Deep Neural Networks

Assignment 10

Assignment due by: 17.07.2018, Discussions on: 24.07.2017

Question 1 MNIST with Convolutions (8 points)

In this exercise we return once more to the MNIST dataset, but this time we will be using convolutions to try to improve the accuracy above what we were able to achieve with just fully connected layers.

Since convolutions are nontrivial to implement efficiently, we will again be using tensorflow for this assignment.

Your task is to implement a neural network comprised of convolutions that surpasses the best accuracy of $\sim 97\%$ for our previous networks which used only fully connected layers. We suggest the following architecture, but you are free to experiment to see whether you can achieve a better accuracy (don't just copy an architecture you found on the internet).

- Convolution: 5×5, 16 features, ReLU
- Max Pooling: 2×2, stride 2
- Convolution: 3×3, 32 features, ReLU
- Max Pooling: 2×2, stride 2
- Convolution: 3×3, 64 features, ReLU
- Global Average Pooling
- Fully Connected: 10 outputs

Notes: Since this model is larger, it will take longer to train than previous models. However, your model should take no more than 30 minutes to train on a CPU (or 5 minutes on a GPU). To avoid problems due to initialization of the weights, you should use the `tf.get_variable` with the default initialization. Moreover it is helpful to reduce the learning rate after about half of your training steps (look into `tf.train.piecewise_constant`). Lastly, `tf.MomentumOptimizer` with a momentum parameter of 0.9 tends to work very well for optimizing this network.

Your code should output training and test errors every 500 training steps to monitor how training is progressing.

Question 2 Backpropagation through a convolution (6 points)

The formula to calculate a one-channel 2D convolution is:

$$\mathsf{Z}_{ij} = \sum_{k,l} \mathsf{V}_{i+k,j+l}\mathsf{K}_{k,l}$$

where $\mathsf{V}_{i,j}$ is the input image and $\mathsf{K}_{k,l}$ is the kernel. Assuming that this convolution appears somewhere in a neural network and given the partial derivative $\frac{\partial \mathcal{L}}{\partial \mathsf{Z}_{ij}}$ calculate the partial derivatives:

$$\frac{\partial \mathcal{L}}{\partial \mathsf{K}_{mn}} \text{ and } \frac{\partial \mathcal{L}}{\partial \mathsf{V}_{mn}}$$

Comment on the resulting formulae.

Hint: You don't need to worry about edge effects (padding etc.) for this question. As a first step you should calculate the partial derivatives $\frac{\partial \mathsf{Z}_{ij}}{\partial \mathsf{K}_{mn}}$ and $\frac{\partial \mathsf{Z}_{ij}}{\partial \mathsf{V}_{mn}}$. Also, for the Kronecker delta: $\delta_{a,b} = \delta_{a-b,0}$

Question 3 Network size and parameters (6 Points)

Suppose the input to a convolution layer has a spatial size 21×21 with 12 channels. We want to convolve this input with a 7×7 kernel with 36 output channels.

(a) What is the spatial size of the output if the convolutions are applied with a stride of 1 and no padding?

(b) How many activations do we have in the output layer?

(c) How many parameters are there in the convolutional weights?

(d) How many weight parameters would there be if we wanted to have an output layer of the same size as computed above, but which was fully connected to the input layer?

(e) How much padding should we use if we want the spatial size of the output to be the same as that of the input (assuming a stride of 1)?

(f) If instead of applying the filters at each location (stride=1), we have a stride of 2, what would be the spatial size of the output layer (assume no padding)?