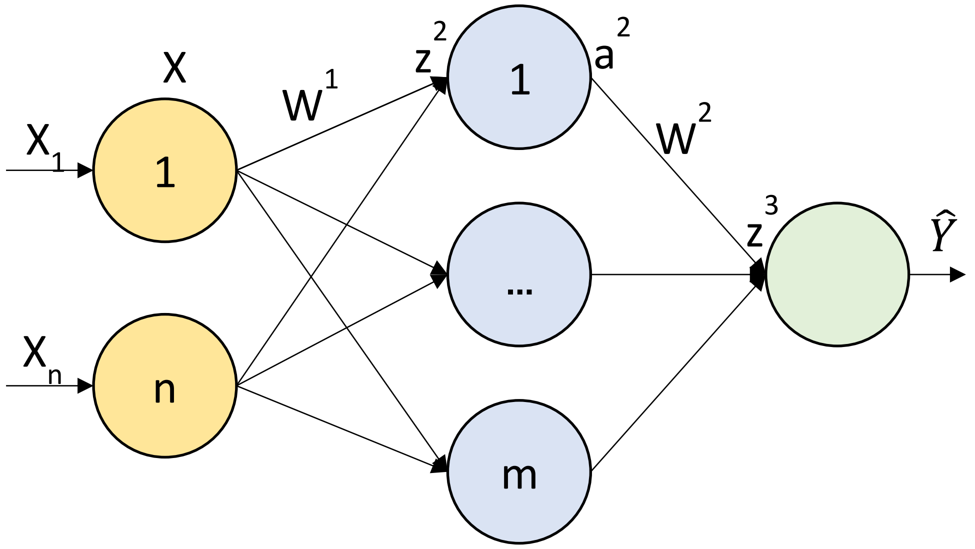
Neural Networks – The Graph Approach

1. Introduction

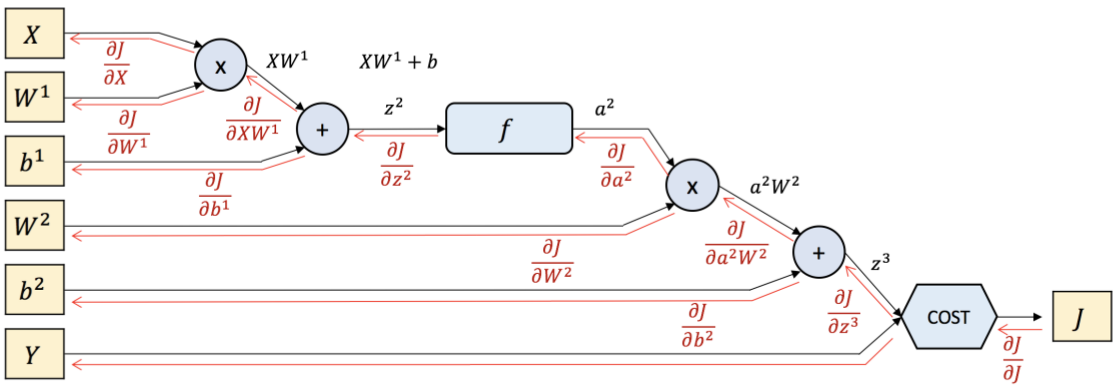
As we always do, let's first introduce the scenario of the neural network we are working with:

Figure 1. Artificial Neural Network with shape [2, 3, 1]



and its graph representation that we achieved in chapter 3 (LINK)

Figure 2. Graph representation of out [2, 3, 1] neural network.



We need to keep track of what we did last chapter, as the outputs of last chapters are the inputs of this one. So, let's also take back the tools we need to proceed.

$X: (4x2)$

$x_1(1)$	$x_2(1)$
$x_1(2)$	$x_2(2)$
$x_1(3)$	$x_2(3)$
$x_1(4)$	$x_2(4)$

$z^2: (4x3)$

$z_1^2(1)$	$z_2^2(1)$	$z_3^2(1)$
$z_1^2(2)$	$z_2^2(2)$	$z_3^2(2)$
$z_1^2(3)$	$z_2^2(3)$	$z_3^2(3)$
$z_1^2(4)$	$z_2^2(4)$	$z_3^2(4)$

$a^2: (4x3)$

$a_1^2(1)$	$a_2^2(1)$	$a_3^2(1)$
$a_1^2(2)$	$a_2^2(2)$	$a_3^2(2)$
$a_1^2(3)$	$a_2^2(3)$	$a_3^2(3)$
$a_1^2(4)$	$a_2^2(4)$	$a_3^2(4)$

$z^3: (4x1)$

$z_1^3(1)$
$z_1^3(2)$
$z_1^3(3)$
$z_1^3(4)$

$W^1: (2x3)$

W_{11}^1	W_{12}^1	W_{13}^1
W_{21}^1	W_{22}^1	W_{23}^1

$W^2: (3x1)$

W_1^2
W_2^2
W_3^2

$J: (4x1)$

$J(1)$
$J(2)$
$J(3)$
$J(4)$

We already know where these values come from as we defined the shapes of does matrices depending on the dimensions of the input, layer, batch and output. We will include also the equation once more to make it easier to follow:

$$\frac{dJ}{dz^3} = -= -\frac{2}{n} \cdot y \cdot (\mathrm{z}^3 - \mathrm{y}) = \boldsymbol{top_{diff}} \quad \text{(Eq. B1)}$$

$$\frac{dJ}{da^3W^2} = \frac{dJ}{dz^3} \cdot \frac{dz^3}{da^3W^2} = \boldsymbol{top_{diff}} \cdot \boldsymbol{local_{diff}} = top_{diff} \cdot 1 \quad \text{(Eq. B2)}$$

$$\frac{dJ}{db^2} = \frac{dJ}{dz^3} \cdot \frac{dz^3}{db^2} = \boldsymbol{top_{diff}} \cdot \boldsymbol{local_{diff}} = top_{diff} \cdot 1 \quad \text{(Eq. B3)}$$

$$\frac{dJ}{da^2} = \frac{dJ}{dz^3} \cdot \frac{dz^3}{da^2W^2} \cdot \frac{da^2W^2}{da^2} = \boldsymbol{top_{diff}} \cdot \boldsymbol{way_{here}} \cdot \boldsymbol{local_{diff}}$$
$$= topp_{diff} \cdot 1 \cdot W^2 = topp_{diff} \cdot W^2 \quad \text{(Eq. B4)}$$

$$\boldsymbol{\frac{dJ}{dW^2}} = \frac{dJ}{dz^3} \cdot \frac{dz^3}{da^2W^2} \cdot \frac{da^2W^2}{dW^2} = \boldsymbol{top_{diff}} \cdot \boldsymbol{way_{here}} \cdot \boldsymbol{local_{diff}}$$
$$= topp_{diff} \cdot 1 \cdot a^2 = topp_{diff} \cdot a^2 \quad \text{(Eq. B5)}$$

$$\frac{dJ}{dz^2} = \frac{dJ}{dz^3} \cdot \frac{dz^3}{da^2W^2} \cdot \frac{da^2W^2}{da^2} \cdot \frac{da^2}{dz^2} = \boldsymbol{top_{diff}} \cdot \boldsymbol{way_{here}} \cdot \boldsymbol{local_{diff}}$$
$$= topp_{diff} \cdot W^2 \cdot f'(\mathrm{z}^2) \quad \text{(Eq. B6)}$$

$$\frac{dJ}{dXW^1} = \frac{dJ}{dz^3} \cdot \frac{dz^3}{da^2W^2} \cdot \frac{da^2W^2}{da^2} \cdot \frac{da^2}{dz^2} \cdot \frac{dz^2}{dXW^1}$$
$$= \boldsymbol{top_{diff}} \cdot \boldsymbol{way_{here}} \cdot \boldsymbol{local_{diff}}$$
$$= topp_{diff} \cdot W^2 \cdot f'(\mathrm{z}^2) \cdot 1 \quad \text{(Eq. B7)}$$

$$\frac{dJ}{db^1} = \frac{dJ}{dz^3} \cdot \frac{dz^3}{da^2W^2} \cdot \frac{da^2W^2}{da^2} \cdot \frac{da^2}{dz^2} \cdot \frac{dz^2}{db^1} = \boldsymbol{top_{diff} \cdot way_{here} \cdot local_{diff}}$$
$$= topp_{diff} \cdot W^2 \cdot f'(z^2) \cdot 1 \qquad \text{(Eq. B8)}$$

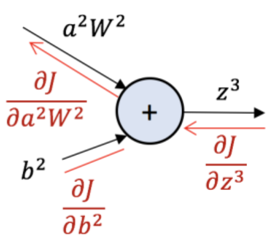
$$\frac{dJ}{dX} = \frac{dJ}{dz^3} \cdot \frac{dz^3}{da^2W^2} \cdot \frac{da^2W^2}{da^2} \cdot \frac{da^2}{dz^2} \cdot \frac{dz^2}{dX} = \boldsymbol{top_{diff} \cdot way_{here} \cdot local_{diff}}$$
$$= topp_{diff} \cdot W^2 \cdot f'(z^2) \cdot W^1 \qquad \text{(Eq. B9)}$$

$$\boldsymbol{\frac{dJ}{dW^1}} = \frac{dJ}{dz^3} \cdot \frac{dz^3}{da^2W^2} \cdot \frac{da^2W^2}{da^2} \cdot \frac{da^2}{dz^2} \cdot \frac{dz^2}{dW^1}$$
$$= \boldsymbol{top_{diff} \cdot way_{here} \cdot local_{diff}}$$
$$= topp_{diff} \cdot W^2 \cdot f'(z^2) \cdot X \qquad \text{(Eq. B10)}$$

2. Dimensionality in backpropagation

What we are going to cover in this last chapter is making the same visualization of chapter 2, over the BackProp process in the Graph we did in chapter 3 described by the previous formulas.

The first thing we encounter backpropagating is an addGate:

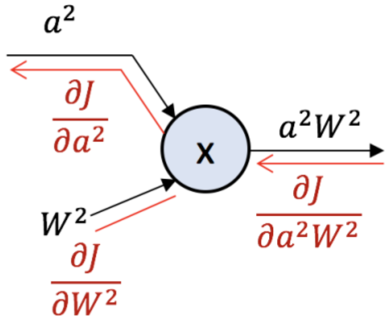


$$\frac{dJ}{db^2} = \frac{dJ}{dz^3} \cdot \frac{dz^3}{db^2} = \begin{bmatrix} dJ(1)/dz^3 \\ dJ(2)/dz^3 \\ dJ(3)/dz^3 \\ dJ(4)/dz^3 \end{bmatrix} \cdot \begin{bmatrix} dz^3/db^2 \\ dz^3/db^2 \\ dz^3/db^2 \\ dz^3/db^2 \end{bmatrix} = \begin{bmatrix} dJ(1)/dz^3 \\ dJ(2)/dz^3 \\ dJ(3)/dz^3 \\ dJ(4)/dz^3 \end{bmatrix} = \underset{(4x1)}{\begin{bmatrix} \delta^3(1) \\ \delta^3(2) \\ \delta^3(3) \\ \delta^3(4) \end{bmatrix}}$$

$$\frac{dJ}{da^2W^2} = \frac{dJ}{dz^3} \cdot \frac{dz^3}{da^2W^2} = \begin{bmatrix} dJ(1)/da^2W^2 \\ dJ(2)/da^2W^2 \\ dJ(3)/da^2W^2 \\ dJ(4)/a^2W^2 \end{bmatrix} \cdot \begin{bmatrix} dz^3/da^2W^2 \\ dz^3/da^2W^2 \\ dz^3/da^2W^2 \\ dz^3/da^2W^2 \end{bmatrix} = \begin{bmatrix} \delta^3(1) \\ \delta^3(2) \\ \delta^3(3) \\ \delta^3(4) \end{bmatrix}$$

We already know that addGates act like distributors, and that the local derivative in both cases is 1. The result we can call it from now on the ‘Back-Propagated Error’ δ.

Next step is a mulGate:



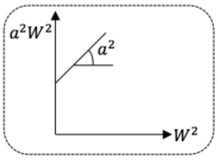
$$\boldsymbol{\frac{dJ}{dW^2}} = \frac{dJ}{dz^3} \cdot \frac{dz^3}{da^2W^2} \cdot \frac{da^2W^2}{dW^2} = \begin{bmatrix} \delta^3(1) \\ \delta^3(2) \\ \delta^3(3) \\ \delta^3(4) \end{bmatrix} \cdot \frac{da^2W^2}{dW^2} =$$

$$\frac{dJ}{da^2} = \frac{dJ}{dz^3} \cdot \frac{dz^3}{da^2W^2} \cdot \frac{da^2W^2}{da^2} = \begin{bmatrix} \delta^3(1) \\ \delta^3(2) \\ \delta^3(3) \\ \delta^3(4) \end{bmatrix} \cdot \frac{da^2W^2}{da^2} =$$

(*) We have applied the whole chain rule since J. From now on we will simplify it with only the top diff and the local gradient.

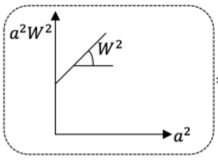
And now we have to determine these derivatives on the right of both equations. Look that, how is represented in the LHS in Figure 3 there is a linear relationship between both terms where we can identify the slope. Mathematically, we achieve that operation by transposing the matrix.

Figure 3. Back-Propagation Error through Connection between Layers



$$(a^2)^T\,\delta^3 = (3x1)$$

$$\Rightarrow (a^2)^T\,\delta^3 = \begin{bmatrix} a_1^2(1) & a_1^2(2) & a_1^2(3) & a_1^2(4) \\ a_2^2(1) & a_2^2(2) & a_2^2(3) & a_2^2(4) \\ a_3^2(1) & a_3^2(2) & a_3^2(3) & a_3^2(4) \end{bmatrix} \cdot \begin{bmatrix} \delta^3(1) \\ \delta^3(2) \\ \delta^3(3) \\ \delta^3(4) \end{bmatrix} = \begin{bmatrix} a_1^2(1)\delta^3(1) + a_1^2(2)\delta^3(2) + a_1^2(3)\delta^3(3) + a_1^2(4)\delta^3(4) \\ a_2^2(1)\delta^3(1) + a_2^2(2)\delta^3(2) + a_2^2(3)\delta^3(3) + a_2^2(4)\delta^3(4) \\ a_3^2(1)\delta^3(1) + a_3^2(2)\delta^3(2) + a_3^2(3)\delta^3(3) + a_3^2(4)\delta^3(4) \end{bmatrix}$$



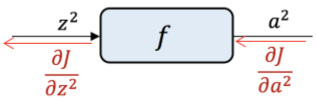
$$\delta^3(W^2)^T = (4x3)$$

$$\Rightarrow \delta^3(W^2)^T = \begin{bmatrix} \delta^3(1) \\ \delta^3(2) \\ \delta^3(3) \\ \delta^3(4) \end{bmatrix} \cdot \begin{bmatrix} W_1^2 & W_2^2 & W_3^2 \end{bmatrix} = \begin{bmatrix} \delta^3(1)W_1^2 & \delta^3(1)W_2^2 & \delta^3(1)W_3^2 \\ \delta^3(2)W_1^2 & \delta^3(2)W_2^2 & \delta^3(2)W_3^2 \\ \delta^3(3)W_1^2 & \delta^3(3)W_2^2 & \delta^3(3)W_3^2 \\ \delta^3(4)W_1^2 & \delta^3(4)W_2^2 & \delta^3(4)W_3^2 \end{bmatrix}$$

If we translate this to human language and try to think about what is happening with Figure 1 in mind, we could say:

1. We are averaging the product $a^2\delta$ over the different iterations to define the ratio of change of a^2W^2 with respect to W^2. That is why we have 3 dimensions, one for each output of the hidden neurons; and each of them is adding the error committed for the different observations.
2. We are splitting the back-propagated error of each observation to the different hidden neurons, that's why we have (4x3) matrix (4 observations and 3 neurons).

Figure 4. Back-Propagation through Activated Neuron



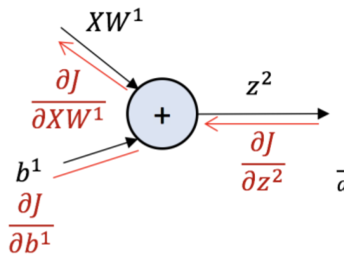
$$\frac{dJ}{dz^2} = \frac{dJ}{da^2} \cdot \frac{da^2}{dz^2} = \delta^3(W^2)^T \cdot f'^{(z^2)} =$$

$$\delta^3(W^2)^T \cdot f'(z^2) = (4x3)$$

$$\begin{bmatrix} \delta^3(1)W_1^2 \\ \delta^3(2)W_1^2 \\ \delta^3(3)W_1^2 \\ \delta^3(4)W_1^2 \end{bmatrix} \begin{bmatrix} d\left(f(z^3(1))\right)/dz^3 & d\left(f(z^3(1))\right)/dz^3 & d\left(f(z^3(1))\right)/dz^3 \\ d\left(f(z^3(2))\right)/dz^3 & d\left(f(z^3(2))\right)/dz^3 & d\left(f(z^3(2))\right)/dz^3 \\ d\left(f(z^3(3))\right)/dz^3 & d\left(f(z^3(3))\right)/dz^3 & d\left(f(z^3(3))\right)/dz^3 \\ d\left(f(z^3(4))\right)/dz^3 & d\left(f(z^3(4))\right)/dz^3 & d\left(f(z^3(4))\right)/dz^3 \end{bmatrix} = \begin{bmatrix} \delta^3(1)\cdot da(1)/dz^3 & \delta^3(1)\cdot da(1)/dz^3 & \delta^3(1)\cdot da(1)/dz^3 \\ \delta^3(2)\cdot da(2)/dz^3 & \delta^3(2)\cdot da(2)/dz^3 & \delta^3(2)\cdot da(2)/dz^3 \\ \delta^3(3)\cdot da(3)/dz^3 & \delta^3(3)\cdot da(3)/dz^3 & \delta^3(3)\cdot da(3)/dz^3 \\ \delta^3(4)\cdot da(4)/dz^3 & \delta^3(4)\cdot da(4)/dz^3 & \delta^3(4)\cdot da(4)/dz^3 \end{bmatrix}$$

It is very important to see that in this occasion, we have done a scalar product instead of a product between matrices. This is because is the forward pass, the activation function was an element-wise operation, so is in the backward pass.

In the next step, we face another addGate:



$$\frac{dJ}{db^1} = \frac{dJ}{dz^2} \cdot \frac{dz^2}{db^1} = \delta^3 (W^2)^T \cdot f'^{(z^2)} \cdot \frac{dz^2}{db^1}$$

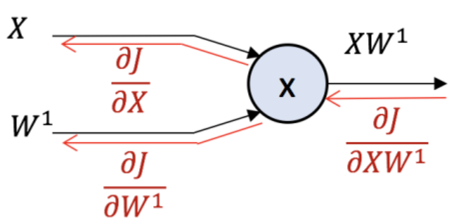
$$\frac{dJ}{dXW^1} = \frac{dJ}{dz^2} \cdot \frac{dz^2}{dXW^1} = \delta^3 (W^2)^T \cdot f'^{(z^2)} \cdot \frac{dz^2}{dXW^1}$$

$$\underset{(4x3)}{\begin{bmatrix} \delta_1^2(1) & \delta_2^2(1) & \delta_3^2(1) \\ \delta_1^2(2) & \delta_2^2(2) & \delta_3^2(2) \\ \delta_1^2(3) & \delta_2^2(3) & \delta_3^2(3) \\ \delta_1^2(4) & \delta_2^2(4) & \delta_3^2(4) \end{bmatrix}}$$

See how, as we know that addGates are distributors in the backward pass, both propagations are going to be the same value. What is happening is that in the case of dJ/dW^1 we are multiplying the top derivative coming by a $(4x3)$ matrix of ones, again element-wise, so scalar product. And for the dJ/dW the top derivative multiplies a $(4x1)$ matrix of ones, leading the same result. We call this step the back-propagated error at layer 2, δ^2.

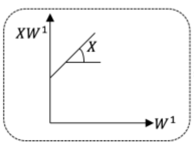
We can find another pattern then! δ^3 was a $(4x1)$ matrix and δ^2 is a $(4x1)$ matrix. Then we can see how the back-propagated error at a layer i is going to have dimensions (kxm^i) where m^i is going to be the number of neurons on that layer i. Great!

We have our last step, where we face again another mulGate, but this is too easy for us now right? We know that they are switchers, and that there are going to be some transposes, but that is not a big deal for us.



$$\boldsymbol{\frac{dJ}{dW^1}} = \frac{dJ}{dz^2} \cdot \frac{dz^2}{dW^1} = \begin{bmatrix} \delta_1^2(1) & \delta_2^2(1) & \delta_3^2(1) \\ \delta_1^2(2) & \delta_2^2(2) & \delta_3^2(2) \\ \delta_1^2(3) & \delta_2^2(3) & \delta_3^2(3) \\ \delta_1^2(4) & \delta_2^2(4) & \delta_3^2(4) \end{bmatrix} \cdot \frac{dz^2}{dW^1} =$$

$$\frac{dJ}{dX} = \frac{dJ}{dz^2} \cdot \frac{dz^2}{dX} = \begin{bmatrix} \delta_1^2(1) & \delta_2^2(1) & \delta_3^2(1) \\ \delta_1^2(2) & \delta_2^2(2) & \delta_3^2(2) \\ \delta_1^2(3) & \delta_2^2(3) & \delta_3^2(3) \\ \delta_1^2(4) & \delta_2^2(4) & \delta_3^2(4) \end{bmatrix} \cdot \frac{dz^2}{dX} =$$



$$=> (X)^T \delta^2 = \begin{bmatrix} x_1(1) & x_1(2) & x_1(3) & x_1(4) \\ x_2(1) & x_2(2) & x_2(3) & x_2(4) \end{bmatrix} \cdot \begin{bmatrix} \delta_1^2(1) & \delta_2^2(1) & \delta_3^2(1) \\ \delta_1^2(2) & \delta_2^2(2) & \delta_3^2(2) \\ \delta_1^2(3) & \delta_2^2(3) & \delta_3^2(3) \\ \delta_1^2(4) & \delta_2^2(4) & \delta_3^2(4) \end{bmatrix} =$$

$x_1(1)\delta_1^2(1) + x_1(2)\delta_1^2(2) + x_1(3)\delta_1^2(3) + x_1(4)\delta_1^2(4)$	$x_1(1)\delta_2^2(1) + x_1(2)\delta_2^2(2) + x_1(3)\delta_2^2(3) + x_1(4)\delta_2^2(4)$	$x_1(1)\delta_3^2(1) + x_1(2)\delta_3^2(2) + x_1(3)\delta_3^2(3) + x_1(4)\delta_3^2(4)$
$x_2(1)\delta_1^2(1) + x_2(2)\delta_1^2(2) + x_2(3)\delta_1^2(3) + x_4(4)\delta_1^2(4)$	$x_2(1)\delta_2^2(1) + x_2(2)\delta_2^2(2) + x_2(3)\delta_2^2(3) + x_4(4)\delta_2^2(4)$	$x_2(1)\delta_3^2(1) + x_2(2)\delta_3^2(2) + x_2(3)\delta_3^2(3) + x_4(4)\delta_3^2(4)$

The procedure for the derivate respect to X you can figure it out. As it is not different and, specially because we are not interested on it, such as we cannot update the value of the input (unfortunately this is not in our hands) and because we are not going to back-propagate it to anywhere else.

The last transpose of X is achieved to average every back-propagated error of each observation with every input at each observation for every neuron in the hidden layer (because each i neuron is telling backward that δ_i^2.

3. Updating the weights

We should now go forward for the last step, the learning. If we bring back our equation from past chapters, the weights were being updated:

$$W^i = W^i - \lambda_i \cdot \frac{dJ}{dW^i}$$

Therefore, applying this to what we have for our two layers of weights:

Figure 5.Updated weights

W^1=

$W_{11}^1 \cdot (1 - \{x_1(1)\delta_1^2(1) + x_1(2)\delta_1^2(2) + x_1(3)\delta_1^2(3) + x_1(4)\delta_1^2(4)\})$	$W_{12}^1 \cdot (1 - \{x_1(1)\delta_2^2(1) + x_1(2)\delta_2^2(2) + x_1(3)\delta_2^2(3) + x_1(4)\delta_2^2(4)\})$	$W_{13}^1 \cdot (1 - \{x_1(1)\delta_3^2(1) + x_1(2)\delta_3^2(2) + x_1(3)\delta_3^2(3) + x_1(4)\delta_3^2(4)\})$
$W_{21}^1 \cdot (1 - \{x_2(1)\delta_1^2(1) + x_2(2)\delta_1^2(2) + x_2(3)\delta_1^2(3) + x_4(4)\delta_1^2(4)\})$	$W_{22}^1 \cdot (1 - \{x_2(1)\delta_2^2(1) + x_2(2)\delta_2^2(2) + x_2(3)\delta_2^2(3) + x_4(4)\delta_2^2(4)\})$	$W_{23}^1 \cdot (1 - \{x_2(1)\delta_3^2(1) + x_2(2)\delta_3^2(2) + x_2(3)\delta_3^2(3) + x_4(4)\delta_3^2(4)\})$

W^2 =

$W_1^2(1 - \{a_1^2(1)\delta^3(1) + a_1^2(2)\delta^3(2) + a_1^2(3)\delta^3(3) + a_1^2(4)\delta^3(4)\})$
$W_2^2(1 - \{a_2^2(1)\delta^3(1) + a_2^2(2)\delta^3(2) + a_2^2(3)\delta^3(3) + a_2^2(4)\delta^3(4)\})$
$W_3^2(1 - \{a_3^2(1)\delta^3(1) + a_3^2(2)\delta^3(2) + a_3^2(3)\delta^3(3) + a_3^2(4)\delta^3(4)\})$