Best Accuracy: ~40% accuracy

Methods used to improve convolutional network performance:

1. Data augmentation: fliplr, random noise, random rotation, random crop
2. Using batch normalization
3. Reduce filter size and augment convolution layers into multiply layers with smaller filters, add 2 fully connected layers at the end of the network

Motivation: The following paper proposes that replacing one conv layer with conv layers with smaller strides and filter size, the network is able to increase nonlinearity as well as decrease computational cost.

<https://arxiv.org/pdf/1409.1556.pdf>

Problem:

Overfitting. My network overfits after epoch 7, even after much data augmentation

Attempt to solve problem

1. Add more data augmentation.
2. Add convolution layer with 1\*1 filter. This is proposed in the paper my neural network is based on to add another layer of linear transformation.
3. Change learning rate, weight decay, and other parameters

Result: none of my approaches solves the overfitting issue with my network.

1 has resulted in delayed overfitting, but the error rate of the training data also increased

2 has no effect on my neural network’s performance

3: increasing learning rate results in even worse overfitting. Decreasing learning rate results in overall worse error rate. Weight decay doesn’t affect the time overfitting occurs.

Architecture:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Layer | type | input | filter | stride | pad | output |
| Layer 1.1 | conv | 32\*32\*3 | 3\*3\*3\*32 | 1 | 1 | 32\*32\*32 |
| Layer 1.2 | relu | 32\*32\*32 | Max(0,x) | 1 | 0 | 32\*32\*32 |
| Layer 2.1 | conv | 32\*32\*32 | 3\*3\*32\*32 | 1 | 1 | 32\*32\*32 |
| Layer 2.2 | relu | 32\*32\*32 | Max(0,x) | 1 | 0 | 32\*32\*32 |
| Layer 3 | pool | 32\*32\*32 | 2\*2\*32\*64 | 2 | 1 | 16\*16\*64 |
| Layer 4.1 | conv | 16\*16\*64 | 3\*3\*64\*64 | 1 | 1 | 16\*16\*64 |
| Layer 4.2 | relu | 16\*16\*64 | Max(0,x) | 1 | 0 | 16\*16\*64 |
| Layer 5.1 | conv | 16\*16\*64 | 3\*3\*64\*64 | 1 | 1 | 16\*16\*64 |
| Layer 5.2 | relu | 16\*16\*64 | Max(0,x) | 1 | 0 | 16\*16\*64 |
| Layer 6 | pool | 16\*16\*64 | 2\*2\*64\*128 | 2 | 1 | 8\*8\*128 |
| Layer 7.1 | conv | 8\*8\*128 | 3\*3\*128\*128 | 1 | 1 | 8\*8\*128 |
| Layer 7.2 | relu | 8\*8\*128 | Max(0,x) | 1 | 0 | 8\*8\*128 |
| Layer 8.1 | conv | 8\*8\*128 | 3\*3\*128\*128 | 1 | 1 | 8\*8\*128 |
| Layer 8.2 | relu | 8\*8\*128 | Max(0,x) | 1 | 0 | 8\*8\*128 |
| Layer 9 | pool | 8\*8\*128 | 2\*2\*128\*128 | 2 | 1 | 4\*4\*128 |
| Layer 10 | conv | 4\*4\*128 | 4\*4\*128\*128 | 1 | 0 | 1\*1\*128 |
| Layer 11 | conv | 1\*1\*128 | 1\*1\*128\*100 | 1 | 0 | 1\*1\*100 |
| Layer 12 | softmaxloss | 1\*1\*100 |  | 1 | 0 | 1\*100 |