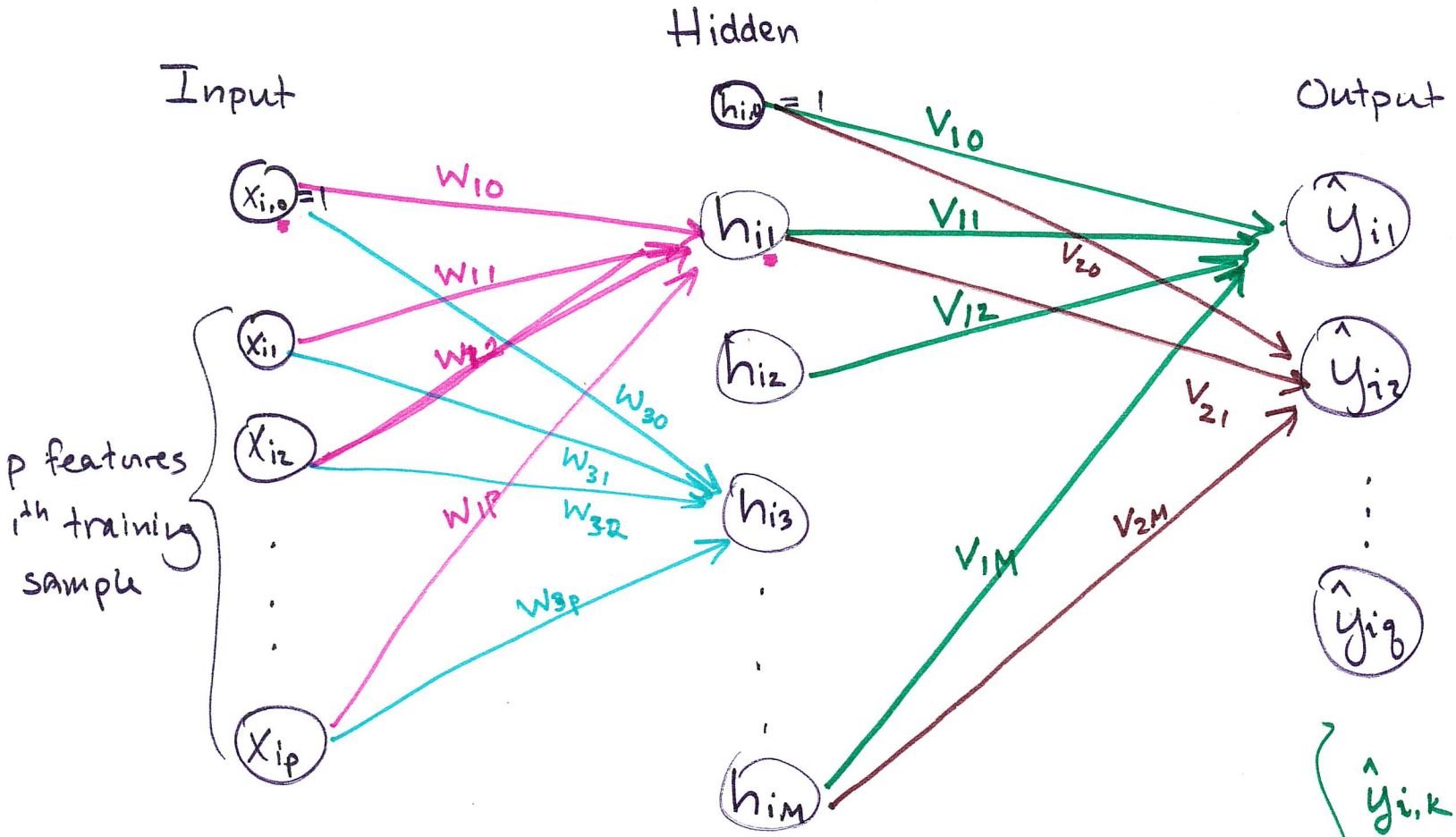
Lecture 23 - Backpropagation in Neural Networks



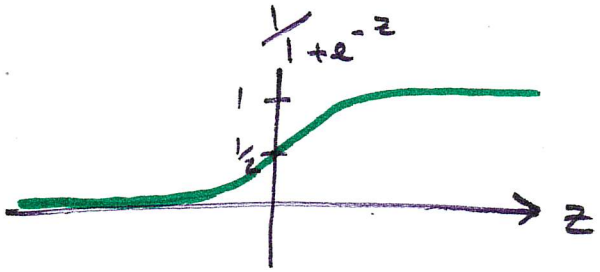
h_{im} = output of m^{th} hidden node if x_i is input

$$= \sigma\left(\sum_{j=0}^{p} w_{m,j}\, x_{ij}\right) \qquad \text{for } m = 1, 2, \ldots, M$$

$$\hat{y}_{i,k} = \sigma\left(\sum_{m=0}^{M} v_{km}\, h_{im}\right)$$

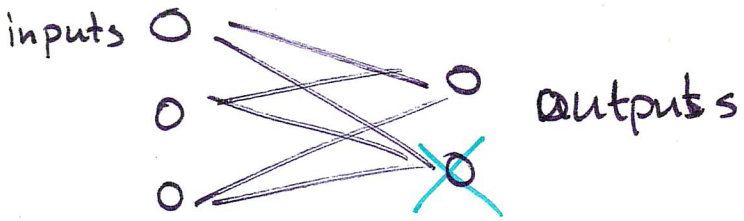
Common activation functions:

- $\sigma(z) = \max(z, 0)$ (ReLU = rectified Linear unit)
- $\sigma(z) = \text{sign}(z)$
- $\sigma(z) = \frac{1}{1+e^{-z}}$ = logistic function



Learning weights:

1st pass: 2-layer network:



$$\hat{y}_{iK} = \sigma\left(\sum_{j=0}^{p} w_{Kj}\, x_{ij}\right)$$

We can learn weights w/ Stochastic Gradient Descent (SGD)

loss function $f(\underline{w}) = \sum_{i=1}^{n} \frac{1}{2}(y_i - \hat{y}_i)^2$ ← want to minimize

$(a-b)^2 = (b-a)^2$

$$= \sum_{i=1}^{n} \underbrace{\frac{1}{2}\left[\sigma\left(\sum_{j=0}^{p} w_j x_{ij}\right) - y_i\right]^2}_{f_i(\underline{w})}$$

$$f(w) = \sum_{i=1}^{n} f_i(w)$$

Today: $\sigma(z) = \frac{1}{1+e^{-z}}$.

$$\sigma'(z) = \frac{d\sigma}{dz} = \sigma(z)(1-\sigma(z))$$

recall: $\hat{y}_i = \sigma\left(\sum_{j=0}^{p} w_j x_{ij}\right)$

algorithm

@ iteration t:

- choose i_t uniformly at random.
- set $w^{(t+1)} = w^{(t)} - \alpha_t \nabla f_{i_t}(w^{(t)})$, α_t = step size (e.g. $\alpha_t = 1/\sqrt{t}$)

$\nabla f_i(w^{(t)})$ (drop i_t use i)

$$\left.\frac{df_i}{dw_j}\right|_{\underline{w}^{(t)}} = \left.\frac{df_i}{d\hat{y}_i}\cdot\frac{d\hat{y}_i}{dw_j}\right|_{\underline{w}^{(t)}}$$

j^{th} weight

$$= (\hat{y}_i - y_i)\,\sigma'\left(\sum_{j'=0}^{p} w_{j'} x_{ij'}\right) x_{ij} \Bigg|_{\underline{w}^{(t)}}$$

$$= (\hat{y}_i - y_i)\underbrace{\sigma\left(\sum_{j'} w_{j'} x_{ij'}\right)}_{\hat{y}_i}\underbrace{\left[1-\sigma\left(\sum_{j'} w_{j'} x_{ij'}\right)\right]}_{1-\hat{y}_i} x_{ij} \Bigg|_{\underline{w}^{(t)}}$$

$$= \underbrace{(\hat{y}_i - y_i)\,\hat{y}_i(1-\hat{y}_i)}_{\text{scalar, independent of } j = S_i}\, x_{ij}$$

$$\Rightarrow \quad \nabla f_{i_t}(\underline{w}^{(t)}) = \delta_{i_t} \underline{x}_{i_t}$$

$$\nabla f_{i_t} = \begin{bmatrix} df_{i_t}/dw_0 \\ df_{i_t}/dw_1 \\ \vdots \\ df_{i_t}/dw_p \end{bmatrix} = \begin{bmatrix} \delta_{i_t} x_{i_t 0} \\ \delta_{i_t} x_{i_t 1} \\ \vdots \\ \delta_{i_t} x_{i_t p} \end{bmatrix} = \delta_{i_t} \underline{x}_{i_t}$$

$\Rightarrow$ SGD:

$$w^{(t+1)} = w^{(t)} - \alpha_t \delta_{i_t} x_{i_t}$$

3-layer network (1 hidden layer)

$$\hat{y}_{i,k} = \sigma\left(\sum_{m=0}^{M} v_{k,m} h_{i,m}\right) = \sigma\left(\sum_{m=0}^{M} v_{k,m} \cdot \sigma\left[\sum_{j=0}^{p} w_{m,j} x_{ij}\right]\right)$$

Again, write i instead of i_t

Backpropagation Algorithm

- choose initial weights $\{V_{k,m}^{(1)}\}_{k,m}$ and $\{W_{m,j}^{(1)}\}_{m,j}$
- for $t = 1, 2, 3, \ldots$
  - choose i_t
  - calculate $\hat{y}_{i_t,k}$ and $h_{i_t,m}$ using $\{V_{k,m}^{(t)}\}_{k,m}$, $\{W_{m,j}^{(t)}\}_{m,j}$
  - update weights for each layer, starting w/ deepest layer (closest to output) and working back to shallowest layer

to update $v_{k,m}$'s :

$$\frac{df_i}{dv_{km}} = \frac{df_i}{d\hat{y}_i} \cdot \frac{d\hat{y}_i}{dv_{k,m}}$$

$$= (\hat{y}_{ik} - y_{ik})\, \sigma'\left(\sum_{m'=0}^{M} v_{k,m'}\, h_{i,m'}\right) h_{i,m}$$

$$= \underbrace{(\hat{y}_{ik} - y_{ik})\, \hat{y}_{ik}\, (1 - \hat{y}_{ik})}_{\delta_{i,k}}\, h_{i,m}$$

$$= \delta_{ik} \cdot h_{i,m}$$

$$\Rightarrow v_{k,m}^{(t+1)} = v_{k,m}^{(t)} - \alpha_t\, \delta_{i_t k}\, h_{i_t m} \quad \text{for } k=1,\dots,q,\ m=0,\dots,M$$

$$\Rightarrow \underline{v}_k^{(t+1)} = \underline{v}_k^{(t)} - \alpha_t\, \delta_{i_t,k}\, \underline{h}_{i_t} \quad \text{for } k=1,\dots,q$$

To update $w_{m,j}$'s :

$$\frac{df_i}{dw_{m,j}} = \sum_{k=1}^{q} \frac{df_i}{d\hat{y}_{ik}} \cdot \frac{d\hat{y}_{ik}}{dh_{i,m}} \cdot \frac{dh_{i,m}}{dw_{m,j}}$$

$$= \sum_{k=1}^{q} \underbrace{(\hat{y}_{ik} - y_{ik})\, \sigma'\left(\sum_{m'=0}^{M} v_{k,m'} h_{i,m'}\right) v_{k,m}}_{\delta_{ik}} \sigma'\left(\sum_{j'=0}^{p} w_{m,j'} x_{ij'}\right) x_{ij}$$

$$= \sum_{k=1}^{q} \underbrace{(\hat{y}_{ik} - y_{ik})\, \hat{y}_{ik}(1-\hat{y}_{ik})\, v_{k,m}}_{\delta_{ik}} h_{i,m}(1-h_{i,m})\, x_{ij}$$

$$= \underbrace{\sum_{k=1}^{q} \delta_{ik} v_{k,m} h_{i,m}(1-h_{im})}_{\text{scalar, indep. of } j,\ \gamma_{i,m}} x_{ij} = \gamma_{m,i}\, x_{i,j}$$

SGD update:

$$w_{m,j}^{(t+1)} = w_{m,j}^{(t)} - \alpha_t \gamma_{i_t,m} x_{i_t,j} \quad \text{for } j=0,\dots,p \quad m=0,\dots,M$$

$$\underline{w}_m^{(t+1)} = w_m^{(t)} - \alpha_t \gamma_{i_t,m} \underline{x}_{i_t} \quad \text{for } m=0,\dots,M$$