Given a matrix of weights: W = $\begin{bmatrix} W_{0,0} & \cdots & W_{0,k} \\ \vdots & \ddots & \vdots \\ W_{m,0} & \cdots & W_{m,k} \end{bmatrix}$

An input vector: x = $\begin{bmatrix} x_0 \\ \vdots \\ x_k \end{bmatrix}$ A bias vector: b = $\begin{bmatrix} b_0 \\ \vdots \\ b_m \end{bmatrix}$ And a target vector of desired outputs: T = $\begin{bmatrix} T_0 \\ \vdots \\ T_m \end{bmatrix}$

(m = # of neurons-1 or outputs–1) (k = # of inputs-1 or weights per neurons-1)

$$\boldsymbol{Weighted\ Sum\ of\ Neuron\ i} = \boldsymbol{S_i = b_i + \sum_{j=0}^{k} W_{i,j} * x_j}$$

$$\boldsymbol{Weighted\ Sum\ of\ all\ Neurons\ (vector) = S = W * x + b}$$

$$\boldsymbol{Softmax\ output\ of\ Neuron\ i = O_i = y(i) = \frac{e^{S_i}}{\sum_{n=0}^{m} e^{S_n}}}$$

$$\boldsymbol{Softmax\ output\ of\ all\ Neurons\ (vector) = O = \begin{bmatrix} O_0 \\ \vdots \\ O_m \end{bmatrix}}$$

$$\boldsymbol{Cross\ Entropy\ Loss\ of\ all\ Neurons = H = -\sum_{f=0}^{m} T_f * \log\left(O_f\right)}$$

$$\boldsymbol{Cost\ Function = J = \frac{1}{m} * H}$$

What I need to do is minimize the Cost Function for all neurons. To do so, I need to find $\frac{\partial J}{\partial W_{r,c}}$ so that I can adjust the value of any weight with a given row (Neuron number) and column (input number) using the gradient of the loss function. I've tried finding the answer myself, and I got this:

$$\boldsymbol{\frac{\partial J}{\partial W_{r,c}} = \frac{-T_r * x_c * (-e^{S_r} + \sum_{n=0}^{m} e^{S_n})}{m * (\sum_{n=0}^{m} e^{S_n}) * \ln(10)}}$$

When comparing this with answers online, it's not the same, but I can't find out where my issue is. I believe I'm using the chain rule correctly, where $\boldsymbol{(f(g(z(x)))` = f`(g\left(\left(z(x)\right)\right) * g`\left(z(x)\right) * z`(x)}$, but my answer seems incorrect somehow, as the answer shown online turns out to be this:

$$\boldsymbol{\frac{\partial J}{\partial W_{r,c}} = \frac{1}{m}\sum_{i=0}^{m}(x_i * (O_i - T_i))}$$

This is where I'm getting lost, because 1.) I'm not sure how they got this answer and 2.) I don't see how it relates to the specific rows or columns which the weight is in. As far as I can tell the gradient will be the same for each weight, even when they have different values, as this function appears to return the same average for any given weight. I'm not sure if I've missed something in my original interpretation of the functions, so if you are interested, here is the link to the solution that I found:
http://rasbt.github.io/mlxtend/user_guide/classifier/SoftmaxRegression/