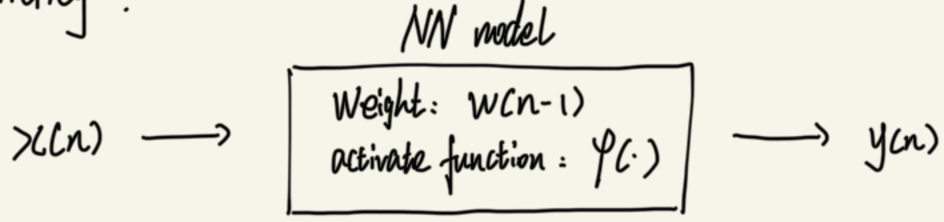
陈羿乔 YiqiaoChen 2024/10/13

① What is Learning ?



Neural Network accepts input and perform mapping through Weights to generate output.

When the model perform poorly . we adjust the weight

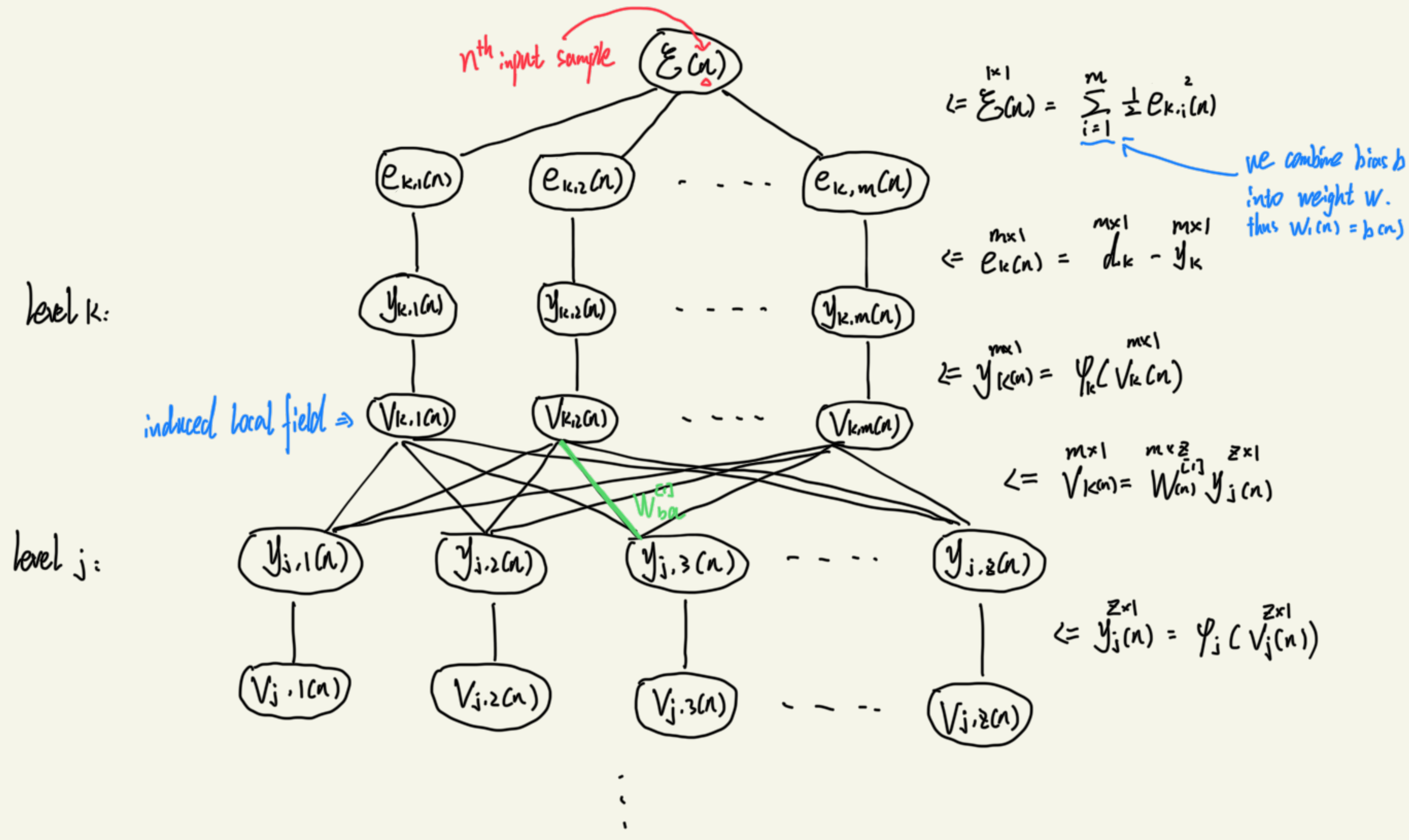
Large $\mathcal{E}(n) = \sum_j \mathcal{E}_j(n)$

gradient descent

$$W(n+1) := W(n) + \Delta W$$
$$= W(n) - \eta \frac{\partial \mathcal{E}(n)}{\partial W}$$

② How we calculate $\frac{\partial \mathcal{E}(n)}{\partial w(n)}$?

measure how changes in weights $w(n)$ affect the training error $\mathcal{E}(n)$



⋮

I. for level "j-to-k" : $\frac{\partial \mathcal{E}(n)}{\partial W_{ba}^{[j]}(n)}$, i.e. $y_a(n) \xrightarrow{W_{ba}^{[j]}(n)} y_b(n)$ ($y_a(n) \in y_j(n)$, $y_b(n) \in y_k(n)$)

Apply chain rule, we have:

$$\frac{\partial \mathcal{E}(n)}{\partial W_{ba}^{[j]}(n)} = \overset{m\times1}{\frac{\partial \mathcal{E}(n)}{\partial e_k(n)}} \cdot \overset{m\times m}{\frac{\partial e_k(n)}{\partial y_k(n)}} \cdot \overset{m\times m}{\frac{\partial y_k(n)}{\partial v_k(n)}} \cdot \overset{m\times1}{\frac{\partial V_k(n)}{\partial W_{ba}^{[j]}(n)}}$$

i. $\frac{\partial \mathcal{E}(n)}{\partial e_k(n)} = \frac{\partial \frac{1}{2} e_k(n)^T e_k(n)}{\partial e_k(n)} = e_k(n)$

ii. $\frac{\partial e_k(n)}{\partial y_k(n)} = \begin{bmatrix} \frac{\partial e_{k,1}(n)}{\partial y_{k,1}(n)} & \frac{\partial e_{k,1}(n)}{\partial y_{k,2}(n)} & \cdots & \frac{\partial e_{k,1}(n)}{\partial y_{k,m}} \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & \cdots & 0 \end{bmatrix} = -I_{m\times m}$

$$\frac{\partial e_k(n)}{\partial y_k(n)} = \begin{bmatrix} \vdots & & & \\ \frac{\partial e_{k,2}(n)}{\partial y_{k,1}(n)} & \frac{\partial e_{k,2}(n)}{\partial y_{k,2}(n)} & \cdots & \frac{\partial e_{k,2}(n)}{\partial y_{k,m}(n)} \\ & & \vdots & \\ \frac{\partial e_{k,m}(n)}{\partial y_{k,1}(n)} & \frac{\partial e_{k,m}(n)}{\partial y_{k,2}(n)} & \cdots & \frac{\partial e_{k,m}(n)}{\partial y_{k,m}(n)} \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 & \cdots & 0 \\ \vdots & & -1 & & \vdots \\ \vdots & & & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & -1 \end{bmatrix}$$

Jacobin Matrix

iii. $$\frac{\partial y_k(n)}{\partial v_k(n)} = J = \begin{bmatrix} \varphi_k'(v_{k,1}(n)) & 0 & \cdots & 0 \\ 0 & \varphi_k'(v_{k,2}(n)) & \cdots & 0 \\ & \vdots & & \vdots \\ 0 & 0 & \cdots & \varphi_k'(v_{k,m}(n)) \end{bmatrix}_{m\times m}$$

iv. know that $V_k(n) = W^{[i]}(n)\, y_k(n)$

i.e. $$\begin{bmatrix} V_{k,1}(n) \\ V_{k,2}(n) \\ \vdots \\ V_{k,m}(n) \end{bmatrix}_{m\times 1} = \begin{bmatrix} \boxed{W_1^{[i]}(n)} \\ \boxed{W_2^{[i]}(n)} \\ \vdots \\ \boxed{W_m^{[i]}(n)} \end{bmatrix}_{m\times m} \begin{bmatrix} y_{j,1}(n) \\ y_{j,2}(n) \\ \vdots \\ y_{j,z}(n) \end{bmatrix} = \begin{bmatrix} [W_1^{[i]}(n)]^T y_j(n) \\ \vdots \\ W_{b,1}^{[i]}(n)\, y_{j,1}(n) + \cdots + W_{b,a}^{[i]}(n)\, y_{j,a}(n) + \cdots + W_{b,m}^{[i]}(n)\, y_{j,m}(n) \\ \vdots \\ [W_m^{[i]}(n)]^T y_j(n) \end{bmatrix}$$

we have $$\frac{\partial V_k(n)}{\partial W_{ba}^{[i]}(n)} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ y_{j,a}(n) \\ \vdots \\ 0 \end{bmatrix}$$

In summary.

$$\frac{\partial \mathcal{E}(n)}{\partial W_{ba}^{[i]}(n)} = \left[-e_{k,1}(n)\varphi_k'(V_{k,1}(n)),\ -e_{k,2}(n)\varphi_k'(V_{k,2}(n)) \cdots, -e_{k,m}(n)\varphi_k'(V_{k,m}(n))\right] \begin{bmatrix} 0 \\ 0 \\ \vdots \\ y_{j,a}(n) \\ \vdots \\ 0 \end{bmatrix} \leftarrow \text{index} = b$$

$$= \boxed{-e_{k,b}(n)\,\varphi_k'(V_{k,b}(n))}\, y_{j,a}(n)$$

define " $-\frac{\partial \mathcal{E}(n)}{\partial V_{k,b}(n)}$ " the local gradient for neuron b. Denote as $\delta_{k,b}(n)$

for change on $W_{ba}^{[i]}(n)$, we have: $\Delta W_{ba}^{[i]}(n) = -\eta \frac{\partial \mathcal{E}(n)}{\partial W_{ba}^{[i]}(n)}$

$$= \eta\, \delta_{k,b}(n)\, y_a(n)$$

II. Local gradient for $y_a(n)$

$$\delta_a(n) = \frac{\partial \mathcal{E}(n)}{\partial V_{j,a}(n)} = \boxed{\frac{\partial \mathcal{E}(n)}{\partial e_k(n)} \cdot \frac{\partial e_k(n)}{\partial y_k(n)} \cdot \frac{\partial y_k(n)}{\partial V_k(n)}} \cdot \boxed{\frac{\partial V_k(n)}{\partial y_j(n)}} \cdot \boxed{\frac{\partial y_j(n)}{\partial V_{j,a}(n)}}$$

$$= \left[+e_{k,1}(n)\varphi_k'(V_{k,1}(n)),\ +e_{k,2}(n)\varphi_k'(V_{k,2}(n)) \cdots, +e_{k,m}(n)\varphi_k'(V_{k,m}(n))\right] \begin{bmatrix} W_{11}^{[i]}(n) & W_{12}^{[i]}(n) & \cdots & W_{1z}^{[i]}(n) \\ W_{21}^{[i]}(n) & W_{22}^{[i]}(n) & \cdots & W_{2z}^{[i]}(n) \\ & \vdots & & \\ W_{m1}^{[i]}(n) & W_{m2}^{[i]}(n) & \cdots & W_{mz}^{[i]}(n) \end{bmatrix}_{m\times z} \begin{bmatrix} 0 \\ 0 \\ \vdots \\ \boxed{\varphi_j'(V_{j,a}(n))} \\ \vdots \\ 0 \end{bmatrix}$$

index a

$$= \sum_{i=1} \underbrace{e_{k,i}(n)\varphi'_k(v_{k,i}(n))}_{\text{local gradient}} W_{i,a}(n) \cdot \varphi'_j(v_{j,a}(n))$$

$$= \left[\sum_{i=1}^{m} \delta_i(n) W_{i,a}^{(\cdot)}(n) \right] \cdot \varphi'_j(v_{j,a}(n))$$

③ Conclusion

$$\text{neuron } (i) \xrightarrow{w_{ji}(n)} (j) \text{ neuron}$$

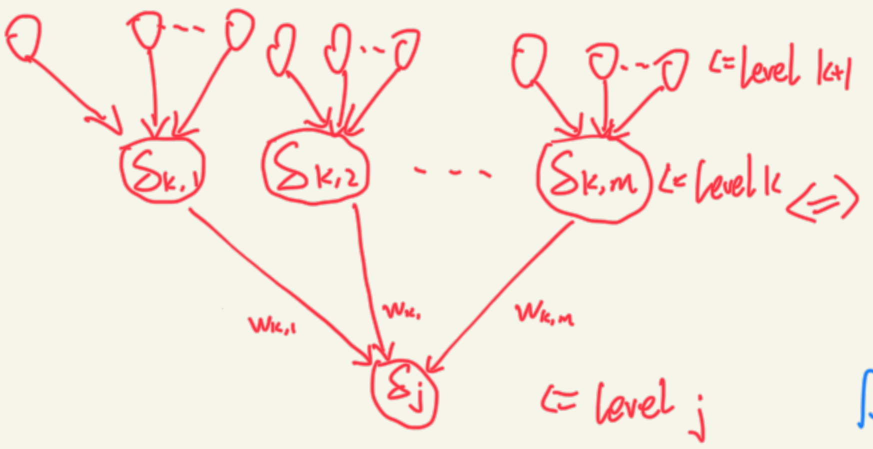
$$\Delta w_{ji}(n) = \boxed{\eta} \cdot \boxed{\delta_j(n)} \cdot \boxed{y_i(n)}$$

2. for output neuron:

$$\delta_j(n) = e_j(n) \cdot \varphi'_j(v_j(n))$$

II. for hidden neuron:

$$\delta_j(n) = \underbrace{\left[\sum_k \delta_k(n) w_{kj}(n) \right]}_{\text{all local gradient of the previous layer}} \cdot \varphi'_j(v_j(n))$$



if k is still not the output neuron

$$\Leftrightarrow \quad \delta_j(n) = \sum_k \Big[\sum_{l(k+1)} \delta_{l(k+1)}(n)\, w_{l(k+1),k}(n) \Big] \cdot \varphi'_j(v_j(n)) \quad \Leftrightarrow$$

↑ high-level layer for k

⇒ Compute the local gradient of output nerons, the the local gradients of hidden neurons layer-by layer.