.2]. A similar inefficiency in mean shift compensation would be expected with Centered Weight Normalization [16], PreLayerNorm [46] or Normalization Propagation [19]. *In short, the properties of the neural network at initialization should not be regarded as fully representative of the whole training.*

6 Results with Proxy Normalization

We finally evaluate the practical performance of our approach of batch-independent normalization. As mentioned in Section 2, we perform all our experiments with two different degrees of regularization to disentangle normalization's effects from regularization's effects. While we focus on ImageNet in the main text of this paper, we report additional results on CIFAR in Appendix A. We also report all our experiments details as well as all our experimental error bars in Appendix A.

For starters, we analyze the effect of adding PN on top of various norms in ResNet-50 (v2).[6] As visible in Table 2, PN is mostly beneficial when added on top of LN or on top of GN with a small number of groups G. The consequence is that the optimal G gets shifted to lower values in GN+PN compared to GN. This confirms the view that PN's benefit lies in addressing LN's failure mode.

Table 2: ResNet-50 (RN50) trained on ImageNet with various norms: BN and LN, GN, GN+WS with G groups, either without or with PN added on top (plain vs. +PN). Results are formatted as X / Y with X, Y the validation accuracies (%) without and with extra regularization, respectively.

		RN50	
	G	plain	+PN
BN		**76.3** / 75.8	76.2 / 76.0
LN	1	74.5 / 74.6	75.9 / 76.5
GN	8	75.4 / 75.4	76.3 / **76.7**
GN	32	75.4 / 75.3	75.8 / 76.1
GN+WS	8	76.6 / 76.7	76.8 / **77.1**

It can also be seen in Table 2 that PN does not provide noticeable benefits to BN. This confirms again the view that PN's benefit lies in addressing the problem — not present with BN — of channel-wise collapse. Importantly, since PN does not entail effects others than normalization that could artificially boost the performance, *GN+PN can be "fairly" compared to BN when evaluating the effectiveness of normalization*. This is unlike WS which has been shown to improve BN's performance [17]. In our results of Table 2, the high performance of GN+WS without extra regularization and the fact that PN still provides benefits to GN+WS suggests that: (i) on top of its normalization benefits, WS entails a form of regularization; (ii) GN+WS is still not fully optimal in terms of normalization.

Next we compare GN+PN's performance to BN's performance across a broad range of models trained on ImageNet. As visible in Figure 4 and Tables 3 and 4, GN+PN outperforms BN in ResNet-50 and

[6] To ensure that activation steps are directly preceded by normalization steps (cf Section 5), we always use v2 instantiations of ResNets and instantiations of ResNeXts having the same block reorderings as ResNets v2 [B].

ResNet-101 [65], matches BN in ResNeXt-50 and ResNeXt-101 [67], and matches BN in EfficientNet-B0 and EfficientNet-B2, both in the original variant with depthwise convolutions and expansion ratio of 6 [50] and in a variant approximately parameter-preserving with group convolutions of group size 16 and expansion ratio of 4 [A.1]. In short, *our approach of batch-independent normalization enables us to match BN not only in behavior but also in performance.*

With regard to matching BN's performance with alternative norms, various positive results have been reported in ResNets or ResNeXts [3, 4, 8, 9, 13, 18, 34] but only few in EfficientNets [13]. We are notably not aware of any batch-independent norm being found in another work to match BN's performance in EfficientNets. As a confirmation, we assess the performance of various batch-independent norms that do not make use of PN: GN [10], GN+WS [17], Evo-S0 [13], FRN+TLU [68, 11]. Unlike GN+PN, none of these norms is found in Table 4 to match BN with consistency in EfficientNets.

Overall, our results suggest that while an efficient normalization is not sufficient in itself to achieve good performance on ImageNet, it is still a necessary condition, together with regularization. In our results, it is always with extra regularization that PN yields the most benefits. Importantly, the fact that PN consistently leads to large improvements in training accuracy [A.3.2] *suggests that additional benefits would be obtained on larger datasets without having to rely on regularization [69, 70].*

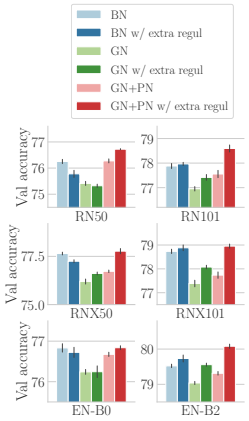

Figure 4: Various models trained on ImageNet with BN, GN, GN+PN without and with extra regularization. The same shorthand notations are used as in Tables 3 and 4. EfficientNets are considered in the variant with group convolutions [A.1].

Table 3: ResNet-50 (RN50), ResNet-101 (RN101), ResNeXt-50 (RNX50), ResNeXt-101 (RNX101) trained on ImageNet with different choices of norm. Results are formatted as in Table 2.

	RN50	RN101	RNX50	RNX101
BN	**76.3** / 75.8	77.9 / **78.0**	**77.6** / 77.2	78.7 / **78.9**
GN	75.4 / 75.3	77.0 / 77.4	76.2 / 76.6	77.4 / 78.1
GN+PN	76.3 / **76.7**	77.6 / **78.6**	76.7 / **77.8**	77.7 / **79.0**

Table 4: EfficientNet-B0 (EN-B0) and EfficientNet-B2 (EN-B2), in variants with depthwise convolutions (left) [50] and with group convolutions [A.1] (right), trained on ImageNet with different choices of norm. Results are formatted as in Table 2.

	depthwise convs		group convs	
	EN-B0	EN-B2	EN-B0	EN-B2
BN	76.9 / **77.2**	79.4 / **80.0**	**76.8** / 76.7	79.5 / **79.7**
GN	76.2 / 76.2	78.9 / 79.4	76.2 / 76.2	79.0 / 79.6
GN+PN	76.8 / **77.0**	79.3 / **80.0**	76.7 / **76.8**	79.3 / **80.1**
GN+WS	74.2 / 74.1	77.8 / 77.8	76.2 / 76.3	79.2 / 79.4
Evo-S0	75.8 / 75.8	78.5 / 78.7	76.2 / 76.5	78.9 / 79.6
FRN+TLU	75.7 / 75.7	78.4 / 78.8	74.9 / 75.1	78.2 / 78.6

7 Summary and broader impact

We introduced a novel framework to finely characterize the various neural network properties affected by the choice of normalization. Using this framework, we showed that while BN's beneficial properties are not retained when solely using the prototypical batch-independent norms, they are retained when combining these prototypical norms with the technique hereby introduced of Proxy Normalization. We demonstrated on an extensive set of experiments that our novel approach of batch-independent normalization enables us to consistently match BN in both behavior and performance.

This work could potentially have wide implications since it unlocks the possibility to retain BN's normalization benefits while removing batch dependence. Firstly, our approach could be used to retain BN's normalization benefits while alleviating the burden of large activation memory stemming from BN's requirement of large enough batch sizes. This is expected to be important in memory-intensive applications such as object detection or image segmentation, but also when using A.I. accelators that leverage local memory to provide extra acceleration and energy savings in exchange for tighter memory constraints. Secondly, our approach could be used to retain BN's normalization benefits while getting rid of BN's regularization when the latter is detrimental. As discussed in Section 6, this is expected to be important in the context — that will likely be prevalent in the future — of large datasets.

References

[1] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 448–456, 2015.

[2] Sergey Ioffe. Batch renormalization: Towards reducing minibatch dependence in batch-normalized models. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 1945–1953, 2017.

[3] Vitaliy Chiley, Ilya Sharapov, Atli Kosson, Urs Köster, Ryan Reece, Sofia Samaniego de la Fuente, Vishal Subbiah, and Michael James. Online normalization for training neural networks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 8431–8441, 2019.

[4] Junjie Yan, Ruosi Wan, Xiangyu Zhang, Wei Zhang, Yichen Wei, and Jian Sun. Towards stabilizing batch statistics in backward propagation of batch normalization. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[5] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016.

[6] Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *CoRR*, abs/1607.08022, 2016.

[7] Mengye Ren, Renjie Liao, Raquel Urtasun, Fabian H. Sinz, and Richard S. Zemel. Normalizing the normalizers: Comparing and extending network normalization schemes. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[8] Ping Luo, Jiamin Ren, Zhanglin Peng, Ruimao Zhang, and Jingyu Li. Differentiable learning-to-normalize via switchable normalization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[9] Ping Luo, Zhanglin Peng, Wenqi Shao, Ruimao Zhang, Jiamin Ren, and Lingyun Wu. Differentiable dynamic normalization for learning deep representation. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 4203–4211. PMLR, 2019.

[10] Yuxin Wu and Kaiming He. Group normalization. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XIII*, pages 3–19, 2018.

[11] Saurabh Singh and Shankar Krishnan. Filter response normalization layer: Eliminating batch dependence in the training of deep neural networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 11234–11243. IEEE, 2020.

[12] Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 12360–12371, 2019.

[13] Hanxiao Liu, Andy Brock, Karen Simonyan, and Quoc Le. Evolving normalization-activation layers. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[14] Cecilia Summers and Michael J. Dinneen. Four things everyone should know to improve batch normalization. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, Apr 26-30, 2020*, 2020.

[15] Tim Salimans and Diederik P. Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, page 901, 2016.

[16] Lei Huang, Xianglong Liu, Yang Liu, Bo Lang, and Dacheng Tao. Centered weight normalization in accelerating training of deep neural networks. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2822–2830. IEEE Computer Society, 2017.

[17] Siyuan Qiao, Huiyu Wang, Chenxi Liu, Wei Shen, and Alan L. Yuille. Weight standardization. *CoRR*, abs/1903.10520, 2019.

[18] Andrew Brock, Soham De, and Samuel L Smith. Characterizing signal propagation to close the performance gap in unnormalized resnets. In *International Conference on Learning Representations*, 2021.

[19] Devansh Arpit, Yingbo Zhou, Bhargava Urala Kota, and Venu Govindaraju. Normalization propagation: A parametric technique for removing internal covariate shift in deep networks. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1168–1176. JMLR.org, 2016.

[20] César Laurent, Nicolas Ballas, and Pascal Vincent. Recurrent normalization propagation. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net, 2017.

[21] Alexander Shekhovtsov and Boris Flach. Normalization of neural networks using analytic variance propagation. *CoRR*, abs/1803.10560, 2018.

[22] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 971–980, 2017.

[23] Twan van Laarhoven. L2 regularization versus batch and weight normalization. *CoRR*, abs/1706.05350, 2017.

[24] Sanjeev Arora, Zhiyuan Li, and Kaifeng Lyu. Theoretical analysis of auto rate-tuning by batch normalization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[25] Elad Hoffer, Ron Banner, Itay Golan, and Daniel Soudry. Norm matters: efficient and accurate normalization schemes in deep networks. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 2164–2174, 2018.

[26] Zhiyuan Li and Sanjeev Arora. An exponential learning rate schedule for deep learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[27] Guodong Zhang, Chaoqi Wang, Bowen Xu, and Roger B. Grosse. Three mechanisms of weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[28] Ruosi Wan, Zhanxing Zhu, Xiangyu Zhang, and Jian Sun. Spherical motion dynamics of deep neural networks with batch normalization and weight decay. *CoRR*, abs/2006.08419, 2020.

[29] Minhyung Cho and Jaehyung Lee. Riemannian approach to batch normalization. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5225–5235, 2017.

[30] Jonas Moritz Kohler, Hadi Daneshmand, Aurélien Lucchi, Thomas Hofmann, Ming Zhou, and Klaus Neymeyr. Exponential convergence rates for batch normalization: The power of length-direction decoupling in non-convex optimization. In *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*, volume 89 of *Proceedings of Machine Learning Research*, pages 806–815. PMLR, 2019.

[31] Boris Hanin and David Rolnick. How to start training: The effect of initialization and architecture. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 569–579, 2018.

[32] Hongyi Zhang, Yann N. Dauphin, and Tengyu Ma. Fixup initialization: Residual learning without normalization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[33] Soham De and Samuel L. Smith. Batch normalization biases residual blocks towards the identity function in deep networks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[34] Jie Shao, Kai Hu, Changhu Wang, Xiangyang Xue, and Bhiksha Raj. Is normalization indispensable for training deep neural network? In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[35] Andreas Veit, Michael J. Wilber, and Serge J. Belongie. Residual networks behave like ensembles of relatively shallow networks. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 550–558, 2016.

[36] George Philipp, Dawn Song, and Jaime G. Carbonell. Gradients explode - deep networks are shallow - resnet explained. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*. OpenReview.net, 2018.

[37] Greg Yang, Jeffrey Pennington, Vinay Rao, Jascha Sohl-Dickstein, and Samuel S. Schoenholz. A mean field theory of batch normalization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[38] Antoine Labatie. Characterizing well-behaved vs. pathological deep neural networks. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 3611–3621. PMLR, 2019.

[39] Mattias Teye, Hossein Azizpour, and Kevin Smith. Bayesian uncertainty estimation for batch normalized deep networks. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4914–4923. PMLR, 2018.

[40] Alexander Shekhovtsov and Boris Flach. Stochastic normalizations as bayesian learning. In *Computer Vision - ACCV 2018 - 14th Asian Conference on Computer Vision, Perth, Australia, December 2-6, 2018, Revised Selected Papers, Part II*, volume 11362 of *Lecture Notes in Computer Science*, pages 463–479. Springer, 2018.

[41] Ping Luo, Xinjiang Wang, Wenqi Shao, and Zhanglin Peng. Towards understanding regularization in batch normalization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.

[42] Johan Bjorck, Carla P. Gomes, Bart Selman, and Kilian Q. Weinberger. Understanding batch normalization. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 7705–7716, 2018.

[43] Arthur Jacot, Franck Gabriel, and Clément Hongler. Freeze and chaos for dnns: an ntk view of batch normalization, checkerboard and boundary effects. *CoRR*, abs/1907.05715, 2019.

[44] David Page. How to train your resnet 7: Batch norm, 2019.

[45] Hadi Daneshmand, Jonas Moritz Kohler, Francis R. Bach, Thomas Hofmann, and Aurélien Lucchi. Batch normalization provably avoids ranks collapse for randomly initialised deep networks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[46] Vinay Rao and Jascha Sohl-Dickstein. Is batch norm unique? an empirical investigation and prescription to emulate the best properties of common normalizers without batch dependence. *CoRR*, abs/2010.10687, 2020.

[47] Boris Hanin and David Rolnick. Complexity of linear regions in deep networks. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2596–2604. PMLR, 2019.

[48] Boris Hanin and David Rolnick. Deep relu networks have surprisingly few activation patterns. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 359–368, 2019.

[49] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 2488–2498, 2018.

[50] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 2019.

[51] Cihang Xie, Mingxing Tan, Boqing Gong, Jiang Wang, Alan L. Yuille, and Quoc V. Le. Adversarial examples improve image recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 816–825, 2020.

[52] Cihang Xie and Alan L. Yuille. Intriguing properties of adversarial training at scale. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[53] Takeru Miyato and Masanori Koyama. cgans with projection discriminator. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[54] Harm de Vries, Florian Strub, Jérémie Mary, Hugo Larochelle, Olivier Pietquin, and Aaron C. Courville. Modulating early visual processing by language. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6594–6604, 2017.

[55] Lucas Deecke, Iain Murray, and Hakan Bilen. Mode normalization. In *International Conference on Learning Representations*, 2019.

[56] Ximei Wang, Ying Jin, Mingsheng Long, Jianmin Wang, and Michael I. Jordan. Transferable normalization: Towards improving transferability of deep neural networks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 1951–1961, 2019.

[57] Woong-Gi Chang, Tackgeun You, Seonguk Seo, Suha Kwak, and Bohyung Han. Domain-specific batch normalization for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 7354–7362. Computer Vision Foundation / IEEE, 2019.

[58] Yuxin Wu and Justin Johnson. Rethinking "batch" in batchnorm, 2021.

[59] Jeffrey Pennington and Pratik Worah. Nonlinear random matrix theory for deep learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 2637–2646, 2017.

[60] Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor S. Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1349–1357. JMLR.org, 2016.

[61] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2414–2423. IEEE Computer Society, 2016.

[62] Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 4105–4113. IEEE Computer Society, 2017.

[63] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[64] Xun Huang and Serge J. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 1510–1519. IEEE Computer Society, 2017.

[65] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016*, 2016.

[66] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[67] Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, 2017.

[68] Sitao Xiang and Hao Li. On the effects of batch and weight normalization in generative adversarial networks, 2017.

[69] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020*, 2020.

[70] Andrew Brock, Soham De, Samuel L. Smith, and Karen Simonyan. High-performance large-scale image recognition without normalization. *CoRR*, abs/2102.06171, 2021.

[71] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2818–2826. IEEE Computer Society, 2016.

[72] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[73] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q. Weinberger. Deep networks with stochastic depth. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV*, volume 9908, pages 646–661. Springer, 2016.

[74] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[75] Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Seong Joon Oh, Youngjoon Yoo, and Junsuk Choe. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 6022–6031. IEEE, 2019.

[76] Ekin D. Cubuk, Barret Zoph, Dandelion Mané, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation strategies from data. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 113–123. Computer Vision Foundation / IEEE, 2019.

[77] Jasmine Collins, Johannes Ballé, and Jonathon Shlens. Accelerating training of deep neural networks with a standardization loss. *CoRR*, abs/1903.00925, 2019.

[78] Yann Dauphin and Ekin Dogus Cubuk. Deconstructing the regularization of batchnorm. In *International Conference on Learning Representations*, 2021.

[79] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] We believe the paper's title and abstract are consistent with the theoretical results of Section 4 and the experimental results of Sections 5 and 6.
   (b) Did you describe the limitations of your work? [Yes] The scope and assumptions of our theoretical results are provided in Sections 3 and 4. The limitation stemming from PN's requirement that each activation step should be directly preceded by a normalization step is stated in Section 4.
   (c) Did you discuss any potential negative societal impacts of your work? [No] Our work directly impacts the advancement of the field of computer vision. Its negative impacts could come only indirectly from the negative impacts of this field in general. We thought that this observation was not needed in the main text, as it is already implied.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] The full assumptions of our theoretical results are provided in Sections 3 and 4.
   (b) Did you include complete proofs of all theoretical results? [Yes] Complete proofs of all our theoretical results are provided in Appendix C, D and E.
3. If you ran experiments...

(a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The experimental results can be reproduced based on Appendix A.1 and B.

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] All the experimental details are reported in Appendix A.1

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] The error bars associated with each of our experiments are reported in Appendix **??**.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No] This information will be included in future versions of the paper.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes]

(b) Did you mention the license of the assets? [No] ImageNet's images are owned by copyright holders. We are unsure about the existence, and potentially the type, of licenses of CIFAR-10 and CIFAR-100.

(c) Did you include any new assets either in the supplemental material or as a URL? [N/A]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]