Optimisation Methods

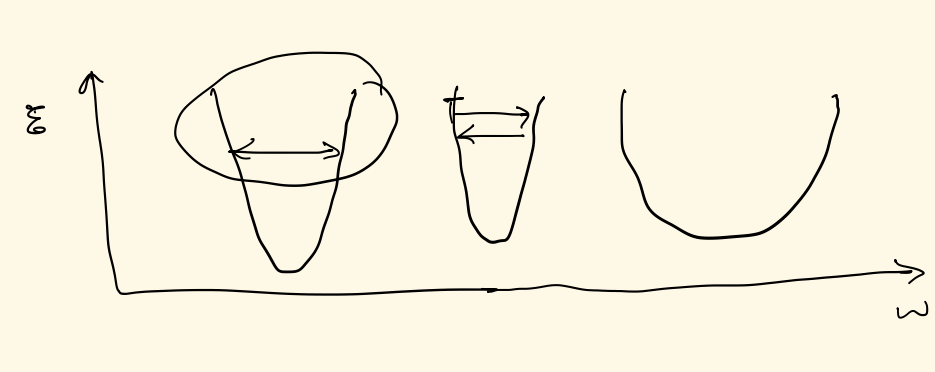
Wednesday, 7 February 2024 06:45

1. Faster Convergence Methods
2. Regularization Methods

Delta Rule : $\Delta w(m) = -\eta \frac{\partial \mathcal{E}}{\partial w}\Big|_{w=w(m)}$

$$w(m+1) = w(m) - \eta \frac{\partial \mathcal{E}}{\partial w}\Big|_{w=w(m)}$$

$$w(m+1) = w(m) - \eta\, g_w(m)$$



Generalized Delta Rule Method:

Momentum term

Momentum factor : α

$$w(m+1) = w(m) - \eta\, g_w(m) + \alpha\, \Delta w(m-1)$$

$\Delta w(m) = w(m) - w(m-1)$

$\Delta w(1) = -\eta g_w(0)$

$\Delta w(2) = -\eta g_w(1) - \alpha \cdot \eta g_w(0)$

$w(0)$

$w(1) = w(0) - \eta g_w(0)$

$w(2) = w(1) - \eta g_w(1) + \alpha(-\eta g_w(0))$

$w(3) = w(2) - \eta g_w(2) - \eta\alpha g_w(1) - \eta\alpha^2 g_w(0)$

$$w(m+1) = w(m) - \eta g_w(m) - \eta\alpha \cdot g_w(m-1) - \eta\alpha^2 g_w(m-2) - \dots - \eta\alpha^m g_w(0)$$

Low curvature region of error surface:

$$w(m+1) = w(m) - \eta\big[g_w(m) + \alpha g_w(m-1) + \alpha^2 g_w(m-2) + \dots + \underbrace{\alpha^m g_w(0)}\big]$$

Exponentially weighted gradients

g_w is approximately the same.

$$w(m+1) \simeq w(m) - \eta g_w\big[\underbrace{1 + \alpha + \alpha^2 + \dots + \alpha^m}_{\frac{1}{1-\alpha}}\big]$$

$0.0 < \alpha < 1.0$

$\alpha \simeq 0.9$

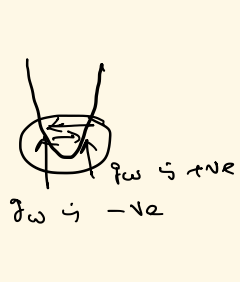
$$w(m+1) = w(m) - \boxed{\frac{\eta}{1-\alpha}}\, g_w$$

Effective Learning Rate $\hat{\eta} = \frac{\eta}{1-\alpha}$

High curvature region:

$$w(m+1) = w(m) - \eta g_w(m) + \alpha\,\Delta w(m-1)$$

$$= w(m) \underbrace{- \eta \underbrace{g_w(m)}_{-ve} - \eta\alpha \cdot g_w \underbrace{(m-1)}_{+ve}}$$



Oscillations are reduced

Learning rate scheduling :

$$w(m+1) = w(m) - \eta(m) \cdot g_w(m)$$

Linear scheduling :

$\eta(m) = \left(1 - \frac{m}{M}\right)\eta(0)$

M: No. of update steps

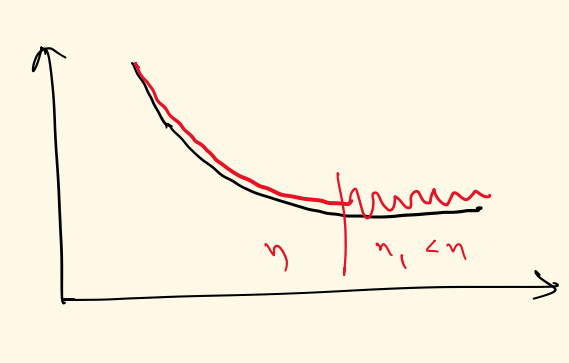
$\eta(1) = \left(1 - \frac{1}{M}\right)\eta(0)$

$\eta(2) = \left(1 - \frac{2}{M}\right)\eta(0)$

$\vdots$

$\eta(m-1) = \frac{1}{m}\eta(0)$

$\eta(m) = \eta(m-1)$ for $m \geq M$



Parameter-Specific Adaptation of Learning Rate:

Adaptive Gradient (AdaGrad) method:

$$\Delta w(m) = -\frac{\eta}{\epsilon + \sqrt{r_w(m)}}\, g_w(m)$$

$$r_w(m) = g_w^2(0) + g_w^2(1) + \dots + g_w^2(m-1)$$

$$\hat{\eta}_w = \frac{\eta}{\epsilon + \sqrt{r_w(m)}}$$

ϵ is a small constant $\epsilon \simeq 10^{-6}$

Root Mean Square Propagation (RMS Prop)

$$r_w(m) = \beta \cdot r_w(m-1) + (1-\beta)\, g_w^2(m)$$

$$w(m+1) = w(m) - \frac{\eta}{\epsilon + \sqrt{r_w(m)}}\, g_w(m)$$

$r_w(0) = g_w^2(0)$

$r_w(1) = \beta \cdot r_w(0) + (1-\beta)\, g_w^2(1)$

$r_w(2) = \beta \cdot \underline{\underline{r_w(1)}} + (1-\beta)\, g_w^2(2)$

$= \beta^2 r_w(0) + \beta(1-\beta)\, g_w^2(1) + (1-\beta)\, g_w^2(2)$

$r_w(3) = \beta^3 r_w(0) + \beta^2(1-\beta)\, g_w^2(1) + \beta(1-\beta)\, g_w^2(2) + (1-\beta)\, g_w^2(3)$

$\vdots$

$r_w(m) = \beta^m r_w(0) + \beta^{m-1}(1-\beta)\, g_w^2(1) + \dots + (1-\beta)\, g_w^2(m)$

Exponentially weighted g_w^2

- Exponentially weighted moving average of g_w^2

$\beta \simeq 0.9$

Adaptive Moments (AdaM)

Combination of RMS Prop and momentum

$q_w(m) = \beta_1\, q_w(m-1) + (1-\beta_1)\, g_w(m)$

$r_w(m) = \beta_2\, r_w(m-1) + (1-\beta_2)\, g_w^2(m)$

g_w : First order moment of gradient

g_w^2 : Second order moment of gradient.

$E[g_w] \simeq \hat{q}_w(m) = \frac{q_w(m)}{1 - \beta_1^m}$

$E[g_w^2] \simeq \hat{r}_w(m) = \frac{r_w(m)}{1 - \beta_2^m}$

$$w(m+1) = w(m) - \eta\, \frac{\hat{q}_w(m)}{\epsilon + \sqrt{\hat{r}_w(m)}}$$

$E[q_w] = (1 - \beta_1^m)\, E[g_w]$

$E[r_w] = (1 - \beta_2^m)\, E[g_w^2]$

$\beta_1 = 0.9, \quad \beta_2 = 0.99$

First order methods

Second order methods

Taylor series

$f(x) \qquad f(x + \Delta x)$

$$f(a + \Delta x) = f(a) + f'(x)\big|_{x=a} \cdot \Delta x + \frac{f''(x)}{2!}\Big|_{x=a} (\Delta x)^2 + \dots + \frac{f^k(x)}{k!}\Big|_{x=a} (\Delta x)^k$$

Second order approximation

$$f(a + \Delta x) \approx f(a) + f'(x)\big|_{x=a} \Delta x + \frac{f''(x)}{2!}\Big|_{x=a} (\Delta x)^2$$

$$\mathcal{E}(w + \Delta w) = \mathcal{E}(w) + \underset{g}{g_w} \cdot \Delta w + \frac{1}{2}\frac{\partial^2 \mathcal{E}}{\partial w^2} (\Delta w)^2$$

$$\mathcal{E}(\bar{\theta} + \Delta\bar{\theta}) = \mathcal{E}(\bar{\theta}) + \left(\frac{d\mathcal{E}}{d\bar{\theta}}\right)^t (\Delta\bar{\theta}) + \frac{1}{2} \cdot (\Delta\bar{\theta})^t \underbrace{\frac{d^2\mathcal{E}}{d\bar{\theta}^2}} (\Delta\bar{\theta})$$

$\bar{\theta} = [w_1\ w_2\ w_3]^t$

$$\frac{d\mathcal{E}}{d\bar{\theta}} = \begin{bmatrix} \frac{\partial \mathcal{E}}{\partial w_1} \\ \frac{\partial \mathcal{E}}{\partial w_2} \\ \frac{\partial \mathcal{E}}{\partial w_3} \end{bmatrix}$$

Hessian matrix $\frac{d^2\mathcal{E}}{d\bar{\theta}^2} = H_{W\times W}$ $\qquad$ W : No. of parameters

$H = [h_{ij}]$

$h_{ij} = \frac{\partial^2 \mathcal{E}}{\partial w_i \partial w_j}$

$$\mathcal{E}(\bar{\theta} + \Delta\bar{\theta}) = \mathcal{E}(\bar{\theta}) + \bar{g}^t \Delta\bar{\theta} + \frac{1}{2}\Delta\bar{\theta}^t H \Delta\bar{\theta}$$

Find $\Delta\bar{\theta}$ such that $(\mathcal{E}(\bar{\theta}) - \mathcal{E}(\bar{\theta} + \Delta\bar{\theta}))$ is maximum

$$\frac{d}{d\Delta\bar{\theta}}\left(\mathcal{E}(\bar{\theta}) - \mathcal{E}(\bar{\theta} + \Delta\bar{\theta})\right) = \bar{0}$$

$$\frac{d}{d\Delta\bar{\theta}}\left(-\bar{g}^t \Delta\bar{\theta} - \frac{1}{2}\Delta\bar{\theta}^t H \Delta\bar{\theta}\right) = \bar{0}$$

$$-\bar{g} - H\Delta\bar{\theta} = \bar{0}$$

$$\Delta\bar{\theta} = -H^{-1}\bar{g}$$

Newton's Steepest Descent Method

Levenberg-Marquardt (LM) method:

Outer product approximation of H

Batch mode $\quad H \approx \frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K} \bar{z}_{nk}\, \bar{z}_{nk}^t$

$\bar{z}_{nk} = \frac{d\, y_{nk}}{d\bar{\theta}}$ $\qquad$ y_{nk} : output of kth node in the output layer when $\bar{x}_n$ is input.

$\mathcal{E}$: Sum-of-squared error

Broyden-Fletcher-Goldfarb-Shanno (BFGS) Method:

Approximation of $H^{-1}(m)$ by a matrix $G(m)$

$\bar{\theta}(m+1) = \bar{\theta}(m) - \eta\, H^{-1}(m)\, \bar{g}(m)$

$\bar{\theta}(m+1) = \bar{\theta}(m) - \eta\, G(m)\, \bar{g}(m)$

$\bar{q} = \bar{g}(m+1) - \bar{g}(m)$

$\bar{s} = \bar{\theta}(m+1) - \bar{\theta}(m)$

$$G(m+1) = G(m) - \frac{\bar{s}\,\bar{s}^t}{\bar{q}^t \bar{q}} - \frac{G(m)\, \bar{q}\, \bar{q}^t G(m)}{\bar{q}^t G(m)\, \bar{q}} + (\bar{q}^t G(m)\, \bar{q}) \cdot (\bar{v}\,\bar{v}^t)$$

$$\bar{v} = \frac{\bar{s}}{\bar{s}^t \bar{s}} - \frac{G(m)\,\bar{q}}{\bar{q}^t G(m)\,\bar{q}}$$

L-BFGS : BFGS with limited memory

✓ $G(0)$: Identity matrix

$\bar{\theta}(0)$: Initialized using an initialization method.

Regularization methods:

L_2 Regularization

$$\hat{\mathcal{E}}_{avg} = \mathcal{E}_{avg} + \frac{\lambda}{2}\|\bar{\theta}\|^2$$

$$\Delta\bar{\theta} = -\eta\, \frac{d\hat{\mathcal{E}}_{avg}}{d\bar{\theta}}$$

$$= -\eta\, \frac{d}{d\bar{\theta}}\left(\mathcal{E}_{avg} + \frac{\lambda}{2}\|\bar{\theta}\|^2\right)$$

$$\Delta\bar{\theta} = -\eta\, \frac{d\mathcal{E}_{avg}}{d\bar{\theta}} \underbrace{- \eta\lambda\bar{\theta}}$$

Dropout and Drop-connect Methods for regularization :

Mini-batch mode of learning

Dropout : A few nodes (in layers other than output layer) are randomly chosen for dropping in a weight update step.

Drop-Connect : A few connections are randomly chosen for dropping in a weight update step.

Thinned network

Ensemble : A number of thinned networks

Committee Machine (CM)

L models

Bagging : Bootstrap Aggregating

Model with only one output

For λ^{th} model : output is $y_\lambda(\bar{x})$

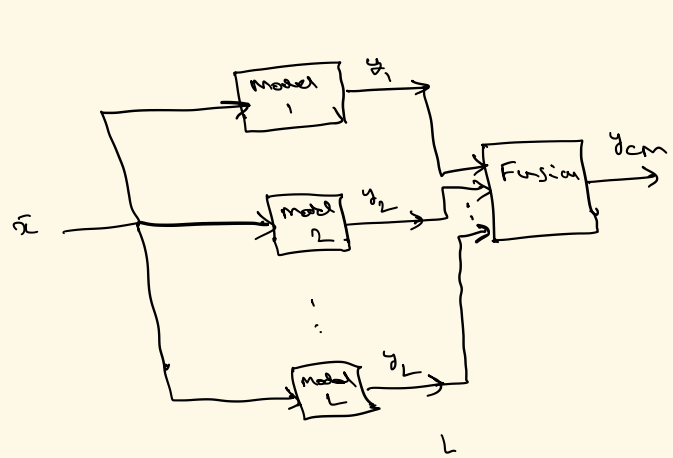
$$y_\lambda(\bar{x}) = \underset{\substack{\uparrow \\ \text{Desired output}}}{t(\bar{x})} + \underset{\substack{\uparrow \\ \text{Error for } \lambda^{th} \text{ model}}}{\mathcal{E}_\lambda(\bar{x})}$$

Average of sum-of-squared error for λ^{th} model :

$$E_{\bar{x}}\left[\{y_\lambda(\bar{x}) - t(\bar{x})\}^2\right] = E_{\bar{x}}\left[(\mathcal{E}_\lambda(\bar{x}))^2\right]$$

Average error by all the model :

$$\mathcal{E}_{Avg} = \frac{1}{L}\sum_{\lambda=1}^{L} \underbrace{E_{\bar{x}}\left[(\mathcal{E}_\lambda(\bar{x}))^2\right]}$$



$$y_{cm}(\bar{x}) = \frac{1}{L}\sum_{\lambda=1}^{L} y_\lambda(\bar{x})$$

$$\mathcal{E}_{cm} = E_{\bar{x}}\left[(y_{cm}(\bar{x}) - t(\bar{x}))^2\right]$$

$$= E_{\bar{x}}\left[\left\{\left(\frac{1}{L}\sum_{\lambda=1}^{L} y_\lambda(\bar{x})\right) - t(\bar{x})\right\}^2\right]$$

$$= E_{\bar{x}}\left[\left\{\left(\frac{1}{L}\sum_{\lambda=1}^{L} y_\lambda(\bar{x})\right) - \left(\frac{1}{L}\sum_{\lambda=1}^{L} t(\bar{x})\right)\right\}^2\right]$$

$$= E_{\bar{x}}\left[\left\{\frac{1}{L}\sum_{\lambda=1}^{L} \underbrace{(y_\lambda(\bar{x}) - t(\bar{x}))}\right\}^2\right]$$

$$= E_{\bar{x}}\left[\left\{\frac{1}{L}\sum_{\lambda=1}^{L} \mathcal{E}_\lambda(\bar{x})\right\}^2\right]$$

$$= E_{\bar{x}}\left[\left(\frac{1}{L}\sum_{\lambda=1}^{L} \mathcal{E}_\lambda(\bar{x})\right)\left(\frac{1}{L}\sum_{j=1}^{L} \mathcal{E}_j(\bar{x})\right)\right]$$

$$= E_{\bar{x}}\left[\frac{1}{L^2}\sum_{\lambda=1}^{L}\sum_{j=1}^{L} \mathcal{E}_\lambda(\bar{x})\, \mathcal{E}_j(\bar{x})\right]$$

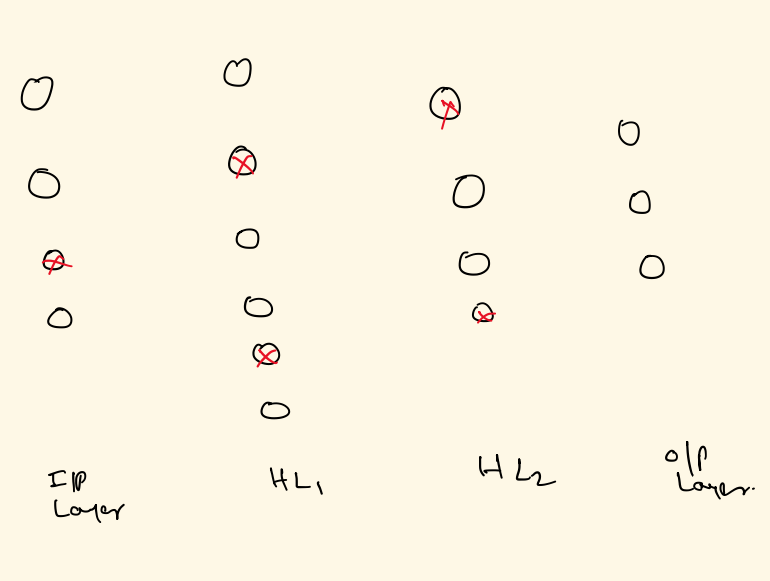
$$= E_{\bar{x}}\left[\frac{1}{L^2}\sum_{\lambda=1}^{L} (\mathcal{E}_\lambda(\bar{x}))^2 + \frac{1}{L^2}\sum_{\lambda}\sum_{\substack{j \\ \lambda \neq j}} \mathcal{E}_\lambda(\bar{x}) \cdot \mathcal{E}_j(\bar{x})\right]$$

$$\mathcal{E}_{cm} = \frac{1}{L} \cdot \underbrace{\frac{1}{L}\sum_{\lambda=1}^{L} E_{\bar{x}}(\mathcal{E}_\lambda(\bar{x}))^2}_{\mathcal{E}_{Avg}} + \frac{1}{L^2}\underbrace{\sum_{\lambda}\sum_{\substack{j \\ \lambda \neq j}} E_{\bar{x}}\left[\mathcal{E}_\lambda(\bar{x}) \cdot \mathcal{E}_j(\bar{x})\right]}$$

When $\mathcal{E}_\lambda(\bar{x})$ and $\mathcal{E}_j(\bar{x})$ are uncorrelated for all pairs of λ and j

$$\mathcal{E}_{cm} = \frac{1}{L} \cdot \mathcal{E}_{Avg}$$

Dropout Method:



Thinned networks

No. of nodes in the input layer and hidden layers : M

No. of thinned network : 2^M

Probability of a node being retained over the training process.

No. of mini batches : B

No. of epochs : L

No. of updates : L·B

n_j : No. of times the jth node is dropped.

$$p_j = \frac{L \cdot B - n_j}{L \cdot B}$$

$$\hat{w}_{jk} = w_{jk} \times p_j$$

Dropout rate :

Probability of retaining a node :

0.8 for input nodes

0.5 for hidden node.

Drop-Connect Method :

A few connections are dropped in every weight update step.

w_{ij}

n_{ij} : No. of times the connection associated with w_{ij} is dropped

$$p_{ij} = \frac{L \cdot B - n_{ij}}{L \cdot B}$$

$$\hat{w}_{ij} = w_{ij} * p_{ij}$$

Batch Normalization : Used in CNNs

Layer Normalization : Used in RNNs, Transformer models.

Input normalization :

Mean-variance normalization :

$x_i \qquad \mu_i,\ \sigma_i^2$ over training examples

$\hat{x}_i = \frac{x_i - \mu_i}{\sigma_i}$

$\hat{\mu}_i = 0$

$\hat{\sigma}_i^2 = 1$

$-3.0 \leq \hat{x}_i \leq 3.0$

Internal Covariant Shift :

Mini-batch mode of learning

$\mathcal{D} = \{\bar{x}_n, t_n\}_{n=1}^{N}$

Batch size : M

$\mathcal{D}_1, \mathcal{D}_2, \dots \mathcal{D}_b, \dots, \mathcal{D}_B$

$B = \frac{N}{M}$

j^{th} node in a hidden layer

$\bar{x}_n \qquad s_{nj} \qquad \phi(a_{nj})$

Mini-batch $\mathcal{D}_b \qquad \mu_{bj} \qquad \sigma_{bj}^2$

↑ Average of s_{nj} for examples in $\mathcal{D}_b$.

$HL_1 \qquad HL_2$

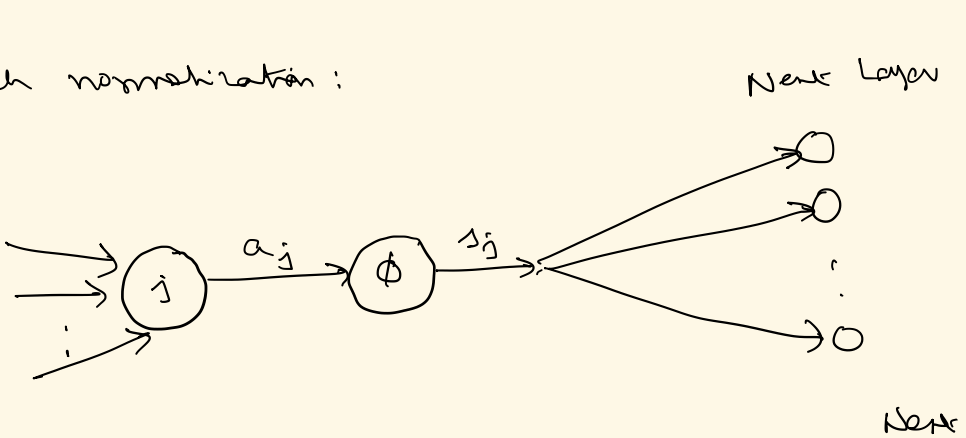
Changes in distributions of hidden layer nodes : Internal covariant shift

Post-activation Batch Normalization : Normalization on outputs of nodes $\quad s_j\ \phi(a_j)$

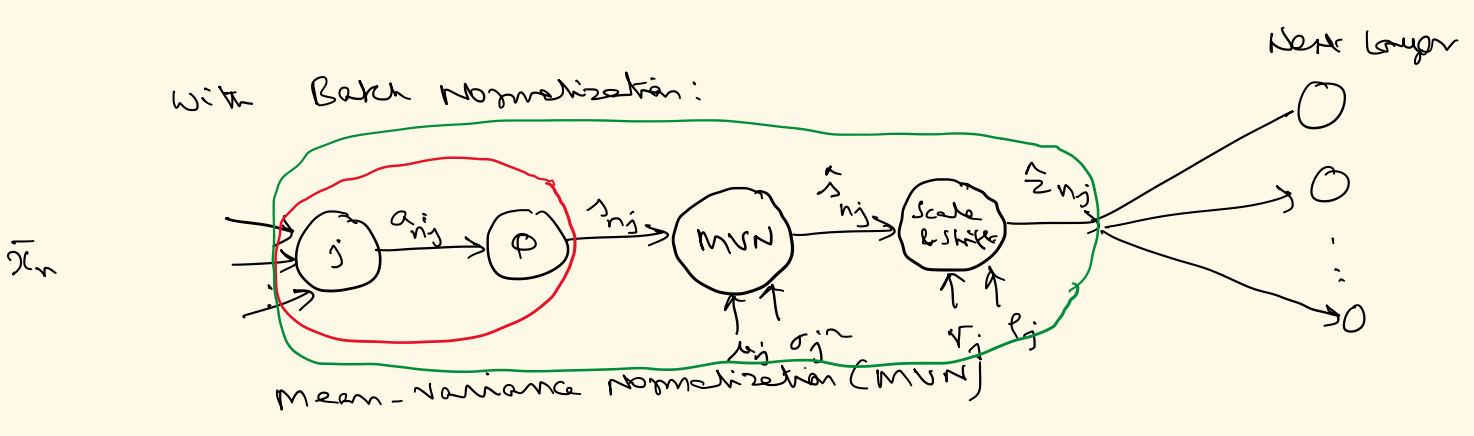
Pre-activation Batch Normalization : Normalization on activation values of nodes $\quad a_j$

Post-activation Batch Normalization :

Without Batch normalization :



With Batch Normalization :



s_{nj} for every example in $\mathcal{D}_b$

μ_j, σ_j^2

$E[\hat{s}_{nj}] = 0 = \hat{\mu}_j$

$E[(\hat{s}_{nj} - \hat{\mu}_j)^2] = 1 = \hat{\sigma}_j^2$

$-3.0 \leq \hat{s}_{nj} \leq 3.0$

Scaling and Shift :

$\hat{z}_{nj} = r_j \hat{s}_{nj} + \rho_j$

r_j and ρ_j are also learnt during training process.

$E[\hat{z}_{nj}] = \rho_j$

$E[(\hat{z}_{nj} - \rho_j)^2] = r_j^2$

Backpropagation with Batch Normalization :

Mini-batch : $\mathcal{D}_b = \{\bar{x}_n, t_n\}_{n=1}^{M}$

Batch size : M

$$w(m+1) = w(m) - \eta\, \frac{\partial \mathcal{E}_b}{\partial w}\Big|_{w=w(m)}$$

$$\mathcal{E}_b = \frac{1}{M}\sum_{\substack{n=1 \\ \bar{x}_n \in \mathcal{D}_b}}^{M} \tilde{\mathcal{E}}_n$$

$$\tilde{\mathcal{E}}_n = \frac{1}{2}\sum_{m=1}^{K} (t_{nm} - s_{nm}^o)^2$$

I/P Layer d $\qquad$ HL J $\qquad$ o/p layer K

Without BN :

$$\frac{\partial \tilde{\mathcal{E}}_n}{\partial w_{jk}^o} = \overbrace{-(t_{nk} - s_{nk}^o)\, \frac{d\phi^o(a_{nk}^o)}{da_{nk}^o}}^{\delta_k^o} \cdot s_{nj}^h$$

$$\frac{\partial \tilde{\mathcal{E}}_n}{\partial w_{ij}^h} = -\left(\sum_{m=1}^{K} w_{jm}^o\, \delta_m^o\right) \underbrace{\frac{d\phi^h(a_{nj}^h)}{da_{nj}^h}} \cdot x_{ni}$$

With BN :

$$\frac{\partial \tilde{\mathcal{E}}_n}{\partial w_{jk}^o} = -(t_{nk} - s_{nk}^o)\, \frac{d\phi^o(a_{nk}^o)}{da_{nk}^o} \cdot \hat{z}_{nj}^h$$

$$\frac{\partial \tilde{\mathcal{E}}_n}{\partial w_{ij}^h} = \underbrace{-\left(\sum_{m=1}^{K} w_{jm}^o\, \delta_m^o\right) \cdot \frac{d\hat{z}_{nj}^h}{da_{nj}^h}}_{\delta_j^h} \cdot x_{ni}$$

$$\frac{d\hat{z}_{nj}^h}{da_{nj}^h} = \frac{d}{da_{nj}^h}\left(r_j \hat{s}_{nj}^h + \rho_j\right)$$

$$= r_j\, \frac{d\hat{s}_{nj}^h}{da_{nj}^h}$$

$$= r_j\, \frac{d}{da_{nj}^h}\left(\frac{s_{nj}^h - \mu_j}{\sqrt{\sigma_j^2 + \epsilon}}\right)$$

$$\boxed{\begin{aligned} \mu_j &= \frac{1}{M}\sum_{n=1}^{M} s_{nj}^h \\ \sigma_j^2 &= \frac{1}{M}\sum_{n=1}^{M} (s_{nj}^h - \mu_j)^2 \end{aligned}}$$

$$\frac{d\hat{z}_{nj}^h}{da_{nj}^h} = r_j \cdot \frac{1}{\sqrt{\sigma_j^2 + \epsilon}} \cdot \frac{d\phi^h(a_{nj}^h)}{da_{nj}^h}\left(1 - \frac{1}{M}\right) \cdot \left[1 - (\hat{s}_{nj}^h)^2\right]$$

$$\frac{\partial \tilde{\mathcal{E}}_n}{\partial r_j} = \frac{\partial}{\partial r_j}\, \frac{1}{2}\sum_{m=1}^{K} (t_{nm} - s_{nm}^o)^2$$

$$= \sum_{m=1}^{K} (t_{nm} - s_{nm}^o)\left(-\frac{\partial s_{nm}^o}{\partial r_j}\right)$$

$$= \sum_{m=1}^{K} (t_{nm} - s_{nm}^o)\left[-\underbrace{\frac{\partial}{\partial r_j}\left[\sum_{l=1}^{J} w_{lm}^o\, \hat{z}_{nl}^h + b_m^o\right]} \cdot \frac{d\phi^o(a_m^o)}{da_m^o}\right]$$

$$-w_{jm}^o \frac{\partial}{\partial r_j}(\hat{z}_{nj}^h)$$

$$-w_{jm}^o \frac{\partial}{\partial r_j}(r_j \hat{s}_{nj}^h + \rho_j)$$

$$-w_{jm}^o\, \hat{s}_{nj}^h$$

$$\frac{\partial \tilde{\mathcal{E}}_n}{\partial r_j} = \sum_{m=1}^{K} (t_{nm} - s_{nm}^o)\left(-w_{jm}^o\, \hat{s}_{nj}^h\right) \cdot \frac{d\phi^o(a_m^o)}{da_m^o}$$

$$\frac{d\tilde{\mathcal{E}}_n}{\partial r_j} = -\sum_{m=1}^{K} w_{jm}^o\, \delta_m^o \cdot \hat{s}_{nj}^h$$

$$\frac{\partial \tilde{\mathcal{E}}_n}{\partial \rho_j} = -\sum_{m=1}^{K} w_{jm}^o\, \delta_m^o$$

Inference phase :

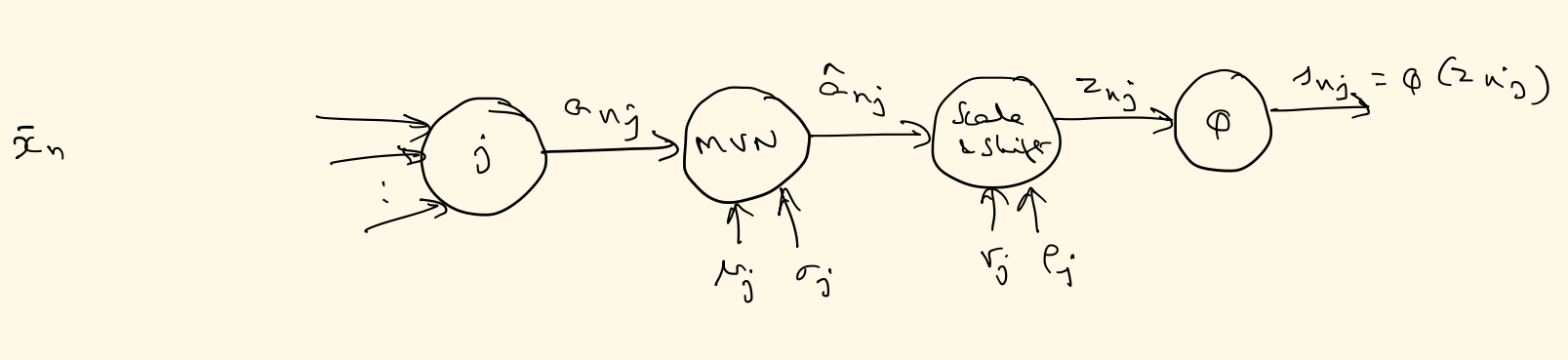
$$\mu_j = \frac{1}{N}\sum_{n=1}^{N} s_{nj}^h$$

$$\sigma_j^2 = \frac{1}{N}\sum_{n=1}^{N} (s_{nj}^h - \mu_j)^2$$

$$\hat{s}_{nj}^h = \frac{s_{nj}^h - \mu_j}{\sqrt{\sigma_j^2 + \epsilon}}$$

$$\hat{z}_{nj}^h = r_j \hat{s}_{nj}^h + \rho_j$$

Batch normalization with pre-activation normalization



Batch normalisation in CNNs

Layer Normalization :

Hidden layer with J nodes

$\bar{x}_n \qquad s_{nj}, \quad j = 1, 2, \dots, J$

$$\mu_j = \frac{1}{J}\sum_{j=1}^{J} s_{nj}$$

$$\sigma_j = \frac{1}{J}\sum_{j=1}^{J} (s_{nj} - \mu_j)^2$$

$$\hat{s}_{nj} = \frac{s_{nj} - \mu_j}{\sqrt{\sigma_j^2 + \epsilon}}$$

Layer normalization is used in pattern mode of training in RNNs and transformer models.