

Assignment – Playing around with CNNs

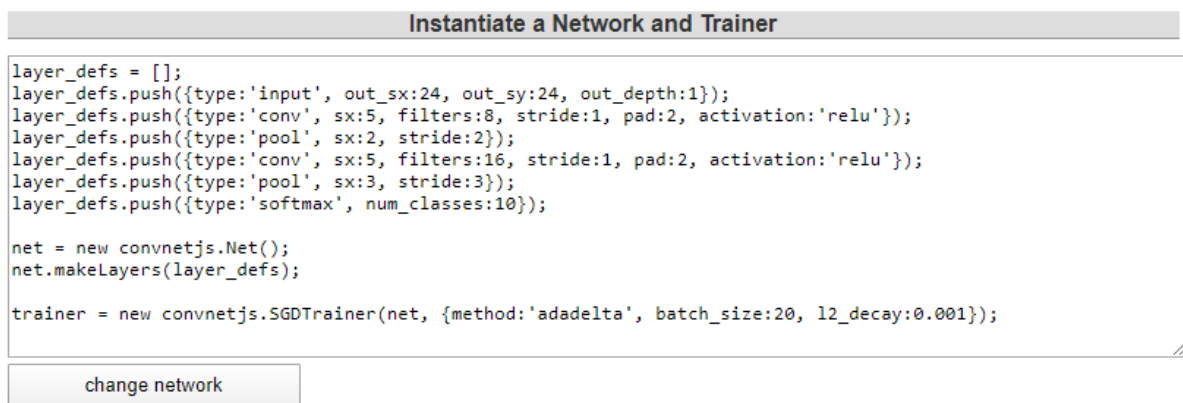
Mattias Borg, 2020-03-24

This assignment will give you a feeling for how to design a good convolutional neural network for image recognition tasks. You will not need much programming skills for this assignment as it will use an online interface that allows you to “magically” build and test a convolutional network.

Go to <https://cs.stanford.edu/people/karpathy/convnetjs/demo/mnist.html>

As soon as you load the page it will start training the default network.

First look closely at the following window:



```
layer_defs = [];  
layer_defs.push({type:'input', out_sx:24, out_sy:24, out_depth:1});  
layer_defs.push({type:'conv', sx:5, filters:8, stride:1, pad:2, activation:'relu'});  
layer_defs.push({type:'pool', sx:2, stride:2});  
layer_defs.push({type:'conv', sx:5, filters:16, stride:1, pad:2, activation:'relu'});  
layer_defs.push({type:'pool', sx:3, stride:3});  
layer_defs.push({type:'softmax', num_classes:10});  
  
net = new convnetjs.Net();  
net.makeLayers(layer_defs);  
  
trainer = new convnetjs.SGDTrainer(net, {method:'adadelta', batch_size:20, l2_decay:0.001});
```

change network

Here the definition of the layers is done, basically it pushes layer descriptions into a list (layer_defs). Also towards the end the network is built, and training is commenced on the last row. Don't mind these last rows, focus on the layer definition rows.

The second row for instance, instantiates the input layer:

```
layer_defs.push({type:'input', out_sx:24, out_sy:24, out_depth:1});
```

The “type” parameter here defines that this is meant to be an input layer. Other definitions that you see for the other layers are ‘conv’, ‘pool’ and ‘softmax’ which corresponds to convolutional layers, pooling layers and a fully connected softmax layer.

The other parameters for the input layer simply decides the size and depth of the images, you can leave these as is.

The second layer is more interesting:

```
layer_defs.push({type:'conv', sx:5, sy:5, filters:8, stride:1, pad:2, activation:'relu'});
```

This convolutional layer has 8 filters (giving eight equivalent feature maps), in each every neuron is connecting to (sx,sy) 5x5 input neurons (5x5 kernel). The stride (i.e. the step that the convolutional kernel takes between each neuron in the filter) is 1. ‘Pad:2’ means that around the input data set it pads with two additional ‘virtual’ input neurons so that the kernel can move its central “pixel” all the way to the edge of the image. With a 24x24 image size, 5x5 kernel, stride=1 and pad=2 this means that each feature map gets the same 24x24 size as the input. The activation function for the layer is defined as ReLU.

The third layer, which is a pooling layer is defined by:

```
layer_defs.push({type:'pool',sx:2, stride:2});
```

sx:2, stride:2 means that it takes a 2x2 kernel and calculates the maximum value of the kernel for each neuron in this layer. Thus, the size of this layer will be only ¼ of the previous layer, i.e. 12x12 neurons.

Then another convolutional layer is defined, this time with 16 filters, 5x5 kernel and same padding, leading to another 12x12 feature map size.

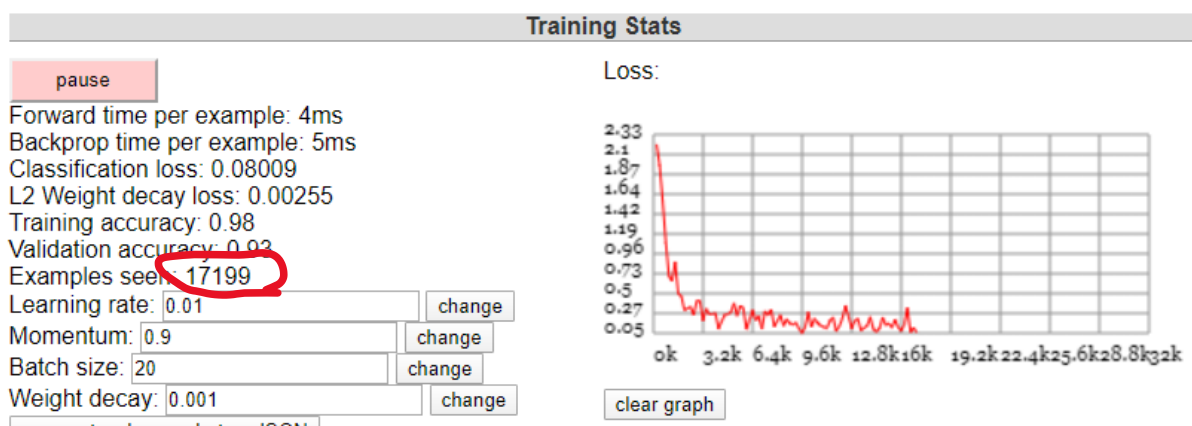
After another pooling layer the network is finished off with a fully connected layer, with softmax activation. 'num_classes' defines the number of classes of the output, which should be 10 for this data set.

Further down on the page, you can see a graphical representation of the layers, and the data as it flows through the layers. You can use these views to evaluate how the network relates to the abstract features of the data.

To make changes to the network, just change the lines in the "Instantiate a Network and Trainer" window, press "change network", and then restart the training in the "Training Stats" window.

Now to the tasks:

1. Let's start simple, just wait until the default network has seen 30000 examples:



- Now note down what the training and validation accuracy was at this point. Training accuracy means how well the network classified on the training data set. Validation accuracy means how well it performed on the validation data set. Look at the graph of the Loss (cost function), what can you say about the shape of this curve? Is the network still learning at a fast rate, or has it saturated? What can you say about the gradient of the cost function?
- Now alter the size of the convolutional kernel in the first and third layers, restart the training and let it again run to 30000 examples (actually use this for all tasks). What is the effect of doing that?
- Alter the stride in a corresponding way. What is the effect?
- Alter the number of filters in the first convolutional layer. What is the effect?

**2. Let's now play around with the number of convolutional/pooling layers.
(the default is two pairs)**

- A) Before changing anything, observe the features that the **feature maps** learn. Compare the first to the last convolutional layer, do they learn the same features? Why or why not?



- B) Let's add another pair of layers. You may have to change the pooling layer s_x and stride so that they don't decrease the data size as fast. What happens to the learning accuracy after 30000 examples? Does the computational time increase or decrease?
- C) Instead now remove all but the first convolutional / pooling layer. What happens to the learning accuracy and so on?
- 3. Try to find the best possible network structure!**
- A) Play around freely and try to optimize the network. Describe your method and thought process, and of course what network you end up with.