Supplementary Material for “Neural Network Quantization with Scale-Adjusted Training”

S1 Detailed Analysis of Clamping Operation

Here we want to give some detailed analysis of the clamping operation to understand the over-fitting phenomenon in Fig. 2b. Our analysis is based on previous work [1], which states that for over-parameterized models, the training error decays exponentially to zero while the generalization error increases monotonically with weight variance. For simplicity, we summarize their results as follows

$$E_g \approx \begin{cases} (1-\alpha)\left(\sigma_w^2+(\sigma_w^0)^2\right)+\sigma_\varepsilon^2, & \alpha<1 \\ \sqrt{\sigma_w^2+(\sigma_w^0)}\sigma_\varepsilon+\sigma_\varepsilon^2, & \alpha=1 \\ \sigma_\varepsilon^2\frac{\alpha}{\alpha-1}, & \alpha>1 \end{cases} \tag{S1}$$

$$\langle E_t(t)\rangle = \frac{1}{\alpha}\int_{\lambda\in\mathbb{R}} f(\lambda)\Big(\lambda\left(\sigma_w^2+(\sigma_w^0)^2\right)+\sigma_\varepsilon^2\Big)\mathrm{e}^{-2\lambda\eta t}\,\mathrm{d}\lambda+\left(1-\frac{1}{\alpha}\right)\sigma_\varepsilon^2\mathbb{1}[\alpha>1] \tag{S2}$$

where E_g denotes the expected generalization error, and $\langle E_t(t)\rangle$ is the average training error at time t. α is the ratio between the number of data and the number of model parameters. $f(\lambda)$ is some distribution function of λ, and η is the learning rate. σ_ε^2, σ_w^2 and $(\sigma_w^0)^2$ are variances of input noise, trained weights and initial weights, respectively.

Now we want to see the effect of weight clamping on the two errors. As already discussed in 3.1.1, the clamped weights

$$\widehat{W}_{ij} = \frac{\tanh(W_{ij})}{\max\limits_{r,s}|\tanh(W_{rs})|} \tag{S3}$$

has a larger variance, so for over-parameterized models where $\alpha<1$, weight clamping hampers the generalization error, and thus the red dashed line is above the black dashed line in Fig. 2b.

For training error, at first glance, larger weight variance should also results in worse performance, as indicated by S2. This seems to contradict the training curve in Fig. 2a. To understand this, we look into the weight updating procedure. During training, the original weights are updated as

$$W_{ij}\to W_{ij}-\eta\frac{\partial\mathcal{L}}{\partial W_{ij}} \tag{S4}$$

This will update the clamped weights as

$$\widehat{W}_{ij}\to\widehat{W}_{ij}-\eta\frac{\partial\mathcal{L}}{\partial W_{ij}}\frac{\partial\widehat{W}_{ij}}{\partial W_{ij}} \tag{S5a}$$

$$=\widehat{W}_{ij}-\eta\cdot\left(\frac{\partial\widehat{W}_{ij}}{\partial W_{ij}}\right)^2\frac{\partial\mathcal{L}}{\partial\widehat{W}_{ij}} \tag{S5b}$$

$$:=\widehat{W}_{ij}-\widehat{\eta}\frac{\partial\mathcal{L}}{\partial\widehat{W}_{ij}} \tag{S5c}$$

In other words, the learning rate experienced by the clamped weights is given by

$$\widehat{\eta} := \eta \cdot \left(\frac{\partial \widehat{W}_{ij}}{\partial W_{ij}} \right)^2 \tag{S6}$$

where the ratio can be estimated as

$$\frac{\partial \widehat{W}_{ij}}{\partial W_{ij}} = \frac{\operatorname{sech}^2(W_{ij})}{\max|\tanh(W_{kl})|} \sim \frac{1}{\max|W_{kl}|} \sim \sqrt{\widehat{n}} \tag{S7}$$

Here we have used the fact that the weights are nearly Gaussian distributed with the variance proportional to the reciprocal of the number of neurons ($\widehat{n} \gg 1$), as typically the training procedure will only have small perturbation on the weights without significant impact on their distribution.

Thus, besides increasing the weight variance, clamping will also increase the learning rate twofold, and as the learning rate impacts the training error exponentially, this effect dominates, resulting in a more rapidly descending training curve (the red solid line).

For the scaled weights

$$W^*_{ij} = \frac{1}{\sqrt{\widehat{n}\mathbb{VAR}[\widehat{W}_{rs}]}} \widehat{W}_{ij} \tag{S8}$$

the weight scale is smaller than the clamped weights, so the generalization error is reduced (the blue dashed line). However, this scale will also eliminate the enlargement in the learning rate, giving a slower descending learning curve than the clamped case (the blue solid line). We also notice that the descending is slower than the vanilla case. This might be due to the fact that the tanh operation results in gradient vanishing for large weights, and thus limits the value range for the final weights.

S2 Comparison of Rescaling Method

In this section, we will compare the constant rescaling methods proposed in the paper and another method named rescaling with standard deviation. It standardizes the effective weights and then rescales them with the standard deviation of the original weights as in Eq. (S9), where $\mathbb{VAR}[\cdot]$ is the sample variance of the elements in the weight matrix, calculated by averaging the square of the weights.

$$W^*_{ij} = \sqrt{\frac{\mathbb{VAR}[W_{rs}]}{\mathbb{VAR}[\widehat{W}_{rs}]}} \widehat{W}_{ij} \tag{S9}$$

We train MobileNet V2 on ImageNet with weight clamping and rescale the last fully-connected layer using the two methods, and plot their learning curves as shown in Fig. S1. We find that the two methods give similar results.

S3 Bitwidths of the First and Last Layers

Here we study the impact of quantization levels of weights in the first and last layers. Using MobileNet V1, we compare the two settings of quantizing these two layers to a fixed 8 bits

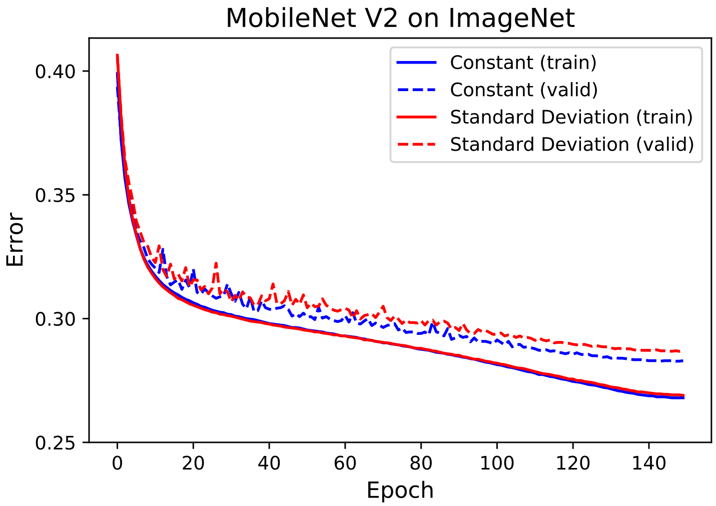

Figure S1: Comparison of constant rescaling and rescaling with standard deviation.

or to the same bit-width as other layers. As shown in Table S1, we find that the accuracy reduction is negligible for quantization levels higher than 4bits. Note that as mentioned in the paper, the input image is always encoded using unsigned 8bit integer (uint8), and the input to the last layer is quantized with the same precision as input to other layers.

Table S1: Impact of precisions of the first and the last layers on MobileNet V1.

Bitwidths of Internal Layers	8bits Both Layers		Uniform Quantization	
	Acc.-1	Acc.-5	Acc.-1	Acc.-5
4bits	71.3	89.9	71.2	89.8
5bits	71.9	90.3	72.1	90.3
6bits	72.3	90.4	72.5	90.5