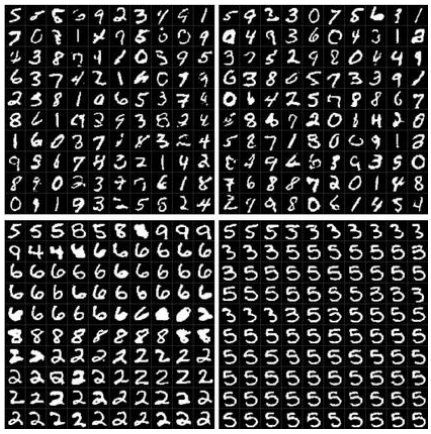
MSSE277b: Machine Learning
Homework assignment #9: Convolutional Neural Networks
Assigned Apr. 3 and Due Apr. 10



1. Convolutional Neural Networks applied to classification. We will again use the MNIST data set to train, validation, and test but this time using a CNN. As described in lecture, 2D convolutional neural nets are specified by various hyperparameters: a receptive field (filter size, DxWxH), number of filters K, stride S, amount of zero padding P, and type of pooling. We will represent our input data, as well as the hidden layers, as 3D-arrays. Since MNIST images are black-and-white and thus have scalar-valued pixels, the depth of the input image is 1.

(a) (4.5pt) Calculate the dimensionality of the output for the following convolutions **sequentially** applied to a black and white MNIST input:
(i). Convolution Filter size of 2x2, number of filters 33, stride of 2, padding of 0
(ii). Convolution Filter size of 3x3, number of filters 55, stride of 1, padding of 1
(iii). Convolution Filter size of 3x3, number of filters 77, stride of 1, padding of 1. Followed by a Max Pooling with filter size of 2x2 and stride 2.
(b) (4.5pt) The MNIST data set was, in fact, in color (RGB). This means the depth of the input image would be 3. Calculate the dimensionality of the output for the following convolutions **sequentially** applied to a RGB MNIST input:
(i). Convolution Filter size of 2x2, number of filters 33, stride of 2, padding of 0
(ii). Convolution Filter size of 3x3, number of filters 55, stride of 1, padding of 1. Followed by a max pooling layer of kernel size 3x3, stride of 1, padding of 0
(iii). Convolution Filter size of 3x3, number of filters 77, stride of 1, padding of 1. Followed by a Max Pooling with filter size of 2x2 and stride 2.
(c) (5pt) Next, implement a CNN to see if we can extract additional features from the MNIST data. For this start with one convolutional layer with a 5x5 kernel, with stride of 1, zero-padding of size 2, and 3 output channels. Flatten the resulting feature maps and add a second layer of fully connected (FC) layer to the 10-neuron output layer. Use ReLU as your activation function. Use the ADAM optimizer with learning rate of 1e-3, batchsize of 128, and 30 epochs (you can also train for longer if time permits). Use mini-batches of data and converge your training to where the loss function is minimal, and choose some regularization techniques. Using 3-fold cross-validation and report your average test accuracy.
(d) (6pt) Now build a deeper (more layers) architecture with two layers each composed of one convolution and one pooling layer. Flatten the resulting feature maps and use two fully connected (FC) layers. Use conv/pooling layers that with kernel, stride and padding size of your choice. Use ReLU as your activation function. Again, use the ADAM optimizer with learning rate of 1e-3, batchsize of 128, and 30 epochs (you can also train for longer if time permits). Use mini-batches of data and converge your training to where the loss function is minimal, and choose some regularization techniques. Using 3-fold cross-validation report and your average test accuracy. You should aim for getting test accuracy above 98.5%.