Wide Residual Networks with advanced augmentation

Rizu Jain
Department of Computer Science
Texas A&M University
College Station, TX 77843
rizujain@tamu.edu

Abstract

Deep residual networks face the issue of diminishing returns. To counter this, Wide Residual Networks increase the number of features while keeping the depth relatively shallow. These networks tend to perform better than their deep counterparts and are faster to train because the widening factor is handled well by the GPUs. In this project, a Wide Residual Network with depth 28 and widening factor 10 was used. However, as the Wide Residual Networks tend to overfit, cutout and optimal image transformation policies for CIFAR 10 learnt using AutoAugment were used to better regularize the network. With hyper-parameter tuning, the network has achieved a accuracy of 97.47% on the public testing dataset.

1 Introduction

The number of layers were gradually increased for deep learning networks. From AlexNet, VGG, Inception to Residual networks each network was deeper than the one before and performed better on image recognition datasets like the CIFAR and SVHN. However, the depth of these neural networks raises difficulties like exploding/vanishing gradients and degradation. This being a crucial issue, it has been studied extensively and numerous workarounds has been suggested like better optimizations, layer-wise training, initialization techniques and skip connections.

Out of the networks mentioned before, Deep Residual Networks has been the most effective. They have been better at generalizing the features thereby giving better accuracies with transfer learning. Future work showed that the residual links make the deep neural network converge faster. Another improvement was done by using identity mappings and pre-activations for Residual networks. Identity mappings allowed deeper networks to train successfully. With that, a 1001 layer ResNet was trained giving the best performance on multiple image classification datasets.

A problem arises with the use of identity mappings though, the problem of diminishing feature reuse. The problem with an identity mapping is that the residual block weights might not learn anything as there is nothing to force the gradient through the ResNet blocks. This results in a only a few blocks learning useful representations. Other blocks make very small contributions to the classification of the images. Naive method of countering this effect is to randomly disable ResNet blocks during training which is equivalent to introducing dropout. The authors obtain improved accuracy with that.

As an alternative, width of the networks is increased instead of its depth to effectively improve performance. The findings are promising showing a Wide Residual Network with 50 times less layers and 2 time faster training times perform comparably with Deep Residual Networks. The increase in speed with the same number of parameters is due to the network's suitability to run on GPUs. As the number of kernels have increased, same input is passed to multiple kernels which gives an opportunity to parallelize.

Preprint. Under review.

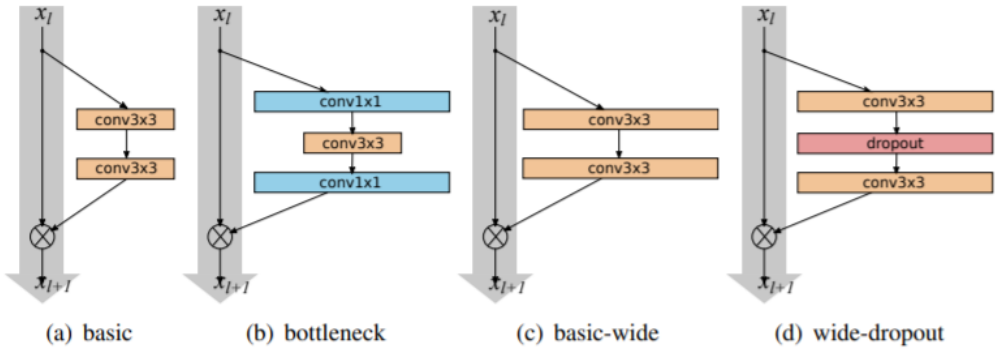

Figure 1: (a) and (b) show the original ResNet blocks. (c) and (d) show the proposed Wide Residual Networks blocks. The width of convolution layer symbolizes the number of feature maps in that layer [1].

Wide Residual Networks however will have many more parameters compared to a conventional deep neural network of same depth. The potential of over-fitting is higher and it increases exponentially with the depth of the network. Thus, regularization techniques yield more improvement than in conventional networks. In our experiments, we compliment the Wide Residual Networks with advanced data augmentation techniques. First, we use Cutout which has demonstrated exceptional improvements in the performance of networks by motivating the networks to learn different aspects of the image and to focus on more salient features. On top of that, we use the best transformation policies learnt by the AutoAugment algorithm. AutoAugment algorithm uses a reinforcement learning agent to identify the best image transformations for a particular dataset. The policies are network agnostic, thus allowing us to use them directly in our network.

The combination worked effectively as the Wide Residual Networks increased the speed of training and the augmentation techniques made the network robust, giving an test accuracy of 97.47% with just 8 hours of training on google colab.

2 Wide Residual Networks [1]

To increase the representational power of residual blocks, one of the following could be done:

- Increase filter sizes in convolutional layers
- Add more convolutional layers per block
- Widen the convolutional layers by adding more feature planes

Various filter sizes has been experimented with and the results show that the small filter sizes work the best. Hence, we stick to a 3x3 filter size in our network.

For the remaining two, let us introduce a factor each. Let l be the deepening factor which denotes the number of convolutions in a block and let k be the widening factor of a residual block which denotes the number of features in convolutional layers. Thus, the basic blocks shown in Figures 1(a) and 1(c) have $l = 2, k = 1$.

The architecture of the Wide Residual Networks is shown in Figure 2. conv1 is the first convolutional layer which is same as ResNet. After that, there are 3 groups of residual blocks, conv2, conv3, and conv4. These groups have N basic blocks where N determines the depth of the network. conv4 is followed by an average pooling layer and finally the classification layer. The widening factor k scales the width of the blocks in the groups. Various combinations for k and N are experimented with in the original paper.

The authors have also experimented with different kind of basic blocks by varying the number of convolution layers and the structures like introducing a 1x1 convolution layer before or after the 3x3 convolution layer. Their results show that using two 3x3 convolution layers works the best. Relying on their results, we use the same in our network.

group name	output size	block type = $B(3,3)$
conv1	32×32	[3×3, 16]
conv2	32×32	$\begin{bmatrix} 3\times3, 16\times k \\ 3\times3, 16\times k \end{bmatrix} \times N$
conv3	16×16	$\begin{bmatrix} 3\times3, 32\times k \\ 3\times3, 32\times k \end{bmatrix} \times N$
conv4	8×8	$\begin{bmatrix} 3\times3, 64\times k \\ 3\times3, 64\times k \end{bmatrix} \times N$
avg-pool	1×1	$[8 \times 8]$

Figure 2: Wide Residual Network architecture. Groups of convolutions are shown together. N is the number of blocks in a group and k is the width multiplier. Final classification layer is not shown [1].

depth	k	# params	CIFAR-10	CIFAR-100
40	1	0.6M	6.85	30.89
40	2	2.2M	5.33	26.04
40	4	8.9M	4.97	22.89
40	8	35.7M	4.66	-
28	10	36.5M	**4.17**	20.50
28	12	52.5M	4.33	**20.43**
22	8	17.2M	4.38	21.22
22	10	26.8M	4.44	20.75
16	8	11.0M	4.81	22.07
16	10	17.1M	4.56	21.59

Figure 3: Test error (%) of various Wide Residual Networks on CIFAR-10 and CIFAR-100 [1].

One typical change that the authors claim to work better is changing the order of batch normalization, activation and convolution in residual block. Changing conv-BN-ReLU to BN-ReLU-conv trained faster and achieved better results than the original version. We use the BN-ReLU-conv order in our network too.

The notation used to represent Wide Residual Network is WRN-depth-k. To select the best depth and k we checked the results reported by the authors. The results shows that increasing the width improves the performance for different depths. We also note that the proportional scaling of width and depth improves performance till the number of parameters become too large. Better regularization is needed in larger Wide Residual Networks like WRN-28-12. As shown in Figure 3, WRN-28-10 gives the best performance on CIFAR 10 dataset. We will use that as our network.

3 Data Augmentation

As the results demonstrate, over-fitting is an problem that Wide Residual Networks face and it is more prominent than the conventional Deep Residual Network. To overcome this, many techniques have been explored. The most effective ones have been applying data augmentation and adding noise to activations, parameters or data. For visual datasets, data augmentation using simple image transformations like cropping and flipping is ubiquitous as it is easy to implement and very effective. Other widely used technique is dropout which stochastically drops neuron activations during training.

Wide Residual Networks' tendency to over-fit gives us an opportunity to try advanced augmentation techniques with this network for gains in performance. We have used two advanced methods for augmentations, Cutout and AutoAugment policies.

3.1 Cutout [2]

Cutout encourages the network to better utilize the full context of the image, rather than relying on the presence of a small set of specific visual features. To achieve this, random square blocks are

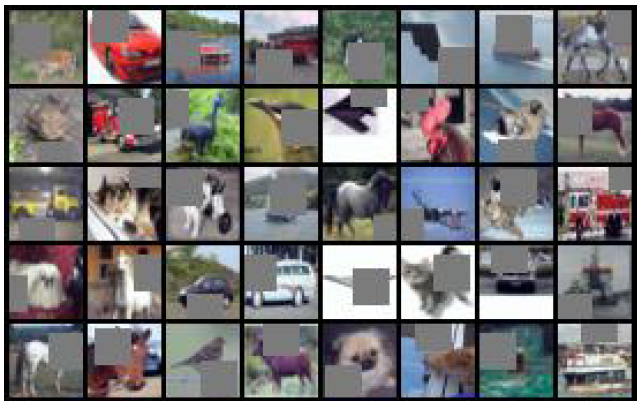

Figure 4: Cutout applied to images from the CIFAR-10 dataset [2].

Method	C10	C10+	C100	C100+	SVHN
ResNet18 [5]	10.63 ± 0.26	4.72 ± 0.21	36.68 ± 0.57	22.46 ± 0.31	-
ResNet18 + cutout	9.31 ± 0.18	3.99 ± 0.13	34.98 ± 0.29	21.96 ± 0.24	-
WideResNet [22]	6.97 ± 0.22	3.87 ± 0.08	26.06 ± 0.22	18.8 ± 0.08	1.60 ± 0.05
WideResNet + cutout	$\mathbf{5.54 \pm 0.08}$	3.08 ± 0.16	$\mathbf{23.94 \pm 0.15}$	18.41 ± 0.27	$\mathbf{1.30 \pm 0.03}$
Shake-shake regularization [4]	-	2.86	-	15.85	-
Shake-shake regularization + cutout	-	$\mathbf{2.56 \pm 0.07}$	-	$\mathbf{15.20 \pm 0.21}$	-

Figure 5: CIFAR and SVHN datasets error rate comparision for different models with cutout. + indicates standard data augmentation (mirror+ crop) [2].

masked out in the images during training. As the masked input propagates through the network, final representation of the image in the network contains no traces of the removed input.

Intuitively, we are removing some of the known features of the images. Suppose the network recognizes a car from its wheels. Essentially, what we are doing is masking out the wheels thereby forcing the network to learn other features of the car like the doors or windows. This makes it more robust at identifying those cars.

Figure 5 shows the results of applying cutout to different models. As we can see, cutout gives a significant improvement over the baselines. It is also noted that it is effective with other augmentation techniques like cropping and flipping. This conceptually and computationally simple augmentation technique is used in our network.

3.2 AutoAugment Policies for CIFAR 10 [3]

In this paper the authors propose a procedure to search for image augmentation policies automatically. For this, they generate a search space where a policy consists of many sub-policies. Each sub-policy consists of two image processing functions such as translation, rotation, or shearing. Sub-policies also contain the probabilities for applying the transformation and the magnitude with which to apply these transformations. One sub-policy is randomly chosen from the policies and the applied to each image in a mini-batch. Examples of sub-policies are shown in Figure 6.

An reinforcement learning agent is used for learning the best policies for a particular dataset. The policies are actions that the agent can take. Policy that the agent selects is then used for training a child model until convergence. The validation accuracy of this child model is then used as a reward by the agent. This way the model learns the best policies for a particular dataset.

One huge drawback of this method is that it is computationally very expensive. To get meaningful results, the child network has to be sufficiently large to have the potential of learning generalizations. This makes each iteration of the reinforcement learning agent computationally expensive.

Despite the computation costs, the polices found by the network achieves excellent results. The policies are used with different models and comparisons are made between accuracies of baseline, with cutout and with AutoAugment policies. As shown in Figure 7, the AutoAugment policies show

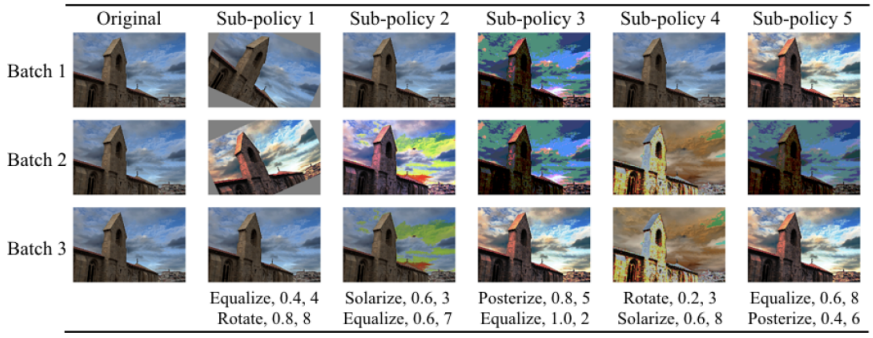

Figure 6: Different Sub-policies used by the AutoAugment algorithm and the effect of their application on images. Note that the probability factor used in both operations gives four different outputs for each sub-policy [3].

Dataset	Model	Baseline	Cutout [12]	AutoAugment
CIFAR-10	Wide-ResNet-28-10 [67]	3.9	3.1	2.6±0.1
	Shake-Shake (26 2x32d) [17]	3.6	3.0	2.5±0.1
	Shake-Shake (26 2x96d) [17]	2.9	2.6	2.0±0.1
	Shake-Shake (26 2x112d) [17]	2.8	2.6	1.9±0.1
	AmoebaNet-B (6,128) [48]	3.0	2.1	1.8±0.1
	PyramidNet+ShakeDrop [65]	2.7	2.3	**1.5 ± 0.1**
Reduced CIFAR-10	Wide-ResNet-28-10 [67]	18.8	16.5	14.1±0.3
	Shake-Shake (26 2x96d) [17]	17.1	13.4	**10.0 ± 0.2**
CIFAR-100	Wide-ResNet-28-10 [67]	18.8	18.4	17.1±0.3
	Shake-Shake (26 2x96d) [17]	17.1	16.0	14.3±0.2
	PyramidNet+ShakeDrop [65]	14.0	12.2	**10.7 ± 0.2**
SVHN	Wide-ResNet-28-10 [67]	1.5	1.3	1.1
	Shake-Shake (26 2x96d) [17]	1.4	1.2	**1.0**
Reduced SVHN	Wide-ResNet-28-10 [67]	13.2	32.5	8.2
	Shake-Shake (26 2x96d) [17]	12.3	24.2	**5.9**

Figure 7: Test set error rates (%) on CIFAR-10, CIFAR-100, and SVHN datasets [3].

significant improvements on all the networks. Added benefit is that these policies has is they can be transferred to new datasets.

The paper's appendix [3] provides the best policies that the agent has found for the CIFAR 10 dataset. We will be using them for data augmentation during training of our model.

4 Experimentation

4.1 Network

We have used Wide Residual Network in our experiments. Specifically, WRN-28-10 (depth = 28, widening factor = 10) has been used owing to its best performance. As stated in the paper we use the BN-ReLU-conv ordering in our blocks. Each block contains two 3x3 convolution layers. A dropout layer with $p = 0.3$ is placed between the two convolutional layers.

4.2 Data Augmentation

Multiple Data augmentation techniques are used. First, the standard techniques of cropping and horizontal flipping are used. After that a sub-policy is selected at random from the best sub-policies identified for CIFAR 10 by the Auto Augment agent. Both the operations of the selected sub-policy are performed with their respective probabilities and magnitude. Finally, Cutout is applied. Best configuration as reported by the paper for CIFAR 10 are 1 square hole of size 16 per image.

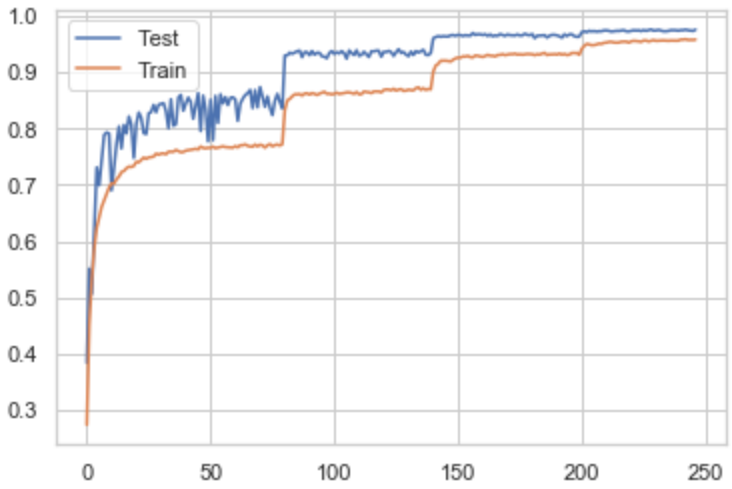

Figure 8: Training curve for WRN-28-10 with cutout and AutoAugment sub-policies. Training accuracy is less as we are using multiple augmentation stratergies.

4.3 Optimizer

In our experiments we use SGD with Nesterov momentum and cross-entropy loss. The initial learning rate is set to 0.1, weight decay to 0.0005, momentum to 0.9 and batch size to 128. Learning rate is dropped by 0.2 at 80, 140 and 200 epochs and we train for total 250 epochs.

4.4 Training

The models were trained on google colab. Each epoch took around 2 minutes to run, with a total of 8 hours of training. The test accuracy and training accuracy graphs are shown in Figure 8.

The training accuracy is lagging in this setup. This is because we use multiple sub-policies and cutout for augmentation. Due to this, it is difficult for the network to predict correctly during training. However, this makes the model more robust and testing accuracies are always better as there are no augmentations used there.

We note that running more epochs for wider and deeper networks has the potential of improving the accuracy.

4.5 Results

The results are summarized in Table 1. WRN-28-10 with standard cropping and flipping gives 96.04%. On adding Cutout regularization, the accuracy improves to 96.91%. Finally, using AutoAugment policies alongside cutout and standard cropping/flipping gives 97.47%.

5 Conclusion

By surveying various networks like PyramidNet, EfficientNet, DenseNet and various augmentation techniques like MixUp, Fmix, etc. we decided on pursuing the combination of Wide Residual

Table 1: Results for Wide Residual Network with different augmentation strategies

Network	Augmentation	Test Accuracy (%)
WRN-28-10	Crop + Flip	96.04
WRN-28-10	Cutout	96.91
WRN-28-10	Cutout + AutoAugment	**97.47**

Networks with cutout and AutoAugment policies. Wide Residual Networks were selected for the improved training speed and to compensate for its over-fitting tendency, we used the augmentation techniques. Thus we conclude that the bigger networks and longer training times aren't the only ways of improving a network. Using better data augmentation techniques and improved network structure could give comparable results.

References

[1] Zagoruyko, Sergey, and Nikos Komodakis. "Wide Residual Networks." Proceedings of the British Machine Vision Conference 2016 (2016).

[2] Terrance V, Graham WT. "Improved Regularization of Convolutional Neural Networks with Cutout." CoRR, abs/1708.04552, 2017.

[3] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. "AutoAugment: Learning Augmentation Policies from Data" In CVPR, 2019.