Amir David Nissan Cohen 200437192
Uzi Cohen 043228311
Izik Golan 201299781

Exercise 2 Report – CIFAR10 classification

Model

We used the Network-in-Network architecture as described in https://github.com/szagoruyko/cifar.torch/blob/master/models/nin.lua, but used smaller filter depth in order to stay within the parameter size constraint. Instead of 192 filters we used 42, Instead of 160 filters we used 32, and instead of 92 we used 19. Choosing these values gave us **49947** parameters in total. One difference from the reference NiN architecture is that we used leaky ReLUs, which are less prone to the "dead relu" problem, and also cause non-sparse activation layers, which help when our layers are relatively think due the parameter constraint. We also fine-tuned the parameter of the dropout layers in the model.

Our best model achieved a score of **85.5%** accuracy on the test set.

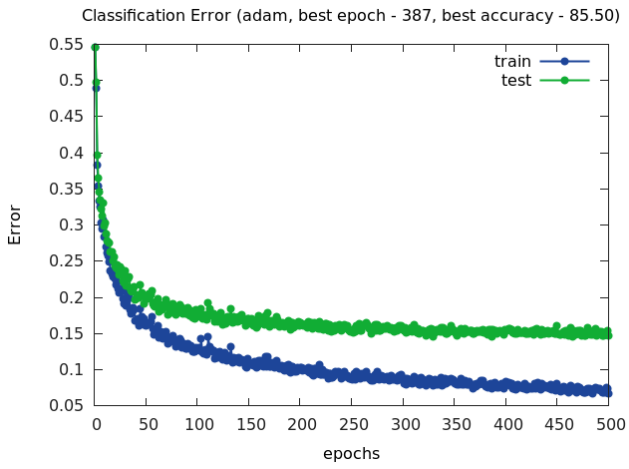
Data Augmentation

We tried 3 different techniques for data augmentation:

- *Random horizontal flip* – This was the only data augmentation technique which helped out model, and increased our test accuracy by 1%-2%. Each sample in each batch was flipped horizontally with probability 0.5.
- *Random crops* – With probability 0.5, each sample in each batch was scaled to size 36x36, and then a crop of 32x32 was taken from that resized image in a random position.
- *Random rotation* – With probability 0.5, each sample was rotated at a random angle in the range $[-\pi/8, \pi/8]$ around the center of the image.

The results from these techniques are presented here:

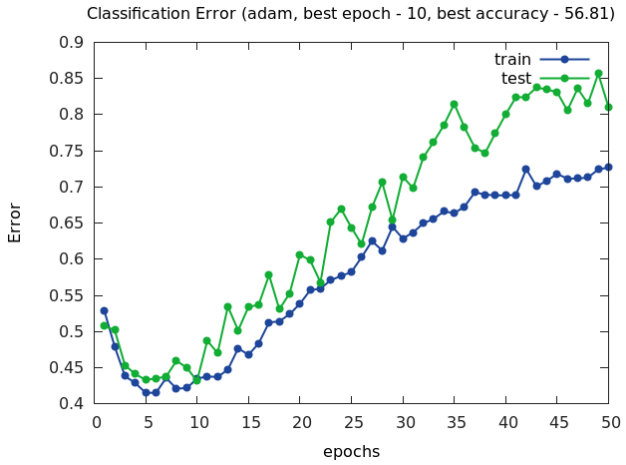
Random horizontal flips



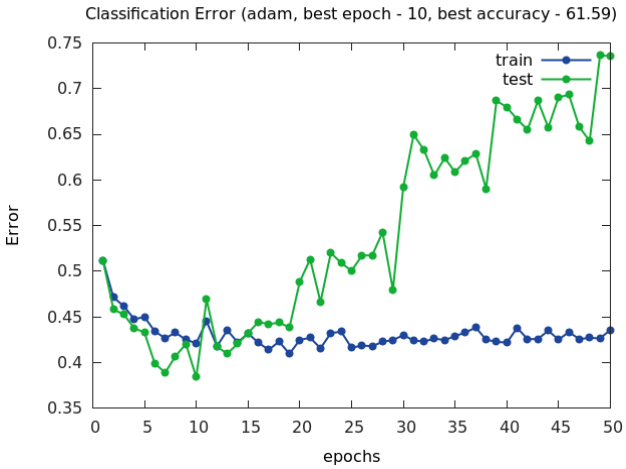
Git repo: https://github.com/izikgo/096260-DeepLearning.git

Amir David Nissan Cohen 200437192
Uzi Cohen 043228311
Izik Golan 201299781

Random crops



Random rotations



The increase in classification error is probably due the fact that there are too many different samples that we are trying to train on (the training procedure keeps getting new samples), and there are not enough parameters in our network. In other words, the model is too simple (because of the small number of parameters) to capture the full complexity of the augmented problem.

Initialization

We used The “He” initialization method for our convolutional layers, which is considered to give good results.

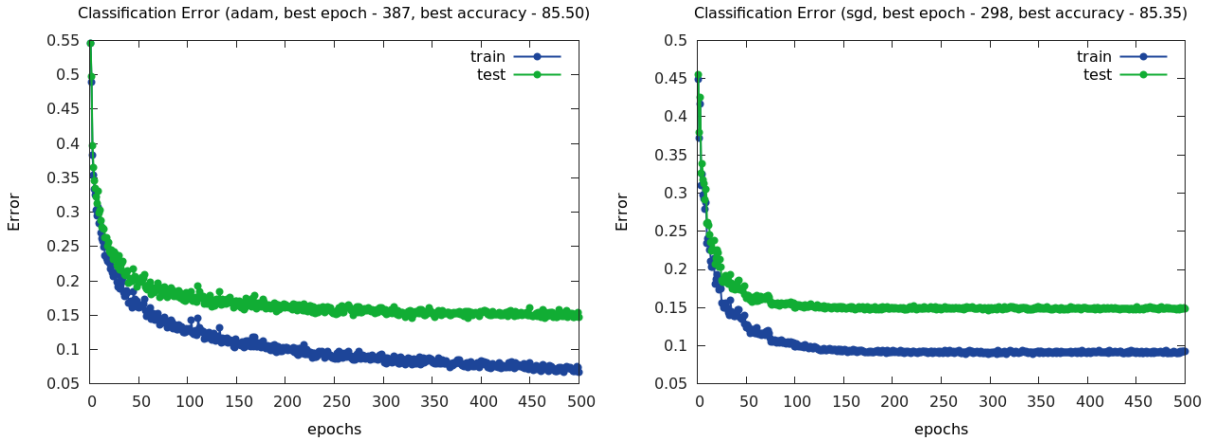
Git repo: https://github.com/izikgo/096260-DeepLearning.git

Amir David Nissan Cohen 200437192
Uzi Cohen 043228311
Izik Golan 201299781

Optimization

- We used the CrossEntropyCriterion, which is recommended for multiclass classification problems.
- We tried two different optimization method:
  - *Adam* – with all the default parameters.
  - *SGD with Nesterov momentum and decreasing learning rate* – We read that this is a good and stable optimization method, and though it didn't reach the very best result we have on the test set, it came very close while the learning was more stable.

Presented here are the plots from both methods (together with horizontal flips):



Things we considered

1. We started from the torch blog post that was mentioned in the tutorial. Our first attempt was trying to reduce the number of layers and size to fit the number of parameters requirement. This proved to give very low precision (around 60-65%)
2. We tried to add large filer layer as the first layer. The intuition was that the number of feature we extract from the image is a bottleneck for the network performance. This improved the results significantly.
3. We removed the fully connected layers to make room for more convolution layers.
4. We added dropouts for regularization.

Git repo: https://github.com/izikgo/096260-DeepLearning.git