Apr 1990 Gradient Descent – Error Surface Estimation

Neil E Cotter

When ~~In~~ learning ~~[illegible]~~ with neural networks, we want to minimize average error over all possible input patterns. We define the true average error to be

$$\mathcal{E} = \frac{1}{A} \int_{\substack{\text{input} \\ \text{patterns} \\ p}} \frac{(t_p - y_p)^2}{2} \, dp$$

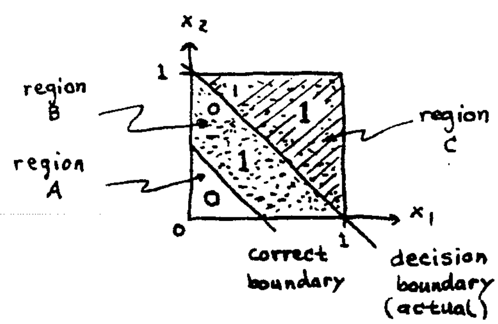
where $A \equiv$ area of input pattern space

$p \equiv$ particular input pattern

$t_p \equiv$ desired output for pattern p

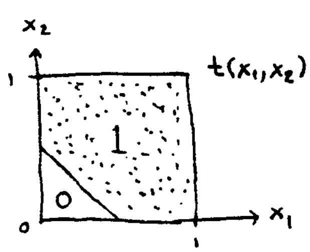
$y_p \equiv$ actual " " " "

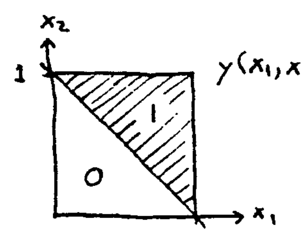
ex:



Perceptron with incorrect decision boundary. Error in dotted, unhatched region, (B).

$$\mathcal{E} = \frac{1}{[0,1] \times [0,1]} \int_0^1 \int_0^1 \frac{[t(x_1,x_2) - y(x_1,x_2)]^2}{2} \, dx_1 dx_2$$





$$\therefore \; \mathcal{E} = \frac{1}{1} \left[\iint_A \frac{(0-0)^2}{2} \, dx_1 dx_2 + \iint_B \frac{(1-0)^2}{2} \, dx_1 dx_2 + \iint_C \frac{(1-1)^2}{2} \, dx \right.$$

$$= \iint_B \frac{1}{2} \, dx_1 dx_2 = \frac{1}{2} \cdot \frac{3}{8} = \frac{3}{16}$$

Neil E Cotter

In most neural network applications, we do <u>not</u> have access to $\mathcal{E}$. This is because $\mathcal{E}$ is really $\mathcal{E}(\vec{w})$ and changes value each time we change $\vec{w}$. To find the new value of $\mathcal{E}$ after each change in $\vec{w}$, we would have to present <u>every</u> input pattern to the network and measure the average output error. This approach is impractical.

Instead, we estimate $\mathcal{E}(\vec{w})$ in a crude way:

$$\mathcal{E}(\vec{w}) \approx E(\vec{x}) \qquad \text{where} \qquad E(\vec{x}) = \tfrac{1}{2}(t-y)^2$$

$t \equiv$ target output for present input, $\vec{x}$.
$y \equiv$ actual " " "

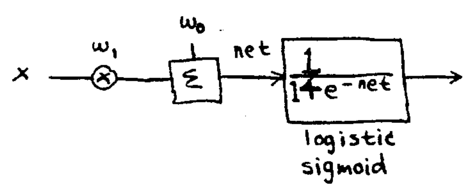
In other words, we use the error, $E(\vec{x})$, for a <u>single</u> input pattern, $\vec{x}$, to estimate the entire error surface. This works fairly well, provided we pick values of $\vec{x}$ at random as gradient descent proceeds.

Note: error acts like no[ise] that helps [us] avoid local min

(or $\nabla_{\vec{w}} E(\vec{x})$)

We use $E(\vec{x})$ because we can easily calculate $\frac{\partial E(\vec{x})}{\partial \vec{w}}$, the gradient of E.

ex: $t = x^2$
$x \in [0,1]$



$$\mathcal{E}(\vec{w}) = \frac{1}{1}\int_0^1 \frac{[t(x)-y(x)]^2}{2}\,dx = \int_0^1 \frac{\left[x^2 - \frac{1}{1+e^{-w_0-w_1 x}}\right]^2}{2}\,dx$$

Very hard to compute

$$E(\vec{x}) = \frac{\left[x^2 - \frac{1}{1+e^{-w_0-w_1 x}}\right]^2}{2}. \qquad \frac{\partial E(\vec{x})}{\partial w_1} = -\overbrace{f(x)[1-f(x)]}^{f'(x)}\overbrace{[x^2-f(x)]}^{t-y}\,w_1$$

Easy comp[ute]

where $f(x) = \dfrac{1}{1+e^{-w_0-w_1 x}}$.