Theo Guidroz
CS 577

Assignment 2

Artificial neurons

1) $g(x_1, x_2) = \theta^T x + \theta_0$
$= (0.2 \times 1) + (0.3 \times 1) + 1$
$= 0.6 \leftarrow$

2) On one side of the boundary, the value should be greater than 0. On the other side, less than 0. On the boundary, 0.

3) θ_0 is the bias, or the negative distance from the origin.
θ_1, θ_2 are the weights and make up for the normal vector

4) normal = $[2, 3]$

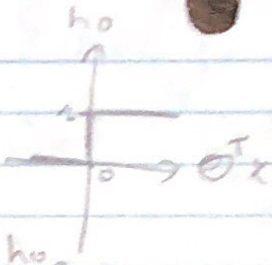
normal normalized = $\frac{1}{\sqrt{13}} [2, 3]$

Distance = $\frac{1}{\sqrt{13}}$

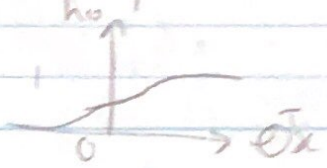
5) Augment the x vector by one and add 1 to the beginning of the vector.

$\theta^T x + \theta_0 = \tilde{\theta}^T$ where $\tilde{\theta} = [\overbrace{\theta_0}^{\theta \text{ bias}}, \ldots \theta_n]$
$\tilde{x} = [1, x_1, \ldots, x_n]$

6) Step function: $h_\theta = \begin{cases} 1 & \text{if } \theta^T x > 0 \\ 0 & \text{otherwise} \end{cases}$



Sigmoid function: $h_\theta = \frac{1}{1+e^{-\theta^T x}}$



Sigmoid function is a continuous function so it outputs values between 0 and 1 whereas step function only output 0 or 1 when θ is small, the graph ressembles more like a straight line increasing from 0 to 1. As θ increases, it ressembles more an more as a step function.

7) $P(y=1|x) > 1 \rightarrow C_1$
$P(y=0|x) < 1 \rightarrow C_0$

let C_0 and C_1 be 2 labels

$$\log\left(\frac{P(y=1|x)}{P(y=0|x)}\right) \begin{matrix} > 0 \rightarrow C_1 \\ < 0 \rightarrow C_0 \end{matrix}$$

$$\log\left(\frac{P(y=1|x)}{P(y=0|x)}\right) = \theta^T x$$

↑ model as a linear function

$$\therefore \frac{P(y=1|x)}{P(y=0|x)} = \exp(\theta^T x)$$

$$P(y=1|x) = (1 - P(y=1|x)) \exp(\theta^T x)$$
$$P(y=1|x)(1+\exp(\theta^T x)) = \exp(\theta^T x)$$
$$P(y=1|x) = \frac{\exp(\theta^T x)}{1+\exp(\theta^T x)} = \frac{1}{1+\exp(-\theta^T x)}$$

↳ Sigmoid

8) $\text{sigmoid}(x) = \frac{1}{1+\exp(-x)}$

$\frac{d}{dx}\text{sigmoid}(x) = \text{sigmoid}(x)(1-\text{sigmoid}(x))$

$\frac{d}{dx}\log(\text{sigmoid}(x)) = \frac{1}{\text{sigmoid}(x)} \cdot \text{sigmoid}(x)(1-\text{sigmoid}(x))$

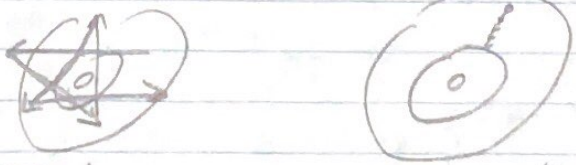
$\frac{d}{dx}\log(\text{sigmoid}(x)) = (1-\text{sigmoid}(x)) \leftarrow$

9) $\theta^{(i+1)} \leftarrow \theta^{(i)} - \eta \nabla J(\theta^{(i)})$ (η: learning rate)

After initially guessing the parameters (θ), take the gradient at the solution and subtract it from the parameter. The size of the update is controlled by the learning rate.

10) Stop : $(J(\theta^{(i+1)}) - J(\theta^{(i)})) < \tau$ where τ is the threshold. It should use the loss change since the goal is to minimize loss, regardless of the change in parameters.

11)

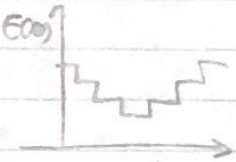


Too large Too small

If the learning rate is too large, the function might not converge or if it does, it might skip the global minimum and settle for a local maximum

If the learning rate is too small, the steps taken by the function will be small and time consuming. The model might settle at a local minima and the value of τ could be reached prematurely.

12) Empirical error loss is (incorrect labels) computed by summing the number of errors at a given parameter. Since we are using count as a feature, the graph would be discrete and would resemble something as such :



Therefore, the gradient at a certain point could be zero and therefore, the formula of gradient descent would be unusable ($\theta^{(i+1)} \leftarrow \theta^{(i)} - \eta \nabla J(\theta^{(i)})$ → 0)

13) $$l(\theta) = \log\left[\prod_{x^{(i)} \in C_1} P(y=1 \mid x^{(i)}) \prod_{x^{(i)} \in C_0} P(y=0 \mid x^{(i)})\right]$$

log ← likelihood

$$= \log \prod_{i=1}^{m} P(y=1 \mid x^{(i)})^{y^{(i)}} P(y=0 \mid x^{(i)})^{1-y^{(i)}}$$

$$= \sum_{i=1}^{m} y^{(i)} \log\left(\overbrace{P(y=1 \mid x^{(i)})}^{h_\theta(x)}\right) + (1-y^{(i)}) \log\left(1 - \overbrace{P(y=1 \mid x^{(i)})}^{h_\theta(x)}\right)$$

$$l(\theta) = \sum_{i=1}^{m} y^{(i)} \log(h_\theta(x^{(i)})) + (1-y^{(i)}) \log(1 - h_\theta(x^{(i)}))$$

↑ Binary Cross-entropy

14) $\frac{d}{d\theta}\left(l(\theta) = \frac{d}{d\theta}\sum_{i=1}^{m} y^{(i)}\log(h_\theta(x)) + (1-y^{(i)})\log(1-h_\theta(x)\right)$

$$= \sum_{i=1}^{m} y^{(i)}\left(1-h_\theta(x^{(i)})\right) + \sum_{i=1}^{m}(1-y^{(i)})\left(-h_\theta(x)\right)x^{(i)}$$

$$= \sum_{i=1}^{m}\left(y^{(i)} - y^{(i)}h_\theta(x)^{(i)}\right) - h(x^{(i)}) + y^{(i)}h_\theta(x^{(i)})x^{(i)}$$

$$= \sum_{i=1}^{m}\left(y^{(i)} - h_\theta(x^{(i)})\right)x^{(i)}$$

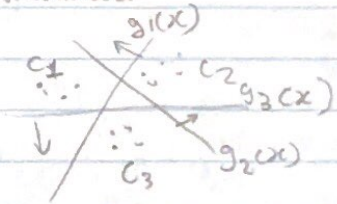
minimize -ve log likelihood:

$$\frac{d}{d\theta}(-l(\theta)) = \sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)x^{(i)}$$

$$\theta^{(i+1)} \leftarrow \theta^{(i)} - h\frac{d}{d\theta}\left(-l(\theta)^{(i)}\right)$$

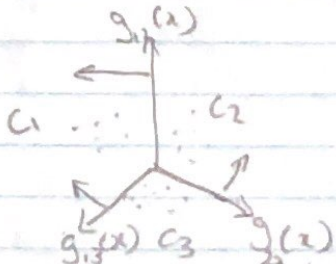
The new parameters are equal to $\theta^{(i)}$ minus the gradient of negative log likelihood.

15) One against all others :



It has k discrminant functions and the label, $\hat{y}$, is calculated by $\hat{y} = \underset{j}{\text{argmax}}\ g_j(x)$

One against each others:

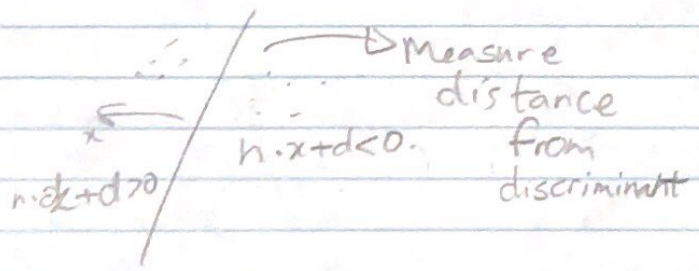


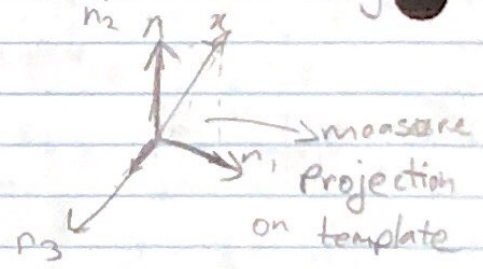
It has $\frac{k(k+1)}{2}$ discriminant functions and the label, $\hat{y}$, is calculatid by $\hat{y} = \underset{i}{\text{argmax}} \sum_{j\neq i} g_{ij}(x)$

One against each other is easier to compare but it is easier to discriminate the data is one against all others.

16) The rows of Θ^T are templates. With K rows of Θ^T, we have K templates. $\Theta^T x$ measures how well x matches each of the K template. Each class has a template and x is labelled with the template with whom it has the highest similarities. In multiple linear discriminant, Θ^T are parameters of K linear discriminant functions and each function separates one class from all others. Each class has a linear discriminant function. An input x is parsed through every function and x is labelled with the function outputting the greatest value (largest distance from discriminant).

multiple discriminant V/S template matching





17) Softmax converts the output to the probability that a certain input belongs to a certain class. The sum of all output after softmax will be equal to 1. The input will be classified with the class which has highest probability for this input.

18) $\text{softmax} = \frac{\exp(\underbrace{\theta_j^T x}_{z_j})}{\sum_{i=1}^{K} \exp(\theta_i^T x)}$

$$\frac{\partial}{\partial \theta_i} \text{softmax}(\theta_j^T x) = \text{softmax}(\theta_j^T x)(\delta_{ij} - \text{softmax}(\theta_i^T x))x$$

$$\frac{\partial}{\partial \theta_i} \log(\text{softmax}(\theta_j^T x)) = (\delta_{ij} - \text{softmax}(\theta_i^T x))x$$

19) $\ell(\theta) = -\log\left[\prod_{i=1}^{m}\prod_{j=1}^{K} p(y^{(i)}=j|x^{(i)})^{1(y^{(i)}=j)}\right)$

↓ -ve log likelihood

$$= -\sum_{i=1}^{m}\sum_{j=1}^{K} 1(y^{(i)}=j)\log p(y^{(i)}=j|x^{(i)})$$

($p(y^{(i)}=j|x^{(i)})$: $\hat{y}_j^{(i)}$)

$$= -\sum_{i=1}^{m}\sum_{j=1}^{K} 1(y^{(i)}=j)\log(h\theta_j(x^{(i)}))$$

categorical cross-entropy

20) $\frac{\partial}{\partial \theta_j}(-\ell(\theta)) = \sum_{i=1}^{m}(\underbrace{h\theta_j(x^{(i)})}_{\hat{y}_j^{(i)}} - 1(y^{(i)}=j))x^{(i)}$

$$\theta_j \leftarrow \theta_j - \eta \sum_{i=1}^{m}(h\theta_j(x^{(i)}) - 1(y^{(i)}=j))x^{(i)}$$

θ_j: new parameter; θ_j: initial parameter; η: learning rate; $\sum_{i=1}^{m}(h\theta_j(x^{(i)}) - 1(y^{(i)}=j))x^{(i)}$: gradient of -ve log likelihood

After training, we have $\theta_1, \ldots, \theta_k$

and $\hat{y} = \underset{j}{\text{argmax}}\ \theta_j^T x$.

Neural networks

1) Dimensionality increase can be useful to allows us to find some relationships that are only available in higher dimensional space. It could also allow the use of linear discriminant.

2) Dimensionality decrease is used to condense the data and extract only the useful features.

3) The hidden layer may not have the same number of inputs as the output layer.
Since $z^{(1)}$, w_0 and V_0 are initially unknown we have to guess the parameters of w_j and V_j until we get a satisfactory $\hat{z}^{(1)}$. Therefore, the same equations cannot be used for both layers

1) Loss: $E = \frac{1}{2} \sum_{t=1}^{n} (\hat{y}^{(t)} - y^{(t)})^2$

Chain rule: $\frac{\partial E}{\partial V} = \frac{\partial E}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial V}$

$= \sum_{t=1}^{n} (\hat{y}^{(t)} - y^{(t)}) z_j^{(t)}$

$\frac{\partial E}{\partial w_j} = \frac{\partial E}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_j} \frac{\partial z_j}{\partial w_j}$

$\sum_{t=1}^{n} (\hat{y}^{(t)} - y^{(t)}) V_j z_j^{(t)} (1 - z_j^{(t)}) x^{(t)}$

Let K be number of outputs.

5) Loss: $E(\{v_i\}, \{w_i\}) = \frac{1}{2} \sum_{i=1}^{m} \sum_{l=1}^{K} (\hat{y}_l^{(i)} - y_l^{(i)})^2$

Chain rule: $\frac{\partial E}{\partial v_j} = \frac{\partial E}{\partial \hat{y}_i} \frac{\partial \hat{y}_i}{\partial v_j}$

$$= \sum_{i=1}^{m} (\hat{y}_j^{(i)} - y_j^{(i)}) z^{(i)}$$

$$\frac{\partial E}{\partial w_j} = \sum_{i=1}^{K} \frac{\partial E}{\partial \hat{y}_i} \frac{\partial \hat{y}_i}{\partial z_j} \frac{\partial z_j}{\partial w_j}$$

$$\sum_{i=1}^{m} \sum_{l=1}^{K} (\hat{y}_l^{(i)} - y_l^{(i)}) V_{lj} z_j^{(i)} (1 - z_j^{(i)}) x^{(i)}$$

6) Loss: $L(\theta) = \prod_{i=1}^{m} P(y=1|x^{(i)})^{y^{(i)}} \frac{(1 - P(y=1|x^{(i)}))^{1-y^{(i)}}}{P(y=0|x^{(i)})}$

$$= -\sum_{i=1}^{m} y^{(i)} \log(\hat{y}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)})$$

$$\frac{\partial L}{\partial v} = -\frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial v} = \sum_{i=1}^{m} \left(y^{(i)} \frac{1}{\hat{y}^{(i)}} - (1 - y^{(i)}) \frac{1}{1 - \hat{y}^{(i)}} \right) \hat{y}^{(i)} (1 - \hat{y}^{(i)}) z^{(i)}$$

$$= \sum_{i=1}^{m} (\hat{y}^{(i)} - y^{(i)}) z^{(i)}$$

$$\frac{\partial L}{\partial w_j} = \sum_{i=1}^{m} (\hat{y}^{(i)} - y^{(i)}) V_j \cdot z_j^{(i)} (1 - z_j^{(i)}) x^{(i)}$$

7) Let k be number of classes:

$$\text{loss} = -\sum_{i=1}^{m}\sum_{j=1}^{k} 1(y^{(i)}=j)\log(\hat{y}_j^{(i)})$$

The gradient is the same as for regression but with different activation. (output activation is softmax for multiclass classification network

$$\frac{\partial E}{\partial v_j} = \frac{\partial E}{\partial \hat{y}_j}\cdot\frac{\partial \hat{y}_j}{\partial v_j} = \sum_{i=1}^{m}(\hat{y}_j^{(i)} - y_j^{(i)})z^{(i)}$$

$$\frac{\partial E}{\partial v_j} = \sum_{i=1}^{k}\frac{\partial E}{\partial \hat{y}_i}\frac{\partial \hat{y}_i}{\partial z_j}\frac{\partial z_j}{\partial w_j}$$

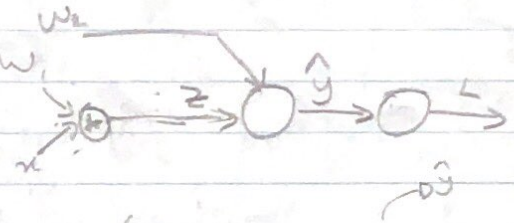
$$\sum_{i=1}^{m}\sum_{i=1}^{k}(\hat{y}_i^{(i)} - y_i^{(i)})v_{ij}\,z_j^{(i)}(1-z_j^{(i)})z^{(i)}$$

8) Since the initial output of the hidden layer is unknown as well as $\{w_j\}$ and $\{v_j\}$, the approach is to guess $\{w_j\}$ and $\{v_j\}$ to compute $\hat{z}^{(i)}$. Then modify the parameters until a satisfactory $\hat{z}^{(i)}$ is computed.

Computation graphs

1) The advantage of computational graph is that each node is simple and has a simple explicit function for derivatives. During the forward pass, the functions at all intermediate nodes are computed using the input. During the backward pass, the gradient of each node with respect to the input is calculated, starting from the end node. Each node should be able to compute the gradient to update the parameters and it should store the value computed during the forward pass to be used when calculating the gradient.

2)



$L = (\hat{y} - y)^2$

$\hat{y} = f_2(w_2, z)$

$z = f_1(w_1, x)$

i) $L = L_2(f_2(w_2, f_1(w_1, x)), y) = (f_2(w_2, f_1(w_1, x)) - y)^2$

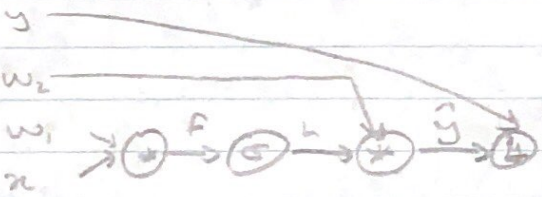
$\frac{\partial L}{\partial \hat{y}} = 2(\hat{y} - y)$ $\quad \frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_2}$ $\quad \frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z} \frac{\partial z}{\partial w_1}$

ii) $L = Cross(\overbrace{f_2(w_2, f_1(w_1, x))}^{\hat{y}}, y) = \prod_{i=1}^{n} \prod_{j=1}^{k} \left(P(y=j | x^{(i)})\right)^{y_j^{(i)}}$

$\frac{\partial L}{\partial \hat{y}} =$ $\quad \frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_2}$ $\quad \frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z} \frac{\partial z}{\partial w_1}$

3)

4)



$L = (y - \hat{y})$

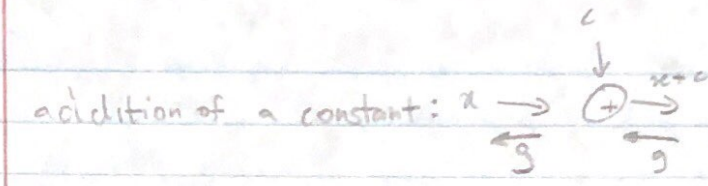
$\hat{y} = h w_2$

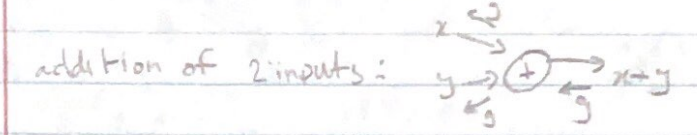
$h = \sigma(x w_1)$

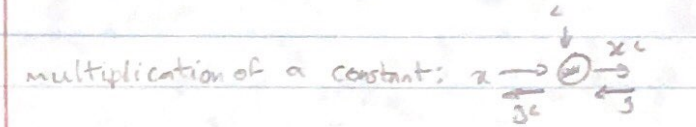
$\frac{\partial L}{\partial \hat{y}} = 2(\hat{y} - y)$ $\quad \frac{\partial \hat{y}}{\partial h} = w_2 \Rightarrow \frac{\partial L}{\partial h} = \frac{\partial L}{\partial \hat{y}} w_2$

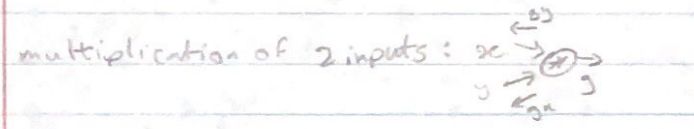
$\frac{\partial h}{\partial w_2} = h \Rightarrow \frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_2} = \frac{\partial L}{\partial \hat{y}} \cdot h$

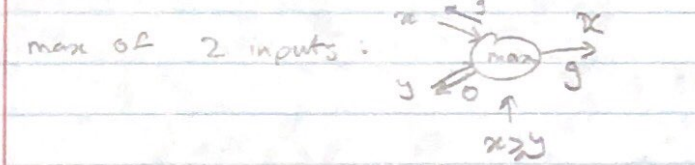
$\frac{\partial h}{\partial w_1} = h(1-h)x \Rightarrow \frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial h} \frac{\partial h}{\partial w_1} = \frac{\partial L}{\partial \hat{y}} w_2 h(1-h)x$

5) addition of a constant:  $\frac{\partial}{\partial x}(x+c) = 1$

addition of 2 inputs:  $\frac{\partial}{\partial x}(x+y) = \frac{\partial}{\partial y}(x+y) = 1$

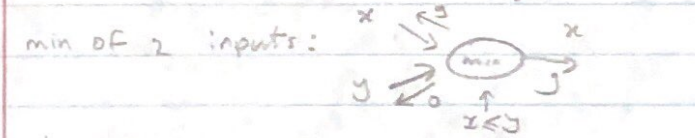
multiplication of a constant:  $\frac{\partial}{\partial x}(xc) = c$

multiplication of 2 inputs:  $\frac{\partial}{\partial x}(xy) = y \quad \frac{\partial}{\partial y}(xy) = x$

max of 2 inputs: 

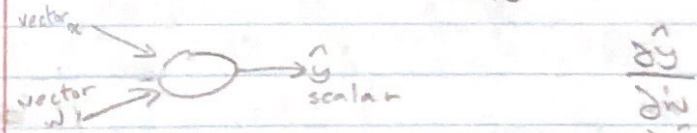
$\frac{\partial}{\partial x}(\max(x,y)) = \begin{cases} 1, & x \geq y \\ 0, & \text{o.w} \end{cases}$

$\frac{\partial}{\partial y}(\max(x,y)) = \begin{cases} 0 & x > y \\ 1 & \text{o.w} \end{cases}$

min of 2 inputs: 

$\frac{\partial}{\partial x}(\min(x,y)) = \begin{cases} 1 & x \leq y \\ 0, & \text{o.w} \end{cases}$

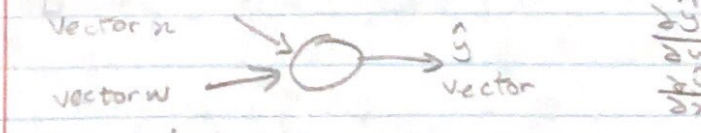
$\frac{\partial}{\partial y}(\min(x,y)) = \begin{cases} 0 & x < y \\ 1 & \text{o.w} \end{cases}$

6) 

$\frac{\partial \hat{y}}{\partial w} \rightarrow$ gradient(vector)

$\frac{\partial \hat{y}}{\partial x} \rightarrow$ gradient(vector)

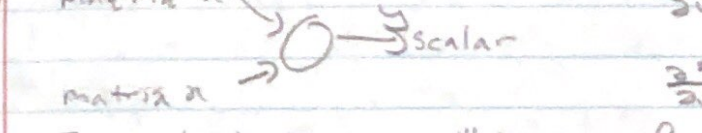
The derivative will be a vector

7) 

$\frac{\partial \hat{y}}{\partial w} \rightarrow$ Rank 2 tensor

$\frac{\partial \hat{y}}{\partial x} \rightarrow$ Rank 2 tensor

The derivative will be a Rank 2 tensor

8) 

$\frac{\partial \hat{y}}{\partial w} \rightarrow$ Rank 2 tensor

$\frac{\partial x}{\partial w} \rightarrow$ Rank 2 tensor

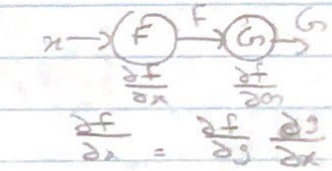
The derivative will be a Rank 2 tensor

9) $f(x) = f(x_1, \dots x_n) = \begin{bmatrix} f_1(x_1, \dots, x_n) \\ f_m(x_1, \dots, x_n) \end{bmatrix}$

$J = \nabla f(x) = \begin{bmatrix} \nabla f_1 \\ \nabla f_m \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_m}{\partial x_1} & & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$ ← Jacobian matrix

$(F \cdot g)(x) = f(g(x))$

$((F \cdot g)(x))' = f'((g)\, g'(x)$



10) $f(x,y,z) = \begin{bmatrix} 3xy \\ y - z \end{bmatrix}$ $\quad G(x,y,z) = \begin{bmatrix} x - 5y \\ xy \\ x + y \end{bmatrix}$

directly: $(F \cdot G)(x,y) = \begin{bmatrix} 3(x-5y)(xy) \\ xy - x + y \end{bmatrix} = \begin{bmatrix} 3x^2y - 15xy^2 \\ xy \quad -x+y \end{bmatrix}$

$\nabla f(x,y,z) = \begin{bmatrix} 3y & 3x & 0 \\ 0 & 1 & -1 \end{bmatrix}$ $\quad \nabla G(x,y) = \begin{bmatrix} 1 & -5 \\ y & x \\ 1 & 1 \end{bmatrix}$

$\nabla(F \cdot G)(x,y) = \begin{bmatrix} \frac{\partial}{\partial x}(3x^2y - 15xy^2) & \frac{\partial}{\partial y}(3x^2y - 15xy^2) \\ \frac{\partial}{\partial y}(xy - x + y) & \frac{\partial}{\partial y}(xy - x + y) \end{bmatrix}$

$= \begin{bmatrix} 6xy - 15y^2 & 3x^2 - 30xy \\ y - 1 & x - 1 \end{bmatrix}$ ←

using chain rule:

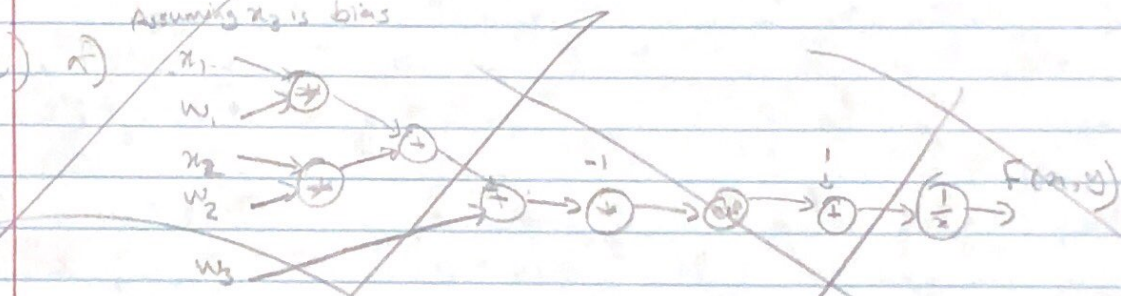
Plug $G(x,y)$ in $\nabla f(G(x,y)) = \begin{bmatrix} 3xy & 3(x-5y) & 0 \\ 0 & 1 & -1 \end{bmatrix}$

Chain rule: $\nabla f(G(x,y) \nabla G(x,y) = \begin{bmatrix} 3xy & 3(x-5y) & 0 \\ 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & -5 \\ y & x \\ 1 & 1 \end{bmatrix}$

$= \begin{bmatrix} 6xy - 15y^2 & 3x^2 - 30xy \\ y - 1 & x - 1 \end{bmatrix}$ ←

11) In the forward pass, the node compute function values and these values can be the input to another function. For example, if node a, is an input to node b, we have to ensure that node a is computed first to be able to compute node b, hence the need of ordering the nodes.
In the backward pass, the inputs become the outputs and vice versa. Instead of computing the function we compute derivatives and pass back the output to previous nodes. The nodes need to be in reverse order of the forward pass to ensure that all derivatives can be computed

12) a) Assuming x_3 is bias



b) f

Assuming $x_0 = 1$ for x_3, y_3

$z_1 = [w_0]^T [x_3] = [0.01, 0.02, 0.03] \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = 0.11$

$z_2 = [w_1]^T [x_3] = [0.03, 0.01, 0.02] \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = 0.09$

$z_3 = [w_2]^T [x_3] = [0.02, 0.03, 0.01] \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = 0.08$

$A_1 = \text{sigmoid}(z_1) = 0.5274$

$A_2 = \text{sigmoid}(z_2) = 0.5224$

$A_3 = \text{sigmoid}(z_3) = 0.5199$

$\hat{y}_3 = V^T A = [0.01, 0.02, 0.03, 0.04] \begin{bmatrix} 1 \\ A_1 \\ A_2 \\ A_3 \end{bmatrix} = 0.0557$

$L_3 = \frac{1}{2}(y_3 - \hat{y}_3)^2 = 49.432 \leftarrow$

c) $\frac{\partial L}{\partial w_1} = (\hat{y} - y) \cdot V_1 \cdot (A)(1 - A))x$

$\frac{\partial L}{\partial V_1} = (\hat{y} - y) \cdot z$

	x_1, y_1	x_2, y_2	x_3, y_3
$\frac{\partial L}{\partial w_1}$	[-0.0397, -0.0397, -0.0793]	[-0.055, -0.055, -0.164]	[-0.0496, -0.0992, -0.0992]
$\frac{\partial L}{\partial w_2}$	[-0.600, 0.600, -0.119]	[-0.082, -0.082, -0.243]	[-0.075, -0.149, -0.149]
$\frac{\partial L}{\partial w_3}$	[-0.080, .080, -0.159]	[-.109, -.164, -.326]	[-.99, -.198, -.198]
$\frac{\partial L}{\partial V}$	[-7.94, -4.09, -4.13, -4.12]	[-10.9, -5.60, -5.75, -5.70]	[-9.95, -5.25, -5.2, -5.2]

d)

13) a) $\nabla f(x,y) = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} = \begin{bmatrix} 4(2x+3y) \\ 6(2x+3y) \end{bmatrix}$

$= \begin{bmatrix} 8x+12y \\ 12x+18y \end{bmatrix}$ ←

b) $f(x,y) = \begin{bmatrix} G_1(x,y) \\ G_2(x,y) \end{bmatrix}$ $G_1(x,y) = x^2+2y$

$G_2(x,y) = 3x+4y^2$

$\Delta f(x,y) = \begin{bmatrix} 2x & 2 \\ 3 & 8y \end{bmatrix}$

$\Delta f(1,2) = \begin{bmatrix} 2 & 2 \\ 3 & 16 \end{bmatrix}$ ←

c) $f'(x,y) = \begin{bmatrix} 2x & 2 \\ 3 & 8y \end{bmatrix}$ $g'(x) = \begin{bmatrix} 1 \\ 2x \end{bmatrix} \rightarrow II$

$f'(g(x)) = \begin{bmatrix} 2x & 2 \\ 3 & 8x^2 \end{bmatrix} - I$

$II \times I = \begin{bmatrix} 2x + 4x \\ 3 + 16x^3 \end{bmatrix} = \begin{bmatrix} 6x \\ 3+16x^3 \end{bmatrix} \Rightarrow \nabla(f \circ G_1)(x)$

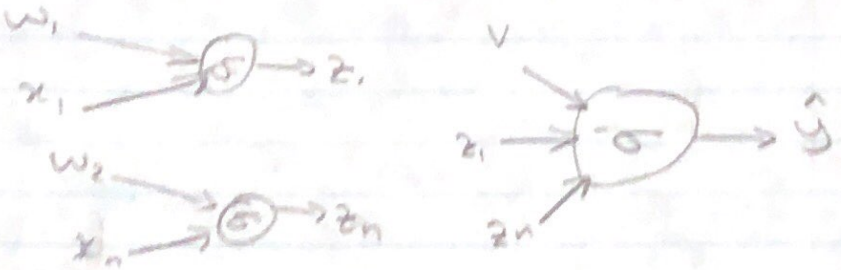
for $x=2$ $\nabla(f \circ G_1)(2) = \begin{bmatrix} 12 \\ 131 \end{bmatrix}$ ←

d) $f \cdot g = \begin{bmatrix} x^2+2x^2 \\ 3x+4(x^2)^2 \end{bmatrix} = \begin{bmatrix} 3x^2 \\ 3x+4x^4 \end{bmatrix}$

$\nabla(f \cdot g) = \begin{bmatrix} 6x \\ 3+16x^3 \end{bmatrix}$

$\nabla(f \cdot g)(2) = \begin{bmatrix} 12 \\ 3+16(8) \end{bmatrix} = \begin{bmatrix} 12 \\ 131 \end{bmatrix}$ ←

11)



$$z_i = h(w_i^T x)$$

Then $\hat{y}_i = p(y=1|x) = h(w^T \cdot z)$

$$L(\theta) = \frac{1}{2} \sum_{i=1}^{m} (y^{(i)} - \hat{y}^{(i)})^2$$

$$\frac{\partial L}{\partial \hat{y}} = \sum_{i=1}^{m} (y^{(i)} - \hat{y}) \qquad \hat{y} = h(V^T z)$$

$$\frac{\partial \hat{y}}{\partial V} = h(V^T z)(1 - h(V^T z)) \cdot z$$

$$\frac{\partial L}{\partial V} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial V} \qquad \frac{\partial L}{\partial V} = -\sum_{i=1}^{m} (y^{(i)} - \hat{y}^{(i)}) \cdot h(V^T z) \cdot (1 - h(V^T z)) \cdot z$$

$$V^+ = V - \eta \left(\frac{\partial L}{\partial V} \right)$$

$$V_i = V_i - \eta \left(-\sum_{i=1}^{m} (y^{(i)} - \hat{y}^{(i)}) \cdot h(V_i^T \cdot z^{(i)}) \cdot (1 - h(V_i^T \cdot z_i)) \right)$$

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z} \cdot \frac{\partial z}{\partial w} \qquad \frac{\partial \hat{y}}{\partial z} = \hat{y}_i (1 - \hat{y}_i) \cdot V_i$$

$$z = h(w^T \cdot x) \qquad \frac{\partial z}{\partial w} = h(w^T \cdot x)(1 - (w^T x)) x_i$$

$$\frac{\partial L}{\partial w} = \sum_{i=1}^{m} z_i \cdot (1 - z_i) x^{(i)}$$

$$\frac{\partial L}{\partial w} = \sum_{i=1}^{m} (y^{(i)} - \hat{y}^{(i)})^2 \cdot \hat{y}_i (1 - \hat{y}_i) \cdot V_i \cdot z_i (1 - z_i) x^{(i)}$$

$$w_i = w_{i-1} + \eta \left(-\sum_{i=1}^{m} (y^{(i)} - \hat{y}^{(i)})^2 \cdot \hat{y}_i (1 - \hat{y}_i) V_i \cdot z_i (1 - z_i) x^{(i)} \right)$$