ON ROLE AND LOCATION OF NORMALIZATION BEFORE MODEL-BASED DATA AUGMENTATION IN RESIDUAL BLOCKS FOR CLASSIFICATION TASKS

Che-Wei Huang, Shrikanth Narayanan

University of Southern California, Los Angeles, CA 90089
cheweihu@usc.edu, shri@sipi.usc.edu

ABSTRACT

Regularization is crucial to the success of many practical deep learning models, in particular in frequent scenarios where there are only a few to a moderate number of accessible training samples. In addition to weight decay, noise injection and dropout, regularization based on multi-branch architectures, such as Shake-Shake regularization, has been proven successful in many applications and attracted more and more attention. However, beyond model-based representation augmentation, it is unclear how Shake-Shake regularization helps to provide further improvement on classification tasks, let alone the baffling interaction between batch normalization and shaking. In this work, we present our investigation on Shake-Shake regularization. One of our findings illustrates the phenomenon that batch normalization in residual blocks is indispensable when shaking is applied to model branches, along with which we also empirically demonstrate the most effective location to place a batch normalization layer in a shaking regularized residual block. Based on these findings, we believe our work is beneficial to future studies on the research topic of refining control for model-based representation augmentation.

Index Terms— Shake-Shake Regularization, Adversarial Training, Discriminative Feature Learning, Control for Representation Augmentation

1. INTRODUCTION

Deep convolutional neural networks have been successfully applied to several pattern recognition tasks such as image recognition [1], machine translation [2] and speech emotion recognition [3]. Currently, to successfully train a deep neural network, one needs either a sufficient number of training samples to implicitly regularize the learning process, or employ techniques like weight decay and dropout [4] and their variants to explicitly keep the model from over-fitting.

However, since the introduction of batch normalization [5], the gains obtained by using dropout for regularization have decreased [5, 6, 7]. A recent work dedicated to study the disharmony between dropout and batch normalization [8] suggests that dropout introduces a variance shift between training and testing, which cripples any following batch normalization layers and severely limits the application of successful architectures such as ResNet/ResNeXt or the application of dropout to the top-most fully connected layers. Yet, multi-branch architectures have emerged as a promising alternative for regularizing convolutional layers.

Regularization techniques based on multi-branch architectures such as Shake-Shake [9] and ShakeDrop [10] have delivered impressive performances on standard image datasets such as the CIFAR-10 [11] dataset. In a clever way, both of them utilize multiple branches to learn different aspects of the relevant information and then a summation in the end follows for information alignment among branches. Also, both Shake-Shake and ShakeDrop regularizations emphasize the important interaction between batch normalization and shaking. In our previous work on acoustic sub-band shaking [12] and stochastic Shake-Shake regularization [13] for affective computing from speech, we found that in a fully pre-activation architecture without a batch normalization layer right before shaking, the shaking mechanism contributes much more to constraining the learning process than to boosting the generalization power. All these findings indicated there is a close interaction between shaking and batch normalization. However, these studies do not give an explanation for the crucial role and location of batch normalization in a shaking regularized architecture, other than a brief discussion of the strength of shaking.

In this work, we study the Shake-Shake regularized ResNeXt for classification tasks to acquire a better understanding of the reported observations. Specifically, we investigate and come up with an explanation, beyond model-based representation augmentation, for the ability of shaking regularization to improve classification tasks and for its crucial interaction with batch normalization. In order to achieve this goal, we conduct two sets of ablation studies on MNIST [14] and CIFAR-10 datasets, respectively, with different configurations of residual blocks. The first set of experiments on the MNIST dataset aims to clarify the role of batch normalization in shaking regularized residual blocks; the second set of experiments on the CIFAR-10 dataset measures the effectiveness of batch normalization with respect to it proximity to the shaking layer.

2. SHAKE-SHAKE REGULARIZATION AND DISCRIMINATIVE FEATURE LEARNING

2.1. Shake-Shake Regularization

Shake-Shake regularization is a recently proposed technique to regularize training of deep convolutional neural networks

for image recognition tasks. This regularization technique based on multi-branch architectures promotes stochastic mixtures of forward and backward propagations from network branches in order to create a flow of model-based adversarial learning samples/gradients during the training phase. An overview of a 3-branch Shake-Shake regularized ResNeXt is depicted in Fig. 1. Shake-Shake regularization adds to the aggregate of the output of each branch an additional layer, called the shaking layer, to randomly generate adversarial flows in the following way:

$$\text{ResBlock}^N(\mathbf{X}) = \mathbf{X} + \sum_{n=1}^{N} \text{Shaking}\left(\{\mathbf{B}_n(\mathbf{X})\}_{n=1}^{N}\right)$$

where in the forward propagation for $\mathbf{a} = [\alpha_1, \cdots, \alpha_N]$ sampled from the $(N-1)$-simplex (Fig. 1 (a))

$$\text{ResBlock}^N(\mathbf{X}) = \mathbf{X} + \sum_{n=1}^{N} \alpha_n \mathbf{B}_n(\mathbf{X}),$$

while in the backward propagation for $\mathbf{b} = [\beta_1, \cdots, \beta_N]$ sampled from the $(N-1)$-simplex and $\mathbf{g}$ the gradient from the top layer, the gradient entering into $\mathbf{B}_n(x)$ is $\beta_n \mathbf{g}$ (Fig. 1 (b)). At testing time, the expected model is then evaluated for inference by taking the expectation of the random sources in the architecture (Fig. 1 (c)).

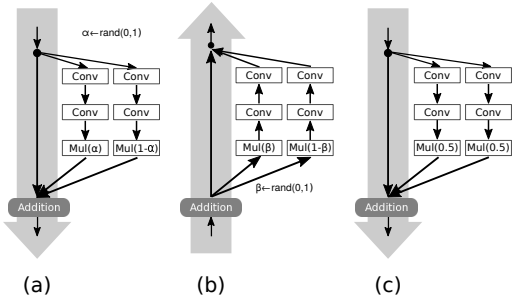

Fig. 1: An overview of a 3-branch Shake-Shake regularized residual block. (a) Forward propagation during the training phase (b) Backward propagation during the training phase (c) Testing phase. The coefficients α and β are sampled from the uniform distribution over $[0, 1]$ to scale down the forward and backward flows during the training phase [9].

It has been shown that shaking with the absence of both batch normalization layers could cause the training process to diverge. One proposed remedy to this situation in [9] is to employ a shallower architecture and more importantly to reduce the range of values α can take on, i.e. to reduce the strength of shaking.

2.2. End-to-End Discriminative Feature Learning

Recently, there has been a trend to focus on the design of loss functions so that a neural network supervised by such a loss function is able to formulate more discriminative features. Inspired by the contrastive loss [16] and the triplet loss [17], a series of works reviewed the interpretation of the softmax loss as a normalized exponential of inner products between feature vector and class center vectors and came up with various modifications, including the large-margin softmax [18], SphereFace [19], CosFace [20], ArcFace [21] and Centralized Coordinate Learning (CCL) [22].

Different from the other four modifications, CCL distributes features dispersedly by centralizing the features to the origin of the space during the learning process so that feature vectors from different classes can be more separable in terms of a large angle between neighboring classes, and ideally symmetrically distributed in the whole feature space. The CCL loss is presented as follows:

$$\mathcal{L} = \frac{-1}{N} \sum_{i}^{N} \log \frac{e^{\Phi(\mathbf{x}_i)\cos(\theta_{y_i})}}{\sum_j^K e^{\Phi(\mathbf{x}_i)\cos(\theta_{y_j})}} \tag{1}$$

where

$$\Phi(\mathbf{x}_i)_j = \frac{\mathbf{x}_{ij} - \mathbf{o}_j}{\sigma_j}. \tag{2}$$

$\Phi(\mathbf{x}_i)_j$ and $\mathbf{x}_{ij}$ are j-th coordinate of $\Phi(\mathbf{x}_i)$ and $\mathbf{x}_i$, respectively, and θ_{y_j} is the angle between $\mathbf{x}_i$ and the class center vector $\mathbf{w}_{y_j}$. It is immediately clear that Eq. (2) resembles the famous batch normalization:

$$\Phi(\mathbf{x}_i)_j = \gamma_j \frac{\mathbf{x}_{ij} - \mathbf{o}_j}{\sigma_j} + \beta_j \tag{3}$$

except that the trainable affine transformation, defined by γ and β, after the normalization are missing in the formulation. In Eq. (2) and (3), σ and $\mathbf{o}$ are running standard deviation and running mean updated per mini-batch during training.

3. SHAKING WITH DIFFERENT CONFIGURATIONS OF RESIDUAL BLOCKS

To demonstrate the close interaction between shaking and batch normalization in representation learning, we present two ablation studies, one on the MNIST dataset and the other on the CIFAR-10 dataset. For both sets of experiments, we employ the ordinary softmax loss to examine the effectiveness of batch normalization in representation learning when it is not coupled with the softmax function.

3.1. Embedding Learning on MNIST and CIFAR-10

The first set aims to visualize representation learning under the influence of batch normalization and shaking. We employ a ResNeXt (20, $2 \times 4d$) architecture, where the last residual block reduces the feature dimension to 2 for the purpose of visualization. Fig. 3 depicts embeddings by four layouts of residual blocks, where the top and bottom rows correspond to embeddings of the training and testing samples, respectively. From left to right, the columns represent embeddings learned by models of fully pre-activation (Fig. 2(c) **PreAct**) without shaking, **PreAct** with shaking, fully pre-activation + BN (Fig. 2(d) **PreActBN**) without shaking and **PreActBN** with shaking, respectively.

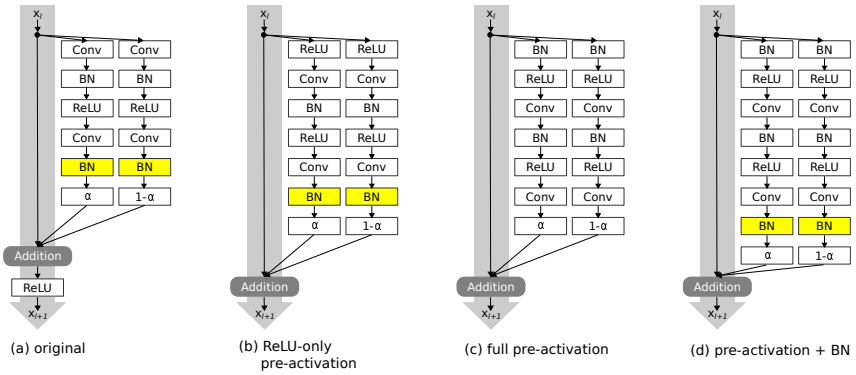

Fig. 2: Shaking regularized ResNeXt architectures with different layouts introduced in [15]

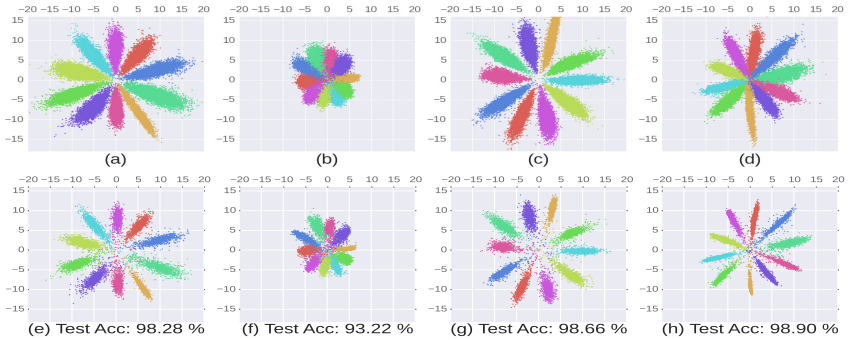

Fig. 3: MNIST embeddings based on different layouts of residual blocks. We set the feature dimension entering into the output layer to be two and train them in an end-to-end fashion. The top and bottom rows depict embeddings of the training samples extracted in the train mode (i.e. $\alpha \in [0, 1]$) without updating parameters, and testing samples extracted in the eval mode ($\alpha = 0.5$), respectively. (a,e) fully pre-activation (Fig. 2(c)) without shaking (b,f) fully pre-activation (Fig. 2(c)) with shaking (c,g) fully pre-activation + BN (Fig. 2(d)) without shaking (d,h) fully pre-activation + BN (Fig. 2(d)) with shaking

The first column serves as the baseline in this set of experiments. Immediately, we can observe a severe degradation in separability when applying shaking to **PreAct**, comparing Fig. 3(a,e) with Fig. 3(b,f). Also notice that shaking without a directly preceding batch normalization could perturb or destroy the symmetric distribution, which is obvious when there is no shaking (the symmetry in Fig. 3(a,e)). This is rather interesting as batch normalization still exists in **PreAct** residual block, only not directly connected to the shaking layer. It seems the exploration encouraged by shaking around each class center has expanded its coverage but without a directly preceding batch normalization to maintain a good dispersion between classes, each class only expands to overlap with neighboring classes, and the resulting distribution is heavily tiled. Consequently, **PreAct** with shaking delivers a much inferior performance compared to **PreAct**. The comparison between **PreAct** (Fig. 3(a,e)) and **PreActBN** (Fig. 3(c,g)), both without shaking, demonstrates the effectiveness of CCL in discriminative feature learning although it is not coupled with the loss function. In Fig. 3(c), not only does it maintain a symmetric distribution of classes, but also it encourages each class to expand outward and to leave more margin between neighboring classes. As a result, **PreActBN** without shaking is able to reach a better performance compared to the baseline.

Finally, **PreActBN** with shaking (Fig. 3(d,h)), on the contrary, does not lead to a larger margin between classes as **PreActBN** without shaking does. On the other hand, it seems that the shaking has expanded the coverage of each class so that all of them are directly adjacent to each other with a minimal or zero margin. We could also observe that, although batch normalization tries to maintain a symmetric distribution of feature vectors, some of the classes in Fig. 3(d,h) are slightly tilted around the outer most region. However, the most salient difference is the distribution of testing samples, where each class becomes more compact. The performance of **PreActBN** is therefore the highest among all of the models. Since in supervised learning we assume testing samples are drawn from a similar or the same distribution as training samples, the majority of testing samples are mapped to embeddings close to the center of class, where most original training samples are mapped to, and thus the distribution seems more compact. In Fig. 4, from the comparison of training embeddings in the train and eval modes, it is visually clear that shaking is expanding the coverage of training embeddings. To quantitatively show that this is the case, for each

layout we calculate distances of the original training embeddings to their respective class center vectors in the eval mode and plot the percentage of class samples within distances relative to the largest distance in the class that is calculated in the train mode, in Fig. 5. It is clear that with shaking most of the original training embeddings are concentrated around the class center. For example, almost 100% of original training embeddings lies within $0.3 \times$ largest distance for **PreActBN** with shaking and just a small number of original training samples are lying close to the boundaries.

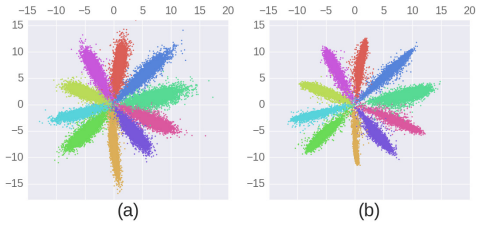

Fig. 4: Embeddings of training samples extracted in the (a) train (b) eval mode from **PreActBN** with shaking

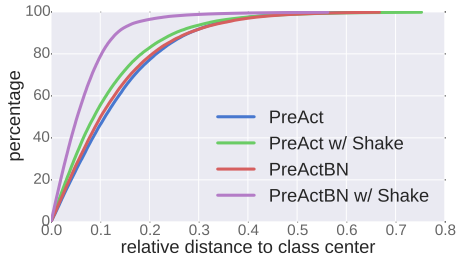

Fig. 5: Percentage of class samples within in a relative distance to class center

The second set of experiments on CIFAR-10 is designed to measure the contribution of batch normalization in residual blocks with respect to it proximity to the shaking layer. In order to do so, we remove the first two batch normalization from the **PreActBN** residual block and rename the new one, the **BN-Shake** residual block (ReLU-Conv-ReLu-Conv-BN-Mul), assuming shaking is applied. Along with **PreActBN** and **PreAct**, by presenting **BN-Shake**, all of them with shaking, we are able to quantitatively demonstrate the crucial location of batch normalization in a shaking regularized architecture.

We modify the open-sourced Torch-based Shake-Shake implementation[1] that was released with [9] to build these three architectures. All of the rest of parameters such as the cosine learning rate scheduling and the number of epochs remain unchanged. Only the part that involves residual block definition is modified to serve our need. We run each experiment for three times to obtain a robust estimate of the performance using different random seeds. Table 1 presents

Model	Depth	Params	Error (%)
ResNeXt (29, $16 \times 64d$) [23]	29	68.1M	3.58
Wide ResNet [6]	28	36.5M	3.80
Shake-Shake (26, $2 \times 96d$) [9]	26	26.2M	2.86
Shake-Shake (26, $2 \times 64d$) [9]	26	11.7M	2.98
ResNeXt (26, $2 \times 64d$) [9]	26	11.7M	3.76
PreActBN (26, $2 \times 64d$)†	26	11.7M	*2.95
BN-Shake (26, $2 \times 64d$)	26	11.7M	*3.65
PreAct (26, $2 \times 64d$)†	26	11.7M	*6.92

* average over three runs
† with shaking

Table 1: Test error (%) and model size on CIFAR-10

the results of our experiments as well as the quoted performances on CIFAR-10 from [9]. Note that Shake-Shake ResNeXt in [9] is based on the ReLu-only pre-activation residual block (Fig. 2(b) **RPreAct**) and is thus different from **PreAct** we have here. Although the ResNeXt-26 $2 \times 64d$ is based on the **RPreAct** structure, with a shallow depth of 26, it should be comparable to one that is based on the **PreAct** when no shaking is applied [15]. Therefore, we also take it as the baseline for the pre-activation layout.

The performance of **PreActBN** with shaking (2.95%, mean of 2.89%, 3.00% and 2.95%) is comparable to the reported performance of **RPreAct** with shaking (2.98%), where both of them have a directly preceding batch normalization layer before shaking. As expected, the performance of **PreAct** with shaking (6.92%, mean of 6.76%, 6.82% and 7.19%) is much worse than every model in Table 1, including the baseline ResNeXt-26 $2 \times 64d$. On the other hand, the result of **BN-Shake** (3.65%, mean of 3.56%, 3.76% and 3.62%) is rather positive. With only one batch normalization layer, it outperforms not only **PreAct** with shaking but also the baseline ResNeXt-26 $2 \times 64d$. This finding highlights the fact that the directly preceding batch normalization plays a crucial role in keeping a good dispersion of intermediate representations when shaking is applied to explore unseen feature space, while the dispersing effect of any other batch normalization that is separated by convolutional layers from the shaking layer is reduced or only auxiliary.

4. CONCLUSION

Based on these two sets of experiments, it is safe to state that in the close interaction between batch normalization and shaking, batch normalization is mainly responsible for keeping a dispersed symmetric distribution of intermediate representations from perturbation by shaking, while the shaking mechanism expands the coverage of augmented training samples to force the distribution of true training samples, and hence that of true test samples, to be more compact, which is exactly the design objective behind end-to-end representation learning. Now that we have learned the essential contribution of shaking to improving classification tasks, one of our future directions would be to apply shaking in an efficient way to reduce the number of epochs (currently 1800 on CIFAR-10) to a reasonable one, without sacrificing the performance.

[1] https://github.com/xgastaldi/shake-shake.

5. 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 (CVPR)*, 2016.

[2] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin, "Convolutional Sequence to Sequence Learning," 2017, arXiv:1705.03122.

[3] Che-Wei Huang and Shrikanth S. Narayanan, "Deep Convolutional Recurrent Neural Network with Attention Mechanism for Robust Speech Emotion Recognition," in *IEEE International Conference on Multimedia and Expo (ICME)*, 2017.

[4] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, Jan. 2014.

[5] Sergey Ioffe and Christian Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, 2015.

[6] Sergey Zagoruyko and Nikos Komodakis, "Wide Residual Networks," in *BMVC*, 2016.

[7] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Weinberger, "Deep Networks with Stochastic Depth," in *ECCV*, 2016.

[8] Xiang Li, Shuo Chen, Xiaolin Hu, and Jian Yang, "Understanding the Disharmony Between Dropout and Batch Normalization by Variance Shift," *arXiv:1801.05134*, 2018.

[9] Xavier Gastaldi, "Shake-Shake Regularization," in *International Conference on Learning Representations Workshop*, 2017.

[10] Yoshihiro Yamada, Masakazu Iwamura, and Koichi Kise, "ShakeDrop Regularization," *arXiv:1802.02375*, 2018.

[11] Alex Krizhevsky, "Learning Multiple Layers of Features from Tiny Images," *Technical Report*, 2009.

[12] Che-Wei Huang and Shrikanth S. Narayanan, "Shaking Acoustic Spectral Sub-bands Can Better Regularize Learning in Affective Computing," in *Proceedings of the IEEE International Conference on Audio, Speech and Signal Processing*, 2018.

[13] Che-Wei Huang and Shrikanth S. Narayanan, "Stochastic Shake-Shake Regularization for Affective Learning from Speech," in *Proceedings of Interspeech*, 2018.

[14] Yann LeCun and Corinna Cortes, "MNIST Handwritten Digit Database," 2010.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Identity Mappings in Deep Residual Networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, Eds., 2016.

[16] Raia Hadsell, Sumit Chopra, and Yann LeCun, "Dimensionality Reduction by Learning An Invariant Mapping," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[17] Florian Schroff, Dmitry Kalenichenko, and James Philbin, "Facenet: A Unified Embedding for Face Recognition and Clustering," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[18] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang, "Large-Margin Softmax Loss for Convolutional Neural Networks," in *Proceedings of The 33rd International Conference on Machine Learning*, 2016, pp. 507–516.

[19] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song, "SphereFace: Deep Hypersphere Embedding for Face Recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[20] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu, "CosFace:Large Margin Cosine Loss for Deep Face Recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[21] Jiankang Deng, Jia Guo, and Stefanos Zafeiriou, "ArcFace: Additive Angular Margin Loss for Deep Face Recognition," *arXiv:1801.07698*, 2018.

[22] Xianbiao Qi and Lei Zhang, "Face recognition via centralized coordinate learning," *arXiv:1801.05678*, 2018.

[23] Saining Xie, Ross Girshick, Piotr Dollr, Zhuowen Tu, and Kaiming He, "Aggregated Residual Transformations for Deep Neural Networks," *arXiv:1611.05431*, 2016.

[24] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz, "mixup: Beyond Empirical Risk Minimization," in *International Conference on Learning Representations*, 2018.

[25] T. DeVries and G. W. Taylor, "Dataset Augmentation in Feature Space," in *International Conference on Learning Representations Workshop*, 2017.