A. Analysis on learned features

In Fig. 6 of the main paper, we visualized the learned features of 11-layer CNN and demonstrated that BC learning has the ability to impose a constraint on the feature distribution. Here, we show the results of more detailed analysis on the learned features. We used the same model as that used in Fig. 6 of the main paper.

A.1. Fisher's criterion

We calculated Fisher's criterion [8] for all combinations of two classes. Let $\{\mathbf{x}_n\}_{n\in C_i}$ and $\mathbf{m}_i$ be features of class C_i and the average of them ($\frac{1}{N_i}\Sigma_{n\in C_i}\mathbf{x}_n$), respectively. Here, Fisher's criterion between class C_1 and C_2 is defined as:

$$\frac{\mathbf{w}^\top \mathbf{S}_B \mathbf{w}}{\mathbf{w}^\top \mathbf{S}_W \mathbf{w}}, \tag{7}$$

where:

$$\begin{aligned} \mathbf{S}_B &= (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^\top, \\ \mathbf{S}_W &= \Sigma_{n\in C_1}(\mathbf{x}_n - \mathbf{m}_1)(\mathbf{x}_n - \mathbf{m}_1)^\top \\ &\quad + \Sigma_{n\in C_2}(\mathbf{x}_n - \mathbf{m}_2)(\mathbf{x}_n - \mathbf{m}_2)^\top, \\ \mathbf{w} &\propto \mathbf{S}_W^{-1}(\mathbf{m}_1 - \mathbf{m}_2). \end{aligned} \tag{8}$$

We show the mean Fisher's criterion in Table 5. We used the activations of the 10-th layer against training data. Fisher's criterion of the features learned with BC learning was larger than that of standard learning. This result shows that a discriminative feature space is learned with BC learning.

Table 5. Comparison of mean Fisher's criterion. BC learning indeed enlarges Fisher's criterion in the feature space.

Learning	Mean Fisher's criterion
Standard	1.76
BC (ours)	**1.97**

A.2. Activation of final layer

We investigated the activation of the final layer against training images. The results are shown in Fig. 7. The (i, j) element represents the mean activation of i-th neuron of the final layer (before the softmax) against the training images of class j. As shown in this figure, each neuron responds only to the corresponding class when using BC learning. The responses against other classes are almost the same level and most of them are negative values, whereas the responses against classes other than the corresponding class have a large variance and some of them are positive values when using standard learning. This result indicates that the features of each class learned with BC learning are distributed in the opposite side of the features of other classes. Such features can be easily separated. It is possible that BC learning indeed regularizes the positional relationship among the feature distributions.

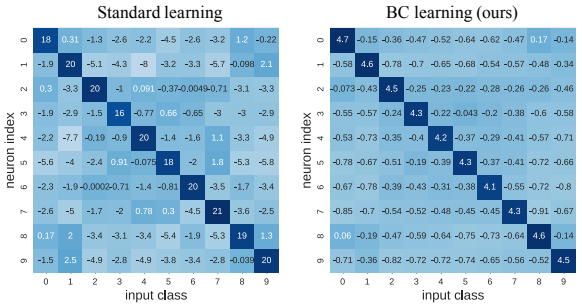

Figure 7. Activation of the final layer. The (i, j) element represents the mean activation of i-th neuron of the final layer (before the softmax) against class j.

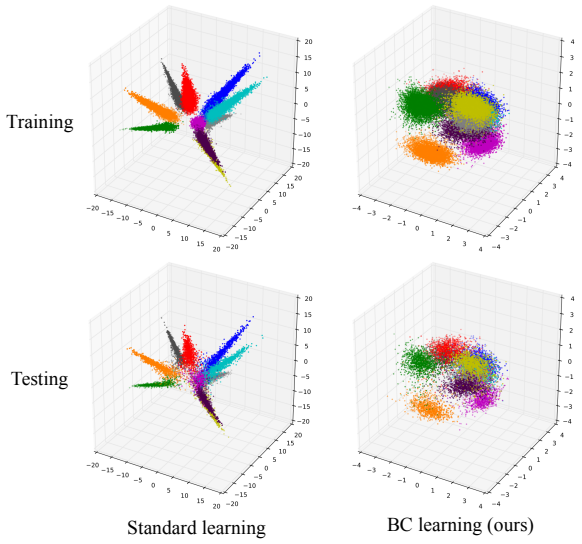

Figure 8. Feature distributions against training and testing data.

A.3. Training features *vs.* testing features

We compared the feature distributions against training and testing data. We visualized the activations of the 10-th layer using 3-D PCA. The results are shown in Fig. 8. The feature distributions of BC learning against training and testing data have similar shapes, whereas some testing examples are projected onto the points near the origin when using standard learning. This result indicates that the model learned with BC learning has a higher generalization ability.

B. Details of CIFAR experiments

B.1. Learning settings

We summarize the learning settings for CIFAR experiments in Table 6. Although most of them follow the original learning settings in [28, 13, 9], we slightly modified the learning settings for ResNet-29, ResNeXt-29 [28], and

Table 6. Learning settings for CIFAR experiments. The figures between brackets indicate the default learning settings.

Model	# of epochs	Initial LR	LR schedule	nGPU	mini-batch size
11-layer CNN	250	0.1	{100, 150, 200}	1	128
ResNet-29 [28]	**375** (300)	**0.0125** (0.1)	{**150**, **225**, **300**} ({150, 225})	**1** (8)	**16** (128)
ResNeXt-29 [28]	**375** (300)	0.1	{**150**, **225**, **300**} ({150, 225})	8	128
DenseNet [13]	**375** (300)	0.1	{**150**, **225**, **300**} ({150, 225})	4	64
Shake-Shake [9] (CIFAR-10)	1800	0.2	cosine	2	128
Shake-Shake [9] (CIFAR-100)	1800	0.025	cosine	2	32

DenseNet [13] in order to achieve a satisfactory performance. We trained the model by beginning with a learning rate of *Initial LR*, and then divided the learning rate by 10 at the epoch listed in *LR schedule*, except that a cosine learning rate scheduling was used for Shake-Shake. We then terminated training after *# of epochs* epochs.

B.2. Configuration of 11-layer CNN

We show the configuration of 11-layer CNN in Table 7. We applied ReLU activation for all hidden layers and batch normalization [14] to the output of all convolutional layers. We also applied 0.5 of dropout [24] to the output of fc4 and fc5. We used a weight initialization of [10] for all convolutional layers. We initialized the weights of each fully connected layer using the uniform distribution $U(-\sqrt{1/n}, \sqrt{1/n})$, where n is the input dimension of the layer. By using BC learning, we can achieve around 5.2% and 23.7% error rates on CIFAR-10 and CIFAR-100, respectively, despite the simple architecture.

Table 7. Configuration of 11-layer CNN.

Layer	ksize	stride	pad	# filters	Data shape
Input					(3, 32, 32)
conv1-1	3	1	1	64	
conv1-2	3	1	1	64	
pool1	2	2			(64, 16, 16)
conv2-1	3	1	1	128	
conv2-2	3	1	1	128	
pool2	2	2			(128, 8, 8)
conv3-1	3	1	1	256	
conv3-2	3	1	1	256	
conv3-3	3	1	1	256	
conv3-4	3	1	1	256	
pool3	2	2			(256, 4, 4)
fc4				1024	(1024,)
fc5				1024	(1024,)
fc6				# classes	(# classes,)