Supplementary Material: Centered Weight Normalization in Accelerating Training of Deep Neural Networks

Lei Huang[†], Xianglong Liu[†], Yang Liu[†], Bo Lang[†], Dacheng Tao [‡]
[†]State Key Laboratory of Software Development Environment, Beihang University, P.R.China
[‡]UBTECH Sydney AI Centre, School of IT, FEIT, The University of Sydney, Australia
{huanglei, xlliu, blonster, langbo}@nlsde.buaa.edu.cn, dacheng.tao@sydney.edu.au

1. Proof of proposition

Proposition 1. *Let* $z = \mathbf{w}^T\mathbf{h}$*, where* $\mathbf{w}^T\mathbf{1} = 0$ *and* $\|\mathbf{w}\| = 1$*. Assume* $\mathbf{h}$ *has Gaussian distribution with the mean:* $\mathbb{E}_{\mathbf{h}}[\mathbf{h}] = \mu\mathbf{1}$*, and covariance matrix:* $cov(\mathbf{h}) = \sigma^2\mathbf{I}$*, where* $\mu \in \mathbb{R}$ *and* $\sigma^2 \in \mathbb{R}$*. We have* $\mathbb{E}_z[z] = 0$*,* $var(z) = \sigma^2$*.*

Proof. It's easy to calculate:

$$\mathbb{E}_z[z] \quad = \mathbf{w}^T\mathbb{E}_{\mathbf{h}}[\mathbf{h}] = \mathbf{w}^T\mu\mathbf{1} = 0 \tag{1}$$

The variance of z is given by

$$\begin{aligned} var(z) &= \mathbb{E}_z[z - \mathbb{E}_z[z]]^2 \\ &= \mathbb{E}_{\mathbf{h}}[\mathbf{w}^T(\mathbf{h} - \mathbb{E}_{\mathbf{h}}[\mathbf{h}])]^2 \\ &= \mathbb{E}_{\mathbf{h}}[\mathbf{w}^T(\mathbf{h} - \mathbb{E}_{\mathbf{h}}[\mathbf{h}])] \cdot \mathbb{E}_{\mathbf{h}}[\mathbf{w}^T(\mathbf{h} - \mathbb{E}_{\mathbf{h}}[\mathbf{h}])]^T \\ &= \mathbf{w}^T\mathbb{E}_{\mathbf{h}}[(\mathbf{h} - \mathbb{E}_{\mathbf{h}}[\mathbf{h}])] \cdot \mathbb{E}_{\mathbf{h}}[(\mathbf{h} - \mathbb{E}_{\mathbf{h}}[\mathbf{h}])]^T\mathbf{w} \\ &= \mathbf{w}^T cov(\mathbf{h})\mathbf{w} \\ &= \mathbf{w}^T\sigma^2\mathbf{I}\mathbf{w} \\ &= \sigma^2\mathbf{w}^T\mathbf{w} = \sigma^2 \end{aligned} \tag{2}$$

□

Proposition 2. *Regarding to the proxy parameter* $\mathbf{v}$*, centered weight normalization makes that the gradient* $\frac{\partial\mathcal{L}}{\partial\mathbf{v}}$ *has following properties: (1)* zero-mean*,* i.e. $\frac{\partial\mathcal{L}}{\partial\mathbf{v}} \cdot \mathbf{1} = 0$*; (2) orthogonal to the parameters* $\mathbf{w}$*,* i.e. $\frac{\partial\mathcal{L}}{\partial\mathbf{v}} \cdot \mathbf{w} = 0$

Proof. As introduced in the paper, the gradient $\frac{\partial\mathcal{L}}{\partial\mathbf{v}}$ is calculated as follows:

$$\frac{\partial\mathcal{L}}{\partial\mathbf{v}} = \frac{1}{\|\hat{\mathbf{v}}\|}[\frac{\partial\mathcal{L}}{\partial\mathbf{w}} - (\frac{\partial\mathcal{L}}{\partial\mathbf{w}}\mathbf{w})\mathbf{w}^T - \frac{1}{d}(\frac{\partial\mathcal{L}}{\partial\mathbf{w}}\mathbf{1})\mathbf{1}^T] \tag{3}$$

where $\hat{\mathbf{v}} = \mathbf{v} - \frac{1}{d}\mathbf{1}(\mathbf{1}^T\mathbf{v})$ is the centered auxiliary parameter. Besides, the centered weight normalization method guarantees that: (1) $\mathbf{w}^T\mathbf{1} = 0$; (2) $\|\mathbf{w}\| = 1$. Based on 3, we have

$$\begin{aligned} \frac{\partial\mathcal{L}}{\partial\mathbf{v}} \cdot \mathbf{1} &= \frac{1}{\|\hat{\mathbf{v}}\|}[\frac{\partial\mathcal{L}}{\partial\mathbf{w}} - (\frac{\partial\mathcal{L}}{\partial\mathbf{w}}\mathbf{w})\mathbf{w}^T - \frac{1}{d}(\frac{\partial\mathcal{L}}{\partial\mathbf{w}}\mathbf{1})\mathbf{1}^T] \cdot \mathbf{1} \\ &= \frac{1}{\|\hat{\mathbf{v}}\|}[\frac{\partial\mathcal{L}}{\partial\mathbf{w}}\mathbf{1} - (\frac{\partial\mathcal{L}}{\partial\mathbf{w}}\mathbf{w})\mathbf{w}^T\mathbf{1} - \frac{1}{d}(\frac{\partial\mathcal{L}}{\partial\mathbf{w}}\mathbf{1})d] \\ &= \frac{1}{\|\hat{\mathbf{v}}\|}[-(\frac{\partial\mathcal{L}}{\partial\mathbf{w}}\mathbf{w})(\mathbf{w}^T\mathbf{1})] = 0 \end{aligned} \tag{4}$$

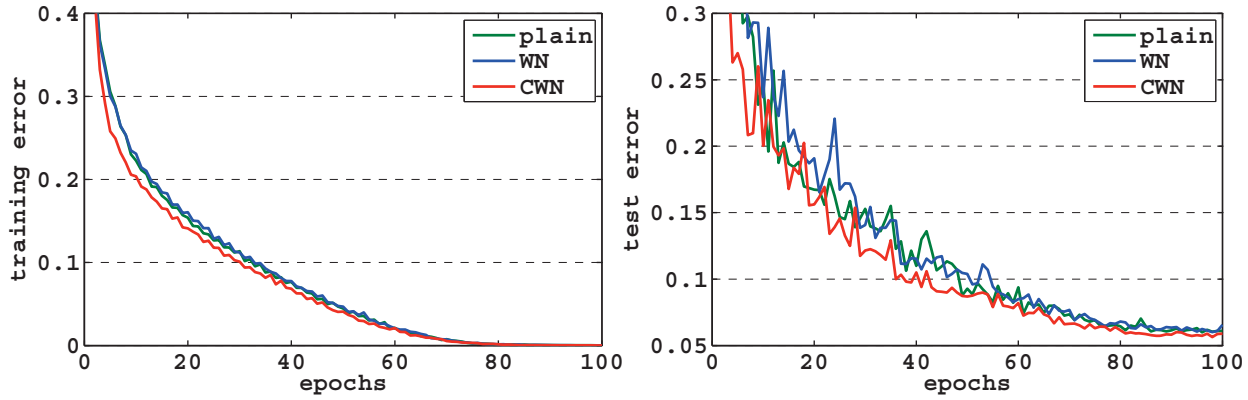
Similarly, we have:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \mathbf{v}} \cdot \mathbf{w} &= \frac{1}{\|\hat{\mathbf{v}}\|}[\frac{\partial \mathcal{L}}{\partial \mathbf{w}} - (\frac{\partial \mathcal{L}}{\partial \mathbf{w}}\mathbf{w})\mathbf{w}^T - \frac{1}{d}(\frac{\partial \mathcal{L}}{\partial \mathbf{w}}\mathbf{1})\mathbf{1}^T] \cdot \mathbf{w} \\ &= \frac{1}{\|\hat{\mathbf{v}}\|}[\frac{\partial \mathcal{L}}{\partial \mathbf{w}} \cdot \mathbf{w} - (\frac{\partial \mathcal{L}}{\partial \mathbf{w}}\mathbf{w})(\mathbf{w}^T \cdot \mathbf{w}) - \frac{1}{d}(\frac{\partial \mathcal{L}}{\partial \mathbf{w}}\mathbf{1})(\mathbf{1}^T \cdot \mathbf{w})] \\ &= \frac{1}{\|\hat{\mathbf{v}}\|}[\frac{\partial \mathcal{L}}{\partial \mathbf{w}} \cdot \mathbf{w} - (\frac{\partial \mathcal{L}}{\partial \mathbf{w}}\mathbf{w})] = 0 \end{aligned} \tag{5}$$

□

2. Experimental results on GoogLeNet over CIFAR-10 dataset

The setup of this experiment is described as in the experiment of GoogLeNet on CIFAR in the paper. The results are shown in Figure 1, from which we can find that CWN also achieves marginal speedup compared to WN and 'plain'. Moreover, CWN also obtains the lowest test error of 5.64%, compared to 'plain' of 6.00% and WN of 6.01%. We conjecture that centering the weight effectively regularizes the neural networks, therefore can achieve improvement in terms of the test performance, while it also improves the conditioning of the optimization problem and obtains net gain in terms of training performance.



(a) training error (b) test error

Figure 1. Performances comparison on GoogLeNet architecture over CIFAR-10 dataset.

3. Details of neural network architectures in the experiments

Figures 2, 3 and 4 show the details of the used VGG-A [2], GoogLeNet [3] and 56 layers residual network [1] in the paper respectively. More details please see the available code at: https://github.com/huangleiBuaa/CenteredWN.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2

[2] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 2

[3] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014. 2

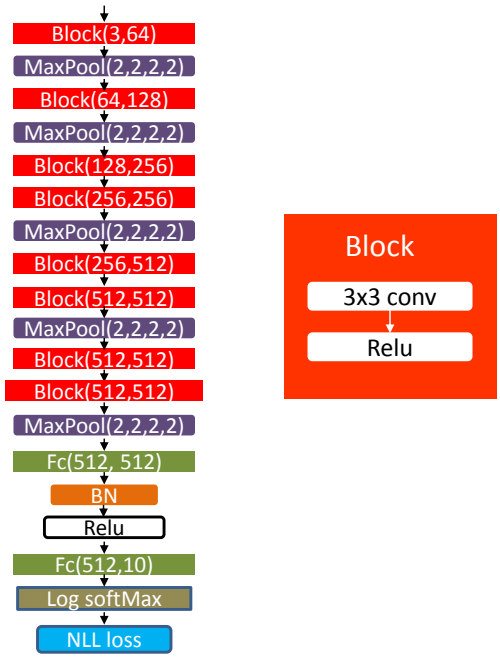


Figure 2. The VGG-A architecture in the experiments. For '*Block(d,n)*', d and n are the sizes of feature maps with respect to the input and output. '3×3 conv' indicates using 3×3 convolutional filter. '*Fc(d, n)*' indicates that the fully connected linear mapping has the input dimension of d and output dimension of n. The *MaxPool(2,2,2,2)* is in 2×2 regions by step size of 2×2.

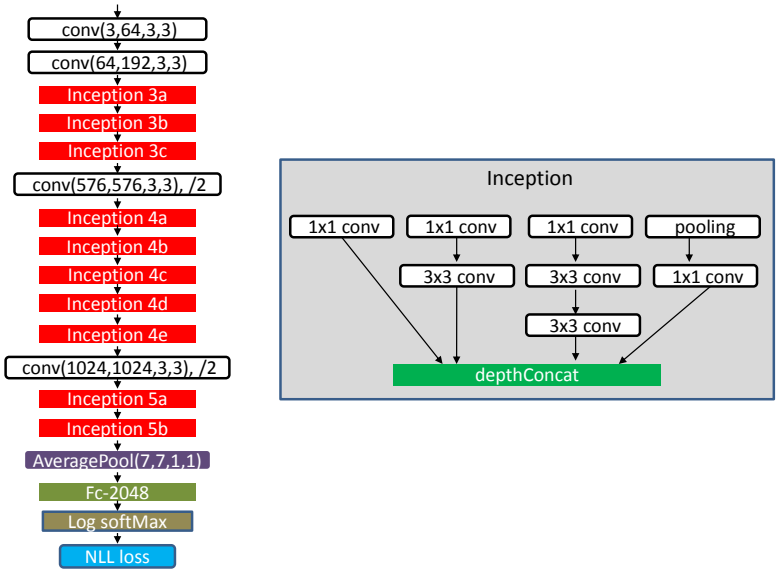


Figure 3. The GoogLeNet architecture. In the left side, we show the overall architecture. Note that 'conv(3,64,3,3)' indicates using 3×3 filter with that the dimensions of input and output feature maps are 3 and 64 respectively. All convolutional layer is followed by batch normalization and Relu nonlinearity. The right side shows the general Inception module.

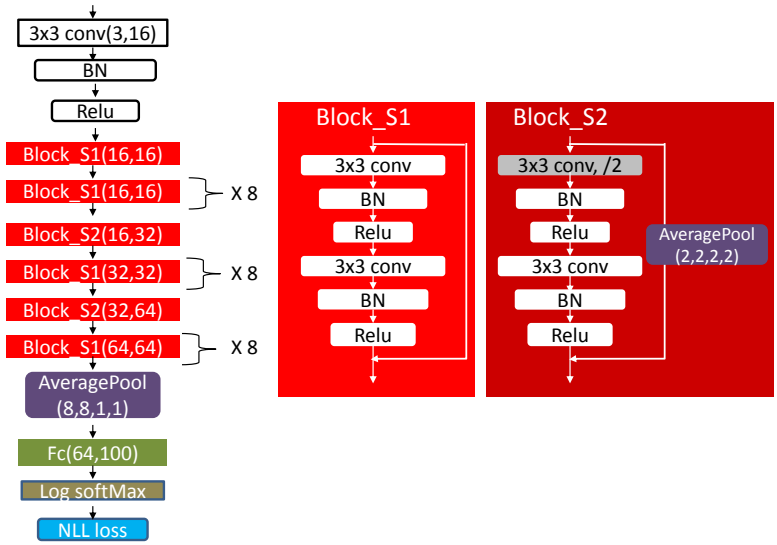


Figure 4. The 56-layers residual neural network. '$\times n$' means there are n stacked blocks.