Image classification on CIFAR 10 dataset using ResNet

Rajesh Satpathy
Department of Computer Science
Texas A&M University
College Station, TX - 77843
rajeshsatpathy@tamu.edu

Abstract

Increasing layers is an attempt to increase the accuracy of a model, but adding layers after a certain level leads to the "Degradation Problem" i.e. the training error increases on adding more layers. A proposed solution to this is using shortcut connections and residual mappings - ResNet[2]. This guarantees increased training accuracy over deeper layers and also avoids the vanishing/exploding gradient problem. My project approaches at using Resnet architecture by Image transformations, Widening the Residual Blocks, Pre-Activation, Batch Normalization, Bottleneck blocks, Model selection based on lower loss values and Dropouts. These strategies help successfully training a model that reaches the accuracy of **93.14%** on the CIFAR 10 [4] test data.

1 Introduction

The aim of an efficient Neural Network Model is to increase the accuracy of prediction. The training accuracy is thought to be directly correlated to adding more number of Convolution layers. But this hypothesis started failing after addition of more layers after reaching a certain accuracy. The normal approach of adding more layers faced with 2 major issues - *Vanishing/Exploding gradient.* and *Degradation problem.* The 1^{st} issue gives a convergence problem and is dealt with using Batch Normalization, Dropouts and ReLU activation. The 2^{nd} issue gives a training error at deeper layers and is dealt with using ResNet[2] architecture of using Identity mappings and Shortcuts. There have been several improvements over increasing the accuracy of a normal ResNet that has been discussed in the survey paper[1]. These improvements help increase the accuracy efficiently without increasing the number of computations required when increasing the depth of the network. The Basic ResNet block used is as shown in Figure 1. To get to a good convergence, Stochastic Gradient Descent with warm Restarts (SGDR)[5] is used. Widening the number of intermediate filters and increasing the number of layers improves the accuracy. However, it is observed that it is possible to get a better accuracy by widening rather than increasing the depth in lesser number of computations. The novelty proposed in my project is saving the checkpoint models based on the *Least Losses* to get the best model. This gives an option to choose good models for lower number of epochs.

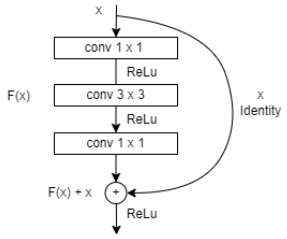

Figure 1: Building block ResNet

2 About the CIFAR 10 dataset

The model proposed in this project is tested on the CIFAR 10 dataset. The dataset provides 50,000 images available for training and 10,000 images for testing. It is a labelled dataset as shown in the Table 1. A snapshot preview of the images is shown in Figure 2.

Table 1: Labels of the dataset

Label Name	Label value
airplane	0
automobile	1
bird	2
cat	3
deer	4
dog	5
frog	6
horse	7
ship	8
truck	9

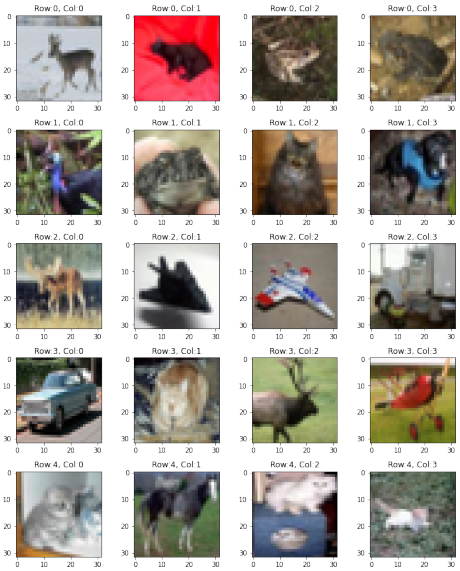

Figure 2: CIFAR 10 images

2.1 Data Augmentation and Transformation

The image data is available as a linear array of values, of shape [3072,], which is reshaped and preprocessed to a [32, 32, 3] shape of *Height* $\times$ *Width* $\times$ *Depth*. The preprocessing includes padding, random cropping, random horizontal flipping and normalization. This helps in getting a larger range of images and a better training. Note that only the training data undergoes padding - 4 pixels on each side, Random Cropping - a *32* $\times$ *32* part of the padded image and Random Horizontal Flipping augmentations and the test data only goes through data normalization. The dataset has 50,000 training images and 10,000 testing images.

3 Model Implementation

3.1 Layers

In this project, a network of a depth of **56** layers is built. Bottleneck blocks are used to reduce the number of computations. The model starts with widening the number of filters from 3 to 128. This scaling has been tested with 16, 32, 64, 128 and 256. Scaling to 128 gives the optimal accuracy in the least amount of time and is chosen for the model. The model contains 3 stacks, of which the initial stack is plain bottleneck and the other two are residual bottleneck blocks. The widening of number of filters is referenced from Wide Residual Networks[6]. Increasing the depth the model has been tested with 29, 38, 47, 56 and 65 layers. The model with 56 layers is chosen. A total number of parameters for the chosen model of 56 layers is calculated to be around **2.37M**. Table 2 shows the architecture of the implemented model. The first layer is a convolutional layer that increases the number of filters from 3 to 128. All the blocks follow pre-activation as mentioned in the referenced paper[3].

Table 2: Model Architecture

Layer Name	Start	Stack 1	Stack 2	Stack 3	Output
Output Map Size	32x32	32X32	16x16	8x8	1x1
No. of layers	1	18	18	18	1
No. of filters	128	128	256	512	10

3.2 Gradient Descent

Stochastic Gradient Descent is used as an optimizer function to reduce loss and improve accuracy. Cross Entropy loss is calculated to track the loss along the epochs. It is observed that an adaptive learning rate performs better compared to the decreasing the learning rate every interval. The initial learning rate is set to 0.01 and then an adaptive learning rate is used, inspired from SGDR[5] - A cosine annealing Learning Rate function to get a good learning rate after each batch. This helps in getting a better convergence for the model. To avoid vanishing gradient and stopping training improvement at a local minimum, a momentum of 0.9 is used. To handle the noise and get a more generalized model, Weight Decay of $1x10^{-4}$ is used.

4 Results and Model Selection

To get the best accuracy among the epochs, the epochs with the least loss are also considered as potential models instead of the ones only at regular intervals.

4.1 Validation and Hyperparameter selection

The validation set is prepared by splitting the training data in an 80:20 ratio i.e. 40,000 training images and 10,000 validation images. The validation set is used to choose a good model and later also used to choose a good set of hyperparameters for a chosen model. The chosen model of modified ResNet gets a maximum accuracy of 92.46% with 100 epochs and gives the best accuracy at the 90th epoch. Figure 3 shows the Loss vs. Epochs and Figure 4 shows Epochs vs. Accuracy for the model.

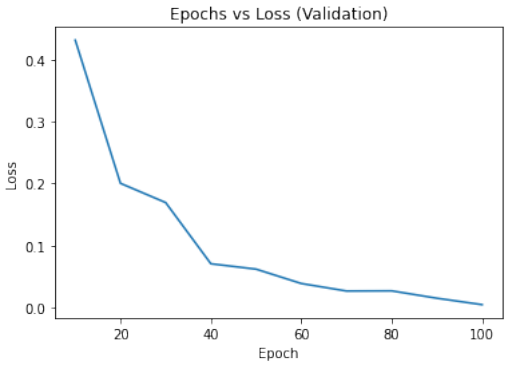

Figure 3: Epochs vs. Loss - Validation

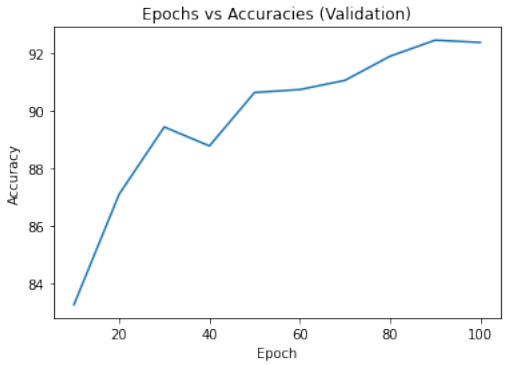

Figure 4: Epochs vs. Accuracy - Validation set

The hyperparameters for the model giving the above accuracy is used to train the whole data of 50,000 images. The chosen hyperparameters are shown in Table 3.

Table 3: Hyperparameters chosen for the model

Parameter name	Description	Value
Depth (n = 6)	Number of layers	9n+2 = 9×6 + 2 = 56
Width	Initial number of filters	128
Batch size	Number of images in each batch	200
Weight Decay	Weight Decay/Regularization	1×10^{-4}
Learning Rate	Initial Learning Rate	0.01
Momentum	Momentum	0.9
Dropout	Probability of dropping a path	0.25
Epochs	Number of epochs	200

4.2 Testing

The model is trained for 200 epochs to get the top accuracy of **93.14%** on epoch 189. Here, we also consider the model with the *Least Loss* is as a potential model for getting a better accuracy. This is done in addition to saving models at regular intervals. Plots are drawn to understand the when the Loss value converges to near zero and Training accuracy does not improve any further. This helps in deciding the number of epochs required and if there is a need to change and try for other hyerparameters. Figure 5 is the plot between Epochs and Loss, and Figure 6 is the plot between Epochs and Accuracy.

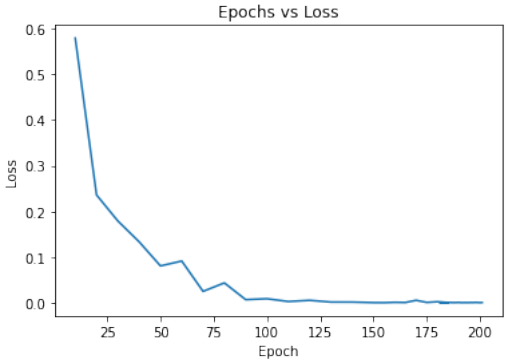

Figure 5: Epochs vs. Loss - Training

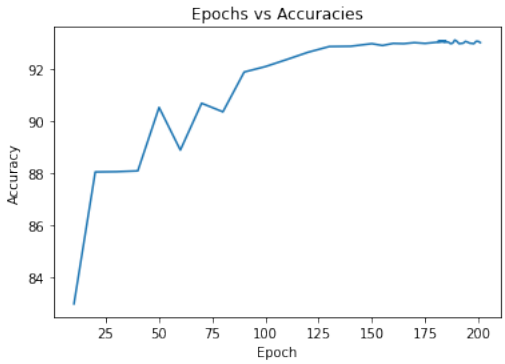

Figure 6: Epochs vs. Accuracy - Test set

5 Conclusion

Several models with different blocks sizes, width and depth were experimented to reach an efficient model. It was observed that a wider ResNet improves performance better compared to increasing depth. A ResNet bottleneck block is used to reach a higher depth with less number of parameters. Data Augmentations are used to train on a more generalized model. Adaptive learning rate gives a better convergence to get the best model at the end of training. For selecting better potential models with optimal hyperparameters, least loss is considered at low number of epochs.

References

[1] I. Bello, W. Fedus, X. Du, E. D. Cubuk, A. Srinivas, T.-Y. Lin, J. Shlens, and B. Zoph. Revisiting resnets: Improved training and scaling strategies, 2021.

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

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

[4] A. Krizhevsky. Learning multiple layers of features from tiny images, 2009.

[5] I. Loshchilov and F. Hutter. Sgdr: Stochastic gradient descent with warm restarts, 2017.

[6] S. Zagoruyko and N. Komodakis. Wide residual networks, 2017.

A Appendix

[1] PyTorch version 1.10.0+cu111 is used in this project and is compatible with PyTorch 1.6

[2] Selected models are stored as checkpoints in models directory.