SHAKEDROP REGULARIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper proposes a powerful regularization method named *ShakeDrop regularization*. ShakeDrop is inspired by Shake-Shake regularization that decreases error rates by disturbing learning. Important and interesting feature of ShakeDrop is that it strongly disturbs learning by multiplying even a negative factor to the output of a convolutional layer in the forward training pass. In addition, in the backward training pass, a different factor from the forward pass is multiplied. As a byproduct, however, learning process gets unstable. Hence, the learning process is stabilized by employing the essence of the Stochastic Depth (ResDrop). Combined with existing techniques (longer training and image preprocessing), ShakeDrop achieved error rates of 2.31% (better by 0.25%) on the CIFAR-10 dataset and 12.19% (better by 2.85%) on the CIFAR-100 dataset.

1 INTRODUCTION

It has been believed that *deeper* networks achieve better performance. In the generic object recognition problem using the ImageNet dataset (Deng et al., 2009), 8-layer AlexNet (Krizhevsky et al., 2012) outperformed conventional shallow methods. VGGNet (Simonyan & Zisserman, 2015) and GoogLeNet (Szegedy et al., 2015) realized 19- and 22-layer deep convolutional neural networks (CNNs), respectively. Subsequently, ResNet (He et al., 2016) opened the door to very deep CNNs of over a hundred layers by introducing the residual block. Error rates are further decreased by PyramdNet (Han et al., 2017a;b) with up to 200 layers. We refer these methods by *deep* network architectures.

A different approach from chasing *deeper* network architectures is to focus on *wider* or *multi-branch* network architectures. We refer them by *non-deep* network architectures. WRN(Zagoruyko & Komodakis, 2016) increased channels of residual blocks and reduced error rates. ResNeXt (Xie et al., 2017) which has residual blocks consisting of multiple thin residual branches achieved lower error rates than WRN.

One may wonder which network architecture, deep or non-deep, better is. Figure 1 shows error rates of representative deep (in red) and non-deep networks (in blue) on the CIFAR-100 datasets (Krizhevsky, 2009) together with the number of parameters. It shows that PyramidNet, which is the best vanilla method of the deep network architecture, was better than ResNeXt, which is the best vanilla method of non-deep network architecture.

Nevertheless, recent advances are mainly brought to networks of the *wide* approach. In particular, Shake-Shake regularization (Gastaldi, 2017) is applied to ResNeXt. In addition, Cutout (DeVries & Taylor, 2017), an image pre-processing method, is applied to Shake-Shake. Hence, a straightforward strategy to achieve better performance is to introduce their essence to PyramidNet.

In this paper, we propose a powerful regularization method, named *ShakeDrop regularization*, designed for the *deep* network architectures. The proposed ShakeDrop is inspired by Shake-Shake, which disturbs learning based on different random variables in forward and backward training passes. It enables longer learning that would help more generalization. ShakeDrop disturbs learning more strongly by multiplying even a negative factor to the output of a convolutional layer. In addition, in the backward training pass, a different factor from the forward pass is multiplied. As a byproduct, however, learning process gets unstable. Hence, the learning process is stabilized by employing the essence of the Stochastic Depth (ResDrop).

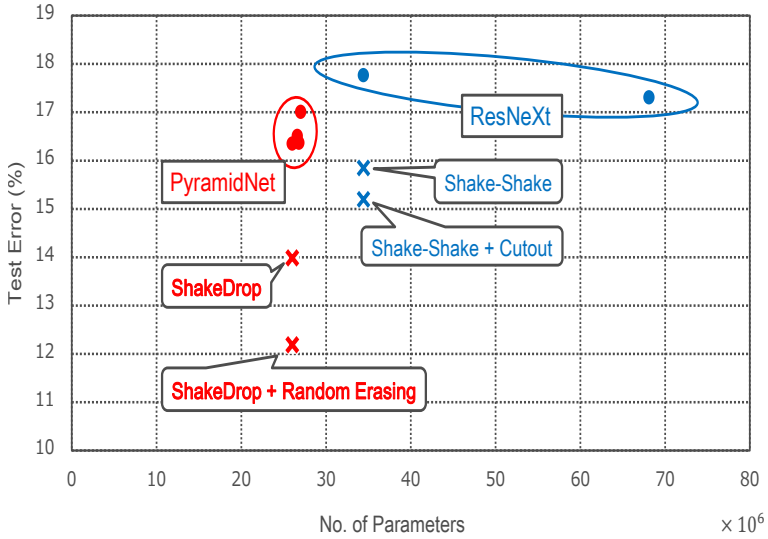


Figure 1: Error rates of representative methods of both deep (in red) and non-deep (in blue) network architectures together with the number of parameters on the CIFAR-100 dataset. PyramidNet, ShakeDrop and ShakeDrop+Random Erasing were better than ResNeXt, Shake-Shake and Shake-Shake+Cutout, respectively.

1.1 Contribution

The contributions of the paper are as follows:

1. This paper proposes ShakeDrop, a powerful and memory efficient regularization method. ShakeDrop combined with longer training achieved error rates of 2.67% (slightly worse than the state-of-the-art by 0.11%) on the CIFAR-10 dataset and 13.99% (better by 1.05%) on the CIFAR-100 dataset. In addition, combined with an existing image preprocessing technique, ShakeDrop achieved error rates of 2.31% (better by 0.25%) and 12.19% (better by 2.85%) on the CIFAR-10/100 datasets, respectively.
2. We showed the linear decay rule of Stochastic Depth is efficient for the stabilization of stochastic regularization like ShakeDrop.

2 Existing Methods Required to Introduce the Proposed Method

2.1 Deep Network architectures

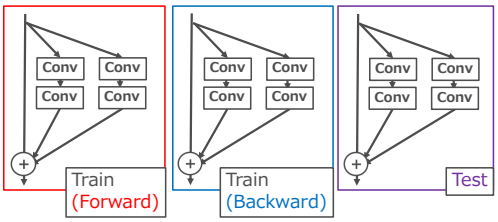
ResNet (He et al., 2016) opened the door to very deep CNNs of over a hundred layers by introducing the residual block, given as

$$G(x) = x + F(x), \tag{1}$$

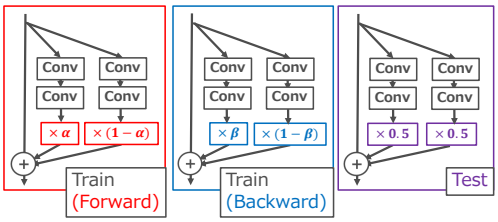
where x is the input of the residual block, $G(x)$ is the output of the residual block and $F(x)$ is the output of residual branch on the residual block.

While ResNet realized deep networks, it is imperfect. He et al. (2016) pointed out that high error rates obtained by the 1202-layer ResNet on the CIFAR datasets is caused by overfitting. Veit et al. (2016) experimentally showed that some residual blocks whose channels greatly increase cause high error rates.

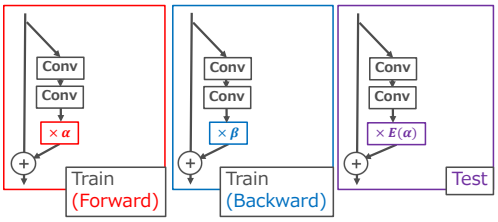
PyramidNet (Han et al., 2017a;b) overcame the problem of ResNet by gradually increasing channels on each residual block. It has almost same residual block as Eqn. (1). It successfully realized *deep* CNNs of up to 272 layers and achieved the lowest error rates among the vanilla residual networks on the CIFAR datasets.



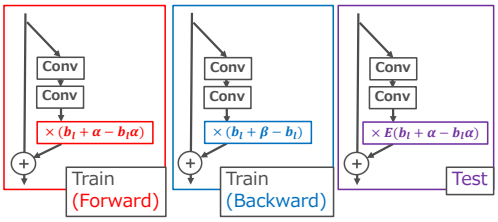
(a) ResNeXt (Xie et al., 2017), in which some proceeding layers omitted for conciseness.



(b) Shake-Shake (Gastaldi, 2017), in which some proceeding layers omitted for conciseness.



(c) PyramidShake, an intermediate network in which essence of Shake-Shake is introduced to PyramiNet.



(d) ShakeDrop, the proposed method, where essence of Stochastic Depth is introduced to PyramidShake.

Figure 2: Network architectures.

2.2 Non-Deep network architectures

WRN (Zagoruyko & Komodakis, 2016) improved error rates by simply increasing channels of ResNet. While it has almost same residual block as Eqn. (1), it has wider and shallower architecture than ResNet. Jastrzebski et al. (2017) experimentally showed that WRN overcomes the problem of ResNet like PyramidNet.

ResNeXt (Xie et al., 2017) achieved the better error rates than WRN on almost same number of parameters. The basic architecture of ResNeXt is given as

$$G(x) = x + F_1(x) + F_2(x), \tag{2}$$

where $F_1(x)$ and $F_2(x)$ are the outputs of the residual branches as shown in Figure 2(a). The number of residual branches is not limited to 2, and the number is the most important factor to control the result.

2.3 Regularization

Stochastic Depth (Huang et al., 2016) is a regularization method which overcame the ResNet problem. It was originally proposed for improving ResNet. It was also applied to PyramidNet; the combination of PyramidNet and Stochastic Depth was named PyramidDrop (Yamada et al., 2016).

On the lth residual block from input layer, the Stochastic Depth process is given as

$$G(x) = x + b_l F(x), \tag{3}$$

where $b_l \in \{0, 1\}$ is a Bernoulli random variable with the probability of p_l. p_l is defined by linear decay rule for stabilizing learning as

$$p_l = 1 - \frac{l}{L}(1 - p_L), \tag{4}$$

where L is the number of all layers and p_L is the initial parameter.

Shake-Shake (Gastaldi, 2017) is a powerful regularization method for improving the ResNeXt architecture. Its architecture is almost same as Eqn. (2) and is given as

$$G(x) = x + \alpha F_1(x) + (1 - \alpha) F_2(x), \tag{5}$$

where α is a coefficient given as $\alpha \in [0, 1]$. As shown Figure 2(b), calculation of the backward pass is disturbed by $\beta \in [0, 1]$ instead of α. As a result, Shake-Shake achieved lower error rates than ResNeXt.

3 PROPOSED METHOD

PyramidNet is the best method on the vanilla residual networks. Thus, it is expected that PyramidNet combined with any powerful regularization method outperforms Shake-Shake (i.e., ResNeXt + a powerful regularization method). Possible regularization methods would be Stochastic Depth and Shake-Shake. As mentioned above, PyramidNet combined with Stochastic Depth, called PyramidDrop, was examined by Yamada et al. (2016). Its error rates on the CIFAR datasets, however, were only slightly better than PyramidNet. On the other hand, Shake-Shake has a memory issue. As shown in Figure 2(b), the Residual block of Shake-Shake includes two branches of aggregated transforms, and each branch has to store the calculation results of the forward pass for gradient calculation (the backward pass). It requires so much memory storage that the regularization is hard to be used on the *deep* network architectures.

Overcoming the memory problem, we focus on the characteristic of ResNet. Veit et al. (2016) suggested that even if one residual block is removed from ResNet, its error rate was almost same as the complete ResNet. In addition, Han et al. (2017b) experimentally showed that PyramidNet is more robust to a defect of a residual block than ResNet. That is, residual functions do not strongly depend on each other and ResNet behaves like ensembles of relatively shallow networks. Hence, it can be regarded that the features of residual functions are blended. It can be also regarded that sum of the aggregated transforms of ResNeXt blends the features. Hence, even if there is only one branch on a residual block, it can simulate the behavior of Shake-Shake with use of random variables for disturbing learning. The new regularization method which has only one branch on each residual block is expected to be powerful but slight memory overhead.

PyramidShake is a method where essence of Shake-Shake is introduced to PyramidNet, given as

$$G(x) = x + \alpha F(x), \tag{6}$$

where α is a coefficient similar to the one of Shake-Shake. Eqn. (6) is equivalent to Eqn. (5) when $F_1(x) = F(x)$ and $F_2(x) = 0$. As shown in Figure 2(c), β is a coefficient similar to the one of Shake-Shake and used in the backward pass instead of α. The PyramidShake is expected to realize powerful generalization like Shake-Shake regularization. However, the result of a simple experiment on the CIFAR-100 dataset, where 110-layer PyramidShake was used with $\alpha \in [0, 1]$ and $\beta \in [0, 1]$ following Shake-Shake, was hopelessly bad (i.e., an error rate of 77.99%).

Actually, Eqn. (6) can be regarded as an extension to Eqn. (3) of Stochastic Depth because these equations are equal if $\alpha = \beta$ and they are Bernoulli random variables with the probability of $p_l = p_L$. Huang et al. (2016) reported that Stochastic Depth with the uniform rule where $p_l = p_L$ did not work well, while the linear decay rule given by Eqn. (4) worked. They indicate that, like Stochastic Depth with uniform rule, PyramidShake is expected to be inherently sensitive to the perturbation by α and β. Therefore, we will propose *ShakeDrop*.

ShakeDrop is a new powerful regularization method which is introduced essence of linear decay rule to PyramidShake for overcoming the sensitiveness. It is given as

$$G(x) = x + (b_l + \alpha - b_l\alpha)F(x), \tag{7}$$

where b_l is a Bernoulli random variable following the linear decay rule. We call the new method "ShakeDrop." According to the value of b_l, Eqn. (7) is deformed as

$$G(x) = \begin{cases} x + F(x), & \text{if } b_l = 1 \\ x + \alpha F(x), & \text{otherwise.} \end{cases} \tag{8}$$

If $b_l = 1$, the output of Eqn. (8) is same as Eqn. (1). If $b_l = 0$, it is the same as Eqn. (6). In other words, If $b_l = 1$, ShakeDrop is equivalent to PyramidNet, and if $b_l = 0$, it is equivalent to PyramidShake. As shown in Figure 2(d), similar to Shake-Shake, β is used in the backward pass instead of α. ShakeDrop is a method aimed at powerful regularization with excellent memory efficiency by combination of linear decay rule like Stochastic Depth and the disturbed residual branch like Shake-Shake.

Table 1: Average Top-1 errors (%) of ShakeDrop with several ranges of parameters of 4 runs at the final (300) epoch on CIFAR-100 dataset in the "Batch" level. In some settings, it is equivalent to PyramidNet and PyramidDrop.

Method	α	β	Error (%)	Note
ShakeDrop (Proposed)	1	1	18.01	Equivalent to PyramidNet
	0	0	17.74	Equivalent to PyramidDrop
	$[0, 1]$	$[-1, 1]$	20.61	
	$[0, 1]$	$[0, 1]$	18.27	
	$[-1, 1]$	1	18.68	
	$[-1, 1]$	0	17.28	
	$[-1, 1]$	$[-1, 1]$	18.26	
	$[-1, 1]$	$[0, 1]$	**16.22**	

Table 2: Average Top-1 errors (%) of ShakeDrop with different levels of 4 runs at the final (300) epoch on CIFAR-100 dataset.

Method	α	β	Level	Error (%)
ShakeDrop (Proposed)	$[-1, 1]$	$[0, 1]$	Batch	16.22
	$[-1, 1]$	$[0, 1]$	Image	16.04
	$[-1, 1]$	$[0, 1]$	Channel	**15.78**
	$[-1, 1]$	$[0, 1]$	Pixel	**15.78**

4 EXPERIMENTS

4.1 IMPLEMENTATION DETAILS

All implementations used in the experiments were based on the publicly available code of *fb.resnet.torch*[1]. ShakeDrop was trained using back-propagation by stochastic gradient descent with an approximate version [2] of Nesterov accelerated gradient (Nesterov, 1983) and momentum method (Rumelhart et al., 1986). Input images were produced in the following manner. An original image of 32×32 pixels was color-normalized, followed by horizontally flipped with a 50% probability. Then, it was zero-padded to be 40×40 pixels and randomly cropped to be an image of 32×32 pixels. The initial learning late was set to 0.5 for both CIFAR-10/100 datasets following the version 2 of Han et al. (2017b), while they use 0.1 for the CIFAR-10 dataset and 0.5 for the CIFAR-100 dataset since version 3. The initial learning rate was decayed by a factor of 0.1 at $1/2$ and $3/4$ of the entire learning process (300 epochs), respectively, following Han et al. (2017b). As the filter parameters initializer, "MSRA" (He et al., 2015) was used. In addition, a weight decay of 0.0001, a momentum of 0.9, and a batch size of 128 were used. On PyramidDrop and ShakeDrop, the linear decay parameter $p_L = 0.5$ was used following Huang et al. (2016). 4 GPUs were used for learning acceleration; due to parallel processing, even the same layer had different values of parameters p_L, α and β depending on GPUs.

4.2 PRELIMINARY EXPERIMENTS: SEARCH FOR PARAMETER SETTINGS OF THE PROPOSED METHOD

The best parameter settings of the proposed method, ShapeDrop, are searched. They include parameter ranges of α and β, and the timing these parameters are updated. In the preliminary experiments, networks were trained on the CIFAR-100 dataset. ShakeDrop had 110 layers, which consisted of a convolution layer, 54 additive pyramidal residual blocks and a fully connected layer. The number of channels at the last residual block was 286 channels. The number of channels at Table 1 shows representative parameter ranges of α and β we tried and their results. The table shows that the ranges

[1] `https://github.com/facebook/fb.resnet.torch`

[2] Please be aware that the implementation in *sgd.lua* of Torch7 with *nesterov mode* is an approximation of the original Nesterov accelerated gradient and momentum method. See `https://github.com/torch/optim/issues/27`.

$\alpha \in [-1, 1]$ and $\beta \in [0, 1]$ was the best. The ranges $\alpha \in [0, 1]$ and $\beta \in [0, 1]$, which was used in Gastaldi (2017), were not comparable.

Next, the best strategy to update the parameters (referred as scaling coefficient) among "Batch," "Image," "Channel" and "Pixel" are explored. "Batch" means that the same scaling coefficients are used for all the images in the mini-batch for each residual block. "Image" means that the same scaling coefficients are used for each image for each residual block. "Channel" means that the same scaling coefficients are used for the each channel for each residual block. "Pixel" means that the same scaling coefficients are used for the each element for each residual block. ShakeDrop was trained using the best parameters found in the experiment above, i.e, ranges $\alpha \in [-1, 1]$, $\beta \in [0, 1]$. Table2 shows that "Channel" level and "Pixel" level were the best. Among them, we selected "Channel" level because of memory efficiency.

Through the preliminary experiments, we found that the efficient range is $\alpha \in [-1, 1]$ and $\beta \in [0, 1]$, and found that the efficient level is "channel" level. We conduct the following experiments using these results.

4.3 Comparison with state-of-the-art methods

The proposed method was compared with state-of-the-arts on the CIFAR-10/100 datasets. State-of-the-art methods introduced some techniques that can be applied to many methods in the learning process. One is *longer learning*. While most of methods related to ResNet use 300-epoch scheduling for learning like the preliminary experiment in Sec. 4.2, Shake-Shake use 1800-epoch cosine annealing, on which the initial learning rate is annealed using a cosine function without restart (Gastaldi, 2017). Another one is *image preprocessing*. DeVries & Taylor (2017) and Zhong et al. (2017) showed that accuracy is improved by data augmentation which randomly fills a part of learning images. For fair comparison with these methods, we also applied them to the proposed method.

ShakeDrop (including PyramidNet and PyramidDrop) had 272 layers, which consisted of a convolution layer, 90 additive pyramidal "bottleneck" blocks ("bottleneck" is a special residual block, which is proposed by He et al. (2016)) and a fully connected layer. The number of channels at the last pyramidal residual block was 864. Table 3 shows the error rates. The proposed method, ShakeDrop, without *longer learning* and *image preprocessing*, was 3.41% on the CIFAR-10 dataset and 14.90% on the CIFAR-100 dataset. We make following comparisons with existing methods. Note that the proposed method used less numbers of parameters than rival methods in all conditions.

(1) Fair comparison with Shake-Shake in the same condition
ShakeDrop combined with *longer training* achieved error rates of 2.67% (better by 0.19%) on the CIFAR-10 dataset and 13.99% (better by 1.86%) on the CIFAR-100 dataset.

(2) Fair comparison with Cutout in the same condition
ShakeDrop combined with both *longer learning* and *image preprocessing* achieved error rates of 2.31% (better by 0.25%) and 12.19% (better by 3.01%) on the CIFAR-10/100 datasets, respectively.

(3) Comparison with the state-of-the-arts, Cutout on CIFAR-10 and Coupled Ensemble on CIFAR-100
ShakeDrop combined with both *longer learning* and *image preprocessing* achieved error rates of 2.31% (better by 0.25% than Cutout) and 12.19% (better by 2.85% than Coupled Ensemble) on the CIFAR-10/100 datasets, respectively.

4.4 Discussion

In order to ascertain why the proposed method achieved the low error rates, we investigate what kind of difference exists with the conventional methods. Under the same conditions as the preliminary experiment in Sec. 4.2, we recorded the change of learning loss on each epoch, the means of gradients on each iteration and the variances of gradients on each iteration. We compared with PyramidNet, PyramidDrop and "Batch" level ShakeDrop. We checked the 2nd convolutional layer of the 3 blocks, "first", "middle" and "final." "first" means the first residual block of the 54 blocks from the input layer. "middle" means the 27th residual block of the 54 blocks. "final" means the final residual block of the 54 blocks.

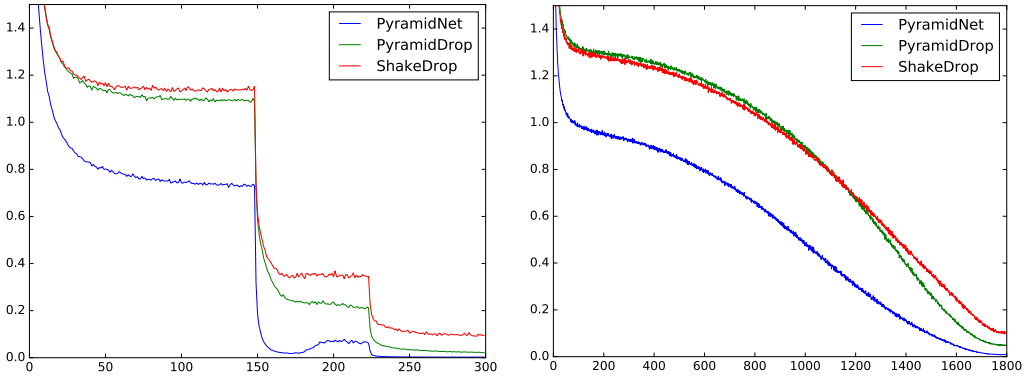
Table 3: Top-1 errors (%) at the final epoch (300 or 1800) on the CIFAR datasets. Representative methods of both *deep* and *non-deep* approaches and the proposed method, ShakeDrop are compared. If "Reg" is checked, any regularization is used. If "Cos" is checked, 1800-epoch cosine annealing schedule (Loshchilov & Hutter, 2016) is used following Gastaldi (2017). Otherwise, 300-epoch multi-step learning rate decay schedule is used following each method. If "Fil" is checked, the data augmentation used in Cutout (DeVries & Taylor, 2017) or Random Erasing (Zhong et al., 2017), which randomly fills a part of learning images, is used. While DeVries & Taylor (2017) used "Cutout" method as "Fil" process for Cutout on their experiment which we cited, we used "Random Erasing (Zhong et al. (2017))" as "Fil" for PyramidNet, PyramidDrop and ShakeDrop. $*$ indicates the result is quoted from the literature. $+$ indicates the result is quoted from Gastaldi (2017). Compared to the same condition of Cutout, the state-of-the-art, the proposed method reduced the error rate by 0.25% on CIFAR-10 and 3.01% on CIFAR-100.

	Method	Reg	Cos	Fil	Depth	#Param	CIFAR -10 (%)	CIFAR -100 (%)
	Coupled Ensemble (Dutt et al., 2017)				118	25.7M	*2.99	*16.18
					106	25.1M	*2.99	*15.68
					76	24.6M	*2.92	*15.76
					64	24.9M	*3.13	*15.95
					-	50M	*2.72	*15.13
					-	75M	*2.68	*15.04
					-	100M	*2.73	*15.05
Non-Deep	ResNeXt (Xie et al., 2017)		✓		26	26.2M	$^+$3.58	-
					29	34.4M	-	$^+$16.34
	Shake-Shake (Gastaldi, 2017)	✓	✓		26	26.2M	*2.86	-
					29	34.4M	-	*15.85
	Cutout (DeVries & Taylor, 2017)	✓	✓	✓	26	26.2M	*2.56	-
					29	34.4M	-	*15.20
Deep	PyramidNet (Han et al., 2017b)				272	26.0M	*3.31	*16.35
			✓	✓	272	26.0M	3.42	16.66
	PyramidDrop (Yamada et al., 2016)	✓			272	26.0M	3.83	15.94
		✓	✓	✓	272	26.0M	2.91	15.48
Deep	ShakeDrop (Proposed)	✓			272	26.0M	3.41	**14.90**
		✓		✓	272	26.0M	2.89	**13.85**
		✓	✓		272	26.0M	2.67	**13.99**
		✓	✓	✓	272	26.0M	**2.31**	**12.19**

The trends of average learning loss are provided in Fig. 3(a) PyramidDrop and ShakeDrop which introduced probabilistic regularization have a lower drop in loss than PyramidNet. While there is no big difference between PyramidDrop and ShakeDrop on the loss, there is big difference on the gradients. The trends of mean gradients are provided in Fig. 4, and the trends of mean gradients are provided in Fig. 5 PyramidDrop and ShakeDrop show significantly different values on the comparison of the gradients, although PyramidDrop and ShakeDrop were similar on the comparison of the loss. Especially, the difference at first is severe, and the difference is large also at middle.

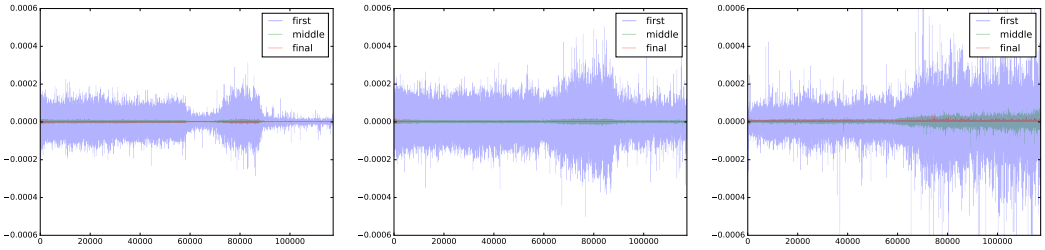
ShakeDrop uses α when extracting features. Since $\alpha \in [-1, 1]$ may take a negative value, each residual function may extract negative features that negate the features that were supposed to be originally extracted. On the other hand, $\beta \in [0, 1]$ doesn't take negative value. Therefore, even if negative features are extracted, the learning is advanced assuming that positive features were extracted. The gradient difference between ShakeDrop and other methods seems to be due to the accumulation of discrepancies by α and β.

What effect does the gradient change in ShakeDrop have? In the CNN where the recognition accuracy is high in the test data, the loss becomes small for any input and the gradient becomes small. In particular, it is thought that it becomes a flat in the neighborhood of the optimum parameters (Hochreiter & Schmidhuber, 1997), and the relationship between flatness and recognition accuracy has been experimentally shown Keskar et al. (2017). Therefore, as the parameters approaches the optimum parameters, the gradient becomes flat and the learning does not progress well. On the



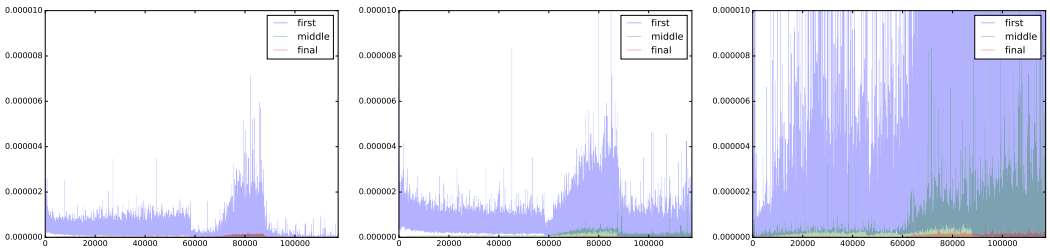
(a) Change of 300 epochs training loss. (b) Change of 1800 epochs training loss.

Figure 3: Change of training loss on PyramidNet, PyramidDrop and ShakeDrop ("Channel" level).



(a) PyramidNet (b) PyramidDrop (c) ShakeDrop

Figure 4: The average of the gradients on each iteration.



(a) PyramidNet (b) PyramidDrop (c) ShakeDrop

Figure 5: The variance of the gradients on each iteration.

other hand, ShakeDrop can learn even if the original gradient becomes small due to the pseudo gradient enlarging effect. Therefore, by searching for a flat state efficiently, it seems that ShakeDrop could have achieved recognition accuracy far superior to conventional methods. In particular, Jastrzebski et al. (2017) reported that ResNet tends to concentrate representation learning behavior in the first few layers, so it is highly probable that the disturbance of the gradient of "First" is involved in expression learning. However, there is also a skeptical view on the relationship between flatness and recognition accuracy Dinh et al. (2017), so it is necessary to verify further effect.

5 Conclusion

Based on PyramidDrop and Shake-Shake, we proposed a new stochastic regularization method ShakeDrop and experimentally examined its effect. As a result, as shown in the table 3, the pro-

posed method ShakeDrop achieved the state-of-the-art accuracy in CIFAR datasets. In particular, recognition accuracy was improved by 3% at the maximum from the state-of-the-art method in CIFAR-100. The future goal is to proceed with the verification of CIFAR-10, to proceed with verification on large scale conditions, and to conduct parameter verification aimed at further lowering the error rate.

REFERENCES

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proc. CVPR*, 2009.

Terrance DeVries and Graham W. Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. *arXiv preprint arXiv:1703.04933*, 2017.

Anuvabh Dutt, Denis Pellerin, and Georges Qunot. Coupled ensembles of neural networks. *arXiv preprint 1709.06053v1*, 2017.

Xavier Gastaldi. Shake-shake regularization. *arXiv preprint arXiv:1705.07485v2*, 2017.

Dongyoon Han, Jiwhan Kim, and Junmo Kim. Deep pyramidal residual networks. In *Proc. CVPR*, 2017a.

Dongyoon Han, Jiwhan Kim, and Junmo Kim. Deep pyramidal residual networks. *arXiv preprint arXiv:1610.02915v4*, 2017b.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proc. ICCV*, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. CVPR*, 2016.

Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural Computation*, 9(1):1–42, 1997.

Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Weinberger. Deep networks with stochastic depth. *arXiv preprint arXiv:1603.09382v3*, 2016.

Stanisaw Jastrzebski, Devansh Arpit, Nicolas Ballas, Vikas Verma, Tong Che, and Yoshua Bengio. Residual connections encourage iterative inference. *arXiv preprint arXiv:1710.04773*, 2017.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *Proc. ICLR*, 2017.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. NIPS*, 2012.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Yurii Nesterov. A method of solving a convex programming problem with convergence rate $o(1/k^2)$. *Soviet Mathematics Doklady*, 27:372–376, 1983.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, oct 1986. doi: 10.1038/323533a0.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proc. ICLR*, 2015.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proc. CVPR*, 2015.

Andreas Veit, Michael J Wilber, and Serge Belongie. Residual networks behave like ensembles of relatively shallow networks. *Advances in Neural Information Processing Systems 29*, 2016.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proc. CVPR*, 2017.

Yoshihiro Yamada, Masakazu Iwamura, and Koichi Kise. Deep pyramidal residual networks with separated stochastic depth. *arXiv preprint arXiv:1612.01230*, 2016.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *Proc. BMVC*, 2016.

Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. *arXiv preprint arXiv:1708.04896*, 2017.