Assignment 4

Mukul Sati [msati3@gatech.edu]

April 29, 2016

I used the CAFFE framework [1] for my implementation.

1 MNIST Dataset

I trained two CNN architectures on the MNIST dataset (Fig. 1). For the first architecture, starting with a baseline training loss of 2.37 and a baseline test accuracy of 0.089, I obtain a final accuracy of 99.01%. Getting good numbers on the first architecture, I wanted to simplify the second architecture. Thus, the second architecture is essentially the first architecture with lesser number of weights being learnt, one max pooling layer removed, and with the addition of a dropout layer. I train this architecture for a greater number of iterations (40,000, in batches of 64, as opposed to 10,000 iterations with the same batch size for the first architecture). For the second architecture, starting with a baseline training loss of 2.31 and a baseline test accuracy of 0.097, I obtain a final accuracy of 99.03%. The training progress is shown in Fig. 2. As I did not manually tweak hyper-parameters, I feel it was sufficient have directly using the provided test set during the testing phase instead of performing hold out validation. The kernels learned from the first and second convolutional layers are visualized in Fig. 3. The confusion matrices of the two architectures are shown in Fig. 4, and a few incorrectly classified images are shown in Fig. 5.

The following are the gradient descent equations:

2 Sunset Dataset

I used the CaffeNet pre-trained network that comes with CAFFE. This is pretty similar to AlexNet, but without the relighting data-augmentation and has a difference in the order of the pooling and normalization layers. The primary tweaks I made to CaffeNet for getting a start on the Sunset dataset are:

1. Editing the last fully connected layer for the binary classification problem at hand. The initial architecture was for the 1000 class ImageNet database.

2. Lowering the learning rate of the solver, while using a higher multiplier for the weights of the modified layer.

Using CaffNet, I arrive to an accuracy of 93.3% (Fig. 6), starting with a baseline accuracy of 46.67%. Some incorrectly classified images are also shown in Fig 6

I use the following schemes for data-augmentation: I increase the

References

[1] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

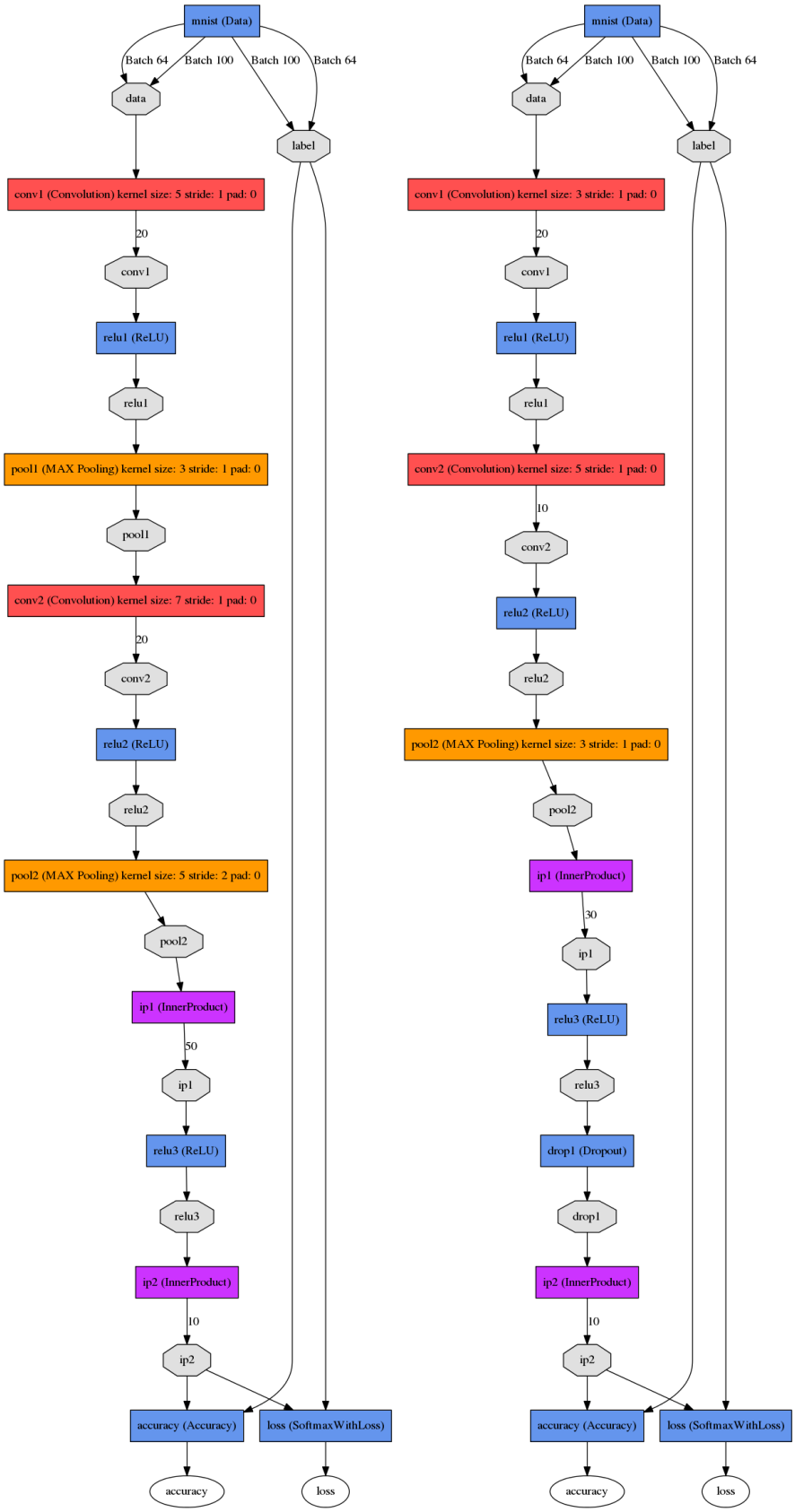

Figure 1: The two architectures trained for the MNIST dataset.

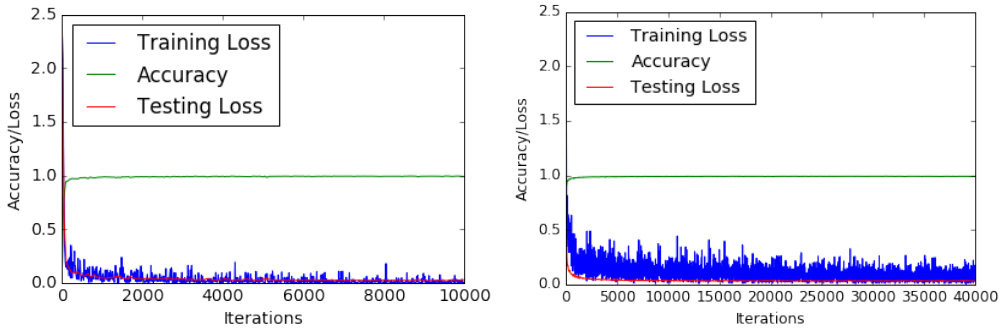

Figure 2: Training loss and testing loss and accuracy versus iterations for architecture 1 (left) and architecture 2 (right).

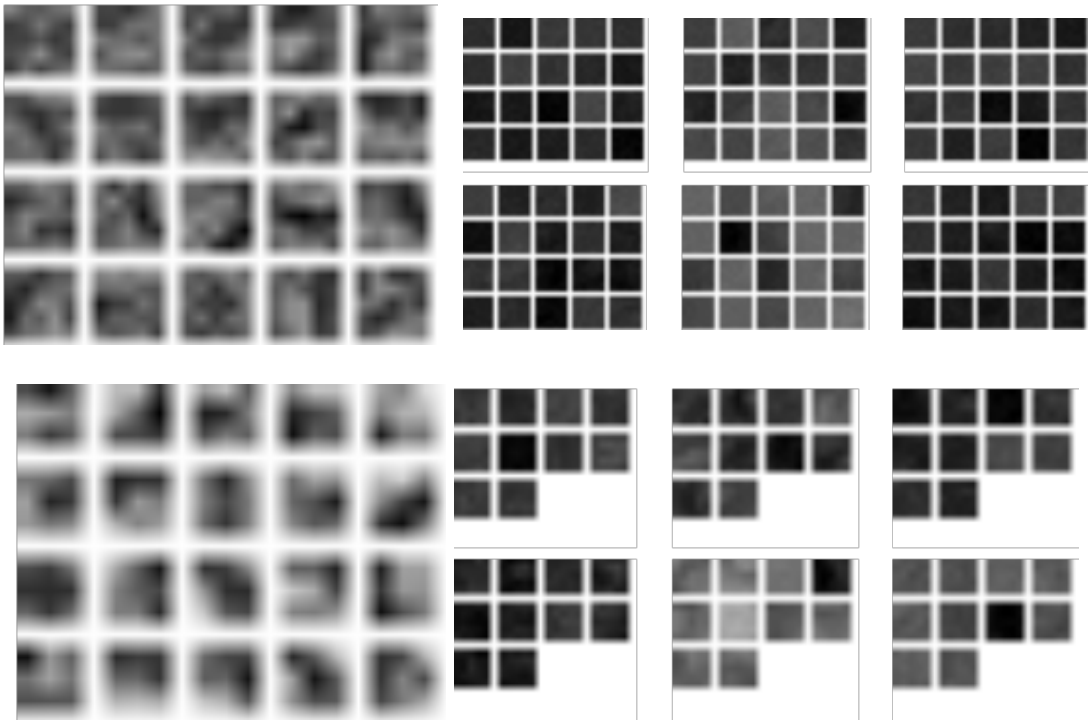

Figure 3: The learned filters for the first and some of the learned filters for the second convolutional layers of the first (top row) and second (bottom row) architectures. Global contrast normalization is performed for the second layer filters for improved visualization.

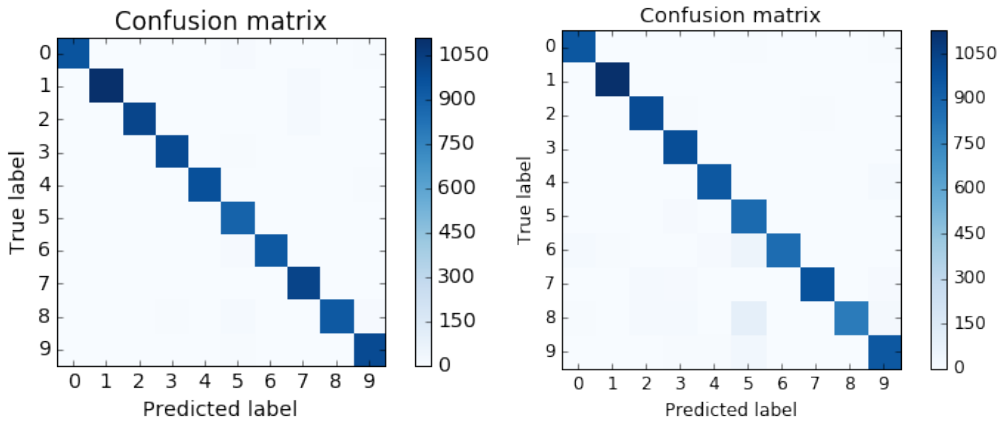

Figure 4: Confusion matrix for the first (left) and second (right) architectures.

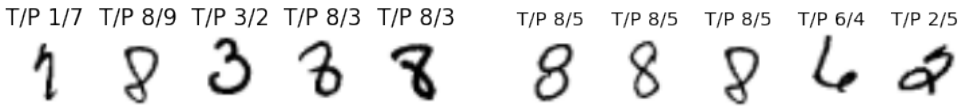

Figure 5: Some incorrectly classified test examples for the first (left) and second (right) architectures, showing also the true (T) and predicted (Pr) labels.

Figure 6: Left: Training loss and testing loss and accuracy versus iterations for the vanilla CaffeNet described above. Right: Some misclassified images with this network.