Backward Query

In this lesson, we will learn what is Backward Query and how does it work?

We already know how to propagate signals through a network, moderating them with link weights, and recombining them at nodes before applying an activation function. All this works for signals flowing backward too, except that the inverse activation is used. If $y = f(x)$ was the forward activation then the inverse is $x = g(y)$. This isn't that hard to work out for the logistic function using simple algebra:

$$y = 1/(1 + e^{-x})$$

$$1 + e^{-x} = 1/y$$

$$e^{-x} = (1/y) - 1 = (1 - y)/y$$

$$-x = ln[(1 - y)/y]$$

$$x = ln[y/(1 - y)]$$

This is called the *logit* function, and the Python `scipy.special` library provides this function as `scipy.special.logit()`, just like it provides `scipy.special.expit()` for the logistic sigmoid function.

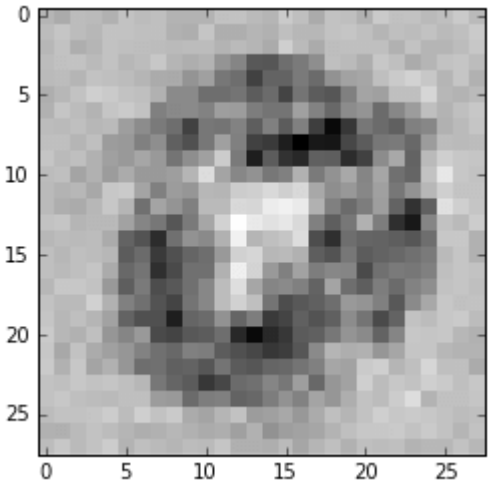
Before applying the `logit()` inverse activation function, we need to make sure the signals are valid. What does this mean? Well, you remember the logistic *sigmoid* function takes any value and outputs a value somewhere between 0 and 1, but not including 0 and 1 themselves. The inverse function must take values from the same range, somewhere between 0 and 1, excluding 0 and 1, and pop out a value that could be any positive or negative number. To achieve this, we simply take all the values at a layer about to have the `logit()` applied, and rescale them to the valid range. I've chosen the range 0.01 to 0.99. The code is always available online at GitHub at the following link:

- https://github.com/makeyourownneuralnetwork/makeyourownneuralnetwork/blob/master/part3_neural_network_mnist_backquery.ipynb

The Label "0"

Let's see what happens if we do a back query with the label 0. That is, we present values to the output nodes which are all 0.01 except for the first node representing the label 0 where we use the value 0.99. In other words, the array $[0.99, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01]$.

The following shows the image that pops out of the input nodes.



That is interesting! That image is a privileged insight into the mind of a neural network. What does it mean? How do we interpret it? The main thing we notice is that there is a round shape in the image. That makes sense because we're asking the neural network what the ideal question is for an answer to be 0. We also notice dark, light and some medium grey areas:

- The *dark* areas are the parts of the question image which, if marked by a pen, make up supporting evidence that the answer should be a 0. These make sense as they seem to form the outline of a zero shape.
- The *light* areas are the bits of the question image which should remain clear of any pen marks to support the case that the answer is a 0. Again these make sense as they form the middle part of a zero shape.
- The neural network is broadly indifferent to the *grey* areas.

So we have actually understood, in a rough sense, what the neural network has learned about classifying images with the label 0. That's a rare insight into

more complex networks with more layers, or more complex problems, may not have such readily interpretable results. You're encouraged to experiment and gave a go yourself!