CS-GY 6953 Project 1

Matthew So[1]

[1]Tandon School of Engineering
New York University
m.so@nyu.edu

Abstract

We aim to achieve the highest possible accuracy on an unknown, custom test set. The task is to train a ResNet-type model (or other convolutional neural network) with under 5M parameters on the CIFAR-10 dataset. In this project, we use an implementation of Wide Residual Networks (Zagoruyko and Komodakis 2017), which achieved state-of-the-art (SOTA) performance on CIFAR-10 in 2017. We reproduce their findings and only add minor data augmentation modifications to increase our performance on the unknown test set, leading to an accuracy of 84.68.

Project code can be found at this link - https://github.com/ms15032/project-1.

Introduction

Traditional ResNets are usually networks with many layers. The purpose of the residual connection was to ensure gradients don't vanish in networks with hundreds or even thousands of layers. In (Zagoruyko and Komodakis 2017), the authors show that competitive performance can be achieved by using "wide" networks as opposed to "deep" networks. In this case, "width" is proportional to the number of convolutional filters used in each block. Each block still contains a residual connection, so these networks are called WRNs (Wide Residual Networks).

This network architecture was chosen because of its high performance on the CIFAR-10 dataset. In 2017, this paper achieved SOTA performance on CIFAR-10 compared to other ResNets. Today's SOTA results only achieve accuracy improvements in the single digits, and almost always involve transformer architectures or pretraining on other datasets, all of which is not in the scope of this project. It is also highly configurable (mainly, the depth and width parameters d and k), so the number of parameters can be adjusted to fit the requirement of 5M parameters.

Methodology

Our implementation was a faithful reproduction of the authors' original implementation. Our model was written as a PyTorch module and trained in a Kaggle notebook with an Nvidia P100 GPU for 200 epochs. The Cross Entropy loss was used with a Stochastic Gradient Descent optimizer with a learning rate of 0.1, momentum of 0.9, and weight decay of 0.0005, all of which are clearly stated by the original authors. A learning rate scheduler started at 0.1 and decreased by 0.2 times at epochs 60, 120, and 160. This is to allow the model to quickly find minima, then refine that guess by using a smaller learning rate after a certain period. This is clearly visible in the training curves (Fig. 1, 2, 3, 4) - loss makes a marked decrease and accuracy makes a marked increase at epochs 60 and 120, with a smaller difference at epoch 160.

d	k	Params	Transforms	Train	Test	Custom
40	2	2.2M	Flip Crop	99.75	95.25	83.52
22	4	4.3M	Flip Crop Color	99.87	95.79	84.68
22	4	4.3M	Flip Crop Color Affine	98.99	95.34	84.58

Table 1: Train set, test set, and custom test set accuracies for 3 model and training variants.

Results

Our first run used a network with depth 40 and width 2, totaling 2.2M parameters. The original authors did not use many data augmentations, only a random horizontal flip and random crop of 32x32. Since the original images are also 32x32, a padding of 4 was used with the "reflect" padding mode applied. The images were then rescaled to values with a mean of (0.491, 0.482, 0.447) and standard deviation of (0.247, 0.244, 0.262), which were also defined in the original paper. We were able to reproduce the results stated in the paper, achieving a train accuracy of 99.75 and test accuracy of 95.25 (Fig. 1, 2). We observed a sharp decrease in accuracy on the custom test set of 83.52, so aimed to achieve further generalization of the model.

Our second run used a larger model with $d = 22$ and $k = 4$. This model is not tested by the original authors, but was a suitable combination to increase the model parameters as close as possible to the limit of 5M. The authors had observed higher performance with higher widths, so we increased the width to 4 and decreased the depth until the num-

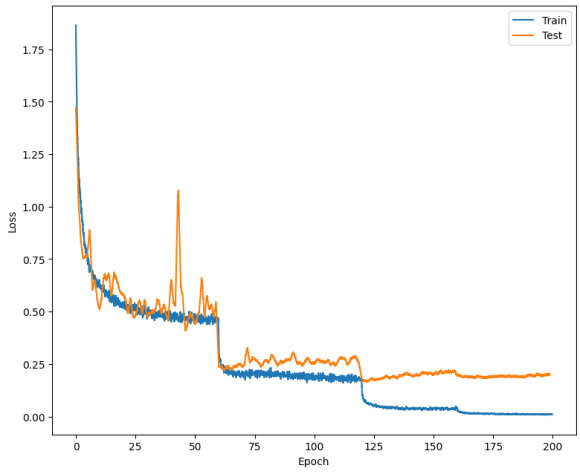

Figure 1: Train and test loss over 200 epochs for the 2.2M model with flip and crop data augmentations.

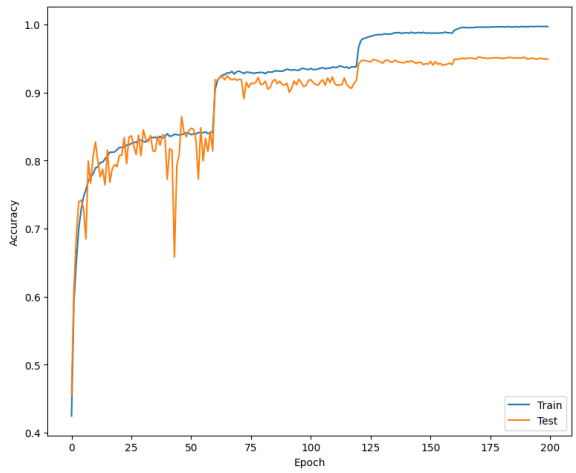

Figure 2: Train and test accuracy over 200 epochs for the 2.2M model with flip and crop data augmentations.

ber of parameters reached less than 5M (in this case, 4.3M). Attempting to increase generalization capabilities, we also added a color data augmentation, with brightness, color, and saturation randomly perturbed by 20%. The model reached its maximum accuracy much quicker, presumably because of its larger size. We achieved a higher custom test set accuracy of 84.68 (Fig. 3, 4).

Lastly, we attempted to add additional data augmentations to further increase generalization capabilities. We added a random affine transformation, which randomly applies a random rotation up to a maximum of 15 degrees, a random vertical or horizontal translation of 20%, a random zoom-in or out by 20%, and a random shear to a maximum of 15 degrees. This made the dataset much harder for the model to learn, though it still reached a train accuracy of 98.99 and a test accuracy of 95.34. This exhibited very slightly less overfitting, but the custom test accuracy was still slightly lower at 84.58.

The best performing model, a WRN with depth 22 and width 4 and totaling 4.3M parameters, achieved a custom test set accuracy of 84.68. The transforms used were horizontal flip, crop, and color perturbations. The loss, optimizer, and learning rate schedule were not changed from the original implementation defined in the paper.

Since it wasn't clear what data is contained in the custom test set, and accuracy on the CIFAR-10 test set was already quite high for models within this scope, we did not experiment further. Based on our 3 runs, we did not foresee any benefit from additional model parameter changes or data augmentations.

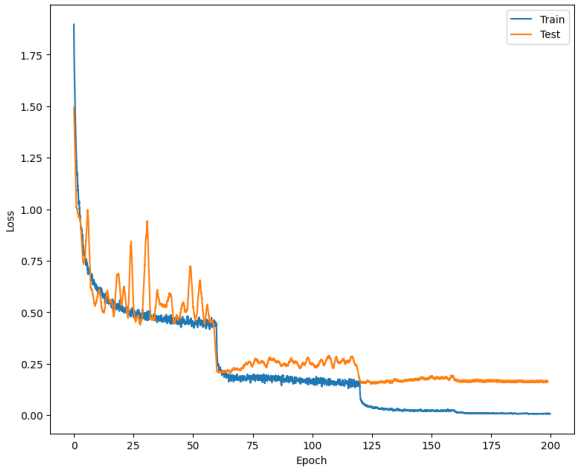

Figure 3: Train and test loss over 200 epochs for the 4.3M model with flip, crop, and color data augmentations.

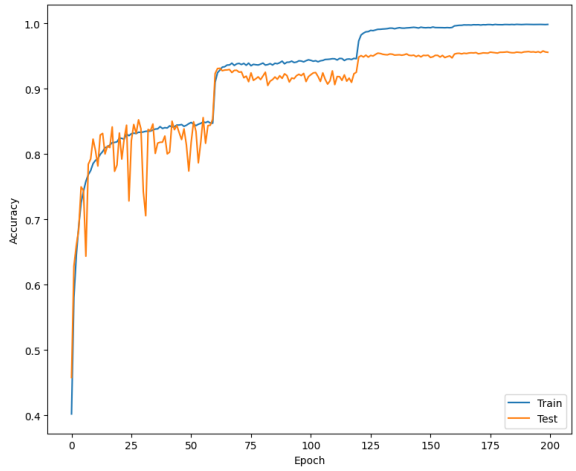

Figure 4: Train and test accuracy over 200 epochs for the 4.3M model with flip, crop, and color data augmentations.

References

Zagoruyko, S.; and Komodakis, N. 2017. Wide Residual Networks. arXiv:1605.07146.