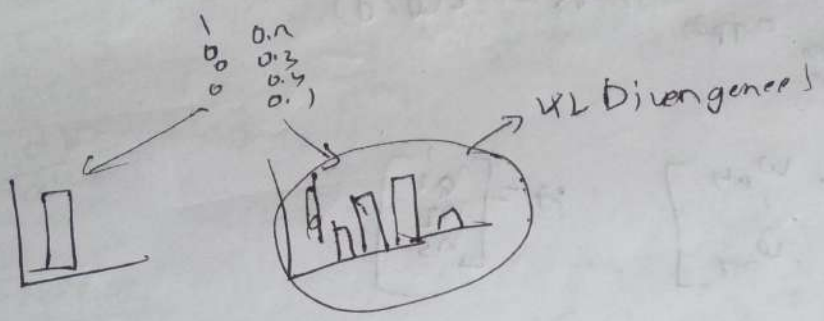
For calculating the loss function in student model

$$L = \alpha f_1(GH(t_i), \vec{P}_{student}) + (1-\alpha) f_2(\vec{P}^{student}, \vec{P}^{teacher})$$

$\alpha \rightarrow$ To assign weight

Softmatching



$m_{Teacher} \rightarrow \vec{z} \rightarrow$ Temp Softmax $\rightarrow \vec{P}$

$(x_i, \vec{P}_i^{teacher})$

$m_{student}$

Backpropagation

It is used to calculate gradients to update parameters.

$$\hat{y} = (AW\vec{x})^2$$

$\vec{x} \in \mathbb{R}^{4\times 1}$

$W \in \mathbb{R}^{3\times 4}$

$A \in \mathbb{R}^{2\times 1}$

$\longrightarrow \mathcal{L} = mse(\hat{y}, y)$

Input layer — Hidden layer — output

$$W = \begin{bmatrix} w_{1,1} & \cdots & w_{1,4} \\ \vdots & & \vdots \\ w_{3,1} & \cdots & w_{3,4} \end{bmatrix} \qquad H = \begin{bmatrix} h_1' \\ h_2 \\ h_3 \end{bmatrix}$$

$$h_i' = \sum_{j=1}^{4} w_{i,j} x_j$$

$$h_2 = \sum_{j=1}^{3} a_{1,j} h_j$$

$$\hat{y} = (h_2)^2$$

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

$$\frac{d\mathcal{L}}{dw_{i,j}} \qquad \hat{x} = (AW\vec{x})^2$$

$$H = [a_{11}, a_{12}, a_{13}]$$

We want partial derivatives

$$\frac{d\mathcal{L}}{da_{i,j}} \quad \frac{d\mathcal{L}}{d\hat{y}} = -2(y - \hat{y})$$

$$\frac{d\mathcal{L}}{dw_{1,1}}$$

$$\hat{y} = (h_2)^2$$

$$\frac{d\hat{y}}{dh_2} = 2h_2$$

$$\frac{dh_2}{da_{1,b}} = h_j'$$

$$\frac{dh}{da_{i,j}} = \frac{dh}{d\hat{y}} \cdot \frac{d\hat{y}}{dh_i} \cdot \frac{dh_i}{da_{i,j}}$$

Same $\frac{dh}{da_{2,j}} / \frac{dh}{a_{3,j}}$

Further Dynamic Programming

$$\frac{dh_i'}{dw_{i,j}} \quad \bar{a} = x_j$$

$$\frac{dL}{dw_i} = \frac{dh}{d\hat{y}} \cdot \frac{d\hat{y}}{dh_2} \cdot \frac{dh_2}{dh_1} \cdot \frac{dh_i}{dw_{2,i}}$$

Exercise

$$\vec{x} = \begin{bmatrix} 1 \\ 0.5 \\ -3 \end{bmatrix} \quad W = \begin{bmatrix} 1 & -1 & 2 & 4 \\ 4 & 3 & 0 & 1 \\ 0.5 & 1 & 2 & 0 \end{bmatrix}$$

$$A = \begin{bmatrix} 0.3 & 0.3 & 0.6 \end{bmatrix}$$

[Forward pass to Backpropagate]

calculate $\hat{y}$

Given y → calculate d

calculate $\frac{dh}{dw_{i,y}} / \frac{dh}{da_{i,y}}$

[Non linearity Needs to be added
Activation function]

$\vec{y} = AWX$

2×3 $\quad 3 \times 100$ $\quad 100 \times 1$ $\quad\quad \vec{y} = AWX$

A $\quad$ W $\quad$ X

2×100 $\quad 100 \times 1$ $\quad\quad \vec{y} = TX$

T

[we can use T as all trainable parameters Also for Non linear Transformation]

Activation ~~fn~~ function (To transform Non-Linear)

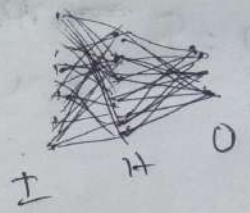
⇒ tanh, sigmoid, Relu, Leaky-Relu, Softmax $\hat{y} = \ldots NL_2(w_2 NL_1(w_1 \vec{x}))$

[It is Non-linear transformation]

$\hat{y} = T\vec{x}$ when,

$T = (NL_2(w_2 NL_2 w_2 \ldots NL_i w_i)$

Activation function

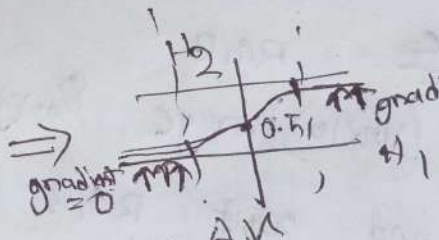


$$NL\left(AL\left(NL\left(A\vec{x}\right)\right)\right)$$

Sigmoid

$$\text{Sgmoid}(q) = \frac{1}{1+e^{q}}$$

$$\sigma(u) = \frac{1}{1+e^{u}}$$



$H_2 = \sigma(H_1)$

So enclosed must be used

$H_a = A_1 x$

$H_2 = \sigma(H_1)$

$y = A_2 H_2$

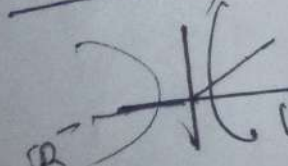
$L = f(y, \hat{y})$

$$\frac{dL}{dA_1} = \frac{dL}{dy} \times \frac{dy}{dH_2} \times \frac{dH_2}{dH_1} \times \frac{dH_1}{dA_1}$$

Vanishing gradient will ~~not cause~~ update parameters.

Exploding gradient descent will not enable to reach ~~optimal~~ optimal parameters.

ReLU



$$\text{ReLU}(u) = \max(x, 0)$$

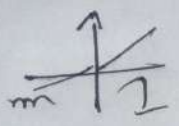
$$R(u) = \begin{cases} 0, & u < 0 \\ u, & u \geq 0 \end{cases}$$

$$\frac{dR(u)}{du} = \begin{cases} 0 & u \leq 0 \\ 1 & u > 0 \end{cases}$$

ReLU → for non-negative

Leaky ReLU

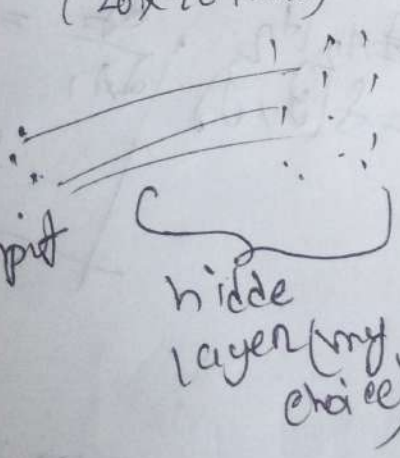
$$\text{Leaky ReLU}(x) = \begin{cases} x(0), & x \leq 0 \\ x, & x > 0 \end{cases}$$



You can use Batch Normalization to ignore Vanishing gradient system

Exercise

Input a (10x10) RGB image fingerprint to non the image and ouput RGB image to the face of the person (20x20 image)



20X20X3 => 1200 (Total number)

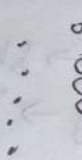
(0-255) number

Transformers

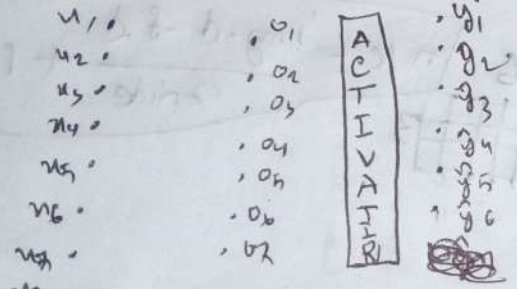
$$V_i = \frac{e^{(x_i/T)}}{\sum_{j=1}^{k} e^{(x_j/T)}}$$

$$\sum_{i=k}^{k} V_i \vec{R}_i = ?$$

Drop out (Used to ON/OFF Neuron, prevent overfitting)



$\hat{y}_i == 0.5$ if $x_i == x_{i+1}$
$\hat{y} > 0.5$ if $x_i < x_{i+1}$
$\hat{y} < 0.5$ if $x_i > x_{i+1}$
} We should use sigmoid



[Background for CNN]

$\hat{y} = \text{activation}(W\vec{x})$

$\xrightarrow{\quad x \quad}$

$$\begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix} \Longrightarrow \begin{bmatrix} 1 \\ -1 \end{bmatrix} \begin{matrix} f_1 \\ f_u \end{matrix}$$

$O_i = f_1 x_i + f_2 x_{i+1}$

16

$\Rightarrow$ Feature extraction