Supplemental Material of Truncating Wide Networks using Binary Tree Architectures

Yan Zhang[†], Mete Ozay[†], Shuohao Li[†‡] and Takayuki Okatani[†§]

[†]Tohoku University, [‡]National University of Defense Technology, [§]RIKEN Center for AIP

{zhang,mozay,lishuohao,okatani}@vision.is.tohoku.ac.jp

In the supplemental material, we provide the proofs of Corollary 3.2 and Proposition 3.3 in Section 3.3 in the main paper. The detailed training and testing setting used in the experiments are also provided. We also show training errors of proposed BitNets used in our experiments.

A. Proofs

Proof of Corollary 3.2. The total depth of the given network is LK. Thus, the parameter size is $\mathcal{O}\big(LKD^2\big)$. Also, according to Corollary 5 of [6], the maximal number of linear region of functions that can be computed by the given network in an n-dimensional ($D > n$) input space is lower bounded by $\mathcal{O}\big((\frac{D}{n})^{n(KL-1)}D^n\big)$, which can be simplified by a looser bound $\mathcal{O}\big((\frac{D}{n})^{nKL}\big)$ and resulting in our corollary.

□

Proof of Proposition 3.3. According to the definition of BitBlock, the width of k^{th} layer in a BitBlock is $\frac{D}{2^{(k-1)}}$. Thus, the parameter size of the given BitBlock can be computed by,

$$\begin{aligned}&\sum_{k=1}^{K}\Big(\frac{D}{2^{(k-1)}}\Big)^2\\ =&\frac{4}{3}\big(1-\frac{1}{4^K}\big)D^2\end{aligned}.$$

Thus, the parameter size of L stacked BitBlocks is $\mathcal{O}\Big(\frac{4}{3}L\big(1-\frac{1}{4^K}\big)D^2\Big)$.

To analyze the expressive capacity, we approximate a BitBlock as a ConvenBlock with depth K and width $\frac{D}{2^k}$ at each k^{th} layer. According to Theorem 4 of [6], the maximal number of linear regions of functions that can be computed by a given BitBlock in an n-dimensional ($D \geq 2^K n$) input space is lower bounded by

$$\begin{aligned}&\prod_{k=1}^{K}\Big(\frac{D}{2^k n}\Big)^n\\ =&\Big(\frac{D}{n}\Big)^{nK}2^{\frac{-n(1+K)K}{2}}\\ >&\Big(\frac{D}{n}\Big)^{nK}2^{-nKK}\\ =&\Big(\frac{D}{2^K n}\Big)^{nK}.\end{aligned}$$

As a result, with L BitBlocks, the maximal number of linear region is bounded by $\mathcal{O}\Big(\big(\frac{D}{2^K n}\big)^{nKL}\Big)$. □

B. Training and Testing Setting

For Cifar-10 and Cifar-100 dataset, we use a common data augmentation method used in previous works [1, 5, 2, 10]. More precisely, 4 pixels with zero values are padded on each side of the original image to make a 40×40 image, from which a 32×32 patch is randomly cropped and randomly flipped horizontally. For testing, the original 32×32 image is used. Batch size is 128 that is split on two GPUs. The initial learning rate is 0.1, and is reduced by 0.2 on the 60^{th}, 120^{th} and 160^{th} epoch. The training is finished at the 200^{th} epoch.

For ILSVRC12 task, data augmentation and image preprocessing methods used during training are as follows. Each image is cropped by scale and aspect ratio augmentation method [9], and then resized to 224×224. A random horizontal flip is also applied. The input images are mean subtracted and variance normalized on each RGB channel. The color distortion methods proposed in [4] and [3] are both used. For validation, the image is resized such that its shorter side is 256, and a center crop of 224×224 are used to test. Batch size is 256 split on 8 GPUs. The initial learning rate is 0.1 and is reduced by 10^{-1} at each 30 epoch. The training is finished at the 90^{th} epoch. Following [2], stochastic gradient descent (SGD) with momentum 0.9 is used as our optimizer and the weight decay is set as

Model	Param.	Error
Wide ResNet (d=4,k=2,n=6)	8.9M	0.018
BitNet (d=4,k=3,n=4)	3.7M	0.018
BitNet (d=4,k=4,n=3)	2.7M	0.022
BitNet (d=4,k=2,n=6)	5.4M	0.026
BitNet (d=4,k=6,n=2)	1.7M	0.028
Wide ResNet (d=10,k=2,n=2)	17.1M	0.018
BitNet (d=10,k=2,n=2)	9.6M	0.018
BitNet (d=10,k=4,n=1)	3.9M	0.020
Wide ResNet (d=10,k=2,n=3)	26.8M	0.018
BitNet (d=10,k=2,n=3)	15.6M	0.018
BitNet (d=10,k=3,n=2)	10.2M	0.018
Wide ResNet (d=12,k=2,n=4)	52.5M	0.018
BitNet (d=12,k=2,n=4)	31.2M	0.018
BitNet (d=12,k=4,n=2)	14.9M	0.018

Table 1. Cifar-100 training error (%) of the BitNets and Wide ResNets used in our experiments.

0.0001. Batch Normalization is used in every convolutional layer before ReLU. We didn't use dropout [8] in any BitNet.

C. Training Errors

Training errors of BitNets used in our experiments for Cifar-100 classification task are given in Table 1. As shown in [7, 6], the expressive capacity reflects the complexity of class decision boundary computable by a DNN. We use training errors to quantify the expressive capacity of a DNN. As we can see from the table, the training errors of BitNets are close to that of the Wide ResNets, which indicates that by using considerably less number of parameters, the proposed BitNets can obtain expressive capacity similar to that of the Wide ResNets.

References

[1] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. C. Courville, and Y. Bengio. Maxout networks. In *ICML*, 2013.

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

[3] A. G. Howard. Some improvements on deep convolutional neural network based image classification. In *ICLR*, 2014.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[5] M. Lin, Q. Chen, and S. Yan. Network in network. In *ICLR*, 2014.

[6] G. F. Montufar, R. Pascanu, K. Cho, and Y. Bengio. On the number of linear regions of deep neural networks. In *NIPS*, 2014.

[7] R. Pascanu, G. Montúfar, and Y. Bengio. On the number of inference regions of deep feed forward networks with piecewise linear activations. In *ICLR*, 2014.

[8] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 15:1929–1958, 2014.

[9] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.

[10] S. Zagoruyko and N. Komodakis. Wide residual networks. In *BMVC*, 2016.