Using mixup as regularization and tuning hyper-parameters for ResNets

Venkata Bhanu Teja Pallakonda

Texas A&M University
College Station, TX, 77801
bhanu@tamu.edu

Abstract

While novel computer vision architectures are gaining traction, the impact of model architectures is often related to changes or exploring in training methods. Identity mapping based architectures ResNets [1] and DenseNets [2] have promised path breaking results in the image classification task and are go to methods for even now if the data given is fairly limited. Considering the ease of training with limited resources this work revisits the ResNets and improves the ResNet50 [1] by using mixup data-augmentation as regularization and tuning the hyper-parameters.

1 Introduction

The performance of a vision model is dependent on both model architecture and training methods. With the findings of attention [3] based architectures the vision transformers [4] achieved the state of art results on the image classification tasks. But training time is expensive and we need to large data-sets for learning. Vision transformers are trained using JFT 300M [5] which happens to be private data. Keeping in view of all these shortcomings and if the provided dataset is not large enough we go back to skip connection and Identity mapping based ResNet [1] and DenseNet [2] architectures which performs reasonably well till date on Image classification tasks.

We use the idea of data augmentation task, mixup as a regularization [6] task to improve our test set validation error (there by increasing accuracy) which would also help us to do better classification if we want to classify a new image that completely out of domain of our train set. We also do Bayes hyper-parameters sweep using wandb [7] to find the best hyper-parameters for our model after applying this regularization task.

2 Methodology

Since the introduction of AlexNet [8] on ImageNet [9] various methods have been proposed to further improve image recognition performance. These improvements typically occur on architecture and training methods.

2.1 Architecture

We experimented and used ResNet50 [1] architecture with the preactivations [10] at every bottleneck block [1] and gelu [11] non linear activation functions. The skip connection block used in the ResNet is shown in Figure 1. We use this skip connection as basic block for the ResNet50 architecture.

Preprint. Under review.

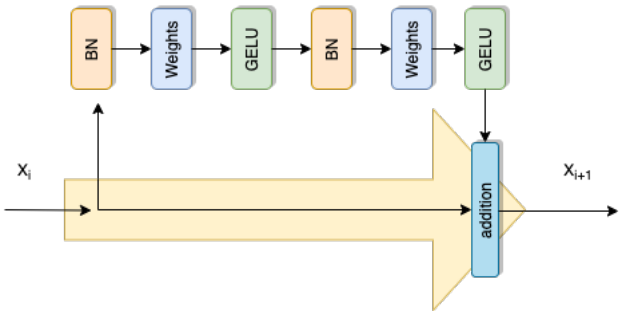


Figure 1: Skip connection block with tweaked activations and order of the activations.

2.2 Augmentations

Data augmentation is an important techinque that will be very handy to imporove out architecture and for improving prediction on out of domain data. In this method we follow mixup (Deatiled explnation in section 3), cropping and random horizontal flipping. As the images dimmensions in CIFAR-10 [12] is small the cropping and flipping would be very useful.

3 Exploring mixup and using it as regularization factor

This section describes how the mixup is performed, how to calculate the loss after mixup for the backpropagation tasks, and how could it can be related it to regularization.

3.1 Mixup calculation

As show in Figure 2 we overlay one image from one class to a different image in another class with an an randomly generated overlay factor of λ from beta distribution [13] during training. A simple algorithmic approach is found below in algorithm 1.

Algorithm 1 mixup mechanism

$s = inputs\ size$
$\lambda = np.random.beta(alpha, alpha, s)$
$index = np.random.permutation(s)$
$x_1, x_2 = inputs, inputs[index, :, :, :]$
$y_1, y_2 = onehot(targets, numclasses), onehot(targets[index,], numclasses)$
$final\_input = \lambda * x_1 + (1 - \lambda) * x_2$
$final\_target = \lambda * y_1 + (1 - \lambda) * y_2$

3.2 Cross entropy loss calculation for mixup

As shown in Figure 3 we take the new true label $\boldsymbol{y}$ and compute the cross entropy loss aganist the predicted probabilities $\hat{\boldsymbol{y}}$ of our model. We follow Algorithm 2 for calculating the effective cross entropy loss for the mix up input images to the model.

$$CE(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\sum_{i=1}^{N_c} y_i \log(\hat{y}_i) \tag{1}$$

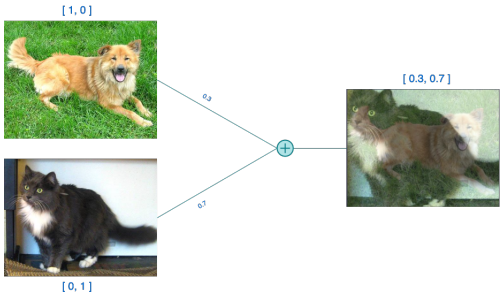


Figure 2: Mixing 2 images ($\lambda = 0.3$)

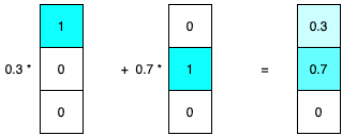


Figure 3: Truelable for mixup

3.3 Relating to regularization

For experimentation we use ResNet50 [1] with GELU [11] non-linear activation function and pre-activation's [10] at every bottleneck layers. With the same given set of hyperparameters and traing the two different models as shown in Figure 4a we could clearly see that loss on the test set is decrease by almost a factor of 1.5 (Also increasing the test accuracy) which could be helpful for predicting the images that are not the same domain of our train set. This could be related to regularization. We can see the the train loss in Figure 4b is higher for this method as we are introducing more uncentrainity to the training samples by mixing up and and forcing our model to do image classification by trying to maximize the single class probability.

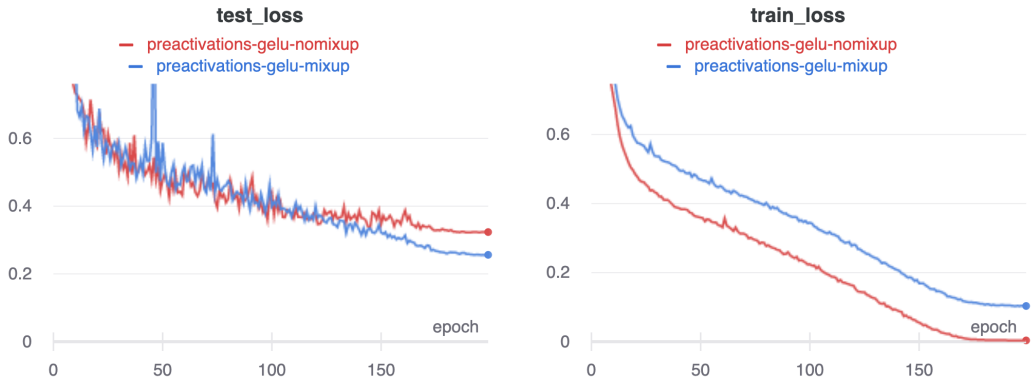
4 Hyperparameter tuning

Using weights and biases [7] parameter sweeps are done with hyperband [14] early stopping. The below figures 5 illustrate the test accuracies over a range of various hyper-parameters and Figure 6 was plotted to check the speed on convergence of the network. From figure 7 we could see the correlation of hyperparameters with test accuracies and their importantance in the sweep.

After runnng the experimentes the optimal hyperparameters are shown in Table 1

Algorithm 2 mixup loss calculation

Require: $\lambda \; from \; mixup \; algorithm$
Require: $final\_input \; from \; mixup \; algorithm$
Require: $final\_target \; from \; mixup \; algorithm$
$x_p = model(final\_input)$
$x_p = torch.log(torch.nn.functional.softmax(x_p, dim = 1).clamp(1e-5, 1))$
$final \; loss = \; -torch.sum(x_p * final\_target)$



(a) Test loss

(b) Train loss

Figure 4: Losses with and without mixup

Table 1: Optimal hyperparameters after experimentation

Hyperparameter	Value
Learning rate	0.05
Batch size	128
Optimizer	SGD
Learning rate scheduler	CosineAnnealing $T_m = 200$ and epochs = 200

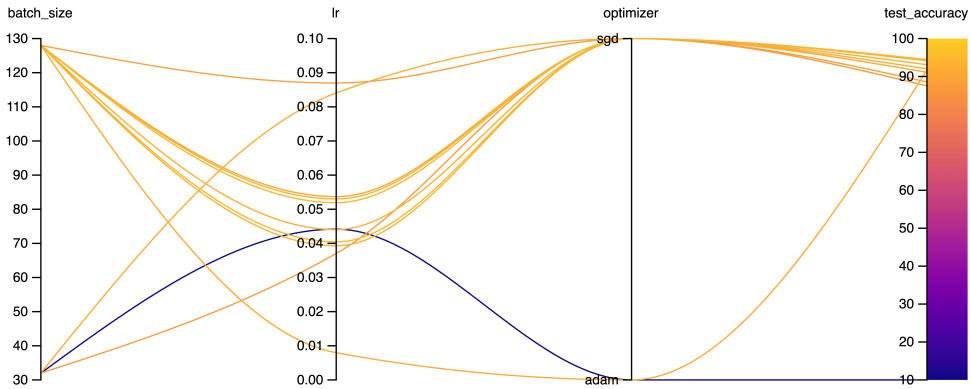


Figure 5: Test accuracies over a range of hyperparameters

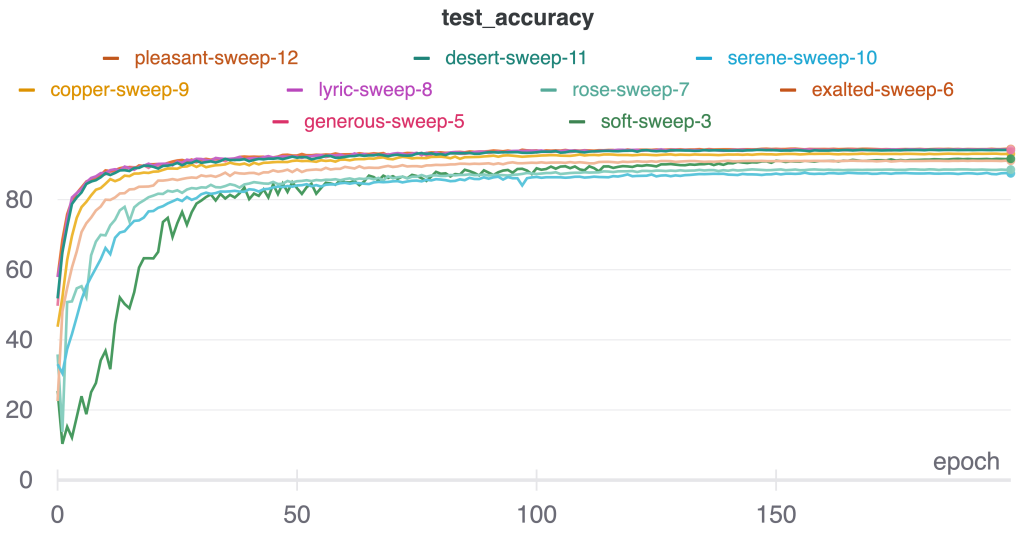


Figure 6: Speed of convergence

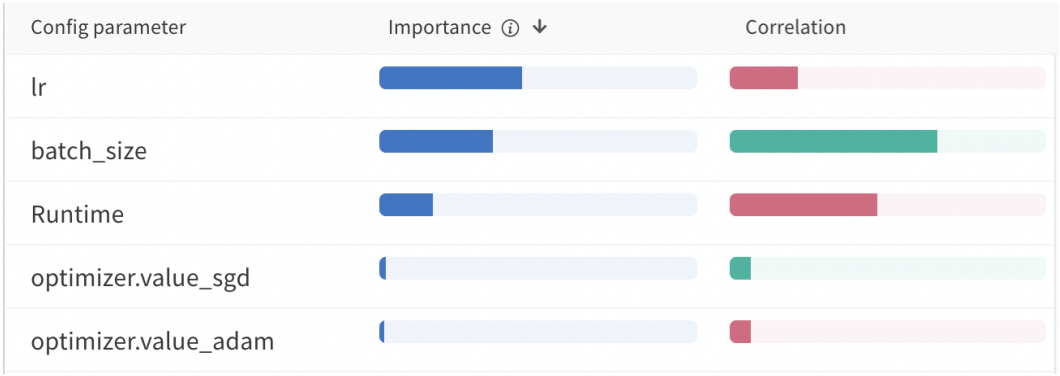


Figure 7: Hyperparameters correlation with test accuracy

5 Results

This method achieves an overall accuracy of **94.57%** on the CIFAR-10 dataset [12]. Below Table 2 shows comparisions with the exisitng ResNet architectures. The ResNet50 could perform well when compared to other higher dept architectures by introducing this idea of mixup.

Table 2: Classification error (%) on the CIFAR-10 test set

network	error(%)
resnet-50	6.97
resnet-110	6.61
resnet-164	5.93
resnet-1001	7.61
This method	**5.43**

References

[1] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” 2015.

[2] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," 2018.

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[4] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," 2021.

[5] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting unreasonable effectiveness of data in deep learning era," 2017.

[6] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," 2018.

[7] L. Biewald, "Experiment tracking with weights and biases," 2020, software available from wandb.com. [Online]. Available: https://www.wandb.com/

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," 2016.

[11] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," 2020.

[12] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-10 (canadian institute for advanced research)." [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html

[13] G. Marsaglia and W. W. Tsang, "A simple method for generating gamma variables," *ACM Trans. Math. Softw.*, vol. 26, no. 3, p. 363–372, sep 2000. [Online]. Available: https://doi.org/10.1145/358407.358414

[14] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A novel bandit-based approach to hyperparameter optimization," 2018.