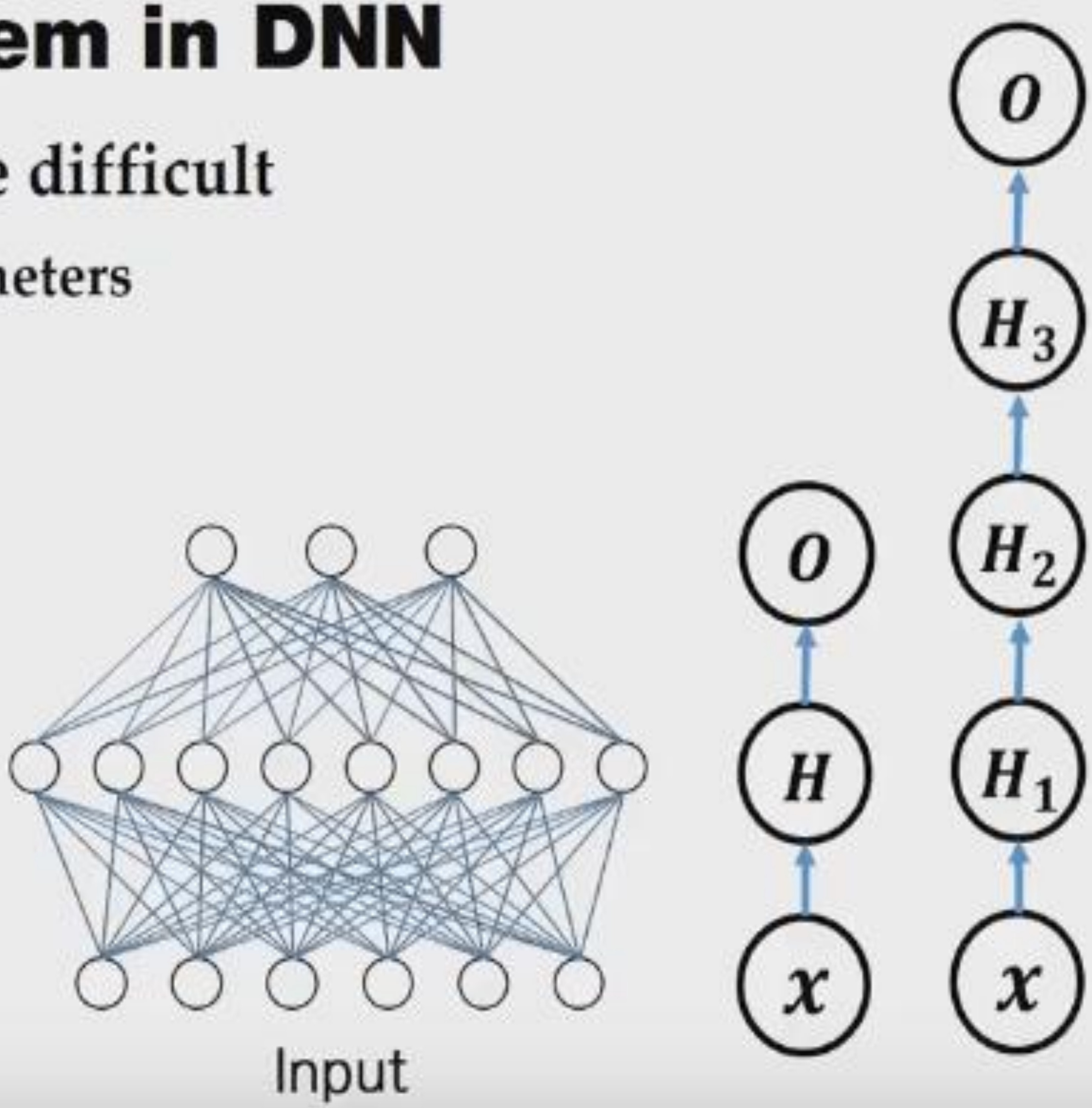
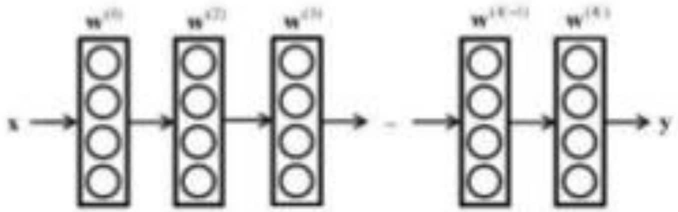
Learning Problem in DNN

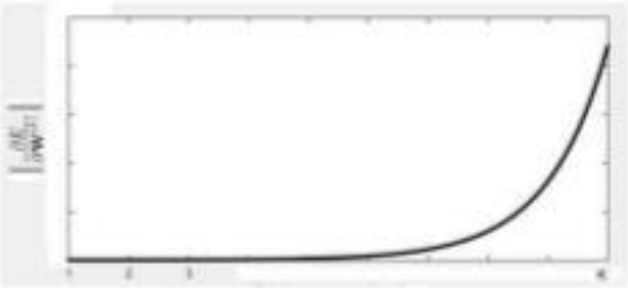
Training DNN is more difficult

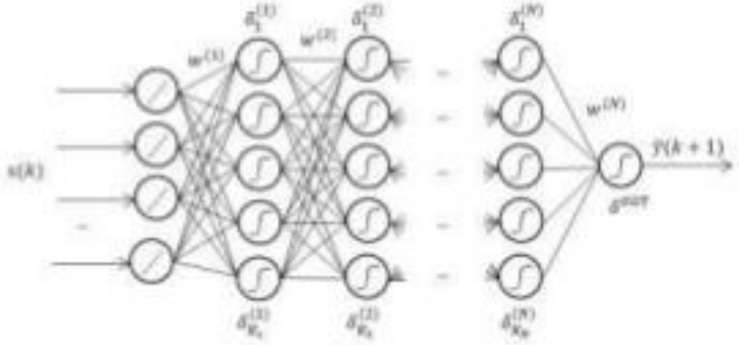
- Due to many parameters

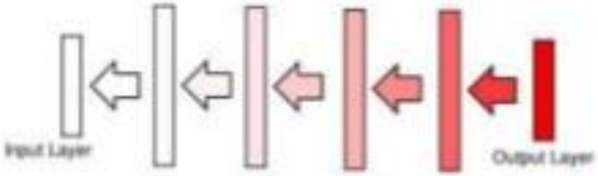


Bad effect of vanishing (exploding) gradients: a problem









$$\delta_j^{(m)} = f_j^{(m-1)\prime} \sum_i w_{ij}^{(m)} \delta_i^{(m+1)},$$

=>

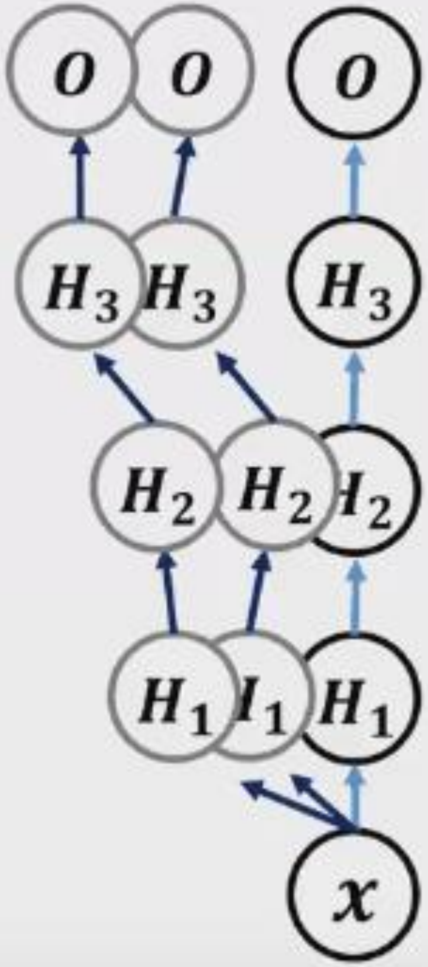
$$\frac{\partial E(k)}{\partial w_{ji}^{(m)}} \to 0 \;\; for\, m \to 1$$

$$\frac{\partial E(k)}{\partial w_{ji}^{(m)}} = \delta_j^{(m)} z_i^{(m-1)},$$

Learning Problem in DNN

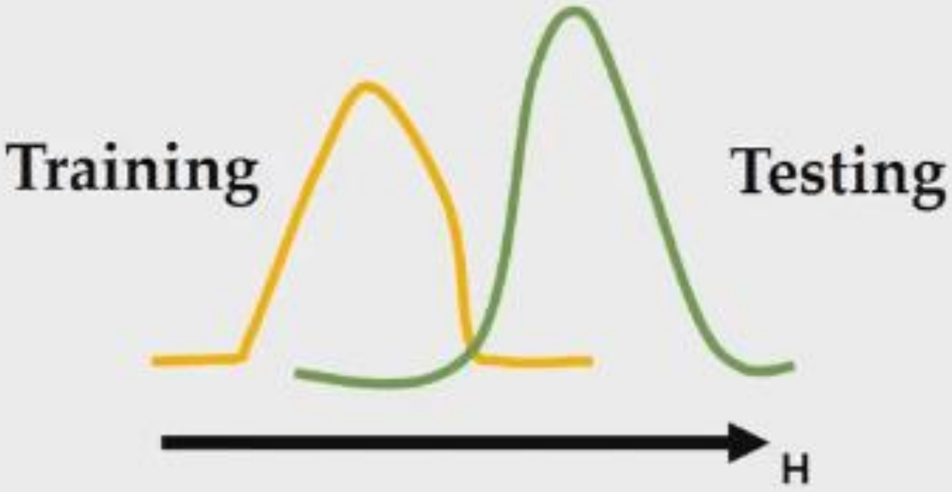
Training DNN is more difficult

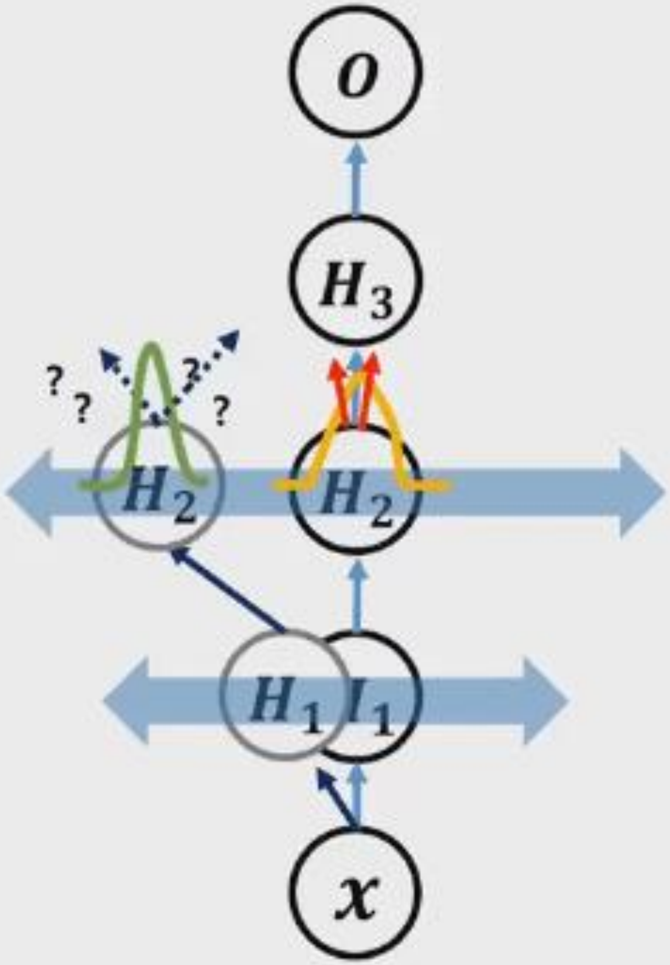
- Due to many parameters
- Small change in all weights could make vary different value in upper layer



Learning Problem in DNN

This variance is called
'Internal Covariate Shift'



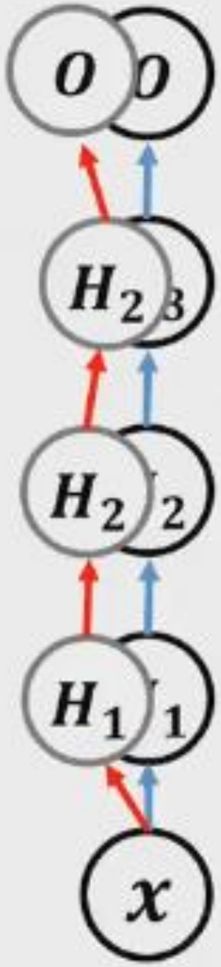


It is similar to the problem
where dist. of training and testing are different

Learning Problem in DNN

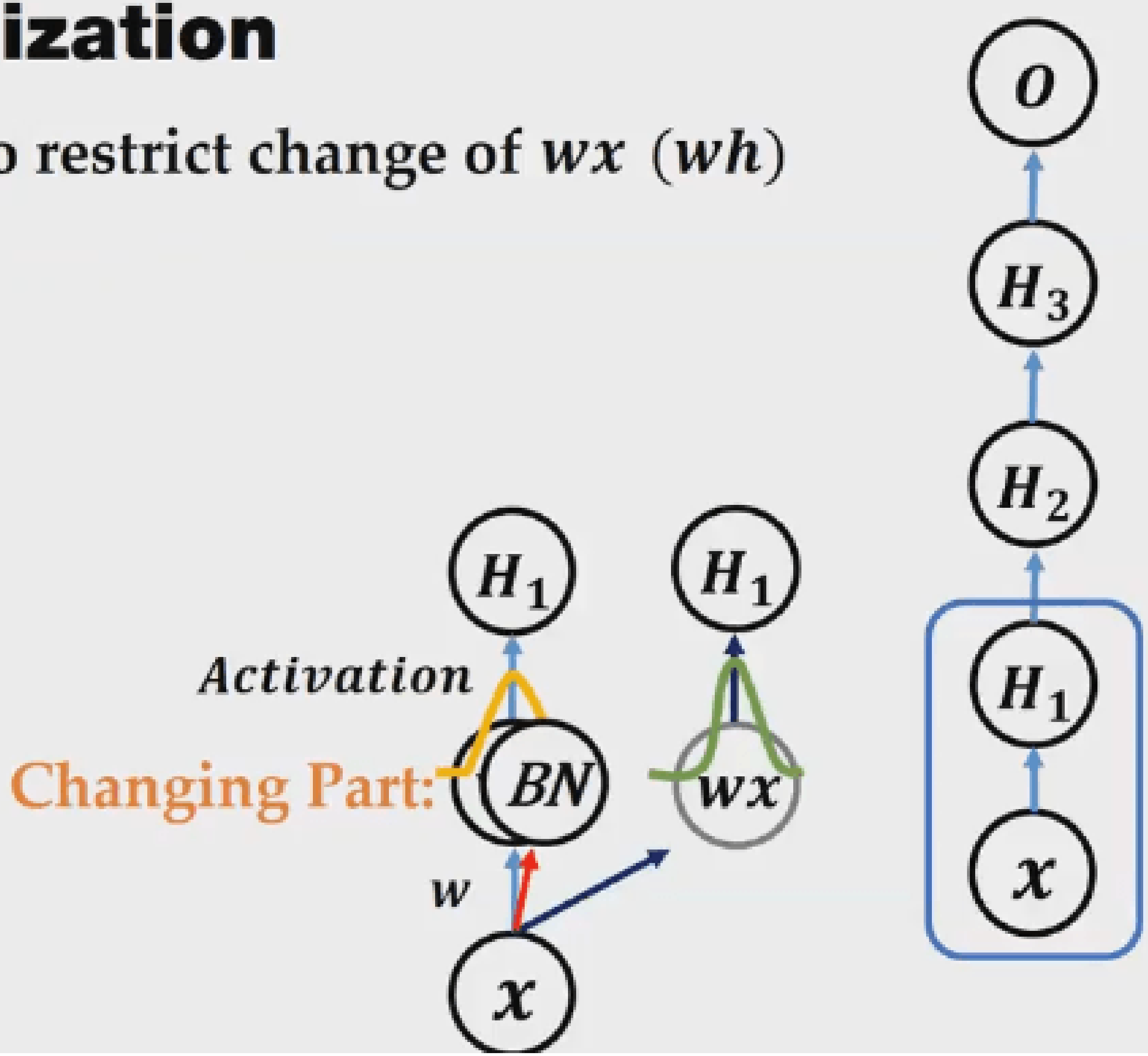
To address this Internal Covariate Shift problem, previous studies used

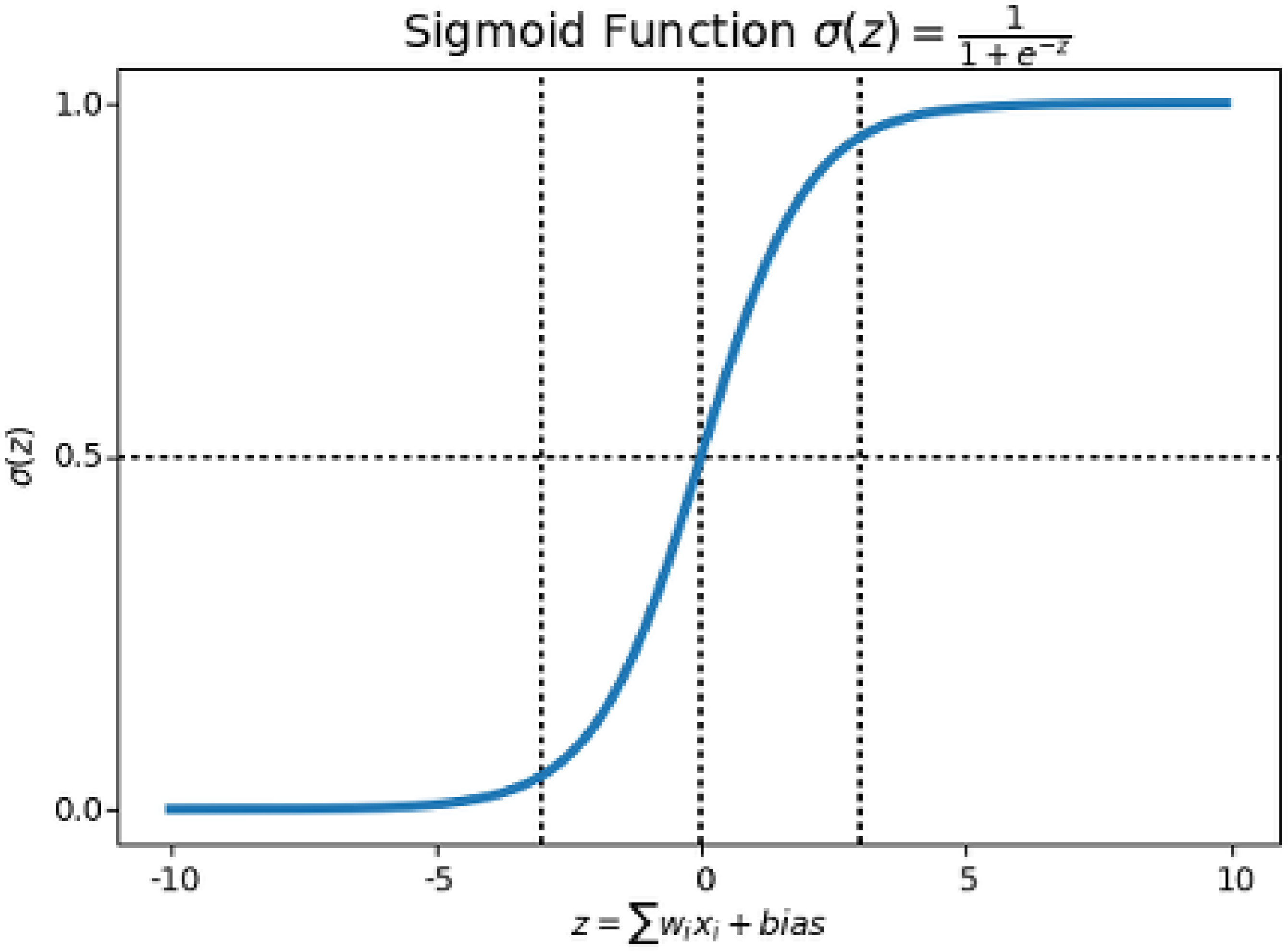
- careful initialization Difficult
- small learning rate Slow

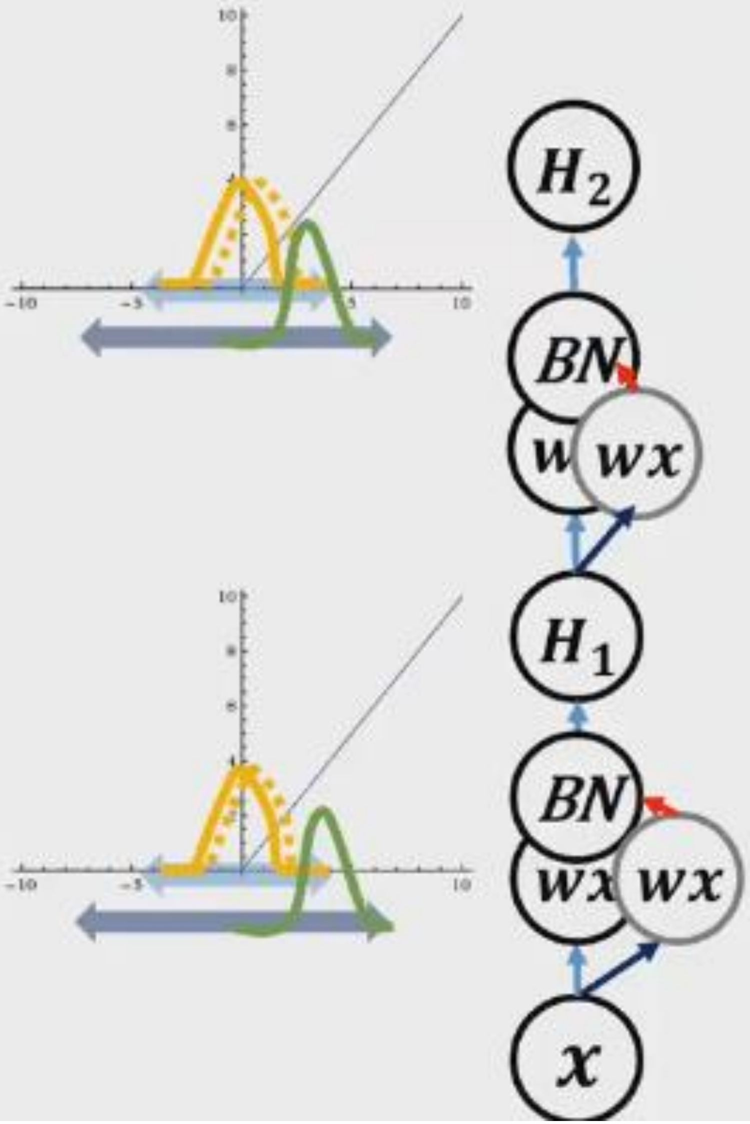


Batch Normalization

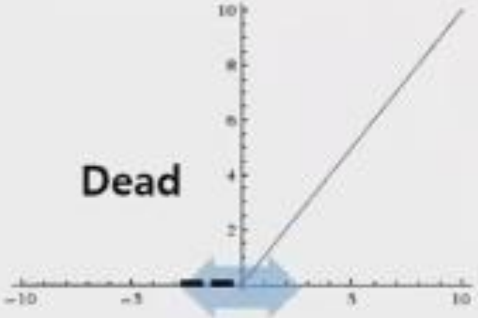
Batch Norm. want to restrict change of wx (wh)

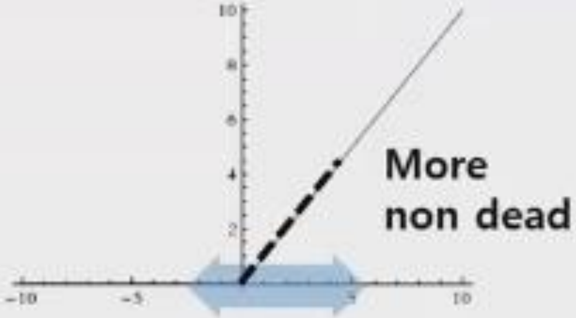



Sigmoid Function $\sigma(z) = \frac{1}{1+e^{-z}}$
1.0
0.5
0.0
$\sigma(z)$
-10
-5
0
5
10
$z = \sum w_i x_i + bias$

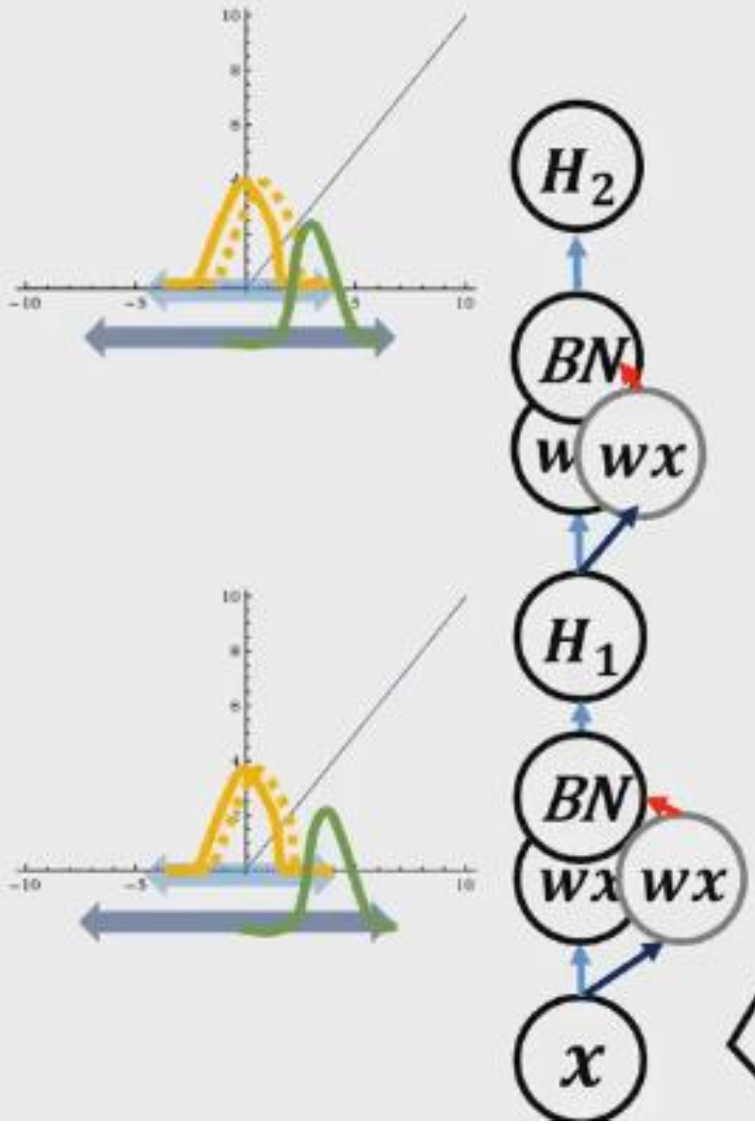


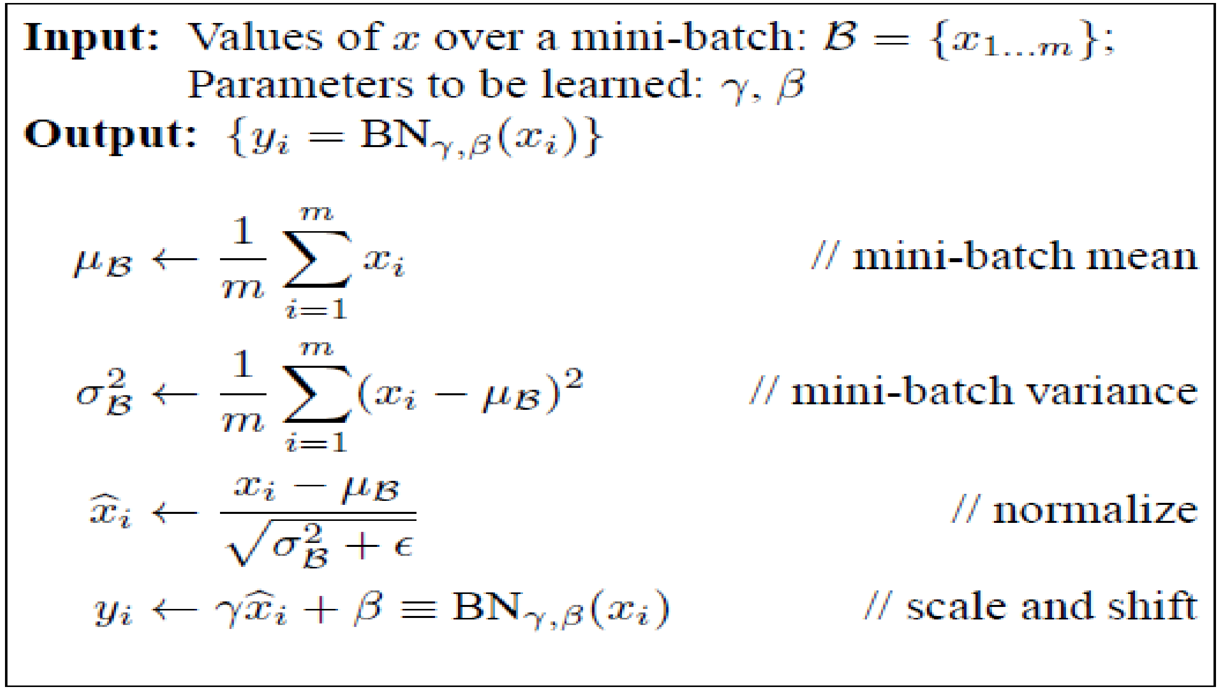
0 mean and 1 variance is preferred
But it is not best





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





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

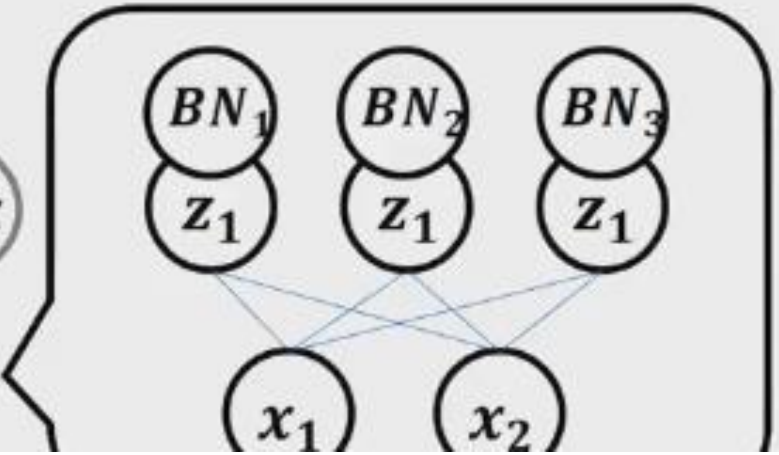
Output: $\{y_i = \text{BN}_{\gamma,\beta}(x_i)\}$

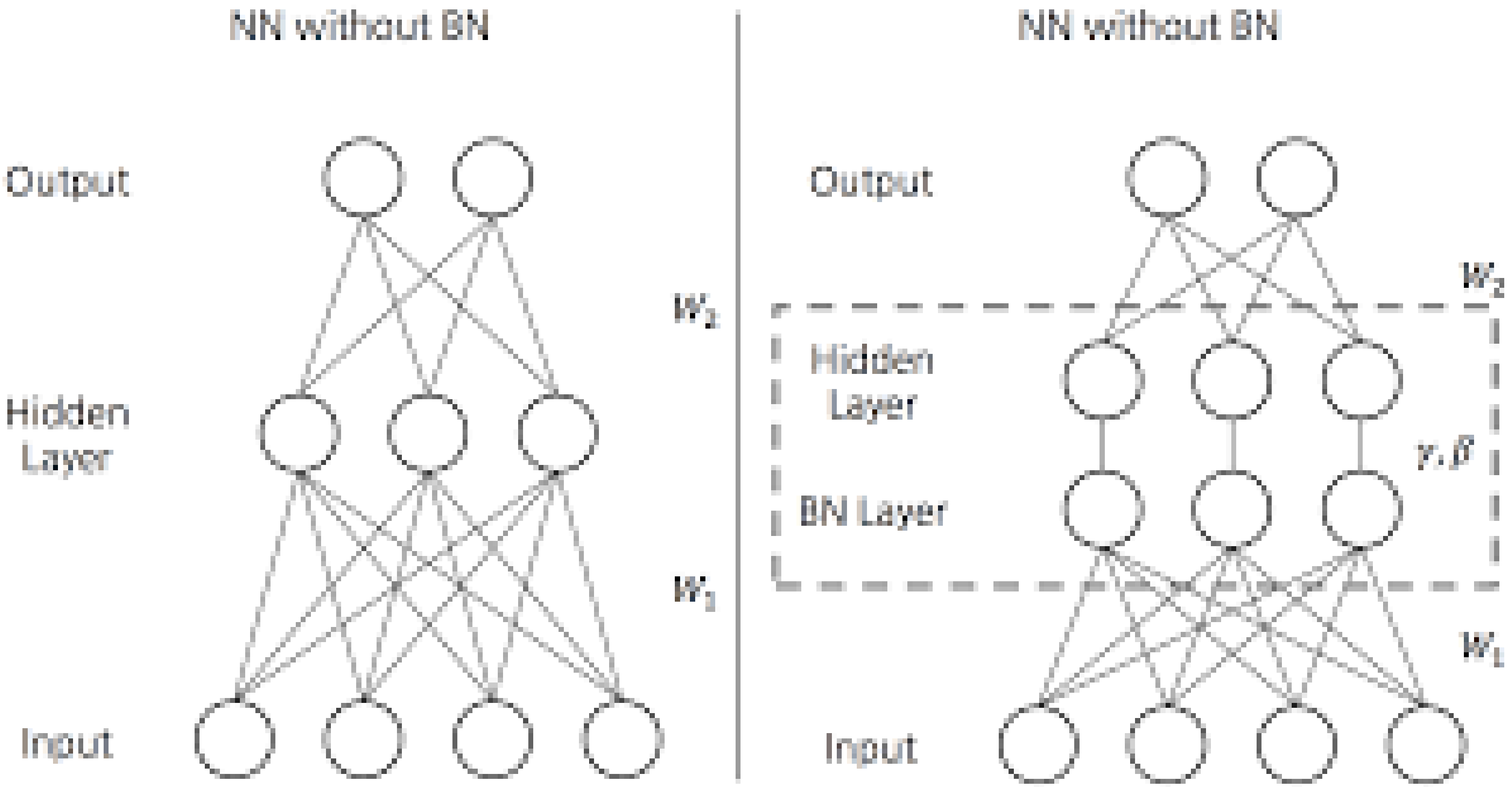
$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

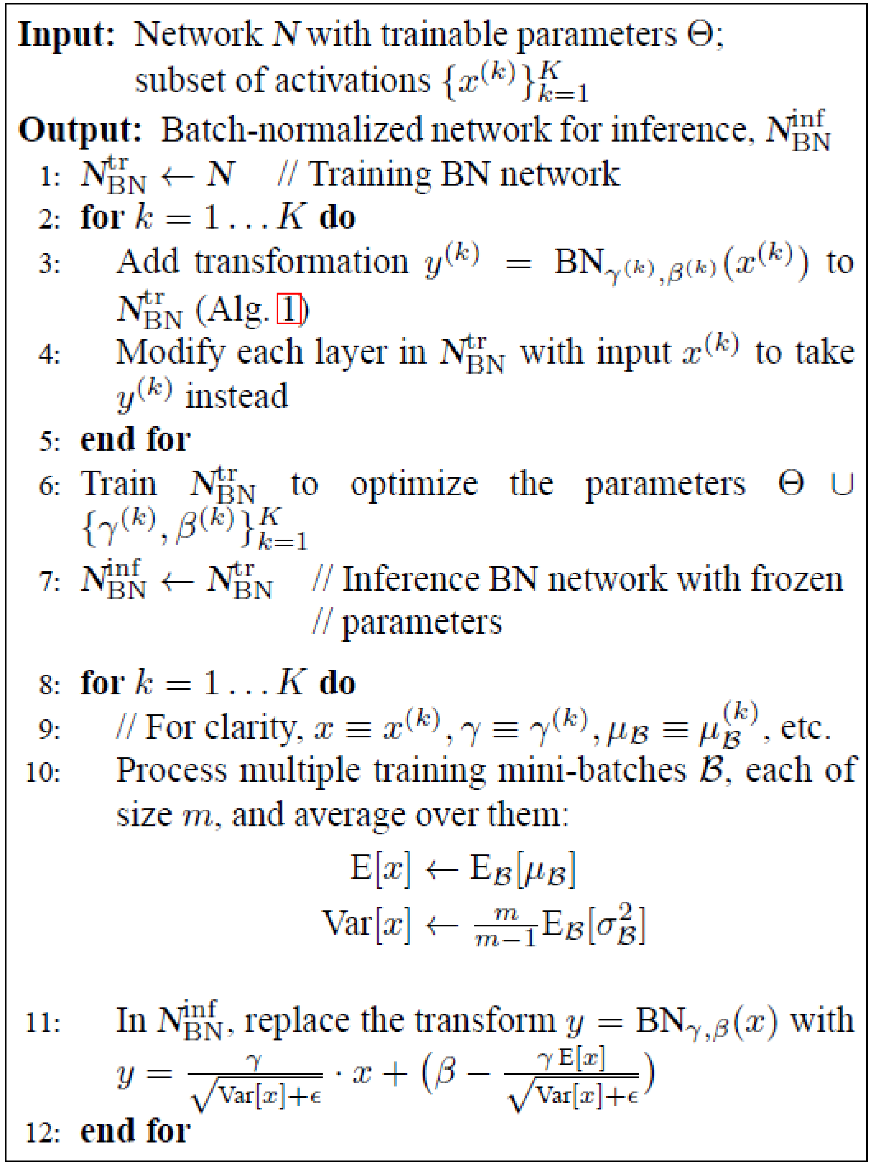
$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

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

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

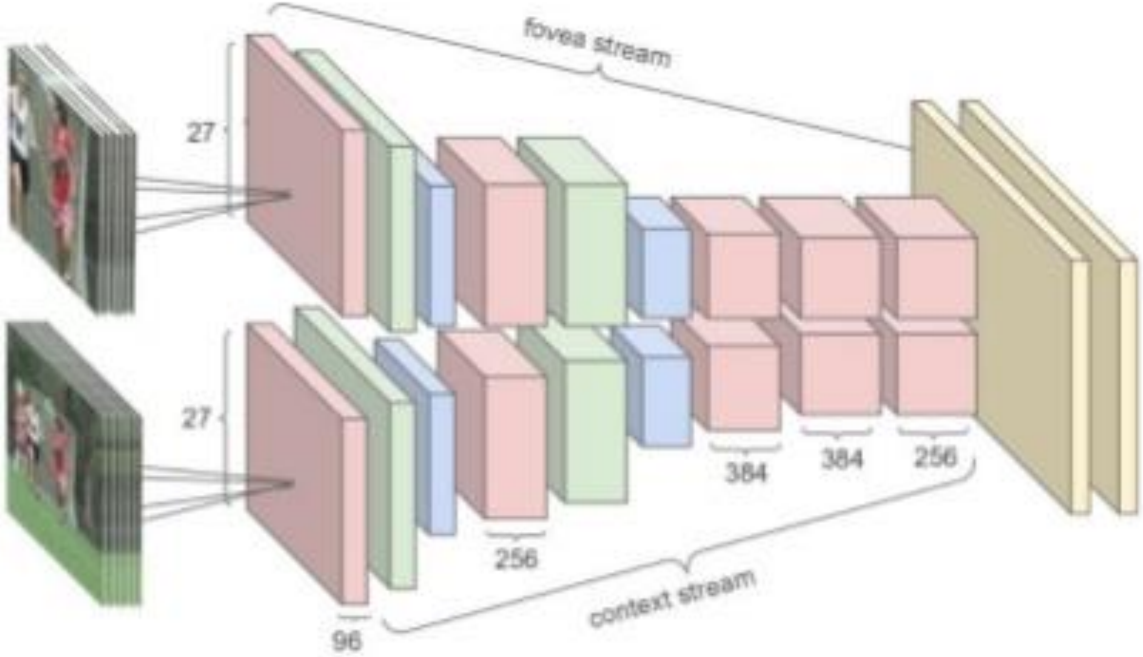



NN without BN
Output
Hidden Layer
Input
W_2
W_1
NN without BN
Output
W_2
Hidden Layer
γ, β
BN Layer
W_1
Input

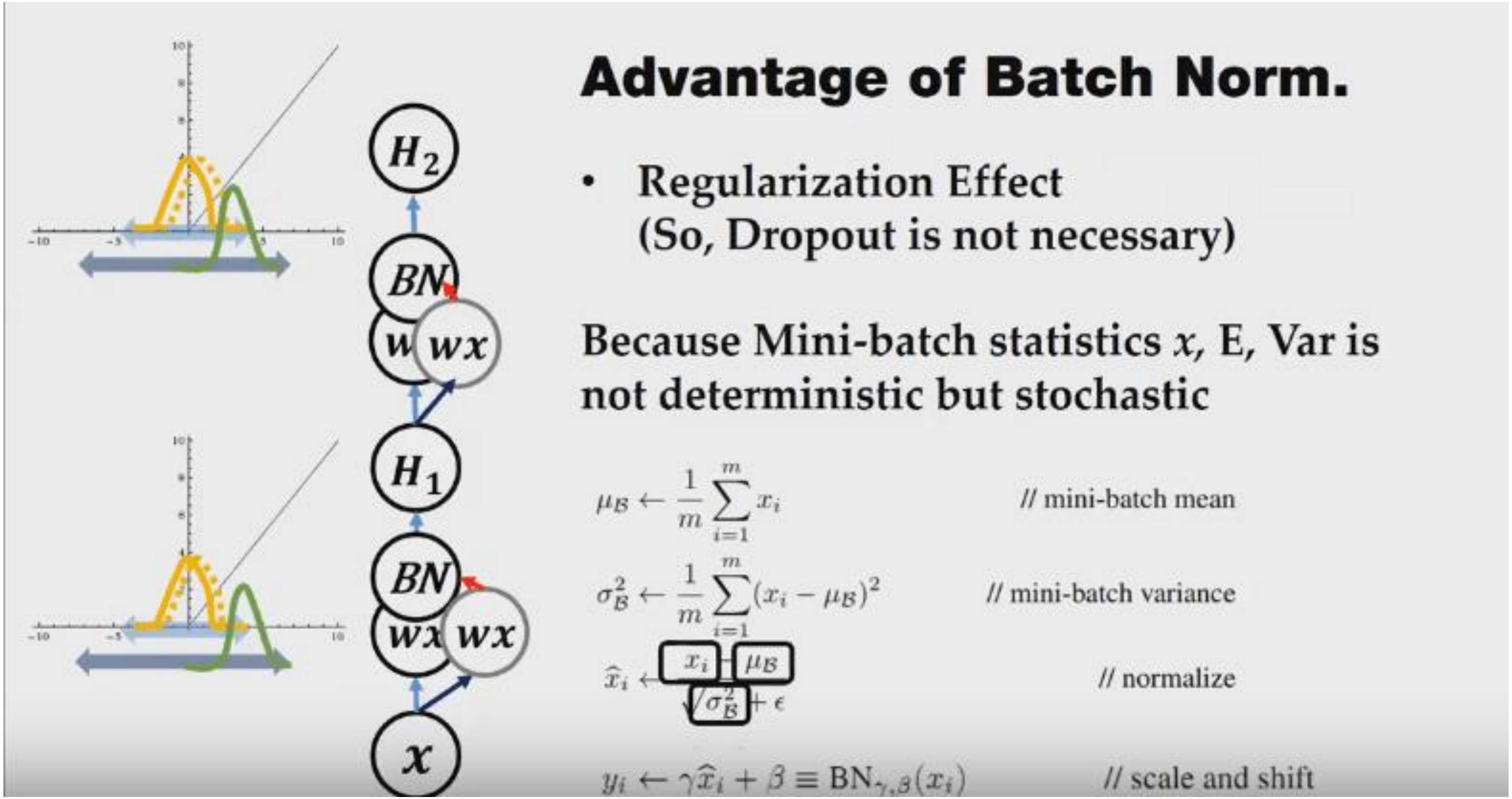


```
Input: Network $N$ with trainable parameters $\Theta$;
       subset of activations $\{x^{(k)}\}_{k=1}^{K}$
Output: Batch-normalized network for inference, $N_{\mathrm{BN}}^{\mathrm{inf}}$
$N_{\mathrm{BN}}^{\mathrm{tr}} \leftarrow N$    // Training BN network
for $k = 1 \dots K$ do
    Add transformation $y^{(k)} = \mathrm{BN}_{\gamma^{(k)},\beta^{(k)}}(x^{(k)})$ to $N_{\mathrm{BN}}^{\mathrm{tr}}$ (Alg. 1)
    Modify each layer in $N_{\mathrm{BN}}^{\mathrm{tr}}$ with input $x^{(k)}$ to take $y^{(k)}$ instead
end for
Train $N_{\mathrm{BN}}^{\mathrm{tr}}$ to optimize the parameters $\Theta \cup \{\gamma^{(k)}, \beta^{(k)}\}_{k=1}^{K}$
$N_{\mathrm{BN}}^{\mathrm{inf}} \leftarrow N_{\mathrm{BN}}^{\mathrm{tr}}$    // Inference BN network with frozen
                                           // parameters
for $k = 1 \dots K$ do
    // For clarity, $x \equiv x^{(k)}, \gamma \equiv \gamma^{(k)}, \mu_{\mathcal{B}} \equiv \mu_{\mathcal{B}}^{(k)}$, etc.
    Process multiple training mini-batches $\mathcal{B}$, each of size $m$, and average over them:
        $\mathrm{E}[x] \leftarrow \mathrm{E}_{\mathcal{B}}[\mu_{\mathcal{B}}]$
        $\mathrm{Var}[x] \leftarrow \frac{m}{m-1}\mathrm{E}_{\mathcal{B}}[\sigma_{\mathcal{B}}^2]$
    In $N_{\mathrm{BN}}^{\mathrm{inf}}$, replace the transform $y = \mathrm{BN}_{\gamma,\beta}(x)$ with
        $y = \frac{\gamma}{\sqrt{\mathrm{Var}[x]+\epsilon}} \cdot x + \left(\beta - \frac{\gamma\,\mathrm{E}[x]}{\sqrt{\mathrm{Var}[x]+\epsilon}}\right)$
end for
```

Recognition: DeepVideo: Multiscale



Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., & Fei-Fei, L. (2014, June). Large-scale video classification with convolutional neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on* (pp. 1725-1732). IEEE.



Advantage of Batch Norm.

- Regularization Effect
  (So, Dropout is not necessary)

Because Mini-batch statistics x, E, Var is not deterministic but stochastic

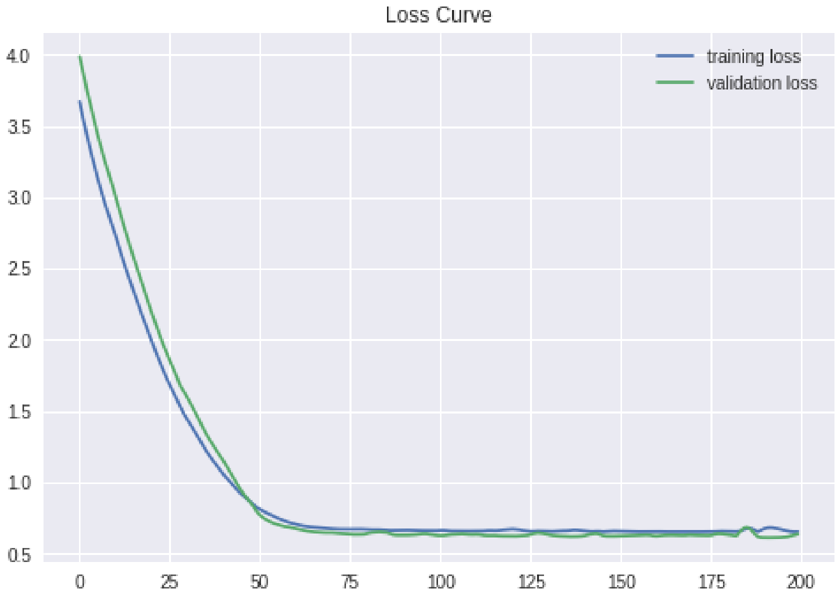
$$\mu_\mathcal{B} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

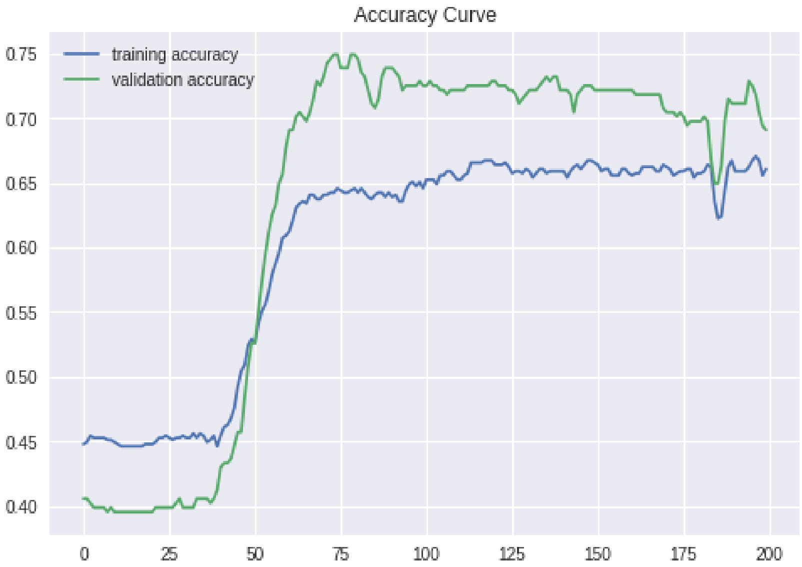
$$\sigma_\mathcal{B}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_\mathcal{B})^2 \qquad \text{// mini-batch variance}$$

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

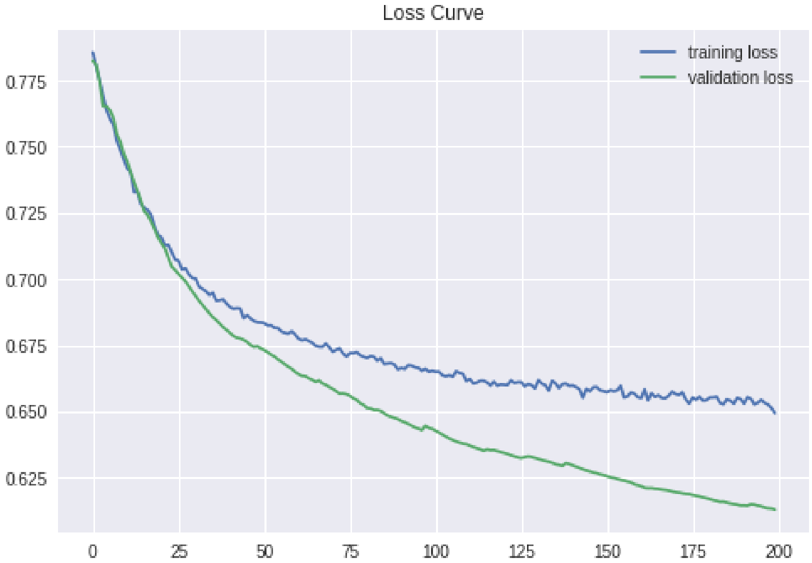
$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

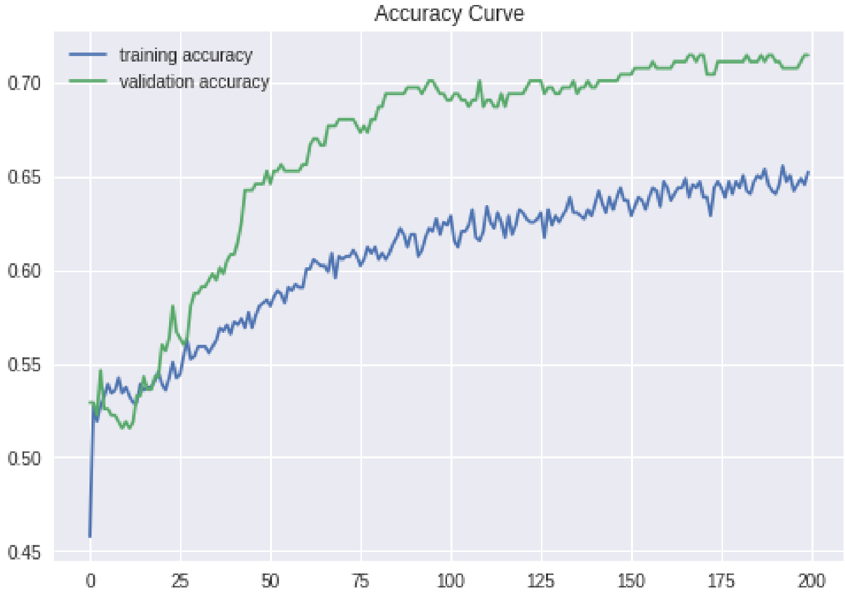
배치 정규화를 적용하지 않은 결과

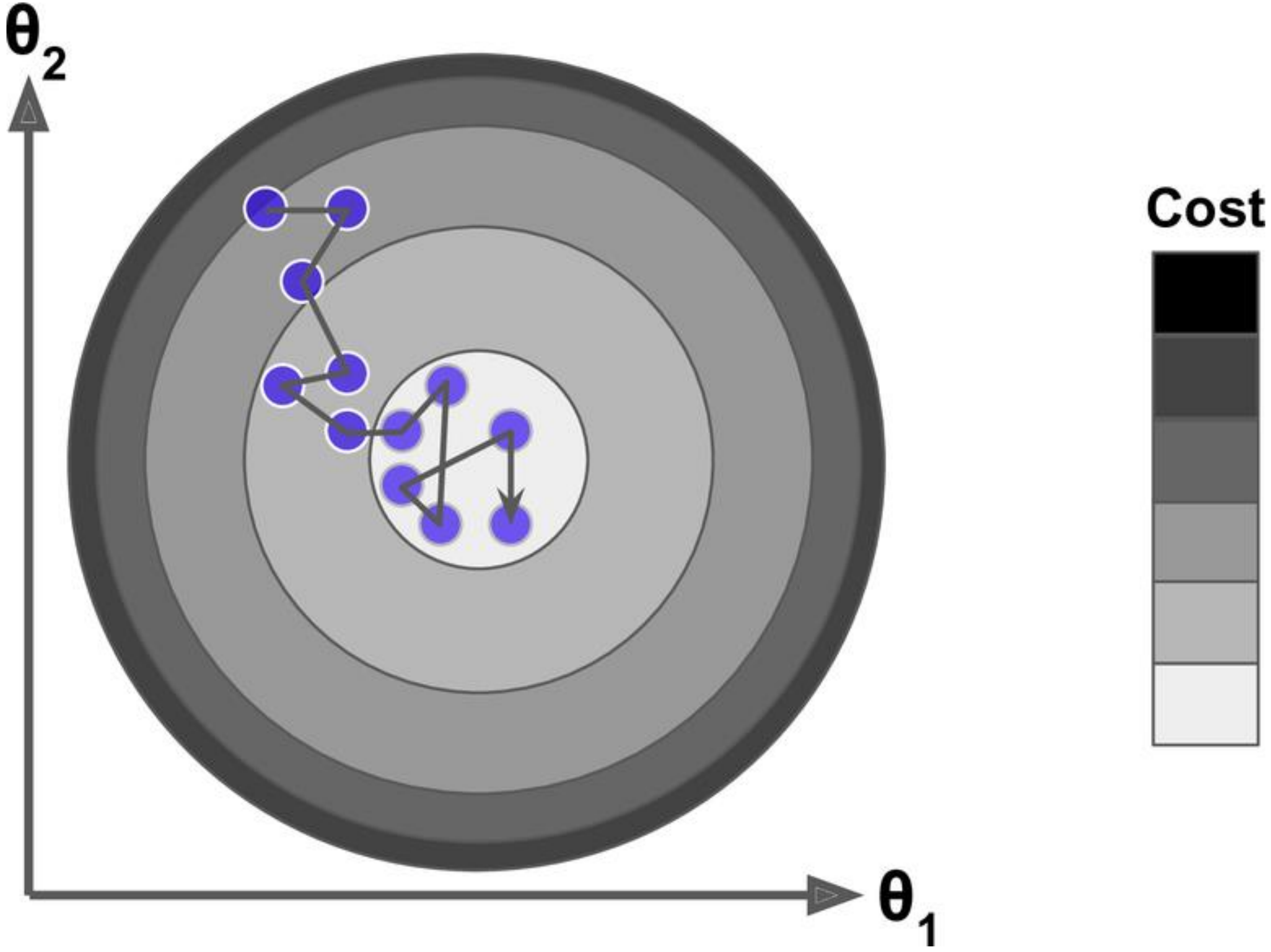

Loss Curve


Accuracy Curve

배치 정규화를 적용한 결과


Loss Curve


Accuracy Curve


θ2
θ1
Cost