Instance Enhancement Batch Normalization: an Adaptive Regulator of Batch Noise

Senwei Liang[1*], **Zhongzhan Huang**[2*], **Mingfu Liang**[3], **Haizhao Yang**[1,4]
[1]National University of Singapore
[2]New Oriental AI Research Academy
[3]Northwestern University
[4]Purdue University
liangsenwei@u.nus.edu, hzz_dedekinds@foxmail.com,
mingfuliang2020@u.northwestern.edu, matyh@nus.edu.sg

Abstract

Batch Normalization (BN) (Ioffe and Szegedy 2015) normalizes the features of an input image via statistics of a batch of images and this batch information is considered as batch noise that will be brought to the features of an instance by BN. We offer a point of view that self-attention mechanism can help regulate the batch noise by enhancing instance-specific information. Based on this view, we propose combining BN with a self-attention mechanism to adjust the batch noise and give an attention-based version of BN called Instance Enhancement Batch Normalization (IEBN) which recalibrates channel information by a simple linear transformation. IEBN outperforms BN with a light parameter increment in various visual tasks universally for different network structures and benchmark data sets. Besides, even if under the attack of synthetic noise, IEBN can still stabilize network training with good generalization. The code of IEBN is available at https://github.com/gbup-group/IEBN

Introduction

Mini-batch Stochastic Gradient Descent (SGD) is a simple and effective method in large-scale optimization by aggregating multiple samples at each iteration to reduce operation and memory cost. However, SGD is sensitive to the choice of hyperparameters and it may cause training instability (Luo, Xiong, and Liu 2019). Normalization is one possible choice to remedy SGD methods for better stability and generalization. Batch Normalization (BN) (Ioffe and Szegedy 2015) is a frequently-used normalization method that normalizes the features of an image using the mean and variance of the features of a batch of images during training. Meanwhile, the tracked mean and variance that estimate the statistics of the whole dataset are used for normalization during testing. It has been shown that BN is an effective module to regularize parameters (Luo et al. 2019), stabilize training, smooth gradients (Santurkar et al. 2018), and enable a larger learning rate (Bjorck et al. 2018; Cai, Li, and Shen 2019) for faster convergence.

Two kinds of noise effects in SGD and BN are concerned in this paper.

Estimation Noise. In BN, the mean and variance of a batch are used to estimate those of the whole dataset; in SGD, the gradient of the loss over the batch is applied to approximate that of the whole dataset. These estimation errors are called estimation noise.

Batch Noise. In the forward pass, BN incorporates batch information to the features of an instance via the normalization with batch statistics. In the back-propagation, the gradient of an instance will be disturbed by the batch information due to BN and SGD. These disturbances to an instance caused by the batch is referred to as batch noise.

The randomness of BN and SGD has been well-known to improve the performance of deep networks and there exists extensive study on optimizing their effeteness via tuning batch sizes. On the one hand, a small batch size will lead to a high variance of statistics and weaken the training stability. On the other hand, a large batch size can reduce the estimation noise but it will cause a sharp landscape of loss (Keskar et al. 2016) making the optimization problem more challenging. Therefore, it is important to choose an appropriate batch size to make a good balance but the noise still exists. These two kinds of noise will finally influence the gradient when performing a forward pass and back-propagation. In fact, the appropriate estimation noise and batch noise can benefit the generalization of the network. BN with the estimation noise can work as an adaptive regularizer of parameters (Luo et al. 2019) and the moderate noise can help escape bad local minima and saddle point (Jin et al. 2017; Ge et al. 2015).

It is an art to infuse a model with the appropriate noise. We argue that self-attention mechanism is an adaptive noise regulator for the model by enhancing instance specificity. The appropriate noise enables a model with BN to ease optimization and benefit generalization, which motivates us to design a new normalization to combine the advantage of BN and self-attention. This paper proposes an attention-based BN which adaptively emphasizes instance information called as Instance Enhancement Batch Normalization (IEBN). The idea behind IEBN is simple. As shown in Fig. 1, IEBN extracts the instance statistic of a channel before BN and applies it to rescale the output channel of BN with a pair of additional parameters. IEBN costs a light

*Equal contribution
Work in progress

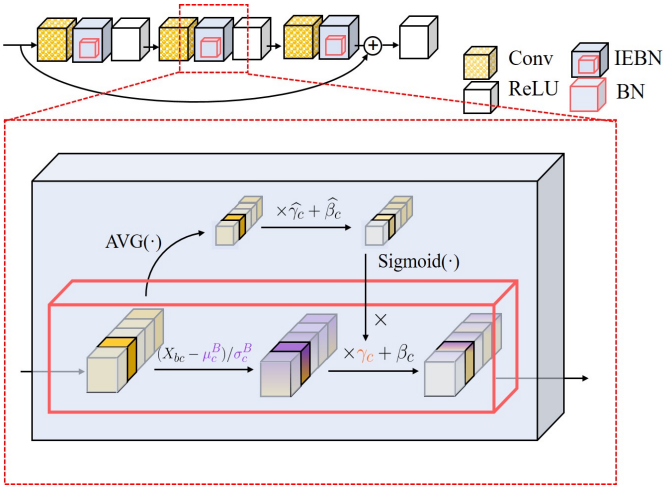

Figure 1: The illustration of IEBN. The top shows a block of ResNet. The bottom is the showcase of IEBN, where the box with red border is the basic flow of BN. AVG(·) means the average pooling over a channel and Sigmoid(·) is sigmoid function.

parameter increment and a low computation complexity increment. The extended experiment shows that IEBN outperforms BN on benchmark datasets over popular architectures for image classification. Our contribution is summarized as followed,

1. We offer a point of view that self-attention mechanism can regulate the batch noise adaptively.
2. We propose a simple-yet-effective and attention-based BN called as Instance Enhancement Batch Normalization (IEBN). We demonstrate empirically the effectiveness of IEBN on benchmark datasets with different network architectures.

Algorithm 1 Instance Enhancement Batch Normalization

Input: X is a batch input of size $B \times C \times H \times W$;
Paramentes: γ_c, β_c, $\hat{\gamma}_c$ and $\hat{\beta}_c$, $c = 1, \cdots, C$;
Output: $\{Y = \textbf{IEBN}_{\gamma_c,\beta_c,\hat{\gamma}_c,\hat{\beta}_c}(X)\}$;

$\hat{\gamma}_c \leftarrow 0; \hat{\beta}_c \leftarrow -1$
for channel c from 1 to C **do**
  $\mu_c^B \leftarrow \frac{1}{B \cdot H \cdot W} \sum_{b=1}^{B} \sum_{h=1}^{H} \sum_{w=1}^{W} X_{bchw}$
  $\sigma_c^B \leftarrow \sqrt{\frac{1}{B \cdot H \cdot W} \sum_{b=1}^{B} \sum_{h=1}^{H} \sum_{w=1}^{W} (X_{bchw} - \mu_c^B)^2 + \epsilon}$
  for instance b from 1 to B **do**
    $\delta_{bc} \leftarrow Sigmoid(\text{AVG}(X_{bc}) \times \hat{\gamma}_c + \hat{\beta}_c)$
    $\hat{X}_{bc} \leftarrow (X_{bc} - \mu_c^B)/\sigma_c^B$
    $Y_{bc} \leftarrow \hat{X}_{bc} \times (\gamma_c \times \delta_{bc}) + \beta_c$
  end for
end for

Related Work

This session reviews related works and mainly focuses on two directions, normalization, and self-attention mechanism. Then we will discuss a work which combines them together.

Normalization. The normalization layer is an important component of a deep network. Multiple normalization methods have been proposed for different tasks. Batch Normalization (Ioffe and Szegedy 2015) which normalizes input by mini-batch statistics has been a foundation of visual recognition tasks (He et al. 2016a). Instance Normalization (Ulyanov, Vedaldi, and Lempitsky 2017a) performs one instance BN-like normalization and is widely used in generative model (Johnson, Alahi, and Fei-Fei 2016a; Zhu et al. 2017). There are some variants of BN, such as, Conditional Batch Normalization (de Vries et al. 2017) for Visual Questioning and Answering, Group Normalization (Wu and He 2018) and Batch Renormalization (Ioffe 2017) for small batch size training, Adaptive Batch Normalization (Li et al. 2018) for domain adaptation and Switchable normalization (Luo, Ren, and Peng 2018) which learns to select different normalizers for different normalization layers. Among them, Conditional Batch Norm and Batch Renorm adjust the trainable parameters in reparameterization step of BN. Both of them are most related to our work which modifies the trainable scaling parameter.

Self-attention Mechanism. Self-attention mechanism selectively focuses on the most informative components of a network via self-information processing and has gained a promising performance on vision tasks. The procedure of attention mechanism can be divided into three parts. First, the added-in module extracts internal information of a networks which can be squeezed channel-wise information (Hu, Shen, and Sun 2018; Li et al. 2019; Huang et al. 2019) or spatial information (Wang et al. 2018; Li, Hu, and Yang 2019). Next, the module processes the extraction and generates a mask to measure the importance of features via fully connected layer (Hu, Shen, and Sun 2018), convolution layer (Wang et al. 2018) or LSTM (Huang et al. 2019). Last, the mask is applied back to features to enhance feature importance (Hu, Shen, and Sun 2018; Li et al. 2019; Huang et al. 2019).

The cooperation of BN and attention dates back to Visual Questioning and Answering (VQA) which inputs an image and an image-related question and then outputs the answer to the question. For this task, Conditional Batch Norm (de Vries et al. 2017) is proposed to influence the feature extraction of an image via the feature collected from the question. A Recurrent Neural Network (RNN) is used to extract the features from the question while a Convolutional Neural Network (CNN), a pre-trained ResNet, performs features selection from the image. The features extracted from the question are conditioned on the shift and scale parameters of the BN in the pre-trained ResNet such that the feature selection of the CNN is question-referenced and the overall networks can handle different reasoning tasks. Note that for VQA, the features from question can be viewed as external attention to guide the training of overall network since those features are external regarding the image. In our work, the

IEBN we proposed can also be viewed as a kind of Conditional Batch Norm but the guidance of the network training is using the internal attention since we use self-attention mechanism to extract the information from the image itself.

Instance Enhancement Batch Normalization

This session first reviews BN and then introduces IEBN.

We consider a batch input $X \in \mathbb{R}^{B\times C\times H\times W}$, where B, C, H and W stand for batch size, number of channels (feature maps), height and width respectively. For simplicity, we denote $X_{bchw} = X[b, c, h, w]$ as the value of pixel (h, w) at channel c of instance b and $X_{bc} = X[b, c, :, :]$ as the tensor at channel c of instance b.

Review of BN

The computation of BN can be divided into two steps: batch-normalized step and reparameterization step. Without loss of generality, we perform BN on the channel c of the instance b, i.e., X_{bc}.

In batch-normalized step, each channel of features is normalized using mean and variance of a batch over the channel,

$$\hat{X}_{bc} = \frac{X_{bc} - \mu_c^B}{\sigma_c^B}, \tag{1}$$

where μ_c^B, σ_c^B are defined in Step 3 and Step 4 as the estimation of mean and standard derivation respectively of the whole dataset.

Then in reparameterization step, a pair of learnable parameters γ_c, β_c scale and shift the normalized tensor $\hat{X}_{bc}$ to restore the representation power,

$$\hat{X}_{bc} \times \gamma_c + \beta_c. \tag{2}$$

As said in Introduction, the batch noise mainly comes from the batch-normalized step where the feature of the instance b is mixed with information from the batch, i.e., μ_c^B and σ_c^B.

Formulation of IEBN

The showcase of IEBN is shown in Fig. 1, where we highlight the instance enhancement process of one channel. The detailed computation can be found in Alg. 1. IEBN is based on the adjustment of the trainable scaling parameter on BN and its implementation consists of three operations: global squeezing, feature processing, and instance embedding.

Global Squeezing. The global reception field of a feature map is captured by average pooling $\text{AVG}(\cdot)$. We obtain a shrinking feature descriptor m_{bc} of the channel c for the instance b by taking average over the channel,

$$m_{bc} = \text{AVG}(X_{bc}) = \frac{1}{H \cdot W} \sum_{h=1}^{H} \sum_{w=1}^{W} X_{bchw}. \tag{3}$$

m_{bc} will serve as a shrinking feature to adjust the c_{th} channel after BN and m_{bc} is exclusive to the instance b.

Feature Processing. The shrinking feature m_{bc} will be processed to generate a weight coefficient ranged in $[0, 1]$ for self-recalibration of channel c. To enhance self-regulating capacity, we introduce an addition pair of parameters $\hat{\beta}_c$, $\hat{\gamma}_c$ for the c_{th} channel, which serve as scale and shift respectively to linearly transform m_{bc}. Then Sigmoid function (i.e., $\sigma(z) = 1/(1 + e^{-z})$) is applied to the value after linear transformation as a gating mechanism:

$$\delta_{bc} = \sigma(\hat{\gamma}_c m_{bc} + \hat{\beta}_c). \tag{4}$$

Specially, the parameters $\hat{\gamma}_c, \hat{\beta}_c$ are initialized by constant 0 and -1 respectively. We will discuss the initialization in Ablation Study.

Instance Embedding. δ_{bc} works as a weight coefficient to adjust the scaling in the reparameterization step of BN for the instance b. We embed the recalibration δ_{bc} to compensate the instance information in Eqn. 2,

$$Y_{bc} = \hat{X}_{bc} \times (\gamma_c \times \delta_{bc}) + \beta_c. \tag{5}$$

The δ_{bc} is composed of nonlinear activation function and an additional pair of parameters which helps improve the nonlinearity of reparameterization of BN.

We conduct IEBN on all channels, i.e., $c = 1, 2, \cdots, C$. Compared with BN, the parameter increment comes from the additional pair parameter for generating coefficient for each channel. The total number of parameter increment is equal to twice the number of channels.

Experiments

In this section, we evaluate the performance of IEBN in image classification task and empirically demonstrate its effectiveness. We conduct experiments on benchmark datasets with popular networks.

Dataset and Model. We conduct experiments on CIFAR10, CIFAR100 (Krizhevsky and Hinton 2009), and ImageNet 2012 (Russakovsky et al. 2015). CIAFR10 or CIFAR100 has 50k train images and 10k test images of size 32 by 32 but has 10 and 100 classes respectively. ImageNet 2012 (Russakovsky et al. 2015) comprises 1.28 million training and 50k validation images from 1000 classes, and the random cropping of size 224 by 224 is used in our experiments. We evaluates our methods with popular networks, ResNet (He et al. 2016a), PreResNet (He et al. 2016b) and ResNeXt (Xie et al. 2017). In our experiments, we replace all the BNs in the original networks with IEBN. The implementation details can be found in the Appendix.

Image Classification. As shown in Table 1, the IEBN improves the testing accuracy over BN for different datasets and different network backbones. For small-classes dataset CIFAR10, the performance of the networks with BN is good enough, so there is not large space for improvement. However, for CIFAR100 and ImageNet datasets, the networks with IEBN achieve a significant testing accuracy improvement over BN. In particular, the performance improvement of the ResNet with the IEBN is most remarkable. Due to the popularity of ResNet and the light additional parameter increment, the IEBN has good application potential in various deep learning tasks.

Analysis

In this session, we explore the role of self-attention mechanism on enhancing instance information and regulating the

	Dataset	BN		SE		IEBN	
		#P(M)	top1-acc.	#P(M)	top1-acc.	#P(M)	top1-acc.
ResNet164	CIFAR100	1.73	74.29	1.93	75.80	1.75	77.09
PreResNet164	CIFAR100	1.73	76.56	1.92	77.41	1.75	77.27
DenseNet100-12	CIFAR100	0.80	77.23	-	-	0.82	78.57
ResNext29,8x64	CIFAR100	34.52	81.47	-	-	34.57	82.45
ResNet164	CIFAR10	1.70	93.93	1.91	-	1.73	95.03
PreResNet164	CIFAR10	1.70	95.01	1.90	95.18	1.73	95.09
DenseNet100-12	CIFAR10	0.77	95.29	-	-	0.79	95.83
ResNext29,8x64	CIFAR10	34.43	96.11	-	-	34.48	96.26
ResNet34	ImageNet	21.81	73.91	21.97	74.39	21.82	74.38
ResNet50	ImageNet	25.58	76.01	28.09	76.61	25.63	77.10
ResNet152	ImageNet	60.27	77.58	66.82	78.36	60.41	79.17
ResNext50	ImageNet	25.03	77.19	27.56	78.04	25.09	77.99

Table 1: Accuracy (%) on benchmark datasets with different architectures using BN, SE module or IEBN.

batch noise. We analysis through the style transfer and experiments with the synthetic noise attack.

Instance Enhancement

We explore the role of self-attention mechanism on instance enhancement through the example of the style transfer task (Gatys, Ecker, and Bethge 2016). We use the style transfer method which generates image by a network called transformation network (Johnson, Alahi, and Fei-Fei 2016b).

It has been empirically shown that the type of normalization in the network has an impact on the quality of image generation (Ulyanov, Vedaldi, and Lempitsky 2017b; Huang and Belongie 2017; Dumoulin, Shlens, and Kudlur 2016). Instance Normalization (IN) is widely used in generative models and it had proved to have a significant advantage over BN in style transfer tasks (Ulyanov, Vedaldi, and Lempitsky 2017b). The formulation of IN is followed,

$$\left(\frac{X_{bc}-\mu(X_{bc})}{\sigma(X_{bc})}\right)\cdot\gamma+\beta=\frac{\gamma}{\sigma(X_{bc})}\cdot X_{bc}+\beta-\frac{\mu(X_{bc})}{\sigma(X_{bc})}\cdot\gamma, \tag{6}$$

where $\mu(X_{bc})$ and $\sigma(X_{bc})$ denote the mean and standard deviation of the instance b at the channel c. Similarly, the formulation of BN can be written in this form,

$$\left(\frac{X_{bc}-\mu_c^B}{\sigma_c^B}\right)\cdot\gamma+\beta=\frac{\gamma}{\sigma_c^B}\cdot X_{bc}+\beta-\frac{\mu_c^B}{\sigma_c^B}\cdot\gamma. \tag{7}$$

γ and β are learned parameters and both are closely related to the target style (Dumoulin, Shlens, and Kudlur 2016). From Eqn. 6 and Eqn. 7, IN or BN directly leads to the scaling of γ that affects the style of images. Different from BN, IN affects the style by self-information instead of batch information. Fig. 2 compares the quality of images generated by the network with BN, IN and SE module. The style transfer task is noise-sensitive, and when the batch noise is added by BN, the style of the generated image becomes more confused. We add the SE module (Hu, Shen, and Sun 2018) to the transformation network with BN to find its effectiveness of regulating batch noise. We can see in Fig. 2 that the attention mechanism (SE) visually improves the effect of style transfer and the quality of the generated images is closer to that of IN. Fig. 3 shows the training loss with respect to the iterations by applying the style Mosaic. The BN network with SE module achieves smaller style loss and smaller content loss than BN, and is closer to IN (see Appendix for more results about the loss by applying other style). Therefore, although the BN can bring the batch information to an instance, it simultaneously introduce batch noise to network training. The attention mechanism such as SE module may be good at alleviating the batch noise and we will investigate it further.

IEBN is a BN equipped with self-attention and Fig. 2 shows the similarity of the generated images of the SE module and IEBN. In fact, we consider IEBN:

$$\left(\frac{X_{bc}-\mu_c^B}{\sigma_c^B}\right)\cdot\gamma\delta_{bc}+\beta=\frac{\gamma\delta_{bc}}{\sigma_c^B}\cdot X_{bc}+\beta-\frac{\mu_c^B}{\sigma_c^B}\cdot\gamma\cdot\delta_{bc}, \tag{8}$$

where δ_{bc} is defined in Eqn. 4 and δ_{bc} contains information from the instance b. It seems like the added-in δ_{bc} is only directly applied to scaling parameter γ of BN, but it does scale the batch information (i.e., μ_c^B, σ_c^B) to regulate the batch information via supplement of instance information. This adjustment of batch information via δ_{bc} makes the Eqn. 8 closer to Eqn. 6 than Eqn. 7 and also leads to the similar results in style transfer between IN and IEBN.

Noise Attack

To further study the ability to regulate the noise of IEBN, two kinds of strategies is used to add the synthetic noise in the batch-normalized step of BN.

Constant Noise Attack. We add constant noise into each BN in the batch-normalized step as followed,

$$\hat{X}_{bc}=\frac{X_{bc}-\mu_c^B}{\sigma_c^B}\cdot N_a+N_b, \tag{9}$$

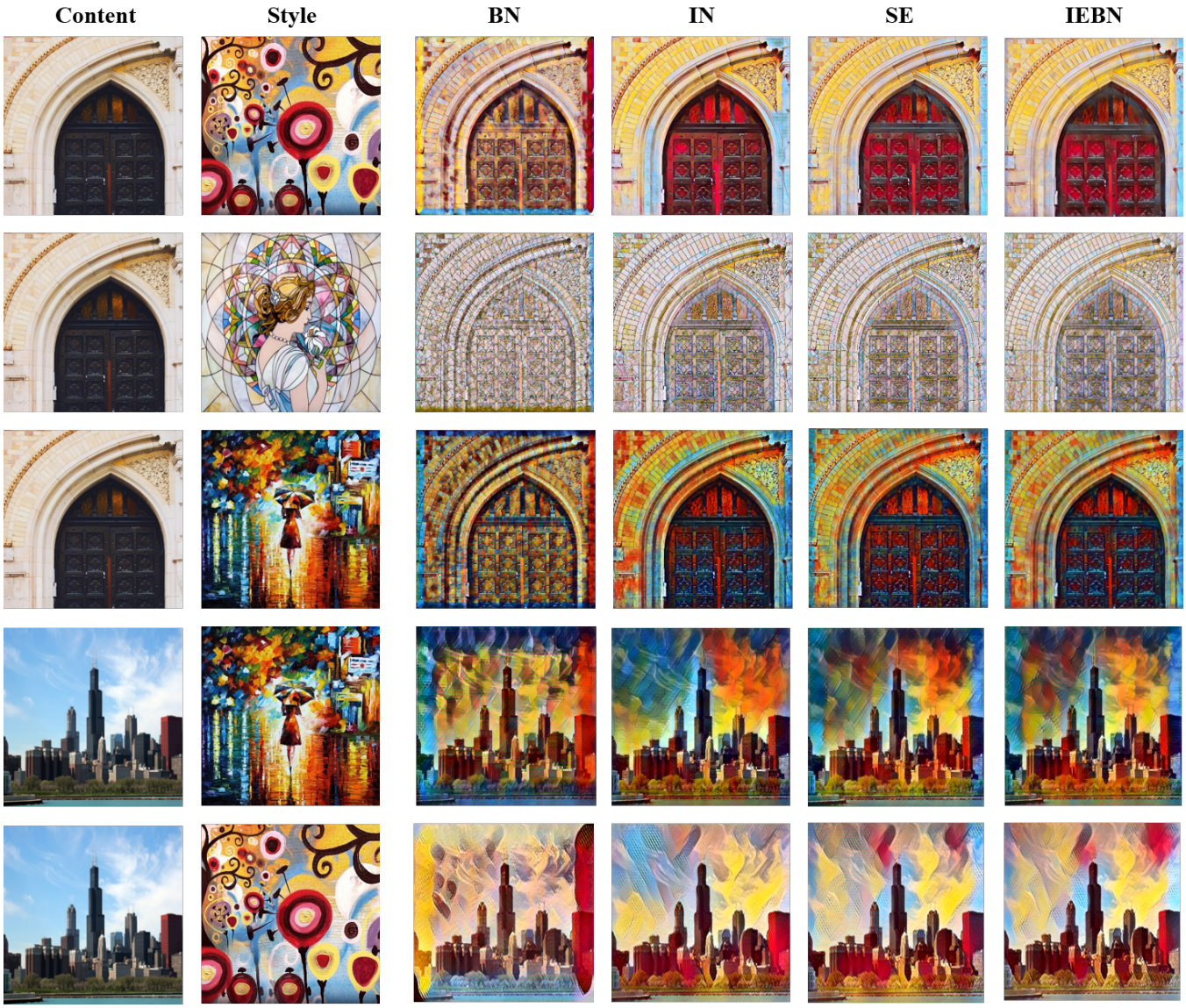

Figure 2: Stylization results obtained by applying style (second column) to content images (first column) with different normalization methods. Specially, "SE" means the transformation network with BN and SE module. The style of the generated images with BN appears more confused, but those with SE or IEBN are quite similar to IN visually.

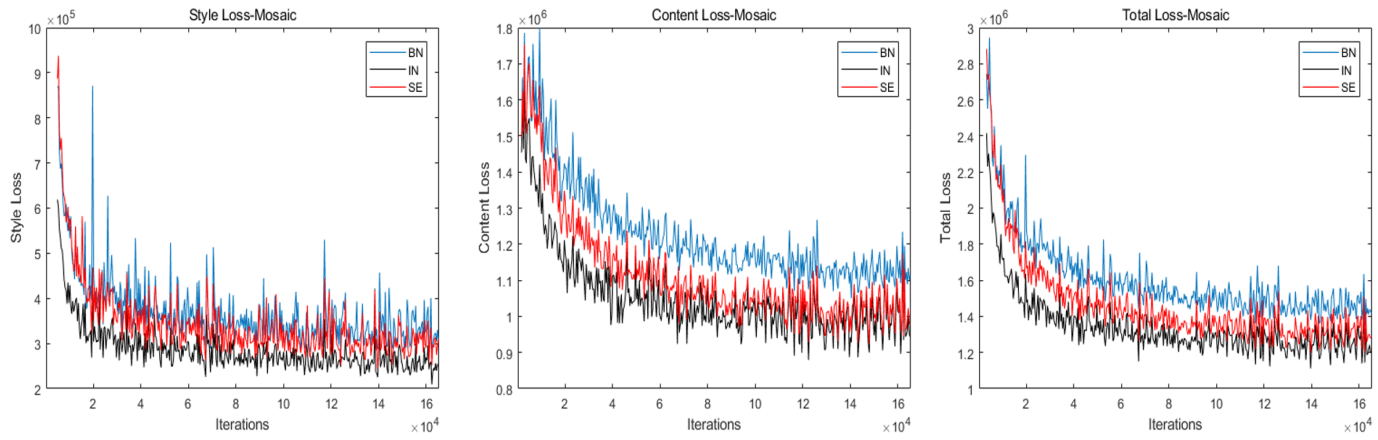

Figure 3: Training curves of style transfer networks with Mosaic style and different normalization methods. Specially, "SE" means the transformation network with BN and SE module.

where (N_a, N_b) are a pair of constant as the constant noise. Table 2 shows the testing accuracy of ResNet164 on CIFAR100 under different pairs of constant noise.

The added constant noise is equivalent to disturbing μ_c^B and σ_c^B such that we can use the inaccurate estimations of mean and variance respectively of the whole dataset in training. This bad estimation can lead to terrible performance. Denote $(X_{bc} - \mu_c^B)/\sigma_c^B$ as Δ. Then in the reparameterization step of BN, we introduce the learnable parameters γ and β and get

$$\begin{aligned}\hat{X}_{bc} &= (\Delta \cdot N_a + N_b) \cdot \gamma + \beta \\ &= \Delta \cdot \underbrace{(N_a \cdot \gamma)}_{\gamma'} + \underbrace{(N_b \cdot \gamma + \beta)}_{\beta'}\end{aligned} \tag{10}$$

From the inference of Eqn. 10, the impact of constant noise can be easily neutralized by the linear transformation of γ and β because N_a and N_b are just constants. However, in Table 2, the network with only BN is not good at handling most constant noise (N_a, N_b). The trainable γ and β of BN does not have enough power to help BN reduce the impact of the constant noise. Due to the forward propagation, the noise will accumulate as the depth increases and a certain amount of noise leads to poor performance and training instability. As shown in Table 2, SE module can partly alleviate this problem, but not enough because of the high variance of the testing accuracy under most pairs of constant noise.

For IEBN, we can rewrite Eqn. 10 as

$$\hat{X}_{bc} = \Delta \cdot \underbrace{N_a \cdot \gamma \cdot \delta_{bc}}_{\gamma''} + \underbrace{N_b \cdot \gamma \cdot \delta_{bc} + \beta}_{\beta''}, \tag{11}$$

where δ_{bc} denotes the attention learned in IEBN. Compared to Eqn. 10, Eqn. 11 with δ_{bc} from IEBN has successfully adjusted constant noise and even achieved better performance under partial noise configuration. If δ_{bc} only excites β, we can rewrite Eqn. 11 as

$$\hat{X}_{bc} = \Delta \cdot \underbrace{N_a \cdot \gamma}_{\gamma'''} + \underbrace{N_b \cdot \gamma + \beta \cdot \delta_{bc}}_{\beta'''}, \tag{12}$$

where δ_{bc} can only adjust the noise in β''' instead of γ'''. But if applied to γ, δ_{bc} can handle the noise of scale and bias simultaneously. It may be the reason why the result about only exciting β is worse than the other in Table 5, but better than the original model with BN in Table 1.

(N_a, N_b)	BN	SE	IEBN
(0.0,0.0)	$74.29_{(\pm 0.64)}$	$75.80_{(\pm 0.25)}$	$77.09_{(\pm 0.15)}$
(0.8,0.8)	$45.42_{(\pm 31.42)}$	$73.18_{(\pm 0.66)}$	$75.42_{(\pm 0.08)}$
(0.8,0.5)	$46.10_{(\pm 31.91)}$	$71.59_{(\pm 1.77)}$	$77.39_{(\pm 0.09)}$
(0.8,0.2)	$71.65_{(\pm 0.22)}$	$71.08_{(\pm 0.52)}$	$76.77_{(\pm 0.22)}$
(0.5,0.5)	$35.77_{(\pm 34.76)}$	$74.61_{(\pm 0.56)}$	$77.00_{(\pm 0.29)}$
(0.5,0.2)	$73.10_{(\pm 1.72)}$	$75.72_{(\pm 1.47)}$	$77.11_{(\pm 0.08)}$

Table 2: The testing accuracy (mean $\pm$ std %) of ResNet164 on CIFAR100. (N_a, N_b) is a pair of constant noise added to BN at the batch-normalized step as stated in Eqn. 9. $(0.0, 0.0)$ means we do not add the noise.

Mix-Datasets Attack. In this part, we consider interfering with μ_c^B and σ_c^B by simultaneously training on the datasets with different distributions in one network. Unlike constant noise which is added to networks directly, this noise is implicit and is generated when BN computes the mean and variance of training data from different distribution. These datasets differ widely in their distribution and causes severe batch noise. Compared with the constant noise, this noise is not easy to eliminate by linear transformation of γ and β.

In our experiments, we train ResNet164 on CIFAR100 but mix up with MINIST (LeCun and Cortes 2010) or Fashion-MINIST (Xiao, Rasul, and Vollgraf 2017) in a batch and compare the performance of BN and IEBN. Table 3 shows the test accuracy on CIFAR100, where "C+$k\times$ M or F" means we sample a batch consisted of 100 images from CIFAR100 (C) and $120 \times k$ images from MNIST (M) or FashionMNIST (F) at each iteration during training. As k increases, the batch noise becomes more severe for CIFAR100 since μ_c^B and σ_c^B contains more information about MNIST

or FashionMnist. In most cases, despite the severe noise like "C+2×", the model with IEBN still performs better than the model with BN training merely on CIFAR100. On the other hand, the drop in accuracy of IEBN is smaller than that of IEBN, and IEBN alleviates the degradation of network generalization. These phenomena illustrate that, although under the influence of MINIST or FashionMINIST, the model with IEBN has a stronger ability to resist the batch noise.

	BN		IEBN	
Dateset	test acc	acc drop	test acc	acc drop
C	74.29	-0.00	77.09	-0.00
C+2× M	73.13	-1.16	76.65	-0.44
C+3× M	71.54	-2.75	76.03	-1.06
C+2× F	71.56	-2.73	75.57	-1.52
C+3× F	71.27	-3.02	74.26	-2.83

Table 3: Test accuracy (%) on CIFAR100 with ResNet-164. "C+$k\times$ M/F" means we samples a batch consisted of 100 images from CIFAR100 (C) and $120 \times k$ images from MNIST (M) or FashionMNIST (F) at each iteration during training. "acc drop" means the drop of accuracy compared with network trained merely on CIFAR100.

Ablation Study

In this section, we conduct experiments to explore the effect of different configurations of IEBN. We study different ways of generating δ_{bc}, the position for applying the attention, initialization of IEBN and activation function used in IEBN. All experiments are performed on CIFAR100 with ResNet164 using 2 GPUs.

The Way of Generating δ_{bc}. This part we study different ways to process the squeezed features to generate δ_{bc}. As shown in Alg. 1, IEBN squeezes the channel through global average pooling AVG$(\cdot)$ and processes the squeezed feature by linear transformation (i.e. AVG$(X_{bc})\times\hat{\gamma}_c$ + $\hat{\beta}_c$) for each channel, denoted as "Linear". We also consider another two methods to process the information. The first one is that we remove the additional trainable parameters $\hat{\gamma}_c$ and $\hat{\beta}_c$ for linear transformation in IEBN and directly apply the squeezed feature after sigmoid function to the channel, denoted as "Identity". The second one is that we use a fully connected layer stacking of a linear transformation, a ReLU layer, and a linear transformation to fuse the squeezed features of all channels $\{\text{AVG}(X_{bc})\}_{c=1}^{C}$, denotes as "FC". "FC" is similar to the configuration as SE module introduced in (Hu, Shen, and Sun 2018).

Table 4 shows the testing accuracy using different ways to process the squeezed features. "FC" operator provides more nonlinearity than "Linear" operator (IEBN), but such nonlinearity may lead to overfitting and the "Linear" operator (IEBN) simplifies the squeezed feature processing and has better generalization ability. Furthermore, the result of "Identity" indicates that it is not enough to simply and directly use instance information to enhance self-information without any trainable parameters. The operators with trainable parameters, such as "Linear" (IEBN) and "FC", are needed to process the instance information such that the adaptive and advantageous noise during training can be regulated to improve the performance.

Operator	Dataset	Test Acc.
Linear (IEBN)	CIFAR100	**77.09**($\pm$0.15)
Identity	CIFAR100	67.53($\pm$**2.49**)
FC	CIFAR100	76.11($\pm$0.28)

Table 4: Testing accuracy (%) with different ways to process the squeezed features. "Linear" means a linear transformation applied to a squeezed feature, which is actually IEBN. "Identity" means removing the parameters for linear transformation in "Linear". "FC" means a fully connected layer is used to fuse all the squeezed features of all channels.

Excitation Position. We study the influence of different positions that δ_{bc} excites. For self-attention mechanism like SENet (Hu, Shen, and Sun 2018), DIANet (Huang et al. 2019) and SGENet (Li, Hu, and Yang 2019), the rescaling coefficient usually excites both the trainable parameter γ and β of BN. In IEBN, the δ_{bc} is only applied to adjust the scaling parameter γ in BN. To differentiate the influence of the excitation positions, Table 5 shows testing accuracy with different positions where the δ_{bc} excites. We show that the performance is unsatisfied when the δ_{bc} is merely exciting β. Moreover, there is a slight difference between exciting only γ and exciting both γ and β, and the former excitation position has better performance. From the point of view of adjusting noise, Eqn. 11 and Eqn. 12 can explain the result shown in Table 5. Therefore, the results suggest that to make IEBN more effective, it is important to carefully choose the position where the δ_{bc} should excite.

Position	Dataset	Test Acc.
γ (IEBN)	CIFAR100	**77.09**($\pm$0.15)
β	CIFAR100	75.03($\pm$0.54)
γ and β	CIFAR100	77.02($\pm$0.08)

Table 5: Testing accuracy (%) with different positions that the δ_{bc} excites. γ and β are the parameters in the reparameterization step of BN.

Initialization of $\hat{\gamma}_c$ and $\hat{\beta}_c$ This part studies the initialization of trainable parameters $\hat{\gamma}_c$ and $\hat{\beta}_c$ which are used to process the squeezed feature in IEBN. According to the experiments in Table 4, the learnable parameters, $\hat{\gamma}_c$ and $\hat{\beta}_c$, are indispensable for IBEN to be effective. Therefore, further study of different initialization configuration is essential to understand IEBN in depth. In order to explore this impact, we use constant 1, 0 and -1 for grid search to find the best pair of initialization for $\hat{\gamma}_c$ and $\hat{\beta}_c$. We find that the initialization of the trainable parameters of IBEN $\hat{\gamma}_c$ and $\hat{\beta}_c$ have

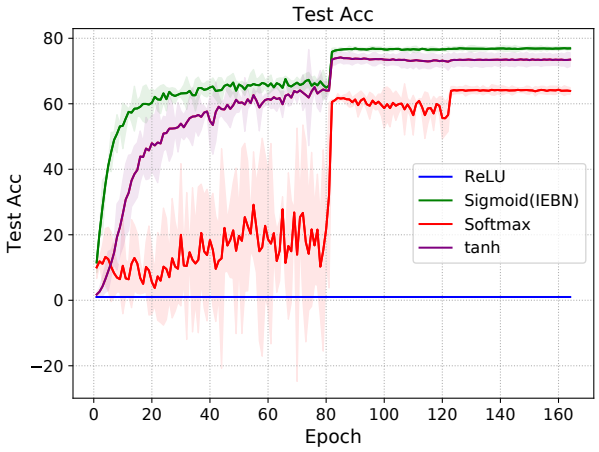

Figure 4: The training curve with different activation functions in IEBN. The sigmoid function outperforms other activation functions as a gating mechanism.

a significant impact on the performance of model: From Table 6, the performance is varying as different initialization is chosen. Note that, the best choice of $\hat{\gamma}_c$ is 0 when we freeze the initialization of $\hat{\beta}_c$. Similarly, the effect of the model is the best when the initialization of $\hat{\beta}_c$ is fixed to be -1. The theoretical nature behind the best initialization configuration will be our future work.

$\hat{\gamma}_c \setminus \hat{\beta}_c$	1	0	-1
1	68.5	66.96	69.53
0	75.86	76.21	**77.09**
-1	74.64	74.73	75.31

Table 6: Test accuracy (%) with different constant initialization for trainable parameters scaling $\hat{\gamma}_c$ and shift $\hat{\beta}_c$ in IEBN.

Activation Function. We explore the choice of activation function in IEBN. We consider four options for activation function: sigmoid, tanh, ReLU and Softmax. The testing accuracy results are reported in Fig. 4. Note that, ReLU may be a terrible choice which maintains only 1% accuracy throughout the training. In addition, the performance of Softmax is evidently worse than that of sigmoid or tanh. The choice of sigmoid can benefit the stability of training and performance. In fact, sigmoid is used in many attention-based methods like SENet (Hu, Shen, and Sun 2018) to generate attention maps as a gate mechanism. The testing accuracy of different choices of activation functions in Table 4 shows that sigmoid helps IEBN as a gate to rescale channel features better. The similar ablation study in the SENet paper (Hu, Shen, and Sun 2018) also shows the performance of different activation functions like: sigmoid, $>$ tanh $>$, and ReLU (bigger is better), which coincides to our reported results.

Conclusion

In this paper, we introduce two kinds of noise brought by BN and offer a point of view that self-attention mechanism can regulate the batch noise adaptively. We propose a simple-yet-effective and attention-based BN called as Instance Enhancement Batch Normalization (IEBN). We demonstrate empirically the effectiveness of IEBN on benchmark datasets with different network architectures and also provide ablation study to explore the effect of different configurations of IEBN.

Acknowledgments

S. Liang and H. Yang gratefully acknowledge the support of National Supercomputing Center (NSCC) Singapore (nscc) and High Performance Computing (HPC) of National University of Singapore for providing computational resources, and the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research. Z. Huang thanks New Oriental AI Research Academy Beijing for GPU resources. H. Yang thanks the support of the start-up grant by the Department of Mathematics at the National University of Singapore, the Ministry of Education in Singapore for the grant MOE2018-T2-2-147.

References

[Bjorck et al. 2018] Bjorck, N.; Gomes, C. P.; Selman, B.; and Weinberger, K. Q. 2018. Understanding batch normalization. In *NeurIPS*, 7705–7716.

[Cai, Li, and Shen 2019] Cai, Y.; Li, Q.; and Shen, Z. 2019. A quantitative analysis of the effect of batch normalization on gradient descent. In *ICML*, 882–890.

[de Vries et al. 2017] de Vries, H.; Strub, F.; Mary, J.; Larochelle, H.; Pietquin, O.; and Courville, A. C. 2017. Modulating early visual processing by language. In *NIPS*, 6597–6607.

[Dumoulin, Shlens, and Kudlur 2016] Dumoulin, V.; Shlens, J.; and Kudlur, M. 2016. A learned representation for artistic style. *arXiv preprint arXiv:1610.07629*.

[Gatys, Ecker, and Bethge 2016] Gatys, L. A.; Ecker, A. S.; and Bethge, M. 2016. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2414–2423.

[Ge et al. 2015] Ge, R.; Huang, F.; Jin, C.; and Yuan, Y. 2015. Escaping from saddle pointsłonline stochastic gradient for tensor decomposition. In *Conference on Learning Theory*, 797–842.

[He et al. 2016a] He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016a. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition*.

[He et al. 2016b] He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016b. Identity mappings in deep residual networks. In *European Conference on Computer Vision*.

[Hu, Shen, and Sun 2018] Hu, J.; Shen, L.; and Sun, G. 2018. Squeeze-and-excitation networks. In *Proceedings of*

the IEEE conference on computer vision and pattern recognition, 7132–7141.

[Huang and Belongie 2017] Huang, X., and Belongie, S. 2017. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, 1501–1510.

[Huang et al. 2019] Huang, Z.; Liang, S.; Liang, M.; and Yang, H. 2019. Dianet: Dense-and-implicit attention network. *CoRR* abs/1905.10671.

[Ioffe and Szegedy 2015] Ioffe, S., and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, 448–456. JMLR.org.

[Ioffe 2017] Ioffe, S. 2017. Batch renormalization: Towards reducing minibatch dependence in batch-normalized models. In *NIPS*, 1942–1950.

[Jin et al. 2017] Jin, C.; Ge, R.; Netrapalli, P.; Kakade, S. M.; and Jordan, M. I. 2017. How to escape saddle points efficiently. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 1724–1732. JMLR. org.

[Johnson, Alahi, and Fei-Fei 2016a] Johnson, J.; Alahi, A.; and Fei-Fei, L. 2016a. Perceptual losses for real-time style transfer and super-resolution. In *ECCV (2)*, 694–711.

[Johnson, Alahi, and Fei-Fei 2016b] Johnson, J.; Alahi, A.; and Fei-Fei, L. 2016b. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, 694–711. Springer.

[Keskar et al. 2016] Keskar, N. S.; Mudigere, D.; Nocedal, J.; Smelyanskiy, M.; and Tang, P. T. P. 2016. On large-batch training for deep learning: Generalization gap and sharp minima.

[Krizhevsky and Hinton 2009] Krizhevsky, A., and Hinton, G. 2009. Learning multiple layers of features from tiny images. Technical report, Citeseer.

[LeCun and Cortes 2010] LeCun, Y., and Cortes, C. 2010. MNIST handwritten digit database.

[Li et al. 2018] Li, Y.; Wang, N.; Shi, J.; Hou, X.; and Liu, J. 2018. Adaptive batch normalization for practical domain adaptation. *Pattern Recognition* 80:109–117.

[Li et al. 2019] Li, X.; Wang, W.; Hu, X.; and Yang, J. 2019. Selective kernel networks.

[Li, Hu, and Yang 2019] Li, X.; Hu, X.; and Yang, J. 2019. Spatial group-wise enhance: Improving semantic feature learning in convolutional networks. *CoRR* abs/1905.09646.

[Luo et al. 2019] Luo, P.; Wang, X.; Shao, W.; and Peng, Z. 2019. Towards understanding regularization in batch normalization. In *International Conference on Learning Representations*.

[Luo, Ren, and Peng 2018] Luo, P.; Ren, J.; and Peng, Z. 2018. Differentiable learning-to-normalize via switchable normalization. *arXiv preprint arXiv:1806.10779*.

[Luo, Xiong, and Liu 2019] Luo, L.; Xiong, Y.; and Liu, Y. 2019. Adaptive gradient methods with dynamic bound of learning rate. In *International Conference on Learning Representations*.

[nscc] nscc.

[Russakovsky et al. 2015] Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; Berg, A. C.; and Fei-Fei, L. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)* 115(3):211–252.

[Santurkar et al. 2018] Santurkar, S.; Tsipras, D.; Ilyas, A.; and Madry, A. 2018. How does batch normalization help optimization? In *NeurIPS*, 2488–2498.

[Ulyanov, Vedaldi, and Lempitsky 2017a] Ulyanov, D.; Vedaldi, A.; and Lempitsky, V. 2017a. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 6924–6932.

[Ulyanov, Vedaldi, and Lempitsky 2017b] Ulyanov, D.; Vedaldi, A.; and Lempitsky, V. 2017b. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 6924–6932.

[Wang et al. 2018] Wang, X.; Girshick, R.; Gupta, A.; and He, K. 2018. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7794–7803.

[Wu and He 2018] Wu, Y., and He, K. 2018. Group normalization. In *ECCV (13)*, 3–19.

[Xiao, Rasul, and Vollgraf 2017] Xiao, H.; Rasul, K.; and Vollgraf, R. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms.

[Xie et al. 2017] Xie, S.; Girshick, R.; Dollár, P.; Tu, Z.; and He, K. 2017. Aggregated residual transformations for deep neural networks. In *Computer Vision and Pattern Recognition*.

[Zhu et al. 2017] Zhu, J.-Y.; Park, T.; Isola, P.; and Efros, A. A. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2242–2251.

	ResNet164	PreResNet164	ResNext29-8x64	Densenet100-12
Batch size	128	128	128	64
Epoch	180	164	300	300
Optimizer	SGD(0.9)	SGD(0.9)	SGD(0.9)	SGD(0.9)
depth	164	164	29	100
schedule	81/122	81/122	150/225	150/225
wd	1.00E-04	1.00E-04	5.00E-04	1.00E-04
gamma	0.1	0.1	0.1	0.1
widen-factor	-	-	4	-
cardinality	-	-	8	-
lr	0.1	0.1	0.1	0.1

Table 7: Implementation detail for **CIFAR10/100** image classification. Normalization and standard data augmentation (random cropping and horizontal flipping) are applied to the training data.

	ResNet34	ResNet50	ResNet152	ResNext50-32x4
Batch size	256	256	256	256
Epoch	120	120	120	120
Optimizer	SGD(0.9)	SGD(0.9)	SGD(0.9)	SGD(0.9)
depth	34	50	152	50
schedule	30/60/90	30/60/90	30/60/90	30/60/90
wd	1.00E-04	1.00E-04	1.00E-04	1.00E-04
gamma	0.1	0.1	0.1	0.1
lr	0.1	0.1	0.1	0.1

Table 8: Implementation detail for **ImageNet 2012** image classification. Normalization and standard data augmentation (random cropping and horizontal flipping) are applied to the training data. The random cropping of size 224 by 224 is used in these experiments.

Appendix

Implementation Detail

The implementation detail is shown in Table 7 and Table 8.

Other Style Transfer Loss

The style transfer loss of different styles can be found in Fig. 5.

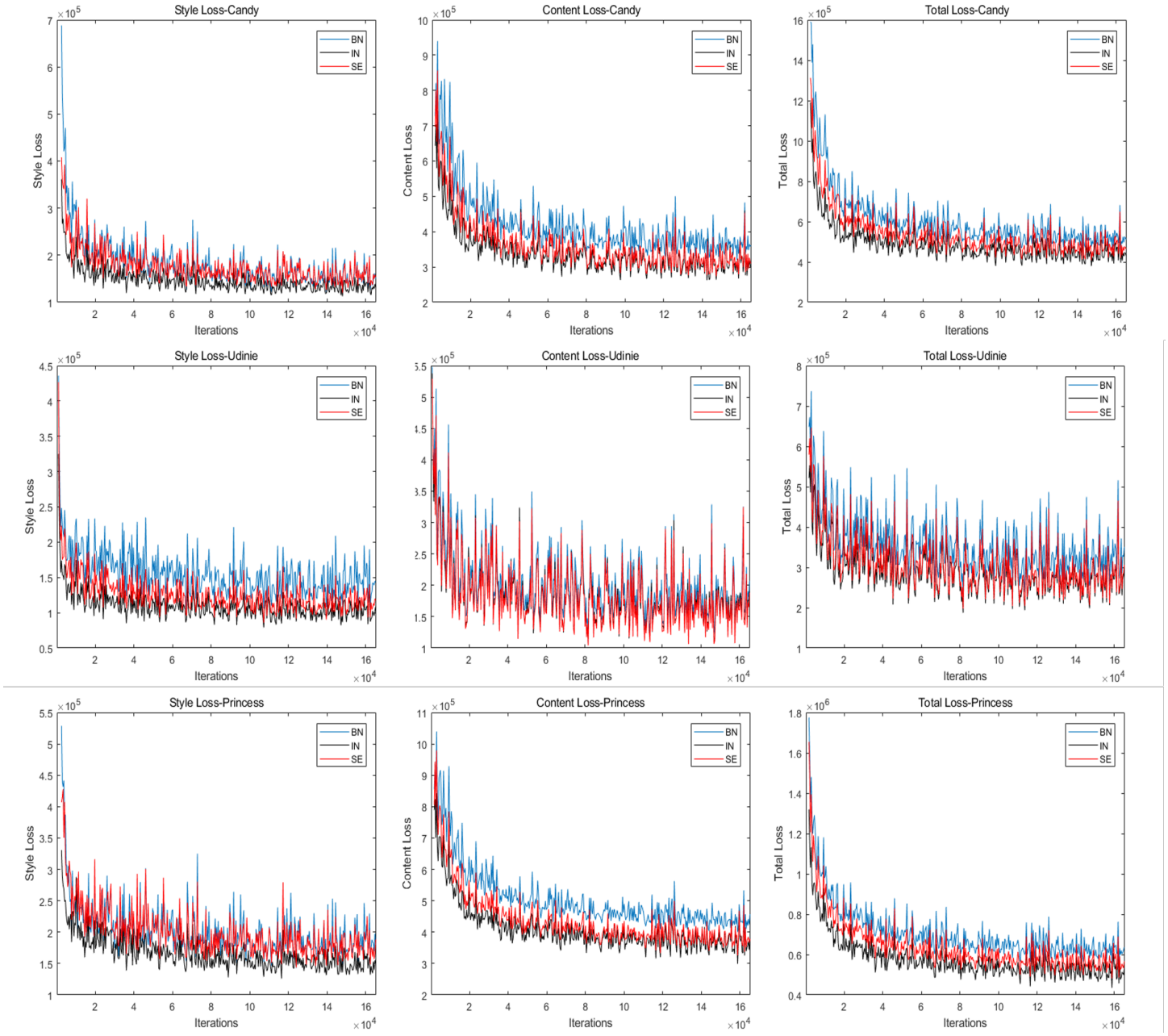

Figure 5: Training curves of style transfer networks with different styles and different normalization methods. Specially, “SE” means the transformation network with BN and SE module.