Today's Agenda

1) Loss Functions

Regression

Mean Squared Error

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y - \hat{y})^2$$

Squares of Errors $\longrightarrow$ large errors will be penalized

Outlier Sensitivity $\longrightarrow$ Sensitive

2) MAE (Mean Absolute Error)

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |y - \hat{y}|$$

Measure the mean absolute difference

1) Robust to outliers
2) Linear Penalty

* Not differentiable at zero*

* Gradient is constant *
Slower Convergence

3) Huber loss (Smooth MAE)

$$\text{Huber} = \begin{cases} \frac{1}{2}(y-\hat{y})^2 & \text{if } |y-\hat{y}| \le \delta \\ \delta|y-\hat{y}| - \frac{1}{2}\delta^2 & \text{otherwise} \end{cases}$$

Tuning $\leftarrow \delta \rightarrow$ threshold (0, 1)

MSE → Small Errors
MAE → Large Errors

Classification

1) Log Loss (Binary Cross Entropy)

* Binary Classification *

$$BCE = -\frac{1}{n}\sum_{i=1}^{n}\left[y \cdot \log(\tilde{y}) + (1-y)\log(1-\tilde{y})\right]$$

Predicted Probability

Penalize → wrong predictions

Activation → Sigmoid

Strong gradients

* Sensitive to mislabeled data *

Outlier Sensitivity → High

Multiclass Classification

1) Categorical Cross Entropy

$$CCE :- -\frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{K} y_{ij} \log(\hat{y}_{ij})$$

n = data points
K = no of classes

Outlier Sensity -> High

Activation => Softmax

* Probabilistic *

3) Hinge loss

$$L = \frac{1}{n} \sum_{i=1}^{n} \max(0, 1 - y \cdot \hat{y})$$

Y_true = {-1, 1}

Outlier Sensity :- Medium

* Binary Classification *

Weight Initialization

1) Zero Weight Ini

$W = 0$, $b = 0$

All neurons learn the same features

* No learning *
* Same gradient values ×

Never, $W = 0$

2) Random Initialization

$W \sim N(0, 0.1)$ $(-0.01, 0.01)$

✓ shallow networks

* Vanishing gradient *

Xavier / Glorot Initialization

n_{in} = No of input neurons

n_{out} = No of output neurons

Uniform

$$W \sim U\left(-\sqrt{\frac{6}{n_{in} + n_{out}}}, \sqrt{\frac{6}{n_{in} + n_{out}}}\right)$$

Normal

$$W \sim N\left(0, \frac{2}{n_{in} + n_{out}}\right)$$

Prevents Vanishing Gradient

Faster Convergence

Use :- Sigmoid, tanh

Not use :- Relu

layer :- Fully Connected / Hidden / Dense / Linear

* ANN *

CNN → Relu ✓

2) He Intialization / Kaiming Initialization

Uniform

$$W = U\left(-\frac{\sqrt{6}}{n_{in}}, \frac{\sqrt{6}}{n_{in}}\right)$$

Normal

$$W = N\left(0, \frac{2}{n_{in}}\right)$$

Best → Relu Family works very good.

Cons :- Sigmoid

Pros :- Prevent V.G.P

Faster Convergance

Lecun Initialization LECUN

$$W \sim N\left(0, \frac{1}{n_{in}}\right)$$

* SELU *

1) Dropout
2) Batch Normalize

Dropout

Regularization Technique

prevent overfitting

Layer → Dropout

$p = (0 - 1)$

10 Neurons

$p = (0.2)$

2 - Neurons

deactivated

20% deactivate

turn - off

Shuffle

* loss of Info *

Overfitting

$(0.1 - 0.5)$ $(.2)$ $.25$ $.3)$

Small - medium

1 → Destructive

* Normalize * → dataset

Pros:-

1) Improve generalization
2) Simple to implement
3) Works good with deep networks

Layer 1 → Input (opening)

mid - closing

⋮

Layer n → Output (closing)

10 →

$$\left\{ \begin{matrix} 6 \\ 7 \\ 8 \\ 9 \end{matrix} \right\}$$

Layer 1 →

2 → 0.1

3 → 0.2

4 → 0.3

5 → 0.3

6 → 0.4

Batch Normalization

Samples, Features

$$X_i = \frac{X_i - \text{Mean}}{\text{Std. Dev}}$$

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\ldots m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \text{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_\mathcal{B} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_\mathcal{B}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_\mathcal{B})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_\mathcal{B}}{\sqrt{\sigma_\mathcal{B}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \text{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

Mean, $\mu_B \rightarrow \frac{1}{m}\sum_{i=1}^{m} x_i$

Variance, $\sigma_B^2 \rightarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_B)^2$

Normalization, $\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma^2 + \epsilon}}$

Scaling & shifting

$y_i \rightarrow \gamma \hat{x}_i + \beta$

$\beta, \gamma \rightarrow$ learnable Parameters

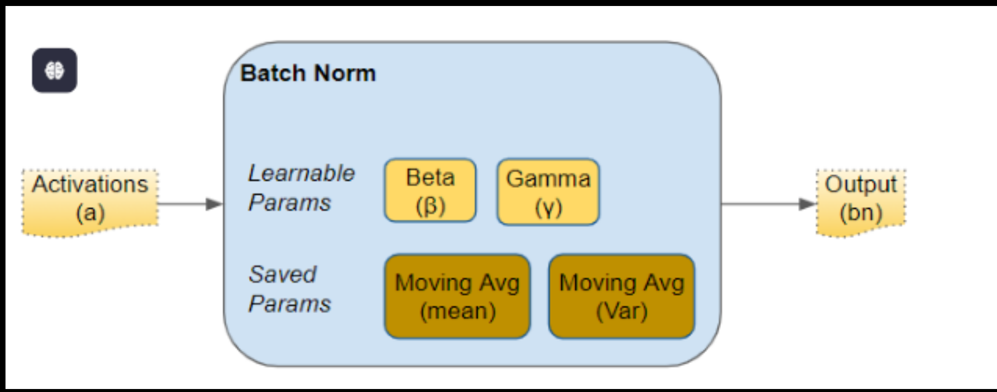
$\beta \rightarrow$ Shift

$\gamma \rightarrow$ scaling

$\beta, \gamma \rightarrow$ optimal activation

w, b, BN $\rightarrow$ learnable

BN $\rightarrow$ non-learnable



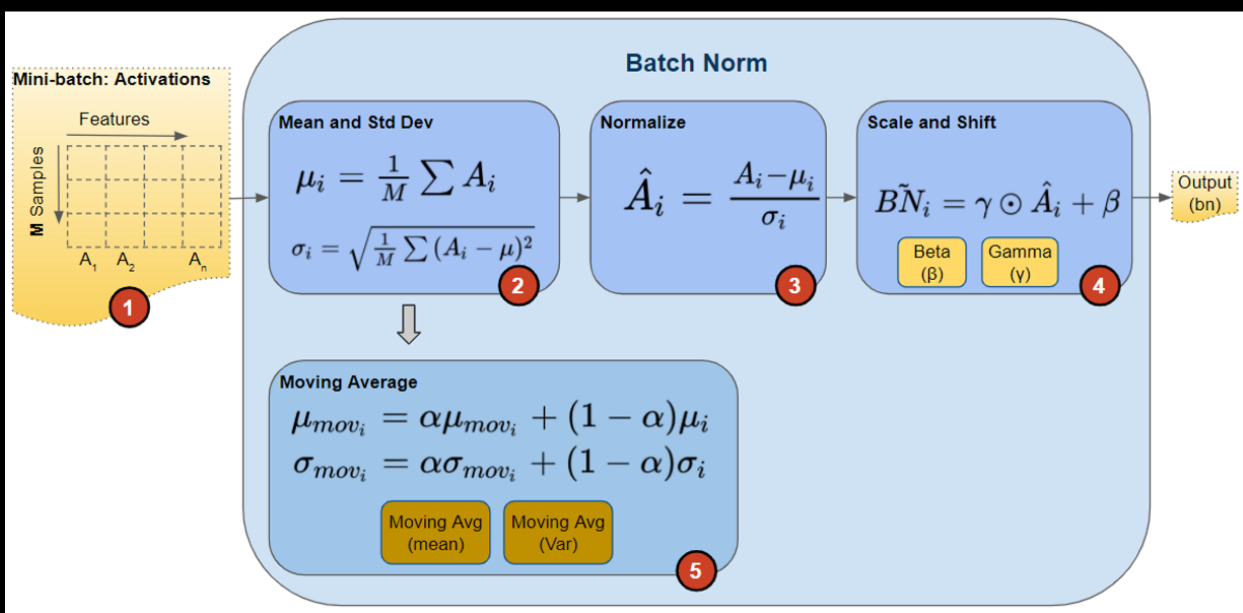
Training

Inference

b.s $n = 32$

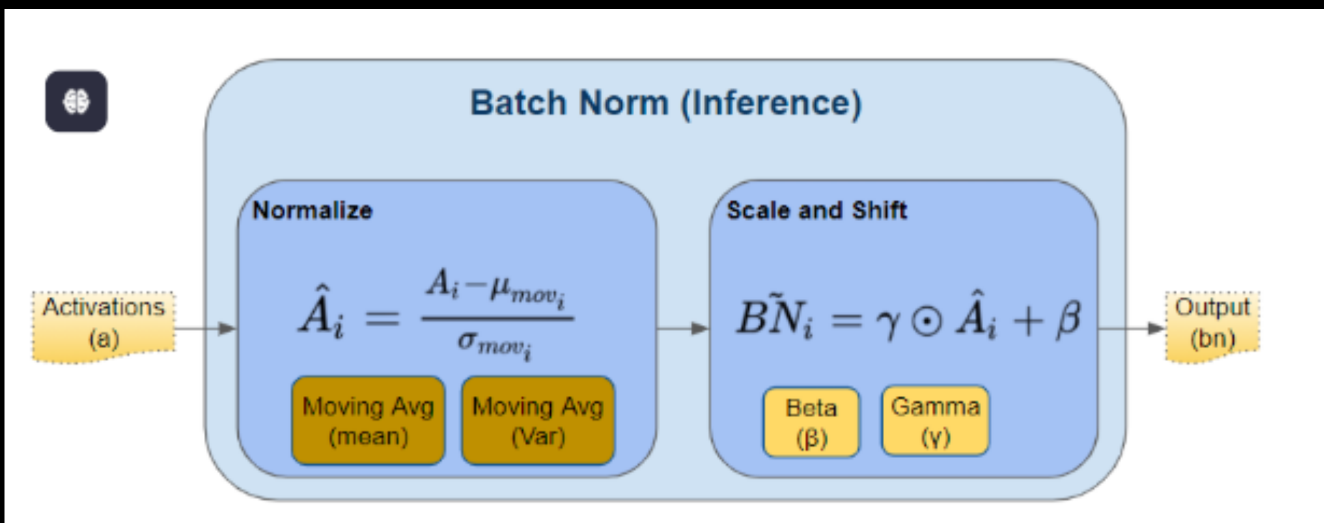
* Mov Avg { Mean, Var } *

* Saved * Model State



$\alpha + (1-\alpha)$

$$\beta = 0.9 = \begin{cases} \beta x + (1-\beta) x \\ 0.9 + (1-0.9) \\ 0.9 + .1 \end{cases}$$



Placement of BN layer

1) Before Activation layer

2) After Activation layer

* Hidden layer → Activation → BN

* Hidden layer → BN → Activation