Gradients on Loss Functions

The formulation for the gradient on the loss function with respect to the weights arises a lot in machine learning as the following:

Consider a score matrix S, a weight matrix W, an input matrix X of compatible dimensions, and a scalar-valued loss function $L(S)$.

That is, we have:

$$\mathbf{S} = \mathbf{W}\mathbf{X}$$

I will provide intuition for the following relationships:

$$\frac{\partial L}{\partial \mathbf{W}_{ij}} = (\nabla_{\mathbf{X}}(L(\mathbf{X}))X^T))_{ij}$$

and

$$\frac{\partial L}{\partial \mathbf{X}_{ij}} = (\mathbf{W^T}\nabla_{\mathbf{W}}(L(\mathbf{W}))_{ij}$$

It is a simple formula but perhaps not so obvious. Here, I show where it comes from.

Also see Wikipedia on Scalar by Matrix.

For the following derivation, we have a bias of 0.

Setup

```
from sympy import *
```

Weight Matrix and Data Matrix

```
#Weight Matrix - 3 classes and dimensionality of two
W = MatrixSymbol('W',3,2)
W.is_real = True
#Data to be trained on - 3 examples
X = MatrixSymbol('X',2,3)
X.is_real = True
```

```
Matrix(W)
```

$$\begin{bmatrix} W_{0,0} & W_{0,1} \\ W_{1,0} & W_{1,1} \\ W_{2,0} & W_{2,1} \end{bmatrix}$$

```
Matrix(X)
```

$$\begin{bmatrix} X_{0,0} & X_{0,1} & X_{0,2} \\ X_{1,0} & X_{1,1} & X_{1,2} \end{bmatrix}$$

Classification

Y holds the classification for each vector in X. That is, vector X_0(column 0 of X) is labelled beforehand as belonging class 0, X_1 belongs to class 1 and so on...

```
Y = Matrix([0,1,2])
```

Score Matrix

```
S = Matrix(MatrixSymbol('s',3,3))
S
```

$$\begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} \\ s_{1,0} & s_{1,1} & s_{1,2} \\ s_{2,0} & s_{2,1} & s_{2,2} \end{bmatrix}$$

Computing the values of the Score Matrix S

```
S_expanded = W*X
Matrix(S_expanded)
```

$$\begin{bmatrix} W_{0,0}X_{0,0} + W_{0,1}X_{1,0} & W_{0,0}X_{0,1} + W_{0,1}X_{1,1} & W_{0,0}X_{0,2} + W_{0,1}X_{1,2} \\ W_{1,0}X_{0,0} + W_{1,1}X_{1,0} & W_{1,0}X_{0,1} + W_{1,1}X_{1,1} & W_{1,0}X_{0,2} + W_{1,1}X_{1,2} \\ W_{2,0}X_{0,0} + W_{2,1}X_{1,0} & W_{2,0}X_{0,1} + W_{2,1}X_{1,1} & W_{2,0}X_{0,2} + W_{2,1}X_{1,2} \end{bmatrix}$$

Loss Function

$L_i = -\log(\frac{e^{S_{y[i],i}}}{\sum_r e^{S_{r,i}}})$

$L = \frac{1}{N}\sum_i L_i$

Softmax computes the above loss on each column and then averages the columns.

The elements in a particular column of the score vector above represent the clasification scores for a particular input vector column from X.

$[S_{0,0}, S_{1,0}, S_{2,0}]$ is the first column in S. $S_{0,0}$ is the score that vector X_0 recieves for class 0. $S_{2,1}$ is the score that vector X_1 recieves for class 2 and so on.

Consider a column S_i from S; a perfect weight matrix would generate a 0 in the rows of that column that do not correspond with the label of the corresponding X column, and a nonzero value in the row that corresponds to the correct label.

The index of the correct label in column S_i is given by $S_{y[i],i}$ where y is defined above.

Compute denominator for each column

```
denoms = []
f_exp = lambda x : exp(x)
for col in range(3):
    denoms += [sum(Matrix(S[:,col]).applyfunc(f_exp))]
```

Column Sum Sanity Check

```
denoms[2]
```

$e^{s_{0,2}} + e^{s_{1,2}} + e^{s_{2,2}}$

Get numerator

The Y vector tells us which row from each column in the score matrix S is the ground truth(a.k.a the correct class).

The numerator of the operand of the -log in the loss formula is is e raised to the ground truth score.

```
truth_scores = []
for col in range(3):
    truth_scores += [S[Y[col],col]]
numers = [exp(val) for val in truth_scores]
```

Compute L_i

```
L_i = []
for i in range(3):
    L_i += [-(log(numers[i]) - log(denoms[i]))]
```

Sanity Check

```
L_i[0]
```

$\log\left(e^{s_{0,0}} + e^{s_{1,0}} + e^{s_{2,0}}\right) - \log\left(e^{s_{0,0}}\right)$

Total Loss

Below is the analytical expression for our total loss - note that we have a bias of 0 in this example.

Also note that the loss function is a function of the entries in the score matrix and the elements in the score matrix are in turn a function of the elements in the weight matrix.

```
from functools import reduce
from operator import add
Loss = Rational(1,3)*reduce(add, L_i)
Loss
```

$$\frac{\log\left(e^{s_{0,0}}+e^{s_{1,0}}+e^{s_{2,0}}\right)}{3}+\frac{\log\left(e^{s_{0,1}}+e^{s_{1,1}}+e^{s_{2,1}}\right)}{3}+\frac{\log\left(e^{s_{0,2}}+e^{s_{1,2}}+e^{s_{2,2}}\right)}{3}-\frac{\log\left(e^{s_{0,0}}\right)}{3}-\frac{\log\left(e^{s_{1,1}}\right)}{3}-\frac{\log\left(e^{s_{2,2}}\right)}{3}$$

Computing the Gradient with respect to:

Note: We substitute S for its expanded $W \times X$ form in the following computations.

```
S_flat = [el for sublist in S.tolist() for el in sublist]
S_expanded_flat = [el for sublist in Matrix(S_expanded).tolist() for el in sublist]
sublist_S_to_W = [(S_flat[i], S_expanded_flat[i]) for i in range(9)]
sublist_W_to_S = [(S_expanded_flat[i],S_flat[i]) for i in range(9)]
```

1 . $\frac{\partial L}{\partial W_{0,0}}$

```
diff(Loss.subs(sublist_S_to_W),W[0,0])
```

$$-\frac{e^{-W_{0,0}X_{0,0}-W_{0,1}X_{1,0}}e^{W_{0,0}X_{0,0}+W_{0,1}X_{1,0}}X_{0,0}}{3}+\frac{e^{W_{0,0}X_{0,2}+W_{0,1}X_{1,2}}X_{0,2}}{3\left(e^{W_{0,0}X_{0,2}+W_{0,1}X_{1,2}}+e^{W_{1,0}X_{0,2}+W_{1,1}X_{1,2}}+e^{W_{2,0}X_{0,2}+W_{2,1}X_{1,2}}\right)}+\frac{e^{W_{0,0}X_{0,1}+W_{0,1}X_{1,1}}X_{0,1}}{3\left(e^{W_{0,0}X_{0,1}+W_{0,1}X_{1,1}}+e^{W_{1,0}X_{0,1}+W_{1,1}X_{1,1}}+e^{W_{2,0}X_{0,1}+W_{2,1}X_{1,1}}\right)}+\frac{e^{W_{0,0}X_{0,0}+W_{0,1}X_{1,0}}X_{0,0}}{3\left(e^{W_{0,0}X_{0,0}+W_{0,1}X_{1,0}}+e^{W_{1,0}X_{0,0}+W_{1,1}X_{1,0}}+e^{}\right.}$$

As you should notice, we have a sum of rational terms with $X_{0,0}$, $X_{0,1}$, and $X_{0,2}$. This seems to imply a sort of dot product between row 0 of X and some other vector to get the gradient with respect to $W_{0,0}$.

2 . $\frac{\partial L}{\partial W_{1,0}}$

```
diff(Loss.subs(sublist_S_to_W),W[1,0])
```

$$-\frac{e^{-W_{1,0}X_{0,1}-W_{1,1}X_{1,1}}e^{W_{1,0}X_{0,1}+W_{1,1}X_{1,1}}X_{0,1}}{3}+\frac{e^{W_{1,0}X_{0,2}+W_{1,1}X_{1,2}}X_{0,2}}{3\left(e^{W_{0,0}X_{0,2}+W_{0,1}X_{1,2}}+e^{W_{1,0}X_{0,2}+W_{1,1}X_{1,2}}+e^{W_{2,0}X_{0,2}+W_{2,1}X_{1,2}}\right)}+$$
$$\frac{e^{W_{1,0}X_{0,1}+W_{1,1}X_{1,1}}X_{0,1}}{3\left(e^{W_{0,0}X_{0,1}+W_{0,1}X_{1,1}}+e^{W_{1,0}X_{0,1}+W_{1,1}X_{1,1}}+e^{W_{2,0}X_{0,1}+W_{2,1}X_{1,1}}\right)}+\frac{e^{W_{1,0}X_{0,0}+W_{1,1}X_{1,0}}X_{0,0}}{3\left(e^{W_{0,0}X_{0,0}+W_{0,1}X_{1,0}}+e^{W_{1,0}X_{0,0}+W_{1,1}X_{1,0}}+e\right.}$$

We again see a dependency on $X_{0,0}$, $X_{0,1}$, and $X_{0,2}$

3 . $\frac{\partial L}{\partial W_{0,1}}$

```
diff(Loss.subs(sublist_S_to_W),W[0,1])
```

$$-\frac{e^{-W_{0,0}X_{0,0}-W_{0,1}X_{1,0}}e^{W_{0,0}X_{0,0}+W_{0,1}X_{1,0}}X_{1,0}}{3}+\frac{e^{W_{0,0}X_{0,2}+W_{0,1}X_{1,2}}X_{1,2}}{3\left(e^{W_{0,0}X_{0,2}+W_{0,1}X_{1,2}}+e^{W_{1,0}X_{0,2}+W_{1,1}X_{1,2}}+e^{W_{2,0}X_{0,2}+W_{2,1}X_{1,2}}\right)}+$$
$$\frac{e^{W_{0,0}X_{0,1}+W_{0,1}X_{1,1}}X_{1,1}}{3\left(e^{W_{0,0}X_{0,1}+W_{0,1}X_{1,1}}+e^{W_{1,0}X_{0,1}+W_{1,1}X_{1,1}}+e^{W_{2,0}X_{0,1}+W_{2,1}X_{1,1}}\right)}+\frac{e^{W_{0,0}X_{0,0}+W_{0,1}X_{1,0}}X_{1,0}}{3\left(e^{W_{0,0}X_{0,0}+W_{0,1}X_{1,0}}+e^{W_{1,0}X_{0,0}+W_{1,1}X_{1,0}}+e\right.}$$

We now see a dependency on $X_{1,0}$, $X_{1,1}$, and $X_{1,2}$

Score Matrix

Lets take a look at the gradient with respect to the score matrix. We accomplish this by substituing $S_{i,j} = W_{i,0}X_{0,j} + W_{i,1}X_{1,j}$

```
diff(Loss.subs(sublist_S_to_W),W[0,1]).subs(sublist_W_to_S)
```

$$-\frac{X_{1,0}}{3}+\frac{e^{s_{0,2}}X_{1,2}}{3\left(e^{s_{0,2}}+e^{s_{1,2}}+e^{s_{2,2}}\right)}+\frac{e^{s_{0,1}}X_{1,1}}{3\left(e^{s_{0,1}}+e^{s_{1,1}}+e^{s_{2,1}}\right)}+\frac{e^{s_{0,0}}X_{1,0}}{3\left(e^{s_{0,0}}+e^{s_{1,0}}+e^{s_{2,0}}\right)}$$

As you can see, in general, the gradient, $\frac{\partial L}{\partial W_{i,j}}$ is a function of the score matrix S, and the input matrix X.

A Closer Look

If you play around a bit more, you'll notice the $-\frac{X_{i,j}}{3}$ term that shows up whenever we differentiate with respect to $W_{i,j}$. This is no coincidence.

Let's take a look at $\frac{\partial L_0}{\partial W_{0,0}}$ - that is, the gradient of the loss on the first column in the score matrix with respect to $W_{0,0}$

```
diff(Rational(1,3)*L_i[0].subs(sublist_S_to_W), W[0,0]).subs(sublist_W_to_S)
```

$$-\frac{X_{0,0}}{3}+\frac{e^{s_{0,0}}X_{0,0}}{3\left(e^{s_{0,0}}+e^{s_{1,0}}+e^{s_{2,0}}\right)}$$

We see that $-\frac{X_{0,0}}{3}$ shows up in the first term!

We know that the entire loss function gradient with respect to $W_{0,0}$ is composed of the sum of the loss gradient on each score column again with respect to $W_{0,0}$. Another way to say this is:

$\frac{\partial L}{\partial W_{0,0}} = \frac{\partial L_0}{\partial W_{0,0}} + \frac{\partial L_1}{\partial W_{0,0}} + \cdot \frac{\partial L_2}{\partial W_{0,0}}$

To understand where that term comes from, we must look more closely at $L_i = -\log(\frac{e^{S_{y[i],i}}}{\sum_r e^{S_r,i}}) = \log(e^{S_{y[i],i}}) - \log(\sum_r e^{S_r,i})$.

Playing with L_0

We have $L_0 = \log(e^{S_{y[0],0}}) - \log(\sum_r e^{S_r,0})$.

$= \log(e^{S_{y[0],0}}) - \log(e^{S_{0,0}} + e^{S_{1,0}} + e^{S_{2,0}})$

$= \log(e^{S_{0,0}}) - \log(e^{S_{0,0}} + e^{S_{1,0}} + e^{S_{2,0}})$

Taking the derivative with respect to $S_{0,0}$ we have:

$\frac{\partial L_0}{\partial S_{0,0}} = 1 - \frac{e^{S_{0,0}}}{e^{S_{0,0}} + e^{S_{1,0}} + e^{S_{2,0}}}$

And taking the derivative with respect to $S_{1,0}$ we have:

$\frac{\partial L_0}{\partial S_{0,0}} = -\frac{e^{S_{0,0}}}{e^{S_{1,0}} + e^{S_{1,0}} + e^{S_{2,0}}}$

So we see that the 1 term shows up when we differentiate with respect to $S_{y[i],i}$.

This 1 becomes $\frac{1}{3}$ when we divide by the number of classes N.

Wrapping it all up

Looking at the expanded expression for $\frac{\partial L}{\partial W_{0,0}}$, $\frac{\partial L}{\partial W_{1,0}}$, and $\frac{\partial L}{\partial W_{0,1}}$ above suggest that there is a dot product between row of X and the gradient of the Loss function over a column of S.

This is indeed true and careful inspection captures our observation in the following nice Matrix operation.

$$\begin{bmatrix} \frac{\partial L}{\partial W_{00}} & \frac{\partial L}{\partial W_{01}} \\ \frac{\partial L}{\partial W_{10}} & \frac{\partial L}{\partial W_{11}} \\ \frac{\partial L}{\partial W_{20}} & \frac{\partial L}{\partial W_{21}} \end{bmatrix} = \begin{bmatrix} \frac{\partial L}{\partial S_{00}} & \frac{\partial L}{\partial S_{01}} & \frac{\partial L}{\partial S_{02}} \\ \frac{\partial L}{\partial S_{10}} & \frac{\partial L}{\partial S_{11}} & \frac{\partial L}{\partial S_{12}} \\ \frac{\partial L}{\partial S_{20}} & \frac{\partial L}{\partial S_{21}} & \frac{\partial L}{\partial S_{22}} \end{bmatrix} X^T$$

Here we confirm that $\frac{\partial L}{\partial W_{0,0}}$ is indeed equal to element $(0, 0)$ of

$$\begin{bmatrix} \frac{\partial L}{\partial S_{00}} & \frac{\partial L}{\partial S_{01}} & \frac{\partial L}{\partial S_{02}} \\ \frac{\partial L}{\partial S_{10}} & \frac{\partial L}{\partial S_{11}} & \frac{\partial L}{\partial S_{12}} \\ \frac{\partial L}{\partial S_{20}} & \frac{\partial L}{\partial S_{21}} & \frac{\partial L}{\partial S_{22}} \end{bmatrix} X^T$$

```
#elementwise derivate of Loss function with respect to elements of
#score matrix as shown above
```

```
dLoss_dX = lambda x : diff(Loss,x)
dL_dS = S.applyfunc(dLoss_dX)
```

```
dW = dL_dS*Matrix(X.T)
```

```
expand(diff(Loss.subs(sublist_S_to_W),W[0,0])) - expand(dW[0,0].subs(sublist_S_to_W))
```

0

Numerically evaluating dW would give us an incremental update to add to our weight matrix in order for us to minimize our loss function.

Closing Remarks

What if we wish to take the derivative of the loss function wit respect to $X_{0,0}$?

We can do the following:

$$\begin{bmatrix} \frac{\partial L}{\partial X_{00}} & \frac{\partial L}{\partial X_{01}} & \frac{\partial L}{\partial X_{02}} \\ \frac{\partial L}{\partial X_{10}} & \frac{\partial L}{\partial X_{11}} & \frac{\partial L}{\partial X_{12}} \end{bmatrix} = W^T \begin{bmatrix} \frac{\partial L}{\partial S_{00}} & \frac{\partial L}{\partial S_{01}} & \frac{\partial L}{\partial S_{02}} \\ \frac{\partial L}{\partial S_{10}} & \frac{\partial L}{\partial S_{11}} & \frac{\partial L}{\partial S_{12}} \\ \frac{\partial L}{\partial S_{20}} & \frac{\partial L}{\partial S_{21}} & \frac{\partial L}{\partial S_{22}} \end{bmatrix}$$

Again, here we confirm that $\frac{\partial L}{\partial X_{0,0}}$ is indeed equal to element $(0,0)$ of $W^T \begin{pmatrix} \frac{\partial L}{\partial S_{00}} & \frac{\partial L}{\partial S_{01}} & \frac{\partial L}{\partial S_{02}} \\ \frac{\partial L}{\partial S_{10}} & \frac{\partial L}{\partial S_{11}} & \frac{\partial L}{\partial S_{12}} \\ \frac{\partial L}{\partial S_{20}} & \frac{\partial L}{\partial S_{21}} & \frac{\partial L}{\partial S_{22}} \end{pmatrix}$

```
dX = (Matrix(W.T)*dL_dS)
```

```
expand(diff(Loss.subs(sublist_S_to_W),X[0,0])) - expand(dX[0,0].subs(sublist_S_to_W))
```

0