# CSE 676: Assignment #1

Name: Koushik Ponugoti
UB ID: kponugot
UBIT#: 50441728

Q.1 $\boxed{\text{Softmax}}$

1.) given,

$$\text{Softmax}(x)_i \triangleq \frac{e^{x_i}}{\sum_{i'} e^{x_{i'}}}$$

we need to prove,

$$\text{Softmax}(x) = \text{Softmax}(x+c)$$

$$\Rightarrow \text{Softmax}(x+c) = \frac{e^{x_i+c}}{\sum_{i'} e^{x_{i'}+c}}$$

$$= \frac{e^{x_i} * \cancel{e^c}}{\cancel{e^c} \cdot \sum_{i'} e^{x_{i'}}}$$

$$= \frac{e^{x_i}}{\sum_{i'} e^{x_{i'}}}$$

$$= \text{Softmax}(x)$$

$$\therefore \text{Softmax}(x) = \text{Softmax}(x+c)$$

2) given,

$$J = \sum_i \log softmax(z_i)$$

we need to find $\frac{\partial J}{\partial W}$, $\frac{\partial J}{\partial c}$

To find $\frac{\partial J}{\partial W}$, we can do that by using the chain Rule

i.e; $$\frac{\partial J}{\partial W} = \frac{\partial J}{\partial z} \cdot \frac{\partial z}{\partial W}$$

$$\frac{\partial J}{\partial z} = \frac{\partial}{\partial z}(\log(softmax(z_i))) - \frac{\partial}{\partial z}\sum_{K \neq i} \log(softmax(z_K))$$

$$\Rightarrow \frac{\partial J}{\partial z} = -\left(\frac{1}{softmax(z_i)} \frac{\partial(softmax(z_i))}{\partial z}\right) - \sum_{K \neq i} y_K \left(\frac{1}{softmax(z_K)} \cdot \frac{\partial(softmax(z_K))}{\partial z}\right)$$

Here $y_K$ is one-hot vector representing true lable.

$$\frac{\partial J}{\partial z} = -(1 - (softmax(z_i))) + \sum_{K \neq i}(softmax(z_i))$$

$$\frac{\partial J}{\partial z} = -1 + (softmax(z_i)) + \sum_{K \neq i}(softmax(z_i))$$

$$\frac{\partial J(z)}{\partial z} = -1 + \left(\sum_{K=1}^{N} 1\right)(softmax(z_i))$$

$$\frac{\partial J(z)}{\partial z} = -1 + N(softmax(z_i))$$

Now, $\frac{\partial J}{\partial W} = \left(-1 + N(\mathrm{Softmax}(z_i))\right) \frac{\partial z}{\partial W}$

$$\boxed{\frac{\partial J}{\partial W} = \left(-1 + N(\mathrm{Softmax}(z_i))\right) x}$$

we can follow same process for $\frac{\partial J}{\partial c}$

$$\frac{\partial J}{\partial c} = \frac{\partial J}{\partial z} \cdot \frac{\partial z}{\partial c}$$

$$\boxed{\frac{\partial J}{\partial c} = \left(-1 + N(\mathrm{softmax}(z_i))\right)}$$

0.2 – Logistic Regression with Regularization.

1.) given,

$$p(y_i; x_i, \theta) = \sigma(\theta^T x_i) \triangleq \frac{1}{1 + e^{-\theta^T x_i}}$$

let us suppose the above equation as $g_\theta(x_i)$ and the Regularization term is

$$R(\theta) = \frac{\lambda}{2} \theta^T \theta$$

we need to come up with our loss function, because if we use MSE as the loss function we will not get the convex function.

the loss function would be

$$Cost(g_\theta(x_i), y_i) = -y_i \log(h_\theta(x_i)) - (1-y_i)\log(1-h_\theta(x_i))$$

The objective function will be,

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N}(cost(h_\theta(x_i), y_i)$$

To make sure that the model does not overfit, we need to add the regularization term to the objective function.

$$\boxed{J(\theta) = \frac{1}{N}\sum_{i=1}^{N} Cost(g_\theta(x_i), y_i) + R(\theta)}$$

2. • The gradient descent algorithm helps us to find the minima of the above derived objective function.

By finding the derivative of the cost function, we can get the gradient equation.

$$\frac{\partial\, \text{cost}(g_\theta(x_i), y_i)}{\partial \theta_j} = -\left(\frac{y_i}{g_\theta(x_i)} - \frac{1-y_i}{1-g_\theta(x_i)}\right) \frac{\partial g_\theta(x_i)}{\partial \theta_j}$$

$$\Rightarrow -\left(\frac{y_i}{g_\theta(x_i)} - \frac{1-y_i}{1-g_\theta(x_i)}\right) g_\theta(x_i)\,(1-g_\theta(x_i))\, \frac{\partial \theta^T x_i}{\partial \theta_j}$$

$$\Rightarrow -\left(y_i(1-g_\theta(x_i)) - (1-y_i)\, g_\theta(x_i)\right) x_{ij}$$

$$\Rightarrow -(y_i - h_\theta(x_i))\, x_{ij}$$

Now, using this, we calculate the complete gradient equation.

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{\partial}{\partial \theta_j}\left(\frac{1}{N}\sum_{i=1}^{N} \text{cost}(g_\theta(x_i), y_i) + R(\theta)\right)$$

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{N}\sum_{i=1}^{N} \frac{\partial}{\partial \theta_j} \text{cost}(g_\theta(x_i), y_i) + \frac{\partial}{\partial \theta_j} R(\theta)$$

$$\frac{\partial J(\theta)}{\partial \theta_j} = -\frac{1}{N}\sum_{i=1}^{N} (y_i - g_\theta(x_i))\, x_{ij} + \frac{\partial}{\partial \theta_j} R(\theta)$$

Now, we need to find the derivative of $R(\theta)$

$$\frac{\partial R(\theta)}{\partial \theta_j} = \frac{\partial}{\partial \theta_j} \frac{\lambda}{2} \theta^T \theta$$

$$\frac{\partial}{\partial \theta_j} R(\theta) = \frac{\lambda}{2} \frac{\partial}{\partial \theta_j} (\theta^T \theta)$$

$$\frac{\partial}{\partial \theta_j} R(\theta) = \frac{\lambda}{2} (2\theta_j) = \lambda \theta_j$$

Now,

$$\frac{\partial J(\theta)}{\partial \theta_j} = -\frac{1}{N} \sum_{i=1}^{N} (y_i - g_\theta(x_i))\, x_{ij} + \lambda \theta_j$$

The equation for updating weights is

$$\theta_j = \theta_j - \alpha \frac{\partial J(\theta)}{\partial \theta_j}$$

$$\theta_j = \theta_j - \alpha \left( -\frac{1}{N} \sum_{i=1}^{N} (y_i - g_\theta(x_i))\, x_{ij} + \lambda \theta_j \right)$$

$$\Rightarrow \boxed{\theta_j = \theta_j \left(1 - \frac{1}{N}\right) - \alpha \left( \sum_{i=1}^{N} (g_\theta(x_i) - y_i)\, x_{ij} \right)}$$

## Q.3 Derivative of the Softmax function.

1.) given, $J(z) = -\sum_{K=1}^{K} y_K \log \tilde{y}_K$

$$\frac{\partial J(z)}{\partial z} = -\frac{\partial}{\partial z} \sum_{K=1}^{K} y_K \log \tilde{y}_K$$

This is for class K, we need to consider the derivative for class i also (useful while using chain rul

$$\frac{\partial J(z)}{\partial z} = -\frac{\partial}{\partial z}(y_i \log(\tilde{y}_i)) - \frac{\partial}{\partial z} \sum_{K \neq i} y_K \log \tilde{y}_K$$

$$\Rightarrow \frac{\partial J(z)}{\partial z} = -\left(y_i \frac{1}{\tilde{y}_i} \frac{\partial y_i}{\partial z}\right) - \sum_{K \neq i} y_K \frac{1}{\tilde{y}_K} \frac{\partial \tilde{y}_K}{\partial z}$$

$$\Rightarrow \frac{\partial J(z)}{\partial z} = -y_i + y_i \tilde{y}_i + \sum_{K \neq i} y_K \tilde{y}_i$$

$$\Rightarrow \frac{\partial J(z)}{\partial z} = -y_i + \left(\sum_{K=1}^{N} y_K\right) \tilde{y}_i$$

$$\Rightarrow \boxed{\frac{\partial J(z)}{\partial z} = -y_i + \tilde{y}_i}$$

② we are given the corss entropy loss function. which is

$$J(z) = -\sum_{K=1}^{K} y_K \log \tilde{y}_K$$

we know the gradient for the last layer, we found this by computing the derivative of the above function w.r.t output of the final layer.

i.e; $\frac{\partial J(z)}{\partial z} = -y_i + \tilde{y}_i$ as $z = W^T h + b$.

Inorder to update $W_{ij}$ which is the value of each weights, we have to find the value of

$$\frac{\partial J(z)}{\partial W_{ij}}$$

so, To find this, we need to propagate the error that ~~backwards~~ we get at the last layer and then we can use the chain rule in the calculus to propagate the error backwards.

Now, $\frac{\partial J(z)}{\partial w_{ij}}$ can be calculated for each weight.

when we calulate the gradient equation, based on the incorrectness of the classes, the incorrect classes also get affected which is not the property of the loss function.

(3)

we have $\frac{\partial J(z)}{\partial z} = -y_i + \tilde{y}_i$

we need to find $\frac{\partial J}{\partial W}$ and $\frac{\partial J}{\partial b}$

we can find $\frac{\partial J}{\partial w}$ by using chain rule.

$$\frac{\partial J(z)}{\partial w} = \frac{\partial J(z)}{\partial z} \cdot \frac{\partial z}{\partial W}$$

$$\frac{\partial J(z)}{\partial w} = (-y_i + \tilde{y}_i) \frac{\partial z}{\partial w}$$

$$\Rightarrow \frac{\partial J(z)}{\partial W} = (-y_i + \tilde{y}_i) \frac{\partial (W^T h + b)}{\partial W}$$

$$\Rightarrow \boxed{\frac{\partial J(z)}{\partial W} = (-y_i + \tilde{y}_i) h}$$

Similarly,

$$\frac{\partial J(z)}{\partial b} = (-y_i + \tilde{y}_i) \frac{\partial (W^T h + b)}{\partial b}$$

$$\boxed{\frac{\partial J(z)}{\partial b} = (-y_i + \tilde{y}_i)}$$

### 0.4 MNIST with FNN [30 points]

1) [10 points] Design an FNN for MNIST classification. Draw the computational graph of your model.
2) [20 points] Implement the model and plot two curves in one figure: i) training loss vs. training iterations; ii) test loss vs. training iterations.

**The Github link for the above code is: https://github.com/Koushik-24/CSE676-DeepLearning**

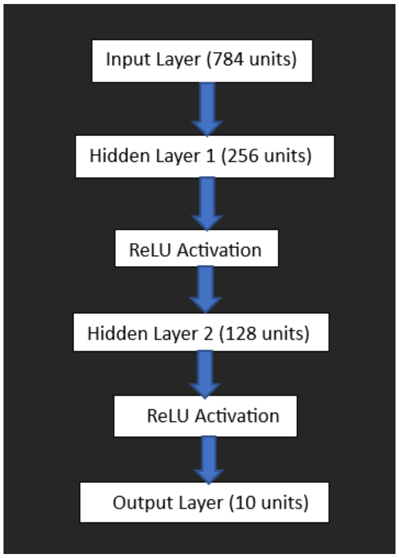


fig1 Computational Graph

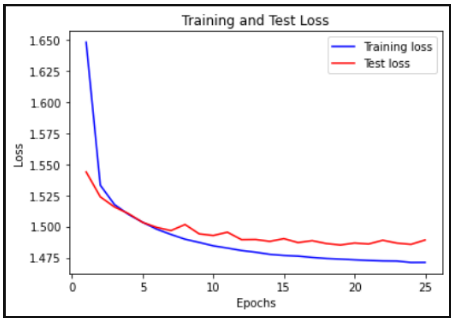


fig2 Training and Testing loss vs Epochs

## References

**1. https://www.youtube.com/watch?v=PcU2RCWuDSI**
**2. https://www.kaggle.com/code/poojandabhi/fnn-mnist**
**3. https://www.baeldung.com/cs/training-validation-loss-deep-learning**