Neural Networks

NN-4 Backpropagation Training

Petr Musilek

University of Alberta

Outline

Summary of the backpropagation algorithm

The correction $\Delta w_{l_j}(k)$ applied to the synaptic weight connecting neuron i to neuron j is given by the delta rule

$$\Delta w_{ji}(k) = \begin{cases} \alpha e_j(k)\phi'_j\big(v_j(k)\big)y_i(k) & \text{if } j \text{ is an output node} \\ \alpha\phi'_j(v_j(k))\sum_l \left(e_l\phi'_j(v_l(k))w_{lj}(k)\right)y_i(k) & \text{if } j \text{ is a hidden node} \end{cases}$$

The computation of $\Delta w_{l_j}(k)$ requires knowledge of the derivative of the activation function $\phi(\cdot)$

- the function $\phi(\cdot)$ must be differentiable (the only requirement)
- for the derivative exists, the function must be continuous
- two representative instances
  - logistic function
  - hyperbolic tangent function

In-class exercise: Backpropagation algorithm

[Adapted from: www.anotsorandomwalk.com/backpropagation-example-with-numbers-step-by-step/]

This is a detailed example of one iteration of the backpropagation algorithm.

High level steps:

(1) Initialize weights and biases

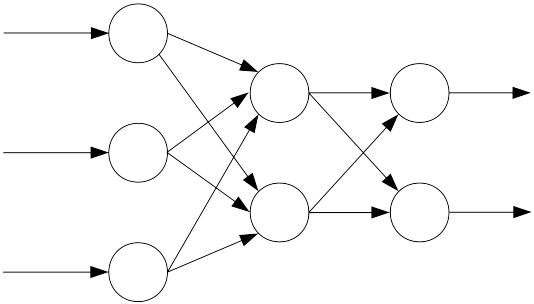
(2) Forward propagate a pattern through the network to get the output values

(3) Determine the network error

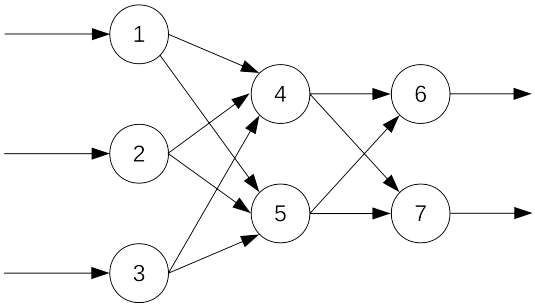
(4) Backpropagate through the network to determine the error corrections

(5) Update the network parameters

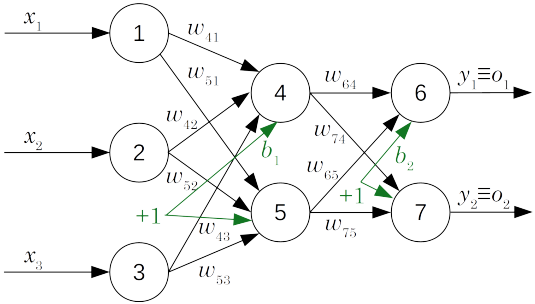
The input and target values for this problem are $\boldsymbol{x} = [1, 4, 5]^{\mathsf{T}}$ and $\boldsymbol{t} = [0.1, 0.05]^{\mathsf{T}}$. This problem requires the following network (3 input and 2 output nodes; # of hidden nodes arbitrary/by experience).



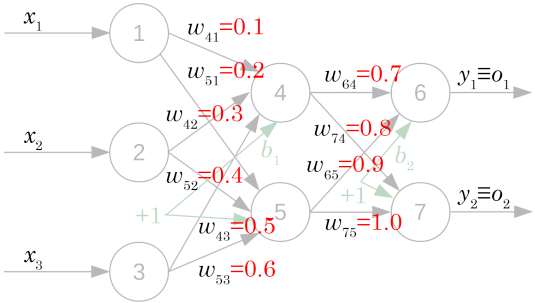
Establish a simple numbering scheme (this works in this small example, not practical for larger networks that would use upper indices to distinguish between layers.)



Add notation for inputs/outputs/weights/biases (biases are joint for individual layers - this is not very common, but certainly a possibility)



Step (1) Initialize values of weights and biases (normally the weights would be assigned randomly, but here we use these arbitrary values for ease of calculations and better visibility)



Step (2) Notation: v_4, v_5, v_6, and v_7 denote the value before the activation function, while y_4, y_5, y_6, and y_7 denote the values after application of the activation function

Mathematically, there are the following relationships between nodes in the networks

Input to hidden layer

$$y_4 = \phi(v_4);\ v_4 = w_{41} \cdot x_1 + w_{42} \cdot x_2 + w_{43} \cdot x_3 + b_1$$

$$y_5 = \phi(v_5);\ v_5 = w_{51} \cdot x_1 + w_{52} \cdot x_2 + w_{53} \cdot x_3 + b_1$$

Hidden layer to output layer

$$y_6 = \phi(v_6);\ v_6 = w_{64} \cdot y_4 + w_{65} \cdot y_5 + b_2$$

$$y_7 = \phi(v_7);\ v_7 = w_{74} \cdot y_4 + w_{75} \cdot y_5 + b_2$$

The formulas above can be used to propagate forward through the network

$$v_4 = w_{41} \cdot x_1 + w_{42} \cdot x_2 + w_{43} \cdot x_3 + b_1 = 0.1 \cdot (1) + 0.3 \cdot (4) + 0.5 \cdot (5) + 0.5 = 4.3$$

$$y_4 = \phi(v_4) = \phi(4.3) = 0.9866$$

$$v_5 = w_{51} \cdot x_1 + w_{52} \cdot x_2 + w_{53} \cdot x_3 + b_1 = 0.2 \cdot (1) + 0.4 \cdot (4) + 0.6 \cdot (5) + 0.5 = 5.3$$

$$y_5 = \phi(v_5) = \phi(5.3) = 0.9950$$

$$v_6 = w_{64} \cdot y_4 + w_{65} \cdot y_5 + b_2 = 0.7 \cdot (0.9866) + 0.9 \cdot (0.9950) + 0.5 = 2.0862$$

$$y_6 = \phi(v_6) = \phi(2.0862) = 0.8896$$

$$v_7 = w_{74} \cdot y_4 + w_{75} \cdot y_5 + b_2 = 0.8 \cdot (0.9866) + 0.1 \cdot (0.9950) + 0.5 = 1.3888$$

$$y_7 = \phi(v_7) = \phi(1.3888) = 0.8004$$

Step (3) Define the sum of squares error

$$E = \frac{1}{2}\left[\left(\frac{\partial E}{d_{y6}}\right)^2 + \left(\frac{\partial E}{d_{y7}}\right)^2\right] = 0.5933$$

$$\frac{\partial E}{d_{y6}} = y_6 - t_1 = 0.8896 - 0.1 = 0.7896$$

$$\frac{\partial E}{d_{y7}} = y_7 - t_2 = 0.8004 - 0.05 = 0.750$$

Step (4) and backpropagate it through the network to compute all the error derivatives with respect to the parameters (a lot of formulas).

- The derivative of the sigmoid function is $\frac{\partial \phi}{dx} = \phi(x)(1 - \phi(x))$.
- Given that $v_6 = w_{64}y_4 + w_{65}y_5 + b_2$ and $v_7 = w_{74}y_4 + w_{75}y_5 + b_2$, we have $\frac{\partial v_6}{\partial w_{64}} = y_4$, $\frac{\partial v_7}{\partial w_{74}} = y_4$, $\frac{\partial v_6}{\partial w_{65}} = y_5$, $\frac{\partial v_7}{\partial w_{75}} = y_5$, $\frac{\partial v_6}{\partial b_2} = 1$, and $\frac{\partial v_7}{\partial b_2} = 1$.
- Now we can calculate $\frac{\partial E}{\partial w_{64}}$, $\frac{\partial E}{\partial w_{74}}$, $\frac{\partial E}{\partial w_{65}}$, and $\frac{\partial E}{\partial w_{75}}$

$$\frac{\partial E}{\partial w_{64}} = \frac{\partial E}{\partial y_6}\frac{\partial y_6}{\partial v_6}\frac{\partial v_6}{\partial w_{64}} = \frac{\partial E}{\partial w_{64}} = (y_6 - t_1)(y_6(1 - y_6))y_4$$
$$\frac{\partial E}{\partial w_{64}} = (0.8896 - 0.1)(0.8896(1 - 0.8896))(0.9866) = 0.0765$$

The details on the next three computations are very similar to the one above.

$$\frac{\partial E}{\partial w_{74}} = \frac{\partial E}{\partial y_7}\frac{\partial y_7}{\partial v_7}\frac{\partial v_7}{\partial w_{74}} = (0.7504)(0.1598)(0.9866) = 0.1183$$

$$\frac{\partial E}{\partial w_{65}} = \frac{\partial E}{\partial y_6}\frac{\partial y_6}{\partial v_6}\frac{\partial v_6}{\partial w_{65}} = (0.7896)(0.0983)(0.9950) = 0.0772$$

$$\frac{\partial E}{\partial w_{75}} = \frac{\partial E}{\partial y_7}\frac{\partial y_7}{\partial v_7}\frac{\partial v_7}{\partial w_{75}} = (0.7504)(0.1598)(0.9950) = 0.1193$$

The error derivative of b_2 is a little bit more involved since changes to b_2 affect the error through both y_6 and y_7.

$$\frac{\partial E}{\partial b_2} = \frac{\partial E}{\partial y_6}\frac{\partial y_6}{\partial v_6}\frac{\partial v_6}{\partial b_2} + \frac{\partial E}{\partial y_7}\frac{\partial y_7}{\partial v_7}\frac{\partial v_7}{\partial b_2}$$

$$\frac{\partial E}{\partial b_2} = (0.7896)(0.0983)(1) + (0.7504)(0.1598)(1) = 0.1975$$

To summarize, we have computed numerical values for the error derivatives with respect to $w_{64}, w_{74}, w_{65}, w_{75}$, and b_2.

Now backpropagate one layer to compute the error derivatives of the parameters connecting the input layer to the hidden layer: $\frac{\partial E}{\partial w_{41}}, \frac{\partial E}{\partial w_{51}}, \frac{\partial E}{\partial w_{42}}, \frac{\partial E}{\partial w_{52}}, \frac{\partial E}{\partial w_{43}}, \frac{\partial E}{\partial w_{53}}, \frac{\partial E}{\partial b_1}$.

Let's calculate $\frac{\partial E}{\partial w_{41}}$, $\frac{\partial E}{\partial w_{42}}$, and $\frac{\partial E}{\partial w_{43}}$ first, since they all flow through node y_4:

$$\frac{\partial E}{\partial w_{41}} = \frac{\partial E}{\partial y_4} \frac{\partial y_4}{\partial v_4} \frac{\partial v_4}{\partial w_{41}}$$

Calculation of the first term on the right hand side of the equation above is a bit more involved than previous calculations since y_4 affects the error through both y_6 and y_7.

$$\frac{\partial E}{\partial y_4} = \frac{\partial E}{\partial y_6} \frac{\partial y_6}{\partial v_6} \frac{\partial v_6}{\partial y_4} + \frac{\partial E}{\partial y_7} \frac{\partial y_7}{\partial v_7} \frac{\partial v_7}{\partial y_4}$$

Now to the numerical values for these error derivatives; they have already been calculated above or are similar in style to those calculated above.

$$\frac{\partial E}{\partial y_4} = (0.7896)(0.0983)(0.7) + (0.7504)(0.1598)(0.8) = 0.1502$$

Plugging the above into the formula for $\frac{\partial E}{\partial w_{41}}$, we get

$$\frac{\partial E}{\partial w_{41}} = (0.1502)(0.0132)(1) = 0.0020$$

Remaining calculations for weights associated with y_4 are below

$$\frac{\partial E}{\partial w_{42}} = \frac{\partial E}{\partial y_4}\frac{\partial y_4}{\partial v_4}\frac{\partial v_4}{\partial w_{42}} = (0.1502)(0.0132)(4) = 0.0079$$

$$\frac{\partial E}{\partial w_{43}} = \frac{\partial E}{\partial y_4}\frac{\partial y_4}{\partial v_4}\frac{\partial v_4}{\partial w_{43}} = (0.1502)(0.0132)(5) = 0.0099$$

Similarly, for node y_5, we can calculate $\frac{\partial E}{\partial w_{51}}$, $\frac{\partial E}{\partial w_{52}}$, and $\frac{\partial E}{\partial w_{53}}$.

For $\frac{\partial E}{\partial w_{51}} = \frac{\partial E}{\partial y_5}\frac{\partial y_5}{\partial v_5}\frac{\partial v_5}{\partial w_{51}}$, we need to determine $\frac{\partial E}{\partial y_5} = \frac{\partial E}{\partial y_6}\frac{\partial y_6}{\partial v_6}\frac{\partial v_6}{\partial y_5} + \frac{\partial E}{\partial y_7}\frac{\partial y_7}{\partial v_7}\frac{\partial v_7}{\partial y_5}$; again, we have a combined term since y_5 affects the error through both y_6 and y_7.

$$\frac{\partial E}{\partial y_5} = (0.7896)(0.0983)(0.9) + (0.7504)(0.1598)(0.1) = 0.0818$$

and thus $\frac{\partial E}{\partial w_{51}} = (0.0818)(0.0049)(1) = 0.0004$

For $\frac{\partial E}{\partial w_{52}}$ and $\frac{\partial E}{\partial w_{53}}$ we have

$$\frac{\partial E}{\partial w_{52}} = \frac{\partial E}{\partial y_5}\frac{\partial y_5}{\partial v_5}\frac{\partial v_5}{\partial w_{52}} = (0.0818)(0.0049)(4) = 0.0016$$

$$\frac{\partial E}{\partial w_{53}} = \frac{\partial E}{\partial y_5}\frac{\partial y_5}{\partial v_5}\frac{\partial v_5}{\partial w_{53}} = (0.0818)(0.0049)(5) = 0.0020$$

The final error derivative we have to calculate is $\frac{\partial E}{\partial b_1}$

$$\frac{\partial E}{\partial b_1} = \frac{\partial E}{\partial y_6}\frac{\partial y_6}{\partial v_6}\frac{\partial v_6}{\partial y_4}\frac{\partial y_4}{\partial v_4}\frac{\partial v_4}{\partial b_1} + \frac{\partial E}{\partial y_7}\frac{\partial y_7}{\partial v_7}\frac{\partial v_7}{\partial y_5}\frac{\partial y_5}{\partial v_5}\frac{\partial v_5}{\partial b_1} =$$

$$= (0.7896)(0.0983)(0.7)(0.0132)(1) + (0.7504)(0.1598)(0.1)(0.0049)(1) = 0.0008$$

Step (5) All error derivatives are now determined and ready for parameter updates after the first iteration of backpropagation. The correction $\delta w_{l_j}(k)$ applied to the synaptic weight connecting neuron i to neuron j is given by the delta rule (cf. equation 3.27 in the textbook)

$$\Delta w_{ji}(k) = \begin{cases} \alpha e_j(k)\phi_j'\big(v_j(k)\big)y_i(k) & \text{if } j \text{ is an output node} \\ \alpha\phi_j'(v_j(k))\sum_l \left(e_l\phi_j'(v_l(k))w_{lj}(k)\right)y_i(k) & \text{if } j \text{ is a hidden node} \end{cases}$$

using the learning rate of $\alpha = 0.01$. See the following page for the results.

$$w_{41} := w_{41} - \alpha \frac{\partial E}{\partial w_{41}} = 0.1 - (0.01)(0.0020) = 0.1000$$

$$w_{51} := w_{51} - \alpha \frac{\partial E}{\partial w_{51}} = 0.2 - (0.01)(0.0004) = 0.2000$$

$$w_{42} := w_{42} - \alpha \frac{\partial E}{\partial w_{42}} = 0.3 - (0.01)(0.0079) = 0.2999$$

$$w_{52} := w_{52} - \alpha \frac{\partial E}{\partial w_{52}} = 0.4 - (0.01)(0.0016) = 0.4000$$

$$w_{43} := w_{43} - \alpha \frac{\partial E}{\partial w_{43}} = 0.5 - (0.01)(0.0099) = 0.4999$$

$$w_{53} := w_{53} - \alpha \frac{\partial E}{\partial w_{53}} = 0.6 - (0.01)(0.0020) = 0.6000$$

$$w_{64} := w_{64} - \alpha \frac{\partial E}{\partial w_{64}} = 0.7 - (0.01)(0.0765) = 0.6992$$

$$w_{74} := w_{74} - \alpha \frac{\partial E}{\partial w_{74}} = 0.8 - (0.01)(0.1183) = 0.7988$$

$$w_{65} := w_{65} - \alpha \frac{\partial E}{\partial w_{65}} = 0.9 - (0.01)(0.0772) = 0.8992$$

$$w_{75} := w_{75} - \alpha \frac{\partial E}{\partial w_{75}} = 0.1 - (0.01)(0.1193) = 0.0988$$

$$b_1 := b_1 - \alpha \frac{\partial E}{\partial b_1} = 0.5 - (0.01)(0.0008) = 0.5000$$

$$b_2 := b_2 - \alpha \frac{\partial E}{\partial b_2} = 0.5 - (0.01)(0.1975) = 0.4980$$

What's next? The described procedure is repeated over and over many times until the error goes down and the parameter estimates stabilize (or converge to some values). In reality, we would not be performing all these calculations manually or using a discrete code, but rather use a machine learning package that is already readily available.