Exploring Mixed Convolutional Kernels in Wide ResNet: Insights and Performance Analysis

Sai Rithvik Addepalli
Department of Electrical and Computer Engineering
Texas A&M University
College Station, TX 77843
rithvik@tamu.edu

Abstract

Residual Networks, known as ResNets, have long been a benchmark in image classification, setting the standard for accuracy and inspiring many architectures aimed at further advancement. Among the diverse architectures built upon ResNets, the Wide Residual Network (WRN) stands out. While the work proposed by Zagoruyko and Komodakis[3] achieved remarkable accuracy with WRN using 3x3 convolutional blocks, their exploration remained constrained. In our paper, we delve deeper, experimenting with mixed convolutional kernels within varying configurations of convolutions in each block of the architecture. We have achieved a top 1 % test error of 5.76 % using the proposed architecture.We also experimented with various augmentation techniques and also performed expirements with different hyperparameters.The models have been trained on CIFAR-10 dataset and experimented also with the same. The results also are reported on the CIFAR-10 dataset.

1 INTRODUCTION

ResNets proposed by He et al.[2] have been a benchmark for a very long time in the field of image classification tasks, it features shortcut connections(known as skip connections to enable training of very deep neural networks effectively.An improved version of ResNets where the skip connections are kept clean without any operations have been proposed by He et al.[1].Scaling neural networks is something which is not uncommon.Scaling can be either done depth wise (adding more convolutional layers and residual blocks) or width wise(increasing the number of filters). The work done by Zagoruyko and Komodakis successfully has shown that wider networks not as deep as the state of the residual networks perform better.Their proposed 16 layer wide residual network (WRN) outperformed a 1000 layer deep residual network.

2 MIXED KERNEL CONVOLUTIONAL WIDE RESIDUAL NETWORK

The work done by Zagoruyko and Komodakis mainly focused on using 2 3x3 convolutional layers as the building blocks for their architecture.Their initial test results showed that both 2 3x3 kernels , and blocks of 3x3 1x1 3x3 kernel performed similarly.This paper experiments with mixed convolution kernels in a block of Wide Residual Net. We also compare our results to the original residual network and also to wide residual network (WRN) with the 3x3 convolution layers as the building block.

2.1 BASIC BLOCKS AND MODEL

The widening factor, denoted by K (authors of [3] refer to the original residual networks with K = 1, and anything greater than 1 is referred to as a wide residual network), determines the factor by which

Group	Output Size	Block Type	No. of Blocks
Conv1	32x32	3×3, 16	1
Conv2	32x32	3x3,16 x K 1x1,16 x K 3x3,16 x K	N
Conv3	16x16	3x3,32 x K 1x1,32 x K 3x3,32 x K	N
Conv4	8x8	3x3,64 x K 1x1,64 x K 3x3,64 x K	N
avg pooling layer	1x1	8X8	1

Table 1: Strucuture of the model without the final classification layer (this table shows teh core of the model)

Basic Conv Block Layers	Time Per Epoch	Test Loss
1x1 3x3 1x1	95s	9.1
3x3 1x1 3x3	65s	8.2
3x3 1x1 3x3	88s	10.9

Table 2: Depth is set to 16, K = 2

the number of filters is increased. Groups of convolutional layers, each repeated N times, are stacked upon each other. There are typically four main convolutional groups in wide residual networks.In experiments, each group consisted of three convolutional layers. The depth of the network is always in the form of 6N+4 , where N for each group is determined by the calculation(depth-4)/6 . Initially, experiments were performed using different combinations of convolutional layer groups:1X1, 3X3, 1X1; 3X3, 1X1, 3X3; and 3X3, 1X1, 1X1 with a depth set to 16 and a widening factor of 2, the model was trained for 200 epochs without any data augmentation other than normalizing the features. The groups with 1X1, 3X3, 1X1 took around 95 seconds for each epoch, whereas those with 3X3 ,1X1 , 1X1 took around 88 seconds per epoch to train. They performed poorer compared to the ones with 3X3, 1X1, 3X3. (The first case had a test error of 9.1, the second one had a test error of 8.2, and the third one had a test error of 10.9). It was decided to focus on the groups with 3X3, 1X1, 3X3 as the convolutional layers(table 2 shows the results).Table 1 shows the architecture overview of the blocks in the proposed model. Figure 1 shows the modified residual block. One more dropout layer has been added after the third convolutional layer. The original work only has dropout layers only between the convolutional layers. Downsampling is performed in layers 2 to 4. Effect of having the third dropout layer is shown in the experiments section.Each convolutional layer is precedded by batch normalisation and then ReLu.

3 Implementation

3.1 Data Pre-processing

In the preprocessing steps , each of the training set of CIFAR-10 image underwent a series of transformations with the aim to enhance the diversity of the training data. Initially, a reflection padding technique was applied, adding four pixels along each side of the image. Then , random cropping was applied to extract a [32, 32] section of the image.The images were randomly flipped. Random adjustments to brightness and contrast were done to simulate varying lighting conditions.Additionally, jittery pixels were introduced to simulate distortion, and random noise was added. Finally, normalization was performed, subtracting the mean and dividing by the standard deviation of pixel

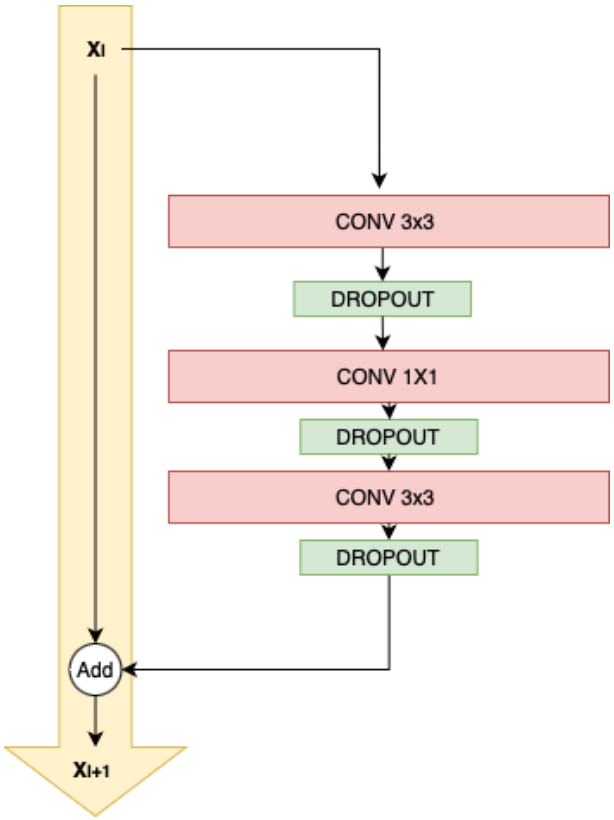

Figure 1: Modified Wide Residual Block based on Zagoruyko and Komodakis's work (Note:These convolutional layers are wider than the standard resnet convolutional layers,they have more filters Batch Normalisation and ReLu have been omitted from the diagram to show neater diagram.)

values to standardize the input data.A train validation split ratio of 0.8 was used to evaluate different experiments.

3.2 Training Procedure

The model was trained for 200 epochs using Stochastic Gradient Descent (SGD) as the optimizer. The model had a initial learning rate of 0.1, it was decreased by a factor of 10 every 75 epochs to ensure smoother convergence. Each epoch processed minibatches of 128 samples.The weight decay was set to 0.0001, providing regularization to prevent overfitting, the dampening factor remained was set to 0.

3.3 Hardware and Software

The model was implemented using PyTorch a python based framework for deep learning. The model was trained on the Nvidia A100 GPU on 1 core.Many experiments were performed.The best performing model took around 61.5 seconds on an average per epoch to train.

4 Experiments and Results

4.1 Widening Factor

Few of the experiments were performed without any data augmentation.Various experiments with multiple widening factor K and depth was performed. As K increases the network becomes more wider.To train efficiently and keep the number of parameters optimal the decision was made to decrease the K as the depth increases. K values were chosen from 2,4,8 and 10. While the depth

Depth	K	Top 1%
16	2	86.4
22	8	90.1
28	10	92.4
34	2	91.1
40	10	89.8

Table 3: Experiment of the Widening factor with various depths

Depth	K	Top 1%
22	10	91.7
28	10	92.4
34	10	93.5
40	10	89.8

Table 4: Experimenting with depth

varied from 16 to 40. The model relatively performed poor on the validation dataset for lower K and lower depth (16 depth , widening factor-2) and higher depth and higher widening factor (depth 40 , widening factor 10).Table 3 shows the results.

4.2 Depth

After experimenting with the widening factor , 10 was chosen as the value of K to experiment with the depth. The model was experimented with varying depths from 16 to 34. The model with depths 22,28 and 34 performed similarly.Also increasing the depth and widening. Also increasing the depth more made the model take more time to train.Table 4 shows the results on the public test dataset.

4.3 Order of BN RELU

The authors of [3] changed the traditional order of convolution-batch normalisation-ReLu from [1] and [2] to the order batch normalisation-ReLu-convolution. They claimed that it trained faster and showed better results. However I experimented with original order also and found that there is no much significant difference in training times and validation accuracy's. I experimented with model with (Depth 28 and Widening Factor 10). Table 5 shows the results on the public test dataset.

4.4 Affect of Extra dropout layer

Adding extra dropout layer after the third convectional layer improved my testing accuracy, it acts as a effective regularisation method.The dropout rate throughout the experiments were set to be 0.3. The testing accuracy for the model with 28 layers depth and widdening factor K as 10 saw an increase of 0.7% in the validation dataset and 0.45 % increase in accuracy for the public test dataset.

5 Results and Conclusion

The final runs were done on the entire training dataset without any train valid split.Image augmentation as described in section 3.1 was performed.The best performing model was with depth 28,widening factor 10. 6 shows our results.The final accuracy on the public test dataset was **94.26 %** The results presented here perform better than the original deep residual networks. They are still a bit less than the state of the art WRN's.This worked showed that even the Wide Residual Mixed Convolution

Depth	Order	Top 1%	Time per Epoch
28	BN-ReLU-Conv	93.14	63s
28	Conv-BN-ReLu	93.2	61s

Table 5: Comparission of Order of Batch Normalisation and ReLu with respective to Conv

Model	Depth	Test Error	Source
ResNet	20	8.75	[1]
Resnet	32	7.51	[1]
Resnet	56	7.17	[1]
Resnet	110	6.43	[1]
Resnet	1202	7.93	[1]
WRN (28-10 ours after image augmentation)	28	**5.74**	-
WRN (34-10ours prior to image augmentation)	34	5.9	-
WRN (Original Work)	28	4.17	[3]

Table 6: Our results vs other works

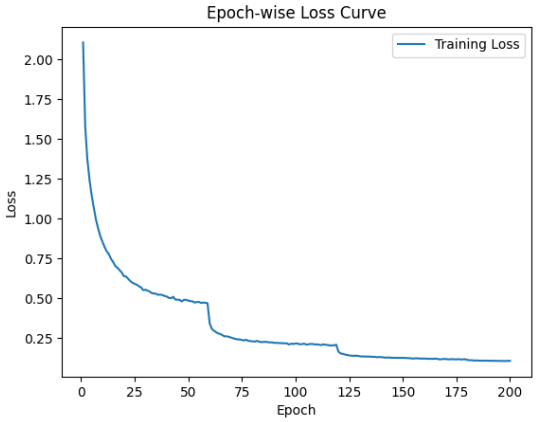

Figure 2: Training loss curve for our final model with depth 28 and widening factor 10.

Kernels can be used effectively for image classification. They can perform better than the original ResNets , training them is not very time and resource consuming.

The project submission zip file contains the checkpoint file for the best model only

The model weights (checkpoint files) to verify other the claimed results can be obtained at : link

6 Future Work

In the future we aim to experiment with better weight initialisation techniques as better learning rate techniques.Since the skip connections are clean we also want to experiment if we add something to the skip connection and make the training even more better. We also plan to experiment with different activation functions and see their effect.From figure 2 it can also be seen that the loss exhibits a sudden drop just before the learning rate is changed both the times. We also aim to examine the reason for this sudden drop and stair case like pattern.

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.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 630–645. Springer, 2016.

[3] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.