Bias Loss for Mobile Neural Networks
Supplimentary Material

Lusine Abrahamyan[1,3]
lusine.abrahamyan@vub.be

Valentin Ziatchin[2]
valentin.ziatchin@picsart.com

Yiming Chen[1,3]
cyiming@etrovub.be

Nikos Deligiannis[1,3]
ndeligia@etrovub.be

[1]ETRO Department, Vrije Universiteit Brussel (VUB), Pleinlaan 2, B-1050 Brussels, Belgium
[2]PicsArt Inc., San Francisco, USA
[3]imec, Kapeldreef 75, B-3001 Leuven, Belgium

1. Training Large CNNs with the Bias Loss

With the purpose to empirically evaluate the relation between the number of parameters of the model and the impact that bias loss can have on the model's performance, we conduct experiments on various models with number of parameters in the range of [1M, 27M]. Specifically, we train the ShuffleNetV2 $0.5\times$ [5], ResNet18 [1], EfficientNet-B4 [7], Inception v3 [6] and DenseNet ($k = 12$), DenseNet ($k = 24$) [2] models with the bias loss and the cross-entropy loss. The trainings are performed on the Tiny ImageNet [4] dataset, which consists of $100,000$ training and $10,000$ validation images from 200 classes. The size of the images is $64 \times 64 \times 3$ pixels. We use a batch size of 128, a learning rate 0.01, which decays by 0.1 every 30 epochs, and train the networks for 90 epochs with the SGD optimizer. Regarding the parameters of the bias loss, we use $\alpha = 0.3$ and $\beta = 0.3$. The results, presented in Table 1, suggest that for the models with a small number of parameters, the bias loss improves the classification accuracy by approximately 1% (e.g. Resnet18 - 1.1%, ShuffleNetV2 $0.5\times$ - 0.9%); however, when the number of the model's parameters increases, the impact of the bias loss starts to become negative. The degradation in the performance is related to that, for large models, even data points with relatively low variance contain enough unique descriptive features. Hence, the problem of random predictions, which hampers the performance of compact models, does not appear in large models. In this case, by performing down-weighting, we limit the number of data points that can help the optimizer to converge to good minima.

Table 1. The accuracy of various CNN models trained on Tiny ImageNet with the bias loss and cross-entropy.

Model	Parameters	Top-1 (%) CE loss	Top-1 (%) Bias loss
ShuffleNetV2 $0.5\times$	1.4M	51.9	**52.8 (+0.9)**
DenseNet ($k = 12$)	7M	57.2	**58.2 (+1.0)**
ResNet 18	11M	57.5	**58.6 (+1.1)**
EfficientNet-B4	19M	**57.7**	56.9 (-0.8)
Inception v3	24M	**65.4**	64.3 (-1.1)
DenseNet ($k = 24$)	27.2M	**67.5**	66.6 (-0.9)

2. Analysis of the Bias Loss

In order to empirically validate the advantage of our approach to up-weight data points with high variance within a training, we conduct an experimental comparison between the current formulation of the bias loss and its modifications.

First, we examine the impact of low variance samples on the model's performance. We set up experiments where we give higher weight to data points with lower variance and vice-versa (see Fig. 1). Specifically, we design the Inverse Bias Loss where we modify our bias function as follows:

$$z(v_i) = \frac{1}{\exp(v_i * \alpha) - \beta}, \tag{1}$$

where v_i is a scaled variance of the ith data point.

We train the models on the Cifar100 [3] dataset using the same setup as described in Section 5.2 of the main manuscript; namely, we use an SGD optimizer with a momentum equal to 0.9 and a weight decay of $5e-4$. The initial learning rate is set to $1e-1$ and then decays at the epochs 60, 120, 160 with a rate of 0.2. For data augmentation, the images are randomly flipped horizontally and rotated between the angles $[-15, 15]$. Regarding the parameters of the bias loss, we use $\alpha = 0.3$ and $\beta = 0.3$. The

Table 2. The accuracy of various compact CNN models trained on CIFAR-100 with cross-entropy and variations of the bias loss.

Model	Parameters	Top-1 (%) CE loss	Top-1 (%) Inv. Bias loss	Top-1 (%) Mean Bias loss	Top-1 (%) Bias loss
ShuffleNetV2 $0.5\times$	1.4M	69.5	68.9	70.2	**71**
MobileNetV2 $0.75\times$	2.6M	68	67.6	68.1	**68.6**
NASNet-A ($N = 4$)	5.3M	77.2	76.8	77.8	**78**
SqueezeNet	1.25M	69.4	69	69.5	**70.4**
DenseNet ($k = 12$)	7M	78.9	78.1	78.9	**79.9**

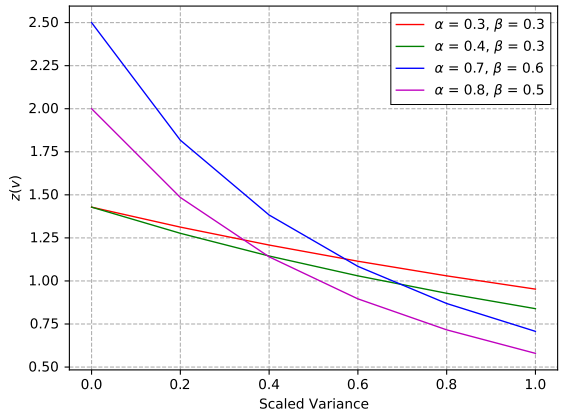

Figure 1. Inverse bias function where data points with lower variance gets higher weight.

results, presented in Table 2, show that concentrating the training on the set of data points with lower variance (i.e., giving them higher weight) can harm the model's final accuracy. In this scenario, the optimizer will give more attention to the random predictions (i.e., predictions made in the absence of enough unique descriptors), which will cause the model to converge to poor minima.

Next, we examine the advantage of using the scaled variance as the metric of diversity of the extracted features in the bias loss. We conduct an experimental comparison between the version of the bias loss function with the scaled variance and its modification where the scaled variance is replaced with the scaled mean; the latter is expressed using the following formulation:

$$z(\mu_i) = \exp(\mu_i * \alpha) - \beta, \tag{2}$$

where μ_i is the scaled mean of the ith data point. This comparison is motivated by the fact that for compact CNNs, the mean and variance of the activations in the feature maps are correlated (see Section 5.5 of the main manuscript). The goal is to assess whether the usage of the mean as a metric for re-weighting the data points can help obtaining better model accuracy. The results presented in Table 2 suggest that the replacement of the variance with the mean value can also boost the performance, albeit significantly less. While a higher mean value can indicate the presence of a large number of non-zero activations (hence, a large amount of extracted features), it cannot describe their diversity, like variance does. We have also attempted to combine both the variance and the mean in the bias loss function, that is, $z_i = z(\mu_i) * z(v_i)$. Our experimentation, however, showed that no additional gain (over that obtained only with the variance) can be achieved when both metrics are used in the process of data points re-weighting.

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 1

[2] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4700–4708, 2017. 1

[3] A. S. Krizhevsky. Learning multiple layers of features from tiny images. *Technical Report*, 2009. 1

[4] Y. Le and X. Yang. Tiny imagenet visual recognition challenge. 2015. 1

[5] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 116–131, 2018. 1

[6] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016. 1

[7] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114, 2019. 1