Project 2: Improving Batch Normalization via Scaling and Shifting Relay

Shizhe Diao, Jincheng Yu, Duo Li, Yimin Zheng

Department of Computer Science and Engineering, HKUST

Introduction

Normalization operation is an indispensable ingredient of modern deep neural networks. Although numerous normalization methods have been explored, there are few techniques that enhance the network performance by dynamically generating scale and shift parameters of normalization layers.

In this work, we encapsulate the normalization scheme into a meta learning framework, giving rise to HyperNormalization (HyperNorm that is conditioned on the progressive feature feed-forward process. On the one hand, HN leverages affine transformation adaptively parameterized by a trimmed meta LSTM to readjust the sample-specific distribution of feature representation in each layer, resolving possibly inaccurate mini-batch statistics estimation under micro-batch scenarios. On the other hand, HN enables scaling and shifting relay which associates feature distributions across the same staged layers for improving the correlations of their representation, inherently promoting the quality of generated parameters. HN could readily enhance existing normalization in a plug-and-play manner. We integrate HN operation into state-of-the-art network architectures and show that our method outperforms BN considerably on ImageNet and CIFAR.

Revisiting Normalization

Considering the input feature map $\mathbf{x} \in \mathbb{R}^{N \times C \times H \times W}$ of a generic normalization layer (N, C, H, W indicate batch size, number of channels, height and width of a tensor respectively)

standardization procedure

$$\hat{\mathbf{x}} = \frac{\mathbf{x} - \boldsymbol{\mu}}{\boldsymbol{\sigma}}$$

affine transformation

$$\mathbf{y} = \boldsymbol{\gamma}\hat{\mathbf{x}} + \boldsymbol{\beta}$$

It is widespread that γ and β are updated with Stochastic Gradient Descent (SGD) without additional constraint.

HyperNormalization

We employ a shared Recurrent Neural Network (RNN) to aggregate and relay information across different building blocks in the same stage, as illustrated in Figure 1.

$$\mathbf{h}_l, \mathbf{c}_l = \mathrm{LSTM}(\mathrm{GAP}(\mathbf{x}_l), \mathbf{h}_{l-1}, \mathbf{c}_{l-1})$$

The hidden and cell state corresponding to the l-th building block are denoted as hl and cl respectively. where GAP(·) represents the operation of Global Average Pooling (GAP) which gathers the contextual information for individual channels of a feature tensor and reduces the follow-up computational consumption.

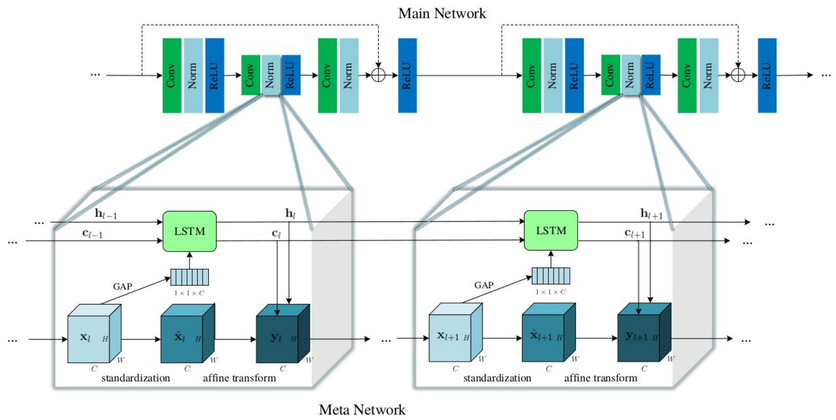

Figure 1. Schematic illustration of HyperNormalization

To guarantee the range of hidden states to vary between zero and one, the hyperbolic tangent activation function -> the sigmoid function

bottleneck structure $\mathbf{v} = \mathbf{W}_1\delta(\mathbf{W}_0\mathbf{u})$

Final Representation $\mathbf{y}_l = \mathbf{h}_l\hat{\mathbf{x}}_l + \mathbf{c}_l$

Experiments - CIFAR-100

Architecture	Norm Type	Top-1 Error(%)
ResNet-164	BN (*official*)†	25.16
	BN (*self impl.*)†	24.876 ± 0.143
	BN (*self impl.*)	26.328 ± 0.587
	BHN	$\mathbf{23.278 \pm 0.306}$
ResNeXt-29 (8x64d)	BN (*official*)	17.77
	BN (*self impl.*)	17.978 ± 0.147
	BHN	$\mathbf{16.974 \pm 0.233}$
ResNeXt-29 (16x64d)	BN (*official*)	17.31
	BN (*self impl.*)	17.538 ± 0.094
	BHN	$\mathbf{16.280 \pm 0.252}$

Table 1. Top-1 error comparisons on the CIFAR-100 test

Experiments - MS-COCO 2017

Backbone	Neck/Head	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
$\mathbb{BN}$	-	36.4	58.4	39.1	21.5	40.0	46.6
SyncBN	-	36.3	58.4	39.0	21.5	40.1	46.8
GN	-	36.6	58.9	39.3	21.6	40.5	47.0
BHN	-	37.5	59.6	40.4	22.0	41.2	48.8
$\mathbb{BN}$	BN	35.1	54.5	38.3	19.2	39.1	47.1
$\mathbb{BN}$	SyncBN	37.7	58.8	41.4	22.1	40.8	49.1
$\mathbb{BN}$	GN	37.8	59.1	41.0	22.6	41.4	48.7
$\mathbb{BN}^\dagger$	$SN^\dagger$	38.0	59.4	41.5	22.7	41.3	48.9
$\mathbb{BN}$	BHN	38.6	59.5	41.9	21.6	42.0	50.4
BN	BN	32.9	51.8	35.6	18.2	36.8	44.1
SyncBN	SyncBN	38.3	59.2	41.8	21.5	41.9	50.5
GN	GN	37.9	59.6	40.7	21.9	41.2	48.5
$SN^\dagger$	$SN^\dagger$	39.3	60.9	42.8	23.5	42.7	50.3
BHN	BHN	38.7	59.1	42.0	22.5	41.3	51.8

Table 2. Bounding-box Average Precision (AP^{bbox}) comparison of Faster R-CNN with ResNet-50 and FPN on the COCO 2017 validation set.

Experiments - ImageNet-1K

We perform extensive experiments on the large-scale ImageNet benchmark, with both heavyweight and lightweight neural architectures, like (SE-)ResNet-50 and ShuffleNetV2.

Architecture	Params	FLOPs	Norm	Top-1/Top-5 Err.(%)
ResNet-18	11.690M	1.814G	BN	29.534 / 10.174
	12.564M	1.816G	BHN	**28.414 / 9.704**
ResNet-50	25.557M	4.089G	BN	23.694 / 6.976
	26.428M	4.092G	BHN	**22.480 / 6.208**
ResNet-101	44.549M	7.801G	BN	22.208 / 6.144
	45.411M	7.807G	BHN	**21.048 / 5.684**
ShuffleNetV2 1×	2.278M	144.916M	BN	30.692 / 11.122
	2.455M	145.578M	BHN	**29.516 / 10.604**
SE-ResNet-50	28.088M	4.091G	BN	22.666 / 6.434
	28.959M	4.095G	BHN	**21.808 / 5.954**

Table 3. ImageNet Classification results

Analyses

Viewed from the training curves in Figure 2, BHN guarantees more stable optimization and faster convergence.

We explore the head-to-head comparison between BN and BHN across a spectrum of mini-batch sizes. The corresponding test errors are visualized in Figure 3.

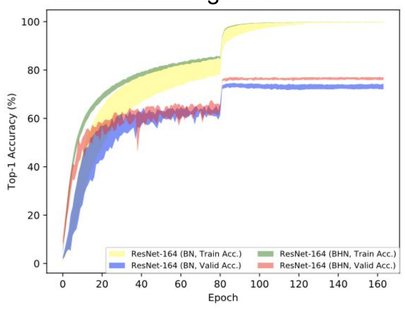

Figure 2. Curves of top-1 training and validation accuracy on CIFAR-100.

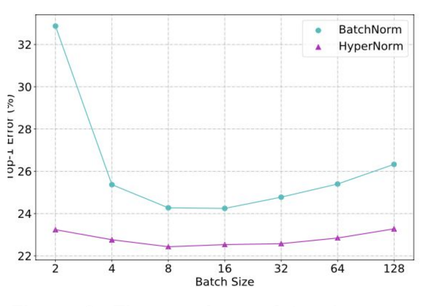

Figure 3. The envelope of top-1 error on CIFAR-100 using ResNet- 164 model with respect to mini-batch size.

Contribution

Shizhe Diao: Implementation, Experiments, Result Analysis, Poster

Jincheng Yu: Experiments, Method, Presentation

Duo Li: Method, Experiments. Poster

Yimin Zheng: Introduction, Method, Experiment. Presentation

Reference

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In ICML, 2015