HW A2

1. SGD updates weights for every single training sample individually. It uses gradient descent so that each weight is updated for a neuron. The update loop will run through each neuron, starting from the output layer ending at the first hidden layer, and it will run through each weight for each neuron. For every training set, the weights are updated.

$$w_{jk} \leftarrow w_{jk} - \eta \frac{\partial}{\partial w_{jk}} Loss(h_w(x))$$

The eta is a chosen learning rate, so the only math is determining how much each weight impacts the loss function and numerically updating. The loss function is a MSE function shown below. It is multiplied by the constant n/2 for simplification of calculation.

$$Loss = \frac{1}{2}\sum_{i=1}^{n}(y_i - \hat{y})^2$$

The partial derivative of this with respect to each weight can be determined by using the chain rule

$$\frac{\partial}{\partial w_{jk}} Loss = \frac{\partial Loss}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial s_k} \frac{\partial s_k}{\partial w_{jk}}$$

This is where $\hat{y}$ and s_k are the following equations

$$\hat{y} = \Phi_k(s_k)$$

$$s_k = w_k \cdot x_k$$

The simplified partial derivative of loss equation for an output neuron is the following:

$$\delta_k = \frac{\partial Loss}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial s_k} = -(y_k - \Phi_k(w_k \cdot x_k))\Phi_k{}'(w_k \cdot x_k)$$

$$\frac{\partial}{\partial w_{jk}} Loss = \delta_k x_{jk}$$

The derivative of the loss function with respect to the function output is a bit more complex for a neuron in a hidden layer, since all neuron's downstream will also impact the change of the output relative to the weight, however, the general method is the same. The only thing that changes is the loss with respect to the calculated output. It becomes the some of all the downstream differences in respected output times the weight that connects them (directly or indirectly for more than 1 hidden layer) to the specific hidden layer neuron.

$$\delta_k = \Phi_k'(w_k \cdot x_k) * \sum_{i \in DS(k)} \delta_i w_{ki}$$

$$\frac{\partial}{\partial w_{jk}} Loss = x_{jk}\delta_k$$

2. The BGD method updates the weights after computing the gradient for the entire data set. The SGD updates weights for every individual training set. The BGD method converges more smoothly, since the numerically determined gradient is closer to the real gradient, but it is much more computationally expensive.

3. Similarly, for BGD, the update method for all the weights is the same as the SGD.

$$w_{jk} \leftarrow w_{jk} - \eta \frac{\partial}{\partial w_{jk}} Loss(h_w(x))$$

The difference is in how the gradient of the loss function is calculated. For the BGD the gradient of the loss function is calculated by averaging the loss gradient for every single training set (or a group of training sets) and *then* updating the weights. The loss is the same.

$$Loss = \frac{1}{2}\sum_{i=1}^{n}(y_i - \hat{y})^2$$

The partial derivative for each individual training set is also calculated the same

$$\frac{\partial}{\partial w_{jk}} Loss_i = \frac{\partial Loss_i}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial s_k} \frac{\partial s_k}{\partial w_{jk}}$$

However, now the gradient will be averaged for each training set for n sets of data

$$\frac{\partial}{\partial w_{jk}} Loss = \frac{1}{n}\sum_{i=1}^{n} \frac{\partial}{\partial w_{jk}} Loss_i$$

Which can be rearranged as the loss for an output neuron

$$\delta_k = \frac{1}{n}\sum_{i=1}^{n} -(y_k - \Phi_k(w_k \cdot x_k))\Phi_k{}'(w_k \cdot x_k)$$

$$\frac{\partial}{\partial w_{jk}} Loss = \delta_k \frac{1}{n}\sum_{i=1}^{n} x_{jk}$$

The above is the loss for an output neuron using BGD. For a hidden layer neuron, it is the same method as the SGD, but it is once again averaged for n training samples.

$$\delta_k = \frac{1}{n}\sum_{i=1}^{n}\left[\Phi_k'(w_k \cdot x_k) * \sum\nolimits_{i \in DS(k)} \delta_i w_{ki}\right]$$

$$\frac{\partial}{\partial w_{jk}} Loss = \delta_k \frac{1}{n}\sum_{i=1}^{n} x_{jk}$$