Instance-Level Meta Normalization

Songhao Jia
National Tsing Hua University
gasoonjia@gapp.nthu.edu.tw

Ding-Jie Chen
Academia Sinica
djchen.tw@gmail.com

Hwann-Tzong Chen
National Tsing Hua University
htchen@cs.nthu.edu.tw

Abstract

This paper presents a normalization mechanism called Instance-Level Meta Normalization (ILM Norm) to address a learning-to-normalize problem. ILM Norm learns to predict the normalization parameters via both the feature feed-forward and the gradient back-propagation paths. ILM Norm provides a meta normalization mechanism and has several good properties. It can be easily plugged into existing instance-level normalization schemes such as Instance Normalization, Layer Normalization, or Group Normalization. ILM Norm normalizes each instance individually and therefore maintains high performance even when small mini-batch is used. The experimental results show that ILM Norm well adapts to different network architectures and tasks, and it consistently improves the performance of the original models. The code is available at https://github.com/Gasoonjia/ILM-Norm.

1. Introduction

The mechanism of normalization plays a key role in deep learning. Various normalization strategies have been presented to show their effectiveness in stabilizing the gradient propagation. In practice, a normalization mechanism aims to normalize the output of a given layer such that the vanishing gradient problem can be suppressed and hence to reduce the oscillation in the output distribution. With appropriate normalization, a deep network would be able to improve the training speed and the generalization capability.

A typical normalization mechanism contains two stages: *standardization* and *rescaling*. The standardization stage regularizes an input tensor $\mathbf{x}$ of feature maps with its mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\gamma}$ by

$$\mathbf{x}_s = \frac{\mathbf{x}-\boldsymbol{\mu}}{\sqrt{\boldsymbol{\gamma}+\epsilon}}\,, \tag{1}$$

where $\mathbf{x}_s$ is a standardized input feature tensor. At the rescaling stage, the standardized feature tensor $\mathbf{x}_s$ is rescaled by a learned weight $\boldsymbol{\omega}$ and a bias $\boldsymbol{\beta}$ to recover the statistics of the features vanished in the standardization stage by

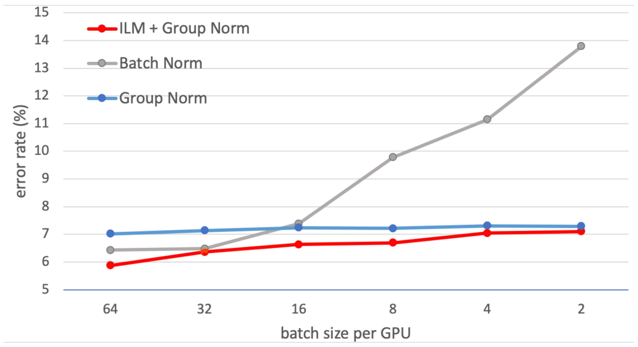

Figure 1. **CIFAR-10 classification error rate versus batch size per GPU.** The evaluation model is ResNet-101. The result shows that ILM Norm applied on Group Normalization has the best error rates in comparison with Batch Normalization [11] and the original Group Normalization [27].

$$\mathbf{x}_n = \boldsymbol{\omega} * \mathbf{x}_s + \boldsymbol{\beta}\,, \tag{2}$$

where $\mathbf{x}_n$ is the final output of the entire normalization process.

Existing normalization techniques mainly focus on studying the standardization stage to improve the training of deep networks under various circumstances. In contrast, as far as we know, the rescaling stage is less investigated and its related improvements remain unexplored. We observe that existing techniques of estimating the rescaling parameters for recovering the standardized input feature tensor often merely rely on the back-propagation process without considering the correlation between the standardization stage and the rescaling stage. As a result, information might be lost while data flow is passing through these two stages. We argue that the lack of correlation between two stages may lead to a performance bottleneck for existing normalization techniques.

The proposed Instance-Level Meta Normalization (ILM Norm) aims to connect the standardization stage and the rescaling stage. The design of ILM Norm is inspired by residual networks [6], which use the previously visited feature maps for guiding the learning of the current feature

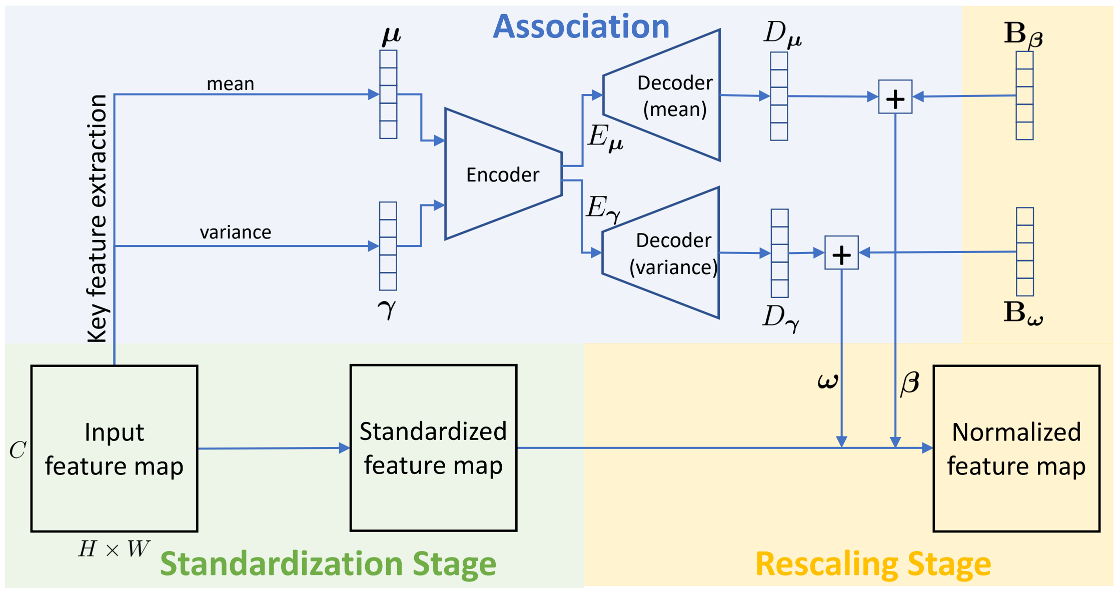

Figure 2. An overview of ILM Norm, the proposed meta learning mechanism for instance-level normalization. As a meta normalization mechanism, ILM Norm can derive its mechanism for the standardization stage and parameters for the rescaling stage from all kinds of instance-level standardization methods. For association, ILM Norm divides the C channels of the input into groups for computing the mean and variance per group as the key features $\boldsymbol{\mu}$ and $\boldsymbol{\gamma}$. An auto-encoder is then used to associate the features $\boldsymbol{\mu}$ and $\boldsymbol{\gamma}$ of the input feature tensor to the rescaling parameters $\boldsymbol{\omega}$ and $\boldsymbol{\beta}$ via the outputs (D_{γ} and D_{μ}) and the original rescaling parameters (B_{ω} and B_{β}) for rescaling the standardized feature map.

maps. The learning of weight $\boldsymbol{\omega}$ and bias $\boldsymbol{\beta}$ in ILM Norm follows the clue from the input feature maps of the standardization stage rather than merely relying on the back-propagation as previous methods do. We link the input tensor $\mathbf{x}$ of feature maps with the weight $\boldsymbol{\omega}$ and the bias $\boldsymbol{\beta}$ in the rescaling stage. In this way, the weight $\boldsymbol{\omega}$ and the bias $\boldsymbol{\beta}$ can not only be optimized better but fit to different inputs more effectively during the forward pass as well.

ILM Norm provides a meta learning mechanism for instance-level normalization techniques. It is handy for combining with existing instance-level techniques such as Instance Normalization [25] or Group Normalization [27]. ILM Norm normalizes the features within each instance individually, *i.e.*, performing the normalization without using the batch dimension. The advantage of this property is that, since the normalization is independent of the batch size, the performance is more robust to various settings of batch sizes that are suitable for particular network architectures on different tasks.

An overview of the proposed meta normalization mechanism is shown in Figure 2. The main ideas, advantages, and contributions of this work are summarized as follows:

1. ILM Norm provides a novel way to associate the rescaling parameters with the input feature maps rather than deciding the parameters merely from back-propagation.
2. ILM Norm can be handily plugged into existing instance-level normalization techniques such as Instance Normalization [25], Layer Normalization [2], and Group Normalization [27]. We show that ILM Norm improves existing instance-level normalization techniques on various tasks.
3. The number of variables in ILM Norm is small and does not add too much computation burden. For ResNet-101, the total number of variables would only increase 0.086% with ILM Norm.
4. The experimental results show that ILM Norm performs stably well under various batch sizes.
5. We conduct extensive experiments on several datasets to analyze and compare the properties of ILM Norm with other normalization techniques.

2. Related Work

2.1. Normalization in Deep Neural Networks

Gradient-based learning may suffer from the well-known problems of exploding gradient or vanishing gradient. It has been demonstrated that normalization provides an effective way to mitigate such problems.

Several popular normalization techniques have been proposed with the developments of deep neural networks.

AlexNet [16] and its follow-up models [21, 22] adopt Local Response Normalization (LRN) [13, 18] to compute the mean and variance of the same spatial locations across several neighboring feature maps for standardizing to the middle one. However, this kind of normalization only focuses on the statistics in a small neighborhood per pixel.

As suggested by its name, Batch Normalization (BN) [11] provides a batch-level normalization method that respectively centers and scales by mean and variance across the whole mini-batch and then rescales the result. Decorrelated Batch Normalization [8] improves Batch Normalization by adding an extra whitening procedure at the standardization stage. For batch-level normalization mechanisms, the calculation of mean and variance relies on the whole mini-batch. The effectiveness of normalization may degrade when the batch size is not sufficient to support the statistics calculation. To ease the issue of degradation, Batch Renormalization [10] suggests adding more learnable parameters in BN.

Several normalization techniques [1, 2, 25, 19, 27] inherit the notion of Batch Normalization but mainly focus on the manipulations of the standardization stage. Layer Normalization (LN) [2] operates along the channel dimension and standardizes the features from a single mini-batch by the mini-batch's own mean and variance. It can be used with batch size 1. Instance Normalization (IN) [25] standardizes each feature map with respect to each sample. Group Normalization (GN) [27] divides the feature channels within each mini-batch into several groups and then performs the standardization for each group. GN's computation is also independent of batch sizes, and we consider it an instance-level normalization technique that can be augmented with ILM Norm.

Another way to do normalization is adjusting the filter weights instead of modifying the feature maps. For example, Weight Normalization [20] and Orthogonal Weight Normalization [7] present this kind of normalization strategy to address some recognition tasks.

We observe that the existing normalization methods merely focus on manipulating the learning of parameters at the *standardization stage*. They do not consider the correlations between the standardization stage and the rescaling stage. The parameters learned for rescaling are based on back-propagation and might be of low correlation with the parameters for standardization. Our experimental results show that taking into account the connection between standardization and rescaling is beneficial.

2.2. Style Transfer with Rescaling Parameters

The goal of a style transfer task is to 'extract' or 'imitate' a visual style from one image and apply that style to another image. Likewise, domain adaptation aims to enable a function learned from one domain to work comparably well in another domain. One solution to this kind of task is manipulating the learned rescaling parameters, and therefore we quickly review some style transfer methods that are related to learning rescaling parameters.

The core idea of using the learned rescaling parameters to address the tasks of style transfer or domain adaptation is similar to the normalization process. The original distribution of one domain is standardized and then mapped to the target distribution in the target domain. Hence, the rescaling parameters learned from the target distribution can be used to recover the original distribution in the target domain.

Adaptive Instance Normalization [9] applies the rescaling parameters generated by another domain to the feature maps of the current domain via Instance Normalization. Dynamic Layer Normalization [14] generates the rescaling parameters by different speakers and environments for adaptive neural acoustic modeling via Layer Normalization.

3. Instance-Level Meta Normalization

This section describes the proposed two-stage learning mechanism for improving instance-level normalization. Our approach is applicable to various techniques that perform instance-level normalization, and hence we call it *Instance-Level Metal Normalization* (ILM Norm). Figure 2 shows an overview of ILM Norm. The first stage is *standardization*, which regularizes the mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\gamma}$ of the input feature tensor $\mathbf{x}$ for standardizing the distribution of the feature tensor. The second stage is *rescaling*, which rescales the standardized feature map $\mathbf{x}_s$ for recovering the representation capability of the feature tensor $\mathbf{x}$. Moreover, we employ an auto-encoder to serve as an *association* between two stages. The rescaling stage uses the auto-encoder to predict the rescaling parameters, *i.e.*, weight $\boldsymbol{\omega}$ and bias $\boldsymbol{\beta}$, with respect to the input tensor $\mathbf{x}$ of feature maps instead of generating the rescaling parameter simply from back-propagation.

3.1. Standardization Stage

The goal of the standardization stage is to regularize the distribution of the input feature map, which is often done by forcing the distribution to have zero mean and unit variance. Existing normalization techniques mostly focus on designing different schemes for this stage.

As a meta learning mechanism, ILM Norm can adopt different standardization processes from different instance-level normalization techniques. Take, for example, Group Normalization's standardization process. Group Normalization (GN) divides the whole layer into several groups along its channel dimension. Each group calculates its own mean and variance for standardization. Many other methods can be considered. It is free to be replaced by others for different purposes.

3.2. Rescaling Stage

The goal of the rescaling stage is to recover the distribution of the input feature maps from its standardized counterpart. Previous approaches usually learn the parameters for recovering the statistics merely via back-propagation. In contrast, ILM Norm predicts the parameters with additional association between the standardization stage and the rescaling stage. In the following, we detail the process of extracting the key features of the input tensor $\mathbf{x}$ of feature maps. The auto-encoder for predicting the rescaling parameters will be presented in Section 3.3.

3.3. Association between Two Stages

The association between the standardization stage and the rescaling stage is achieved by a coupled *auto-encoder*, which is of little computational cost. An overview of the components is shown in Figure 2. ILM Norm contains an auto-encoder for predicting the rescaling parameters $\boldsymbol{\omega}$ and $\boldsymbol{\beta}$ concerning the pre-computed mean $\boldsymbol{\mu}$ and the variance $\boldsymbol{\gamma}$ of the input feature tensor $\mathbf{x}$. In comparison with existing methods that simply learn the parameters $\boldsymbol{\omega}$ and $\boldsymbol{\beta}$ via back-propagation, our experiments show that the meta parameter learning mechanism with additional information from $\boldsymbol{\omega}$ and $\boldsymbol{\beta}$ is more effective.

3.3.1 Key Feature Extraction

ILM Norm uses an auto-encoder to predict the weights $\boldsymbol{\omega}$ and bias $\boldsymbol{\beta}$ as the rescaling parameters for recovering the distribution of the tensor $\mathbf{x}$ of feature maps. We have observed that directly encoding the entire input feature tensor $\mathbf{x}$ would degrade the prediction accuracy, which might be due to overfitting. Instead of using the entire feature tensor $\mathbf{x}$ as the input for the auto-encoder, we propose to use the mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\gamma}$ of $\mathbf{x}$ for characterizing its statistics. Here we define the key features as the mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\gamma}$ extracted from the feature tensor $\mathbf{x}$. The experimental results demonstrate that, ILM Norm, which uses a light-weight auto-encoder, can effectively predict $\boldsymbol{\omega}$ and $\boldsymbol{\beta}$ for recovering the distribution of input tensor $\mathbf{x}$ of feature maps.

Furthermore, for better performance and lower computation burden, we extract the key features from each group of input feature maps rather than a single feature map. For a specific layer comprising C channels as a tensor of feature maps $f_1, f_2, \ldots, f_C$, we evenly partition these feature maps into N groups $\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_N$. The mean and variance of the whole layer are hence denoted as a vector of length N, namely $\boldsymbol{\mu} = [\mu_1, \mu_2, \ldots, \mu_N]$ and $\boldsymbol{\gamma} = [\gamma_1, \gamma_2, \ldots, \gamma_N]$. ILM Norm computes the mean μ_n and variance γ_n for a given feature-map group $\mathbf{f}_n$ as

$$\left\{ \begin{array}{l} \mu_n = \frac{1}{H \times W \times C/N} \sum_{f \in \mathbf{f}_n} \sum_{i=1}^{H} \sum_{j=1}^{W} f^{i,j} \,, \\ \gamma_n = \frac{1}{H \times W \times C/N} \sum_{f \in \mathbf{f}_n} \sum_{i=1}^{H} \sum_{j=1}^{W} (f^{i,j} - \mu_n)^2 \,, \end{array} \right. \tag{3}$$

where C/N is the number of feature maps in a group, f denotes a feature map of group $\mathbf{f}_n$. Further discussions for key feature extraction can be found in Section 4.5.2.

3.3.2 Encoder

The goal of the encoder in ILM Norm is to summarize the information of an input tensors key features through an embedding. Besides, we expect the subsequent rescaling parameters can be jointly learned from the same embedded information.

In our implementation, the encoder comprises one *fully connected layer* ($\mathbf{W}_1$) and one *activation function*. The fully connected layer can model not only the individual elements of the key features but also the correlations between elements. Using an activation function allows us to extract non-linear information. The embedded vectors, which encode the mean and the variance of the grouped input feature maps, are obtained by

$$\left\{ \begin{array}{l} E_{\boldsymbol{\mu}} = \mathrm{ReLU}(\mathbf{W}_1 \boldsymbol{\mu}) \,, \\ E_{\boldsymbol{\gamma}} = \mathrm{ReLU}(\mathbf{W}_1 \boldsymbol{\gamma}) \,, \end{array} \right. \tag{4}$$

where $E_{\boldsymbol{\mu}}$ and $E_{\boldsymbol{\gamma}}$ respectively denote the embedded vectors of $\boldsymbol{\mu}$ and $\boldsymbol{\gamma}$, $\mathrm{ReLU}(\cdot)$ represents the activation function, and the encoding matrix $\mathbf{W}_1 \in \mathbb{R}^{M \times N}$ with the embedded vector of length M and key feature vectors of length N.

3.3.3 Decoder

The decoder in ILM Norm aims to decode the embedded vectors $E_{\boldsymbol{\mu}}$ and $E_{\boldsymbol{\gamma}}$ into $D_{\boldsymbol{\mu}}$ and $D_{\boldsymbol{\gamma}}$ respectively. In a sense, $D_{\boldsymbol{\mu}}$ and $D_{\boldsymbol{\gamma}}$ propagate the correlations from the original feature maps to the rescaling parameters $\boldsymbol{\omega}$ and $\boldsymbol{\beta}$.

In our implementation, we use two different *fully connected layers* ($\mathbf{W}_2$ and $\mathbf{W}_3$) and two *activation functions*. The fully connected layers for decoding aim to summarize the information-rich embedded vectors for predicting the rescaling parameters. By accompanying the decoded vector with an activation function, ILM Norm shifts the vector values into a suitable range. The decoded vectors, which yield the mean and variance of the embedded vector, are obtained as

$$\left\{ \begin{array}{l} D_{\boldsymbol{\mu}} = \tanh(\mathbf{W}_2 E_{\boldsymbol{\mu}}) \,, \\ D_{\boldsymbol{\gamma}} = \mathrm{sigmoid}(\mathbf{W}_3 E_{\boldsymbol{\gamma}}) \,, \end{array} \right. \tag{5}$$

where both $\mathrm{sigmoid}(\cdot)$ and $\tanh(\cdot)$ represent the activation functions, and the decoding matrices $\mathbf{W}_2$, $\mathbf{W}_3 \in \mathbb{R}^{N \times M}$. Further discussions about choosing activation functions can be found in Section 4.5.1.

3.3.4 Alignment

Notice that each decoded vector of $D_{\boldsymbol{\mu}}$ and $D_{\boldsymbol{\gamma}}$ predicted from the auto-decoder needs to align with a corresponding rescaling parameters of the underlying normalization module in which the ILM Norm is plugged, , *e.g.*, Instance Normalization. We obtain the final rescaling parameters $\boldsymbol{\omega}$ and $\boldsymbol{\beta}$ as follow:

$$\begin{cases} \boldsymbol{\omega} = D_{\boldsymbol{\gamma}} \uparrow + B_{\boldsymbol{\omega}} \, , \\ \boldsymbol{\beta} = D_{\boldsymbol{\mu}} \uparrow + B_{\boldsymbol{\beta}} \, , \end{cases} \tag{6}$$

where $B_{\boldsymbol{\omega}}$ and $B_{\boldsymbol{\beta}}$ denote the rescaling parameters of the underlying normalization module that is augmented by ILM Norm. The dimension of either $B_{\boldsymbol{\omega}}$ or $B_{\boldsymbol{\beta}}$ is C, *i.e.*, the number of channels. The operator $\uparrow$ means duplicating the vector components so that the dimension of $D_{\boldsymbol{\gamma}}$ and $B_{\boldsymbol{\omega}}$ can match, and same for $D_{\boldsymbol{\mu}}$ and $B_{\boldsymbol{\beta}}$.

4. Experiments

In the experiments we evaluate ILM Norm using different datasets on various tasks. We apply ILM Norm to several state-of-the-art instance-level normalization techniques, including Layer Normalization (LN), Instance Normalization (IN), and Group Normalization (GN), and we show that the ILM Norm enhanced versions steadily outperform the original ones.

4.1. Image classification with Large Batch Size

4.1.1 Implementation Details

We use ResNet-50 and ResNet-101 [6] as a backbone model for evaluating the experiments on classification tasks. For CIFAR-10 and CIFAR-100 datasets, we change the first conv layer to "3×3, stride 1, padding 1", remove the max-pooling layer, and change the kernel size of average-pooling to 4 to adapt to the input size. We initialize all parameters using the standard normal distribution except the rescaling parameters of the corresponding underlying normalization module $B_{\boldsymbol{\omega}}$ and $B_{\boldsymbol{\beta}}$, which are assigned as 1 and 0, respectively.

Unless otherwise stated, ILM Norm set the size of group equals to 16 (*i.e.*, $C/N = 16$), the number N of groups of GN is set to 32, and the batch size of all normalization methods is 64. We use SGD as the optimizer with momentum 0.9 and weight decay 0.0005. All experiment are conducted on only one GPU. For CIFAR-10 and CIFAR-100 datasets, each normalization method is trained for 350 epochs. The learning rate is initialized with 0.1 and decreased by 0.1 at the 150th and 250th epoch. For ImageNet, each normalization method is trained for 100 epochs. We set learning rate as 0.025 according to the suggestion of [4]. The learning rate is decreased by 0.1 at 30th, 60th and 90th epoch.

Top-1 Error (%)	Method			
	BN	GN	IN	LN
Original	6.43	7.02	7.00	9.98
with ILM	-	5.88	6.50	7.35
with ILM vs. Original	-	−1.14	−0.50	−2.63
with ILM vs. BN	-	−0.55	+0.07	+0.92

Table 1. Comparison of different normalization methods on CIFAR-10.

Top-1 Error (%)	Method			
	BN	GN	IN	LN
Original	26.28	26.94	26.08	41.61
+ ILM	-	23.31	23.97	25.43
+ILM vs. Original	-	−3.63	−2.11	−16.18
+ILM vs. BN	-	−2.97	−2.31	−0.85

Top-5 Error (%)	Method			
	BN	GN	IN	LN
Original	9.37	7.02	7.60	15.26
+ILM	-	6.47	6.71	6.88
+ILM vs. Original	-	−0.55	−0.89	−8.38
+ILM vs. BN	-	−2.90	−2.66	−2.49

Table 2. Comparison of different normalization methods on CIFAR-100.

4.1.2 CIFAR-10

We compare several instance-level normalization methods (GN, IN, LN) with their ILM Norm extensions for image classification on CIFAR-10 dataset [15, 23]. The underlying architecture is ResNet-101. We also present the result of Batch Normalization (BN) trained under the same configuration as a strong baseline. The results are shown in Figure 3 and Table 1.

Figure 3 shows the comparisons of different instance-level normalization techniques. We plot the validation error rate against the number of training epochs. ILM Norm is applied to IN, GN and LN, and all the three normalization methods can be improved to achieve lower validation error rates.

Table 1 shows the error rates of different methods with 350 training epochs. Notice that, in the last two rows of Table 1 we compare the change in performance after applying ILM Norm. We show the relative error rate w.r.t. the original normalization and w.r.t. BN. As can be seen, all instance-level normalization methods can achieve a lower error rate after being equipped with ILM Norm. Furthermore, the combination of ILM+GN can even outperform BN. It is worth mentioning that, to our best knowledge, no existing state-of-the-art instance-level normalization methods have outperformed BN when a large batch size is used on the CIFAR-10 classification task.

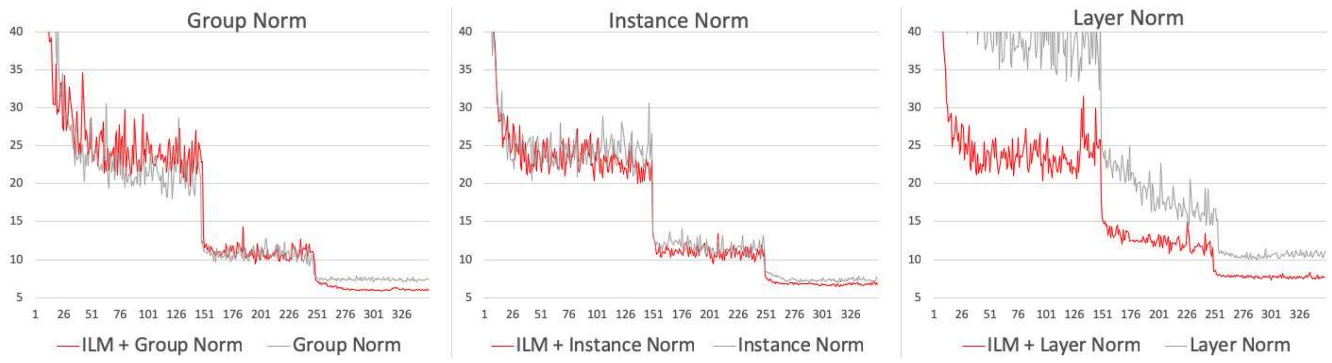

Figure 3. Comparisons of different instance-level normalization techniques. We show the validation error rate (%) against the number of training epochs. The batch size is 64. The performances of the original normalization techniques are improved after applying ILM Norm.

ImageNet	Method (Error Rate %)				
	BN	ILM+GN	GN	IN	LN
Top-1	23.85	23.57	24.06	28.40	25.30
vs. BN	-	-0.28	$+0.21$	$+4.55$	$+1.45$

Table 3. Comparison of different normalization methods on ImageNet.

4.1.3 CIFAR-100

We conduct another similar experiment to compare different normalization methods on CIFAR-100 [15, 23] image classification task. All models are trained on the training set of $50{,}000$ images and evaluated on the validation set of $10{,}000$ images for 350 epochs. The results are shown in Table 2. Similar improvements on the CIFAR-100 classification task for different methods can be observed.

4.1.4 ImageNet

We also use ImageNet to evaluate the setting of ILM Norm plus GN (ILM+GN), in comparison with other normalization methods including BN, IN, LN, and the original GN. The underlying network architecture is ResNet-50. The ImageNet dataset contains over one million images with 1000 different classes. All of the models are trained on the ImageNet training set and evaluated on the validation set. The results are in Table 3.

Table 3 shows the error rates after 100 training epochs for different normalization methods. We can find that ILM+GN achieves a 0.49% lower error rate than the original GN. Moreover, ILM+GN achieves a 0.28% lower error rate than Batch Normalization as well, while the basic instance-level normalization methods cannot outperform BN on this task.

In sum, the experiments on classification tasks with CIFAR-10, CIFAR-100, and ImageNet demonstrate that instance-level normalization methods, such as GN, IN, and LN, can be improved if they are equipped with ILM Norm. Furthermore, ILM+GN is able to achieve better performance than cross-instance normalization like Batch Normalization for a large-batch-size setting on ImageNet, which has never been reported before according to our best knowledge. The advantage of ILM Norm for various instance-level normalization methods is therefore evident, and the improvement can be handily achieved with a negligible computation overhead.

4.2. Image Classification with Various Batch Sizes

The batch size is an issue to be taken into consideration when apply normalization techniques. We conduct an experiment to evaluate ILM Norm plus GN for various batch sizes on CIFAR-10. We test the batch sizes of $\{64, 32, 16, 8, 4, 2\}$ per GPU, without changing other hyper-parameters. For comparison, we also include the results of BN. The error rates are shown in Table 4 and Figure 1.

Figure 1 clearly illustrates that both GN and ILM+GN are not sensitive to the batch size. Furthermore, ILM+GN gets lower validation error rates than GN among all kinds of batch sizes. In contrast, BN obviously requires a larger batch size and gets considerable large error rates when the batch size is small.

Table 4 shows that ILM+GN has the lowest error rates among all batch sizes. On average, ILM plus GN achieves a lower error rate than GN by 0.58% and also a lower error rate than BN by 2.55% among the evaluated batch sizes.

Discussion. Table 4 shows that ILM+GN outperforms GN among all batch sizes. Since all hyper-parameters of ILM+GN are set the same as BN, it is reasonable to consider that the improvement is owing to the association mechanism of ILM Norm that connects the standardization stage and the rescaling stage. As a result, it is helpful to leverage both the cross-stage association and the back-propagation process while learning the rescaling parameters for normalization.

CIFAR-10: Error Rate $(\%)$		Batch Size					
		64	32	16	8	4	2
Method	GN	7.02	7.14	7.24	7.22	7.31	7.29
	BN	6.43	6.48	7.39	9.78	11.15	13.79
	ILM+GN	**5.88**	**6.36**	**6.64**	**6.70**	**7.05**	**7.11**
Improvement	ILM+GN vs. GN	-1.14	-0.78	-0.60	-0.52	-0.26	-0.18
	ILM+GN vs. BN	-0.55	-0.12	-0.75	-3.08	-4.10	-6.68

Table 4. Evaluations on CIFAR-10 dataset with different batch sizes.

Batch Size	Box Head	AP^{bbox}	AP^{bbox}_{50}	AP^{bbox}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}
2	GN	22.40	37.65	23.56	20.88	35.43	21.57
	ILM + GN	**22.68**	**38.28**	**23.68**	**21.19**	**36.08**	**21.92**
16	GN	39.10	60.33	42.51	34.77	56.88	36.79
	ILM + GN	**39.42**	**60.63**	**42.95**	**35.03**	**57.25**	**36.92**

Table 5. Evaluations on MS-COCO dataset for detection and segmentation tasks with different batch sizes.

Metric	Method (Generator / Discriminator)		
	IN / IN	ILM+IN / IN	ILM+IN / ILM+IN
RMSE	108.17	105.82	**105.46**
LPIPS	0.441	0.435	**0.428**
SSIM	0.372	**0.390**	0.372

Table 6. Evaluation on the Facades dataset for the style transfer task. Note that, typically, a higher SSIM score means higher similarity, while a lower RMSE or LPIPS value implies better performance.

4.3. Object Detection and Segmentation

Object detection and segmentation are important tasks in computer vision. We evaluate ILM Norm on Mask R-CNN [5] using MS-COCO dataset [17]. All models are trained on the training set for 90,000 with batch size per GPU equal to 2 using 1 GPU and 8 GPUs. The backbone used in all models are pretrained with GN. We test the models on the test set. All other configurations are just the same as R-50-FPN in Detectron [3]. The results are shown in Table 5.

Table 5 shows that only changing GN layers in the box head can improve the detection and segmentation performance under different batch sizes. To be more specific, we increase AP^{bbox} and AP^{mask} by 0.28 and 0.31 when the batch size is equal to 2, and by 0.32 and 0.26 when batch size is equal to 16. It indicates that ILM+GN can transfer the feature from backbone more efficiently than GN, compared with the GN baseline, with different training lengths.

4.4. Image Transfer

Image transfer is a popular and interesting task in computer vision. We evaluate ILM Norm on pix2pix [12] using the CMP Facades [24] dataset with size of group equals to 1. The Facades dataset contains 400 architectural-labels-to-photo data. We train the model in 200 of them and evaluate on the rest 200 data for 200 epochs. To evaluate the performance, we use SSIM [26], RMSE, and the LPIPS metric [28] as the similarity measures. Typically, a higher SSIM score means higher similarity, while a lower RMSE or LPIPS value implies better quality. The results are shown in Table 6.

Table 6 clearly shows that changing all IN layers in the model or only changing IN in the generator can both improve the similarity between the output of the model and the target. Since LPIPS focuses on not only the structural similarity but also the perceptual similarity, using ILM+IN can produce style transfer results with better structural and perceptual quality than original IN.

4.5. Ablation Study

4.5.1 Different Activation Functions for D_μ and D_γ

Since the rescaling parameters are the only part in the model that is modified during forward propagation, it is critical to control the extent of their variations. Excessive variations in rescaling parameters lead to instability of the model. Moreover, the domain of μ and γ are different; it is reasonable to control D_μ and D_γ within a different range. To verify our assumptions, we evaluate ILM Norm with several different activation functions applied on D_μ and D_γ. The results can be found in Table 7.

Table 7 shows that the model cannot converge without appropriate constraints on D_μ and D_γ. Applying the same activate function, *e.g.* sigmoid, to both D_μ and D_γ may make the model converge, but the performance is even worse than the original group normalization, indicating that the association has a negative impact on the normalization. Only by deploying different activation functions, tanh to D_μ and sigmoid D_γ can we achieve positive impact and the best performance among these configurations.

	Activation Functions for (D_μ, D_γ)						GN
	ours	(sigmoid, sigmoid)	(tanh, tanh)	(†, ⋆)	(⋆, †)	(⋆, ⋆)	
Error Rate %	**5.88**	8.63	8.04	n/a	n/a	n/a	7.02

Table 7. Comparison of different activation functions for (D_μ, D_γ) on CIFAR-10. The symbol † means the activation function is either tanh or sigmoid, while ⋆ means the activation function can be ReLU, Leaky ReLU, ReLU6, or Identity. The entry 'n/a' indicates that the model cannot converge.

Increment Ratio of Parameters	Model				
	ResNet-18	ResNet-34	ResNet-50	ResNet-101	ResNet-152
Group Normalization	0.015%	0.015%	0.086%	0.086%	0.086%
Instance Normalization	2.792%	2.462%	20.696%	20.313%	20.178%

Table 8. The increment in the number of parameters concerning different key-feature extraction strategies. ILM Norm chooses to use GN's scheme for key-feature extraction since the increment in number of additional parameters is less than 0.1% for most of the ResNet models.

4.5.2 Alternative Strategies for Key-Feature Extraction

ILM Norm divides the input channels into groups for computing the mean and variance per group. As mentioned in Section 2.1, the existing normalization methods, such as BN, LN, IN, and GN have their own scheme to extract the mean $\boldsymbol{\mu}$ and variance γ from the input $\mathbf{x}$. To make our normalization mechanism robust to various batch sizes, we do not consider the scheme of BN. Moreover, we also exclude LN's scheme, since LN only generates one pair of mean and variance for all feature maps in a layer, and such few data is not suitable for training our auto-encoder-like network. Therefore, our key-feature extraction strategy considers the implementations as IN and GN. Figure 4 shows a comparison of the key feature extraction strategies derived from GN and IN. Table 8 provides the ratio of increment in the number of parameters concerning different key-feature extraction strategies.

Figure 4 shows that the performance of using a key-feature extraction strategy as GN is usually better than the performance of using IN. From the perspective of the increment in the number of parameters, Table 8 provides further information for choosing the strategy of key feature extraction. In Table 8, the number of additional parameters due to the use of the key-feature extraction strategy as GN is quite small. The lower requirement of additional parameters for GN is because it partitions the C channels into N groups, where $N = C/K$ and the size of a group is fixed to K, and hence the increment in the number of parameters depends on the ratio C/K instead of C. In contrast, the increment in the number parameters using IN's scheme depends on C.

To sum up: The experiments demonstrate that using the key-feature extraction strategy as GN should be the best option. It not only achieves a lower error rate but also requires less increment in the number of parameters.

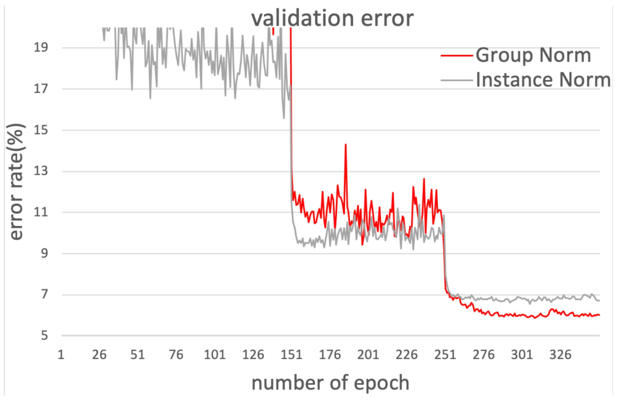

Figure 4. Comparison on the performance of using a key-feature extraction strategy as GN or IN. The evaluation is based on CIFAR-10 validation error.

5. Conclusion

We have presented ILM Norm, a meta learning mechanism for various instance-level normalization techniques. ILM Norm extracts the key features from the input tensor and associates the standardization parameters with the rescaling parameters for deep network normalization. As a result, ILM Norm provides an easy way to predict the rescaling parameters via both the update from back-propagation and the association with input features. ILM works well with state-of-the-art instance-level normalization methods, and meanwhile, improves the performance in most cases. The experiments demonstrate that a deep network equipped with ILM Norm is able to achieve better performance for different batch sizes with just a little increase in the number of parameters.

Acknowledgement: This work was supported in part by MOST Grants 108-2634-F-001-007 and 106-2221-E-007-080-MY3. We thank Ting-I Hsieh, Yi-Chuan Chang, Wen-Chi Chin, and Yi-Chun Lin for insightful discussions. We are also grateful to the *National Center for High-performance Computing* for providing computational resources and facilities.

References

[1] D. Arpit, Y. Zhou, B. U. Kota, and V. Govindaraju. Normalization propagation: A parametric technique for removing internal covariate shift in deep networks. In *ICML*, pages 1168–1176, 2016. 3

[2] L. J. Ba, R. Kiros, and G. E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016. 2, 3

[3] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár, and K. He. Detectron. https://github.com/facebookresearch/detectron, 2018. 7

[4] P. Goyal, P. Dollár, R. B. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He. Accurate, large minibatch SGD: training imagenet in 1 hour. *CoRR*, abs/1706.02677, 2017. 5

[5] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick. Mask R-CNN. In *ICCV*, pages 2980–2988, 2017. 7

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 1, 5

[7] L. Huang, X. Liu, B. Lang, A. W. Yu, Y. Wang, and B. Li. Orthogonal weight normalization: Solution to optimization over multiple dependent stiefel manifolds in deep neural networks. In *AAAI*, 2018. 3

[8] L. Huang, D. Yang, B. Lang, and J. Deng. Decorrelated batch normalization. In *CVPR*, 2018. 3

[9] X. Huang and S. J. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, pages 1510–1519, 2017. 3

[10] S. Ioffe. Batch renormalization: Towards reducing minibatch dependence in batch-normalized models. In *NIPS*, pages 1942–1950, 2017. 3

[11] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456, 2015. 1, 3

[12] P. Isola, J. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *CoRR*, abs/1611.07004, 2016. 7

[13] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun. What is the best multi-stage architecture for object recognition? In *ICCV*, pages 2146–2153, 2009. 3

[14] T. Kim, I. Song, and Y. Bengio. Dynamic layer normalization for adaptive neural acoustic modeling in speech recognition. In *Interspeech*, pages 2411–2415, 2017. 3

[15] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009. 5, 6

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1106–1114, 2012. 3

[17] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 7

[18] S. Lyu and E. P. Simoncelli. Nonlinear image representation using divisive normalization. In *CVPR*, 2008. 3

[19] M. Ren, R. Liao, R. Urtasun, F. H. Sinz, and R. S. Zemel. Normalizing the normalizers: Comparing and extending network normalization schemes. In *ICLR*, 2017. 3

[20] T. Salimans and D. P. Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *NIPS*, page 901, 2016. 3

[21] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *ICLR*, 2014. 3

[22] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, pages 1–9, 2015. 3

[23] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(11):1958–1970, 2008. 5, 6

[24] R. Tylecek and R. Sára. Spatial pattern templates for recognition of objects with regular structure. In *Pattern Recognition - 35th German Conference, GCPR 2013, Saarbrücken, Germany, September 3-6, 2013. Proceedings*, pages 364–374, 2013. 7

[25] D. Ulyanov, A. Vedaldi, and V. S. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *CoRR*, abs/1607.08022, 2016. 2, 3

[26] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Processing*, 13(4):600–612, 2004. 7

[27] Y. Wu and K. He. Group normalization. In *ECCV*, 2018. 1, 2, 3

[28] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. *CoRR*, abs/1801.03924, 2018. 7