CSE 411: Machine Learning

Backpropagation

Dr Muhammad Abul Hasan

Department of Computer Science and Engineering
Green University of Bangladesh
muhammad.hasan@cse.green.edu.bd

Fall 2023

Outline

"A journey of a thousand miles begins with a single step.

– Lao Tzu"

Next Up ...

Gradient Descent

Gradient descent is an optimization algorithm commonly used to train machine learning models and neural networks.

More technically

Gradient descent is a first-order iterative optimization algorithm for finding a local minimum of a differentiable function.

Univariate Chain Rule

- Recall: if $f(x)$ and $x(t)$ are univariate functions, then

$$\frac{d}{dt} f(x(t)) = \frac{df}{dx}\frac{dx}{dt}.$$

Univariate Chain Rule

Univariate logistic least squares model

$$
\begin{aligned}
z &= \theta_1 x + \theta_2 \\
y &= \sigma(z) \\
J(\theta) &= \frac{1}{2}(y - t)^2
\end{aligned}
$$

Let us compute the loss derivatives.

Univariate Chain Rule

$$
\begin{aligned}
J(\theta) &= \frac{1}{2}(\sigma(\theta_1 x + \theta_2) - t)^2 \\
\frac{\partial J(\theta)}{\partial \theta_1} &= \frac{\partial}{\partial \theta_1}\left[\frac{1}{2}(\sigma(\theta_1 x + \theta_2) - t)^2\right] \\
&= \frac{1}{2}\frac{\partial}{\partial \theta_1}(\sigma(\theta_1 x + \theta_2) - t)^2 \\
&= (\sigma(\theta_1 x + \theta_2) - t)\frac{\partial}{\partial \theta_1}(\sigma(\theta_1 x + \theta_2) - t) \\
&= (\sigma(\theta_1 x + \theta_2) - t)\sigma'(\theta_1 x + \theta_2)\frac{\partial}{\partial \theta_1}(\theta_1 x + \theta_2) \\
&= (\sigma(\theta_1 x + \theta_2) - t)\sigma'(\theta_1 x + \theta_2)x
\end{aligned}
$$

Univariate Chain Rule

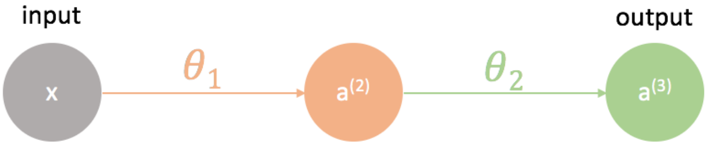
$$
\begin{aligned}
\frac{\partial J(\theta)}{\partial \theta_2} &= \frac{\partial}{\partial \theta_2}\left[\frac{1}{2}\left(\sigma(\theta_1 x + \theta_2) - t\right)^2\right] \\
&= \frac{1}{2}\frac{\partial}{\partial \theta_2}\left(\sigma(\theta_1 x + \theta_2) - t\right)^2 \\
&= \left(\sigma(\theta_1 x + \theta_2) - t\right)\frac{\partial}{\partial \theta_2}\left(\sigma(\theta_1 x + \theta_2) - t\right) \\
&= \left(\sigma(\theta_1 x + \theta_2) - t\right)\sigma'(\theta_1 x + \theta_2)\frac{\partial}{\partial \theta_2}(\theta_1 x + \theta_2) \\
&= \left(\sigma(\theta_1 x + \theta_2) - t\right)\sigma'(\theta_1 x + \theta_2)
\end{aligned}
$$

Next Up ...

Toy Exercise

To figure out how to use gradient descent in training a neural network, let us start with the simplest neural network which has:

- one input neuron
- one hidden layer neuron
- and one output neuron



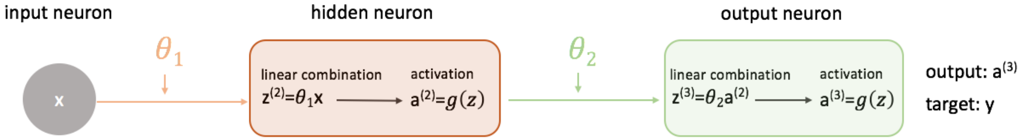
$$a^{(2)} = g(\theta_1 x) \qquad a^{(3)} = g(\theta_2 a^{(2)})$$

Toy Exercise

To show a more complete picture of what is going on, each neuron is expanded to show:

1. the *linear combination* of inputs and weights, and
2. the *activation* of this linear combination.

It is easy to see that the *forward propagation* step is simply a series of functions where the output of one node feeds as the input to the next node.



Relating the weights to the cost function

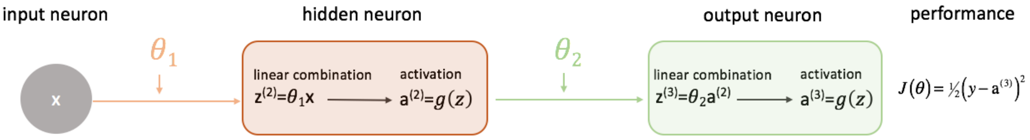
In order to minimize the difference between our neural network's output and the target output, we need to know how the model performance changes with respect to each parameter in our model. We can then update these weights in an iterative process using gradient descent.

$$\frac{\partial J(\theta)}{\partial \theta_1} = ?$$

$$\frac{\partial J(\theta)}{\partial \theta_2} = ?$$

Relating the weights to the cost function

Let us look at $\frac{\partial J(\theta)}{\partial \theta_2}$ first. Keep the following figure in mind as we progress.



Let us take a moment to examine how we could express the relationship between $J(\theta)$ and θ_2. Carefully look at the diagram above, how θ_2 is an input to $z^{(3)}$, which is an input to $a^{(3)}$, which is an input to $J(\theta)$. When we are trying to compute a derivative of this sort, we can use the chain rule to solve.

Relating the weights to the cost function

Let us apply the chain rule to solve for $\frac{\partial J(\theta)}{\partial \theta_2}$

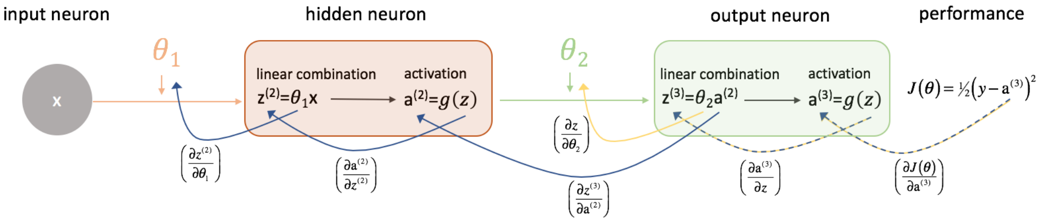
$$\frac{\partial J(\theta)}{\partial \theta_2} = \left(\frac{\partial J(\theta)}{\partial \mathrm{a}^{(3)}}\right)\left(\frac{\partial \mathrm{a}^{(3)}}{\partial z}\right)\left(\frac{\partial z}{\partial \theta_2}\right) \tag{1}$$

Relating the weights to the cost function

By similar logic, we can find $\frac{\partial J(\theta)}{\partial \theta_1}$.

$$\frac{\partial J(\theta)}{\partial \theta_1} = \left(\frac{\partial J(\theta)}{\partial a^{(3)}}\right)\left(\frac{\partial a^{(3)}}{\partial z^{(3)}}\right)\left(\frac{\partial z^{(3)}}{\partial a^{(2)}}\right)\left(\frac{\partial a^{(2)}}{\partial z^{(2)}}\right)\left(\frac{\partial z^{(2)}}{\partial \theta_1}\right) \tag{2}$$

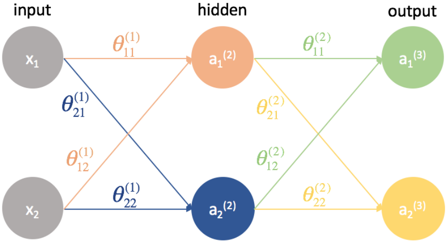
For better understanding, the following diagram is used to visualize these chains.



Next Up ...

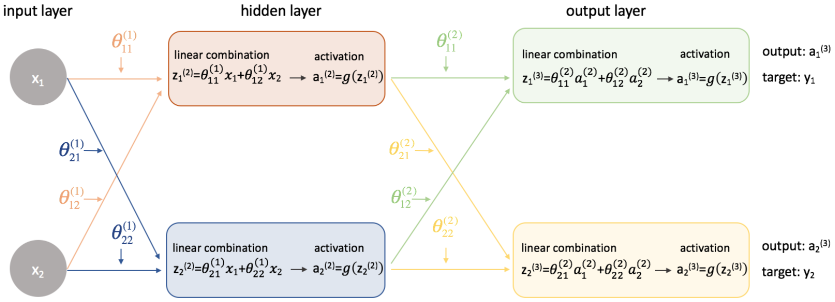
Generalizing Concept: Multiple Units in Each Layer

Let us take a slightly more complicated example. Now, we will look at a neural network with two neurons in our input layer, two neurons in one hidden layer, and two neurons in our output layer. For now, we will disregard the bias neurons that are missing from the input and hidden layers.



Generalizing Concept: Multiple Units in Each Layer

Let us expand this network to expose all of the math that is going on.



We will go through the process of finding one of the partial derivatives of the cost function with respect to one of the parameters only. If it is understood clearly, the rest of the calculations can be done effortlessly.

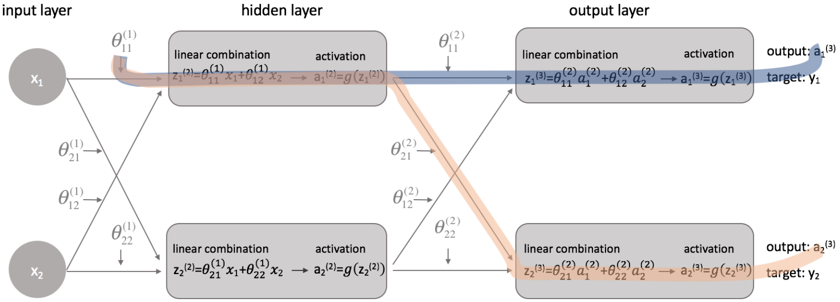
Generalizing Concept: Multiple Units in Each Layer

Initially, we will need to revisit our cost function now that we are dealing with a neural network with more than one output. Let us now use the mean squared error as our cost function.

$$J(\theta) = \frac{1}{2m} \sum_i \left(y_i - a_i^{(2)} \right)^2 \tag{3}$$

Generalizing Concept: Multiple Units in Each Layer

Let us calculate $\frac{\partial J(\theta)}{\partial \theta_{11}^{(1)}}$ in this example. Looking at the diagram, $\theta_{11}^{(1)}$ affects the output for both $a_1^{(3)}$ and $a_2^{(3)}$. Because our cost function is a summation of individual costs for each output, we can calculate the derivative chain for each path and simply add them together.



Generalizing Concept: Multiple Units in Each Layer

The derivative chain for the blue path is:

$$\left(\frac{\partial J\left(\theta\right)}{\partial \mathrm{a}_1^{(3)}}\right)\left(\frac{\partial \mathrm{a}_1^{(3)}}{\partial z_1^{(3)}}\right)\left(\frac{\partial z_1^{(3)}}{\partial \mathrm{a}_1^{(2)}}\right)\left(\frac{\partial \mathrm{a}_1^{(2)}}{\partial z_1^{(2)}}\right)\left(\frac{\partial z_1^{(2)}}{\partial \theta_{11}^{(1)}}\right) \tag{4}$$

Generalizing Concept: Multiple Units in Each Layer

The derivative chain for the orange path is:

$$\left(\frac{\partial J(\theta)}{\partial \mathrm{a}_2^{(3)}}\right)\left(\frac{\partial \mathrm{a}_2^{(3)}}{\partial z_2^{(3)}}\right)\left(\frac{\partial z_2^{(3)}}{\partial \mathrm{a}_1^{(2)}}\right)\left(\frac{\partial \mathrm{a}_1^{(2)}}{\partial z_1^{(2)}}\right)\left(\frac{\partial z_1^{(2)}}{\partial \theta_{11}^{(1)}}\right) \tag{5}$$

Generalizing Concept: Multiple Units in Each Layer

Combining these, we get the total expression for $\frac{\partial J(\theta)}{\partial \theta_{11}^{(1)}}$.

$$\frac{\partial J(\theta)}{\partial \theta_{11}^{(1)}} = \left(\frac{\partial J(\theta)}{\partial a_1^{(3)}}\right)\left(\frac{\partial a_1^{(3)}}{\partial z_1^{(3)}}\right)\left(\frac{\partial z_1^{(3)}}{\partial a_1^{(2)}}\right)\left(\frac{\partial a_1^{(2)}}{\partial z_1^{(2)}}\right)\left(\frac{\partial z_1^{(2)}}{\partial \theta_{11}^{(1)}}\right) + \left(\frac{\partial J(\theta)}{\partial a_2^{(3)}}\right)\left(\frac{\partial a_2^{(3)}}{\partial z_2^{(3)}}\right)\left(\frac{\partial z_2^{(3)}}{\partial a_1^{(2)}}\right)\left(\frac{\partial a_1^{(2)}}{\partial z_1^{(2)}}\right)\left(\frac{\partial z_1^{(2)}}{\partial \theta_{11}^{(1)}}\right)$$

Generalizing Concept: Multiple Units in Each Layer

Now we describe how changing each parameter affects the cost function which is done using partial derivatives. Let's calculate Layer 2 Parameters.

$$\frac{\partial J(\theta)}{\partial \theta_{11}^{(2)}} = \left(\frac{\partial J(\theta)}{\partial \mathrm{a}_1^{(3)}}\right)\left(\frac{\partial \mathrm{a}_1^{(3)}}{\partial z_1^{(3)}}\right)\left(\frac{\partial z_1^{(3)}}{\partial \theta_{11}^{(2)}}\right) \tag{6}$$

$$\frac{\partial J(\theta)}{\partial \theta_{12}^{(2)}} = \left(\frac{\partial J(\theta)}{\partial \mathrm{a}_1^{(3)}}\right)\left(\frac{\partial \mathrm{a}_1^{(3)}}{\partial z_1^{(3)}}\right)\left(\frac{\partial z_1^{(3)}}{\partial \theta_{12}^{(2)}}\right) \tag{7}$$

$$\frac{\partial J(\theta)}{\partial \theta_{21}^{(2)}} = \left(\frac{\partial J(\theta)}{\partial \mathrm{a}_2^{(3)}}\right)\left(\frac{\partial \mathrm{a}_2^{(3)}}{\partial z_2^{(3)}}\right)\left(\frac{\partial z_2^{(3)}}{\partial \theta_{21}^{(2)}}\right) \tag{8}$$

$$\frac{\partial J(\theta)}{\partial \theta_{22}^{(2)}} = \left(\frac{\partial J(\theta)}{\partial \mathrm{a}_2^{(3)}}\right)\left(\frac{\partial \mathrm{a}_2^{(3)}}{\partial z_2^{(3)}}\right)\left(\frac{\partial z_2^{(3)}}{\partial \theta_{22}^{(2)}}\right) \tag{9}$$

Generalizing Concept: Multiple Units in Each Layer

Similarly, Layer 1 Parameters are calculated as follows.

$$\frac{\partial J(\theta)}{\partial \theta_{11}^{(1)}} = \left(\frac{\partial J(\theta)}{\partial a_1^{(3)}}\right)\left(\frac{\partial a_1^{(3)}}{\partial z_1^{(3)}}\right)\left(\frac{\partial z_1^{(3)}}{\partial a_1^{(2)}}\right)\left(\frac{\partial a_1^{(2)}}{\partial z_1^{(2)}}\right)\left(\frac{\partial z_1^{(2)}}{\partial \theta_{11}^{(1)}}\right) + \left(\frac{\partial J(\theta)}{\partial a_2^{(3)}}\right)\left(\frac{\partial a_2^{(3)}}{\partial z_2^{(3)}}\right)\left(\frac{\partial z_2^{(3)}}{\partial a_1^{(2)}}\right)\left(\frac{\partial a_1^{(2)}}{\partial z_1^{(2)}}\right)\left(\frac{\partial z_1^{(2)}}{\partial \theta_{11}^{(1)}}\right) \tag{10}$$

$$\frac{\partial J(\theta)}{\partial \theta_{12}^{(1)}} = \left(\frac{\partial J(\theta)}{\partial a_1^{(3)}}\right)\left(\frac{\partial a_1^{(3)}}{\partial z_1^{(3)}}\right)\left(\frac{\partial z_1^{(3)}}{\partial a_1^{(2)}}\right)\left(\frac{\partial a_1^{(2)}}{\partial z_1^{(2)}}\right)\left(\frac{\partial z_1^{(2)}}{\partial \theta_{12}^{(1)}}\right) + \left(\frac{\partial J(\theta)}{\partial a_2^{(3)}}\right)\left(\frac{\partial a_2^{(3)}}{\partial z_2^{(3)}}\right)\left(\frac{\partial z_2^{(3)}}{\partial a_1^{(2)}}\right)\left(\frac{\partial a_1^{(2)}}{\partial z_1^{(2)}}\right)\left(\frac{\partial z_1^{(2)}}{\partial \theta_{12}^{(1)}}\right) \tag{11}$$

$$\frac{\partial J(\theta)}{\partial \theta_{21}^{(1)}} = \left(\frac{\partial J(\theta)}{\partial a_1^{(3)}}\right)\left(\frac{\partial a_1^{(3)}}{\partial z_1^{(3)}}\right)\left(\frac{\partial z_1^{(3)}}{\partial a_2^{(2)}}\right)\left(\frac{\partial a_2^{(2)}}{\partial z_2^{(2)}}\right)\left(\frac{\partial z_2^{(2)}}{\partial \theta_{21}^{(1)}}\right) + \left(\frac{\partial J(\theta)}{\partial a_2^{(3)}}\right)\left(\frac{\partial a_2^{(3)}}{\partial z_2^{(3)}}\right)\left(\frac{\partial z_2^{(3)}}{\partial a_2^{(2)}}\right)\left(\frac{\partial a_2^{(2)}}{\partial z_2^{(2)}}\right)\left(\frac{\partial z_2^{(2)}}{\partial \theta_{21}^{(1)}}\right) \tag{12}$$

$$\frac{\partial J(\theta)}{\partial \theta_{22}^{(1)}} = \left(\frac{\partial J(\theta)}{\partial a_1^{(3)}}\right)\left(\frac{\partial a_1^{(3)}}{\partial z_1^{(3)}}\right)\left(\frac{\partial z_1^{(3)}}{\partial a_2^{(2)}}\right)\left(\frac{\partial a_2^{(2)}}{\partial z_2^{(2)}}\right)\left(\frac{\partial z_2^{(2)}}{\partial \theta_{22}^{(1)}}\right) + \left(\frac{\partial J(\theta)}{\partial a_2^{(3)}}\right)\left(\frac{\partial a_2^{(3)}}{\partial z_2^{(3)}}\right)\left(\frac{\partial z_2^{(3)}}{\partial a_2^{(2)}}\right)\left(\frac{\partial a_2^{(2)}}{\partial z_2^{(2)}}\right)\left(\frac{\partial z_2^{(2)}}{\partial \theta_{22}^{(1)}}\right) \tag{13}$$

Next Up ...

Backropagation

- Backpropagation is simply a method for calculating the partial derivative of the cost function with respect to all of the parameters.
- The actual optimization of parameters (training) is done by the gradient descent technique.
- Generally, we established that you can calculate the partial derivatives for layer l by combining δ terms of the next layer forward with the activations of the current layer.

$$\frac{\partial J(\theta)}{\partial \theta_{ij}^{(l)}} = \left(\delta^{(l+1)}\right)^{T} a^{(l)} \tag{14}$$

Putting it all together

After we have calculated all of the partial derivatives for the neural network parameters, we can use gradient descent to update the weights.
In general, we defined gradient descent as

$$\theta_i := \theta_i + \Delta\theta_i$$

where $\Delta\theta_i$ is the "step" we take walking along the gradient, scaled by a learning rate, η.

$$\Delta\theta_i = -\eta \frac{\partial J(\theta)}{\partial \theta_i}$$

we will use this formula to update each of the weights, recompute forward propagation with the new weights, backpropagate the error, and calculate the next weight update. This process continues until we have converged on an optimal value for our parameters.

Putting it all together

During each iteration we perform forward propagation to compute the outputs and backward propagation to compute the errors; one complete iteration is known as an epoch. It is common to report evaluation metrics after each epoch so that we can watch the evolution of our neural network as it trains.

Thank You!