Supplementary material
A Comprehensive Overhaul of Feature Distillation

A. margin evaluation

We calculated margin of each channel and use margin ReLU with channel margin $m_{\mathcal{C}}$. The margin is the expectation of the negative value of the feature, which can be obtained directly during training or using a batch normalize layer. We explains how to obtain the margin value using a batch norm layer. For a channel $\mathcal{C}$ and the i-th element of teacher's feature F_t^i, the margin value of a channel $m_{\mathcal{C}}$ is set to an expected value over training images.

$$m_{\mathcal{C}} = E[F_t^i | F_t^i < 0, i \in \mathcal{C}]. \tag{1}$$

In general, we can't know the distribution of F_t^i, so expectation must be obtained through average operation over training process. However, when a batch-norm layer prior to ReLU, the batch-norm layer determines the distribution of feature F_t^i in a batch. Batch norm layer normalizes the feature for each channel to a gaussian distribution with a specific mean μ and variance σ. In other words,

$$F_t^i \sim \mathcal{N}(\mu, \sigma). \tag{2}$$

The value of mean and variance (μ, σ) of each channel correspond to the parameters (β, γ) of the batch-norm layer. So, it can be obtained by analyzing the teacher network. Using the distribution of F_t^i, we can directly calculate the margin value.

$$m_{\mathcal{C}} = \frac{1}{Z} \int_{-\infty}^{0} \frac{x}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx \tag{3}$$

The expectation can be obtained from integration using pdf of gaussian distribution, where the range is smaller than zero. The result of the integration can be expressed in simple form using the cdf function $\Phi(\cdot)$ of normal distribution.

$$m_{\mathcal{C}} = \mu - \frac{\sigma e^{-\mu^2/2\sigma^2}}{\sqrt{2\pi}\Phi(-\mu/\sigma)} \tag{4}$$

Using Eq. 4, the proposed method obtains channel-wise margin value without sampling and averaging on training process. In the experiment of the paper, if the ReLU is followed by batch normalize, the margin is obtained by using Eq. 4. Otherwise, the expectation is obtained from average operation on training process.

B. implementation details

Features for distillation are selected just before downsampling layers, which total three layers for CIFAR and four layers for ImageNet. In the loss function of our method, we sum the values in the entire layer rather than averaging them. When one moves from the top layer to the bottom layer, the total size of the feature is increased by twice the amount as spatial resolution increases. Therefore, the loss is divided in half accordingly. The parameter α is 10^{-3} for CIFAR, 10^{-4} for ImageNet and detection, and 10^{-5} for segmentation. For CIFAR, we used a batch size of 128 for (a) to (d) and a batch size of 64 for (e) and (f). Detection has an extra layer behind the backbone and all extra layers were also used for distillation. Detector were trained over a 120k iteration with a batch size of 32. The learning rate started at 10^{-3} and was multiplied by 0.1 at iteration 80k and 100k. In the case of segmentation, we use an additional distillation layer at the *atrous spatial pyramid pooling* and a layer just before output layer. Output stride was set to 16 and all dropout layers are not used for distillation. When using a pre-trained network, we initialized the student transform at the start of training. Initialization proceeded for 500 iteration for detection and 1 epoch for segmentation.

C. additional experiments

We measured the performance of other distillation methods at our preReLU position. We conducted this experiment in setting (c). As shown in the Table 1, the preReLU improves the performance of most algorithms.

Position	FitNets	AT	Jacobian	FT	AB	Proposed
Block	26.30	26.42	26.71	25.91	-	-
preReLU	26.22	26.45	26.27	25.11	26.02	24.08

Table 1. Performance of other distillation methods in preReLU.

We also measured performance of proposed method in a single-layer setting. As shown in the Table 2, single-layer version is not significantly different from the multi-layer version, which implies that our method outperforms the existing methods in any settings.

Setting	(a)	(b)	(c)	(d)	(e)	(f)
Multi-layer	20.89	21.98	24.08	24.44	17.80	18.89
Single-layer	20.90	22.03	24.14	24.78	18.17	18.99

Table 2. Comparison between single-layer and multi-layer implementation of proposed method.