**Unit Saturation in Neural Networks**

A neural network classifier has a set of input nodes, a set of hidden processing nodes, and a set of output nodes. The hidden nodes usually have values between -1.0 and +1.0 if you use the tanh activation function. The output nodes will have values between 0.0 and 1.0 if you use the softmax activation function.

A saturated neural network is one where most of the hidden nodes have values close to -1.0 or +1.0 and the output nodes have values close to 0.0 or 1.0. Saturation is not a good thing. If hidden nodes are saturated then that means their pre-activation sum-of-products is relatively large (typically greater than 4.0) or small (typically smaller than -4.0).

Saturated nodes lead to a situation where a small change in the input-to-hidden weights during training will likely not change the sum-of-products very much, and then after activation, the node value will still be -1.0 or +1.0 — in other words, training stalls out or moves very slowly. Additionally, saturated models are often overfitted — meaning the model predicts well on training data but poorly on new, unseen data.

You’d expect that saturation in the output nodes would be a good thing. For example, if you’re trying to classify the Iris dataset, there are three species to predict, so the target output node values are (1,0,0) for setosa, (0,1,0) for versicolor, (0,0,1) for virginica. So saturated output node values, for example (0.0000, 1.0000, 0.0000), would seem to be desirable, indicating a high confidence that the predicted species is versicolor. However, this is only partially true because saturated output nodes could just mean that training stalled at a non-optimal set of weights and biases.

So, when creating a neural network model, you should keep an eye saturation as well as model error (typically squared error or cross entropy error) on the training and test data, and on model accuracy on the data.

There are several ways to monitor NN saturation. The simplest is to just display and eyeball the values of the hidden and output nodes. This is OK for very small networks but with large networks, you probably need a metric of some sort. Note that saturation of a NN will be different for every data item, making it quite difficult to examine visually.

The technique I usually use to measure saturation is to count the number of nodes that have extreme values — typically greater than +0.95 or less than -0.95. With that count, I can return an average of some sort, such as the average (over all data items) percentage of extreme/saturated nodes. I usually look at hidden nodes and output nodes separately, but sometimes I examine the two node types together.

For example, suppose I have just three data items. And suppose my neural network has four hidden nodes.

data[0] has 3 saturated nodes, 1 unsaturated

data[1] has 0 saturated nodes, 4 unsaturated

data[2] has 2 saturated nods, 2 unsaturated

total saturated nodes = 5

num items \* num nodes = 3 \* 4 = 12

saturation metric = 5 / 12 = 0.4167

Saturation metric values closer to 0 indicate (good) unsaturated models. Metric values closer to 1 indicate (bad) saturated models.