HW 2

Garrison Neel
CSCE 636

October 1, 2020

I Answers to the non-programming part

6 Deep Residual Networks for CIFAR-10 Image Classification

(a) CIFAR-10 dataset format

The python version of the dataset was used, downloaded from this URL:
https://www.cs.toronto.edu/ kriz/cifar-10-python.tar.gz

II Results and Analysis

The programming part was completed after reading the references listed in the references section.

1 Network Structure

The network was implemented as described in [1]. The cifar dataset contains images of shape (32,32,3). For our network, the input layer is a 3x3 convolution with 16 filters. Following this are 3 stacks of residual blocks, with increasing number of filters. The stacks have [16, 32, 64] filters respectively. The output layer consists of a global average pooling operation and a fully connected layer with 10 neurons, one representing each image class.

Version 1 uses the standard residual block and activations described in [1]. Version 2 uses bottleneck blocks, with pre-activation described in [2]. Pre-activation refers to the placement of the activation functions in the block. The order used is shown in figure 1

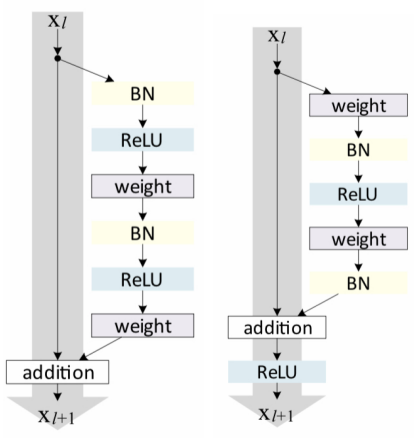

Figure 1: Residual Block Structure

2 HyperParameter Selection

The following hyperparameters were explored:

- ResNet version
- Number of residual blocks per stack layer
- Number of training iterations
- Batch size
- Learning rate

(a) ResNet Verison

[1] and [2] discuss the benefits of pre-activation and bottleneck blocks extensively. To confirm these findings, I trained a network of size n=18.

Network	Validation Loss	Validation Error (%)
ResNet-18v1	0.780658	7.64
ResNet-18v2	0.423452	**5.42**

For this reason, version 2 was used in the final model.

(b) ResNet Size

Again [1] and [2] provide substantial evidence that deeper networks can achieve higher accuracies, but can be harder to train. The authors found that a stack layer size of 18 was already reaching the limits of training feasibility. Below is a table comparing validation error.

Network	Validation Error (%)
ResNet-3v1	8.74
ResNet-18v1	**7.64**

For the final model, a size of 18 was chosen. Larger networks were considered but not evaluated due to training time considerations and concern for training effectiveness. The higher loss value seen in ResNet-18 is likely due to the regularization term.

(c) Number of training iterations

According to [1], 64k iterations were sufficient to train the largest residual network. This is consistent with my findings. At a batch size of 128, this corresponds to approximately 180 training epochs.

After 64000 iterations there is a plateau in both accuracy and loss. Validation accuracy is no longer increasing and has potentially peaked. For this reason, the final model will be trained for 170 epochs (this is where the validation accuracy peaked).

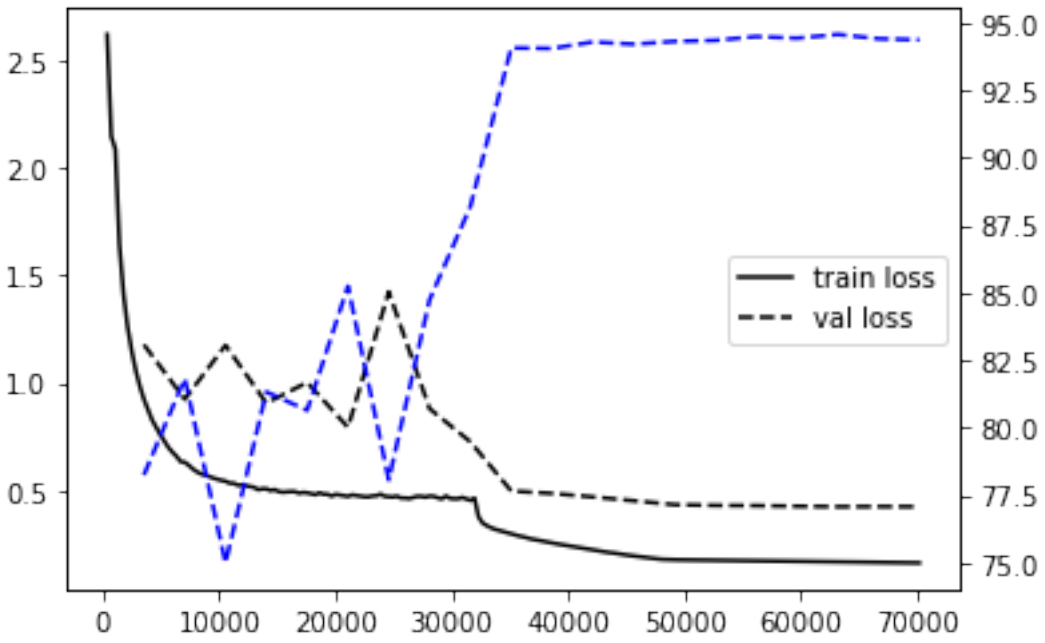

Figure 2: Training and Validation Loss vs training iterations for ResNet-18v2. Accuracy is shown in blue.

(d) Batch Size

Selecting batch size is a tradeoff between training generality and training speed. There exists some minimum batch size for which the samples are still representative of the training dataset. Using this batch in training allows us to update the network more frequently because processing a batch requires fewer computations than the entire training set. Smaller batches means faster training, but if batches aren't generally representative, the network might not ever move in a proper direction to generalize for the whole dataset. The authors of [1] used a batch size of 128. I have trained ResNet-3v1 with batch sizes of (64, 128, 256, 512).

Batch Size	Validation Error (%)
512	9.02
256	**8.48**
128	8.74
64	8.82

A batchsize of 256 yielded the highest validation accuracy. All models took a similar number of training iterations to reach their lowest loss. Higher batch size corresponded to higher time per iteration, meaning that larger batch sizes take extremely long to train. Based on this, a batch size of 128 was chosen for the final model. A batch size of 256 was considered but not ultimately used because the training time would take over 10 hours.

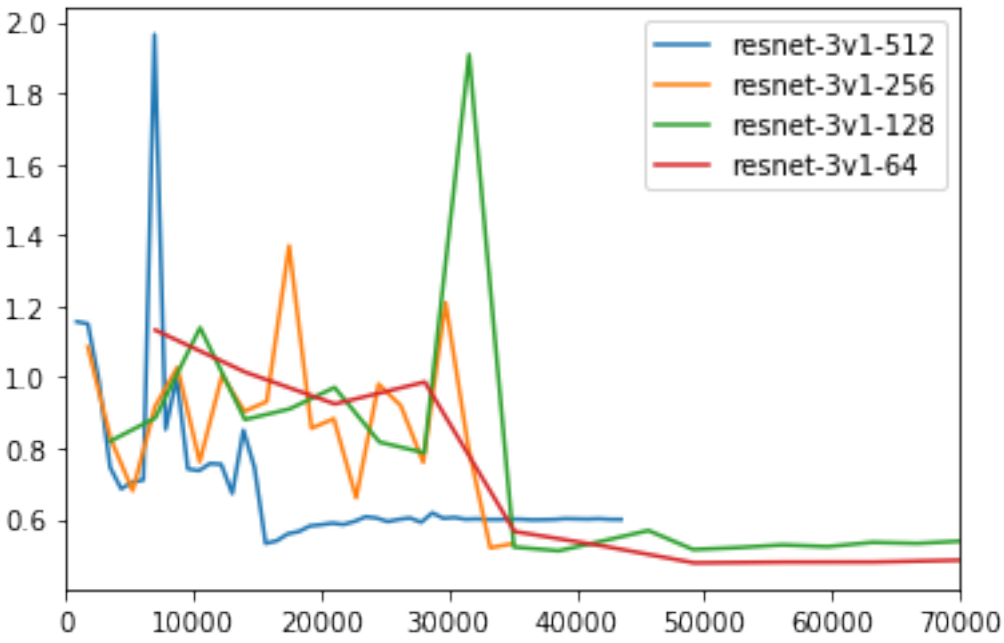

Figure 3: Plot comparing batch size on ResNet-3v1.

3 Final Result

The final model trained was ResNet v2, with n=18 for a total of 164 layers. A batch size of 64 was used, along with the learning rate schedule from [1].

Training Epochs	170
Test Loss	0.4444
Test Error (%)	**5.75**

References

[1] Deep Residual Learning for Image Recognition (https://arxiv.org/abs/1512.03385)
[2] Identity Mappings in Deep Residual Networks (https://arxiv.org/abs/1603.05027)
[3] Batch Normalization: Accelerating Deep Network Training by Reducing Internal CovariateShift (https://arxiv.org/abs/1502.03167)