Neural Network Assignment 2

ELEC 5404

Jonathan Levine

Part 1: The values and the corresponding computations for E, the derivatives $\partial E/\partial \mathbf{w}$ and $\mathbf{w}$ in the first epoch are shown below. Note that the function $\sigma(x)$ is the sigmoid function. The Python code used for the calculations for Part 1 can be found at: `https://github.com/JonathanALevine/ELEC5404/blob/main/assignment2/part1.ipynb`.

$$
\begin{aligned}
\mathbf{w} &= [v_0, v_1, v_2, w_{10}, w_{20}, w_{11}, w_{21}] = [0, 4, -1.0, 0, 1.0, 2.0, 0.1] \\
E &= \frac{1}{2}\sum_{i=1}^{2}(y(x_i, \mathbf{w}) - d_i) \\
&= \frac{1}{2}\left[(1.2689 - 2)^2 + (2.7729 - 4)^2\right] \\
&= 1.0201
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial E}{\partial v_0} &= \sum_{i=1}^{2}(y(x_i, \mathbf{w}) - d_i) = -1.9581 \\
\frac{\partial E}{\partial v_1} &= \sum_{i=1}^{2}(y(x_i, \mathbf{w}) - d_i)z_1 = -1.4463 \\
\frac{\partial E}{\partial v_2} &= \sum_{i=1}^{2}(y(x_i, \mathbf{w}) - d_i)z_2 = -1.4551 \\
\frac{\partial E}{\partial w_{10}} &= \sum_{i=1}^{2}(y(x_i, \mathbf{w}) - d_i)v_1\sigma(x_i w_{11} + w_{10})(1 - \sigma(x_i w_{11} + w_{10})) = -1.2464 \\
\frac{\partial E}{\partial w_{20}} &= \sum_{i=1}^{2}(y(x_i, \mathbf{w}) - d_i)v_2\sigma(x_i w_{21} + w_{20})(1 - \sigma(x_i w_{21} + w_{20})) = 0.3737 \\
\frac{\partial E}{\partial w_{11}} &= \sum_{i=1}^{2}(y(x_i, \mathbf{w}) - d_i)v_1 x_i\sigma(x_i w_{11} + w_{10})(1 - \sigma(x_i w_{11} + w_{10})) = -0.5153 \\
\frac{\partial E}{\partial w_{21}} &= \sum_{i=1}^{2}(y(x_i, \mathbf{w}) - d_i)v_2 x_i\sigma(x_i w_{21} + w_{20})(1 - \sigma(x_i w_{21} + w_{20})) = 0.2299
\end{aligned}
$$

The updated values of $\mathbf{w}$ at the end of the epoch are:

$$\begin{aligned} w = w + \eta\frac{\partial E}{\partial \mathbf{w}} &= [0, 4, -1.0, 0, 1.0, 2.0, 0.1] \\ &\quad + 0.1 \times [-1.9581, -1.4463, -1.4551, -1.2464, 0.3737, -0.5153, 0.2299] \\ &= [0.1958, 4.1446, -0.8545, 0.1246, 0.9626, 2.0515, 0.0770] \end{aligned}$$

Interestingly, if I continue the computation for another 249 epochs the weight values are:

$$w = [0.3880, 4.5391, -0.6990, -0.1377, 0.9321, 2.3806, -0.0698]$$

And the MLP outputs $y(0, \mathbf{w}) = 2.0000$ and $y(1, \mathbf{w}) = 4.0000$; back-propagation works and the MLP is trained perfectly!

Part 2: (a): The MATLAB code used for the computations for this part can be found at: `https://github.com/JonathanALevine/ELEC5404/blob/main/assignment2/part2a.m`. The contour plot of $E(\mathbf{w})$ is shown below. For the first epoch:

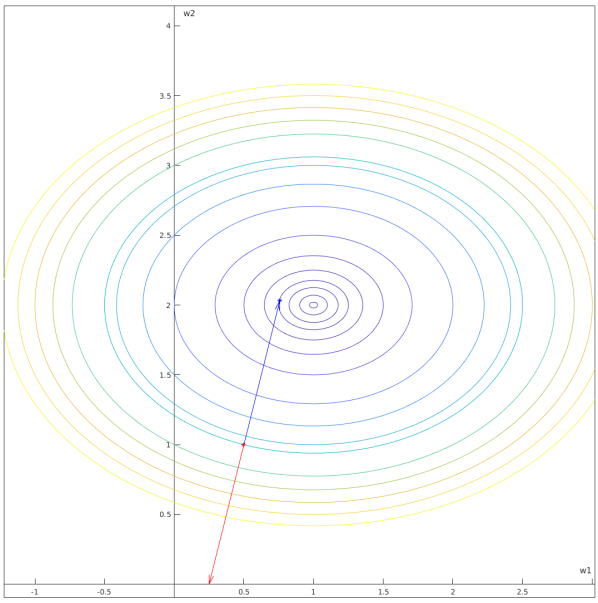

Figure 1: Initial point (w_1, w_2) (red asterisk) and updated point (w_1', w_2') (blue asterisk) with gradient vector (red) and update direction vector (blue). Note that the gradient vector has been scaled down in magnitude so that it can be displayed nicely on the plot.

For the second epoch:

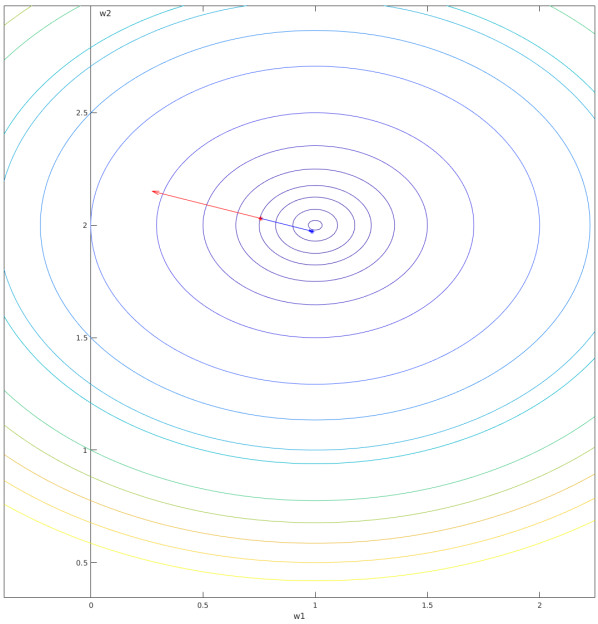

Figure 2: (w_1, w_2) (red asterisk) from first epoch and updated point (w_1', w_2') (blue asterisk) for the second epoch with gradient vector (red) and update direction vector (blue). Note that the gradient vector has been scaled down in magnitude so that it can be displayed nicely on the plot.

Part 2: (b) The MATLAB code used for the computations for this part can be found at: `https://github.com/JonathanALevine/ELEC5404/blob/main/assignment2/part2b.m`. The number of epochs needed to find the optimal solution so that the training error is reduced to zero is: epochs $= N_w = length(\mathbf{w}) = 2$. The contour plot of $E(\mathbf{w})$ is shown below. For the first epoch:

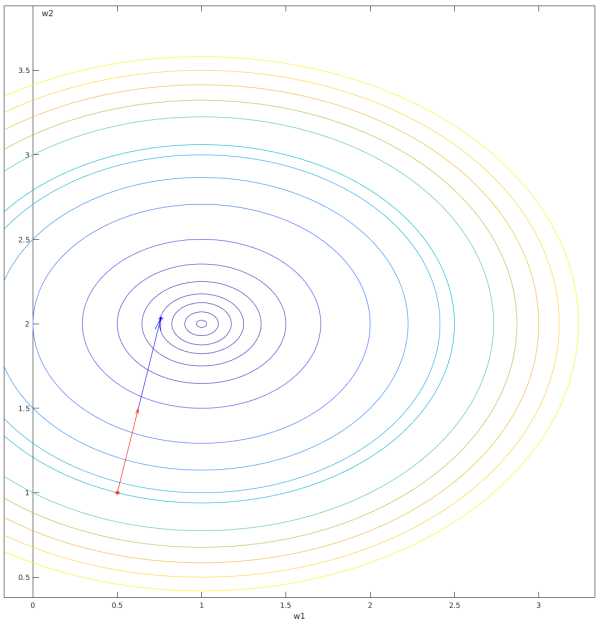

Figure 3: Initial point (w_1, w_2) (red asterisk) from first epoch and updated point (w_1', w_2') (blue asterisk) with negative gradient vector (red) and update direction vector (blue). Note that the negative gradient vector has been scaled down in magnitude so that it can be displayed nicely on the plot.

For the second epoch:

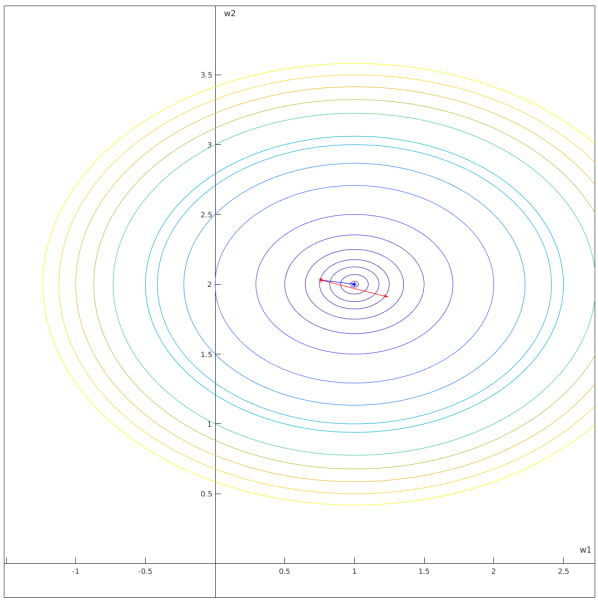

Figure 4: (w_1, w_2) (red asterisk) from first epoch and updated point (w_1', w_2') (blue asterisk) for the second epoch with negative gradient vector and update direction vector (blue). Note that the negative gradient vector has been scaled down in magnitude so that it can be displayed nicely on the plot.