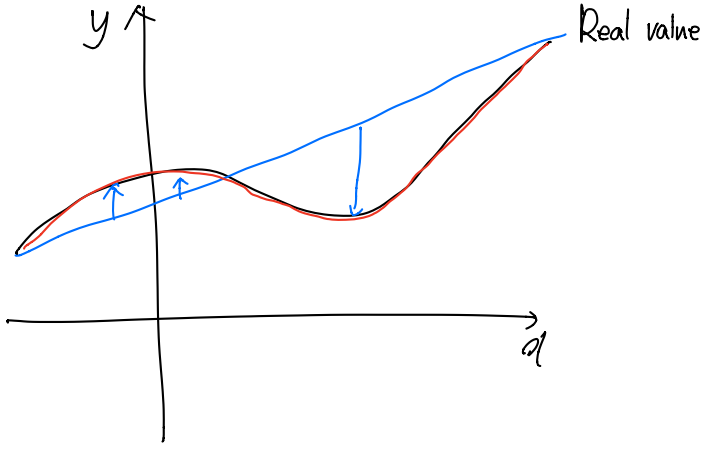
Problems of MLP
Overfitting

Model Capacity

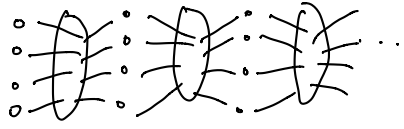


$y = wx + b$

$y = w_1x^3 + w_2x^2 + w_3x + w_4$ } Parameter의 개수가 차이남

Parameter의 개수가 많아지면 더 복잡한 모델을 예측가능?

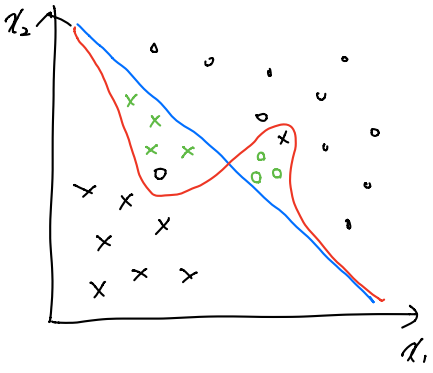
ex) MLP



Layer를 늘리면 Parameter 개수증가.

계속 늘리면 무조건 좋은가? → No

Overfitting

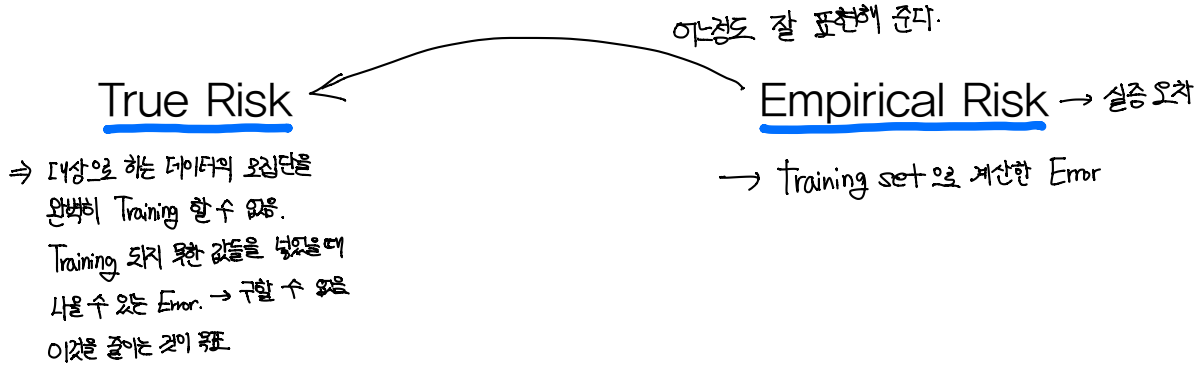


— : Simple한 예측

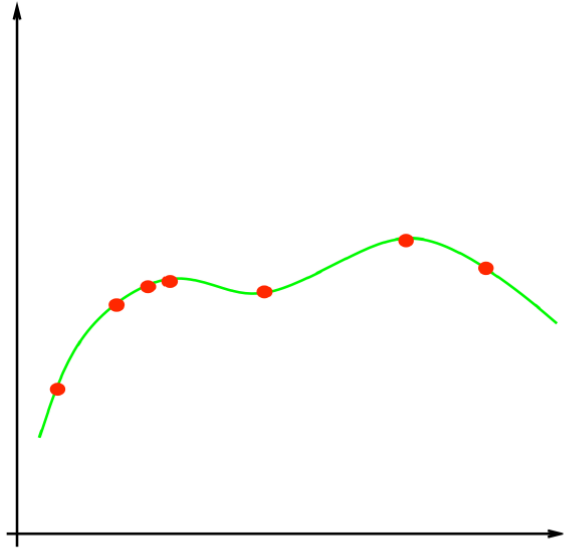
— : Layer를 늘려 세세한 부분까지 고려.

x.o : 실제 데이터들.
overfitting의 경우 이 친구들이 Error로 잡힘

Overfitting

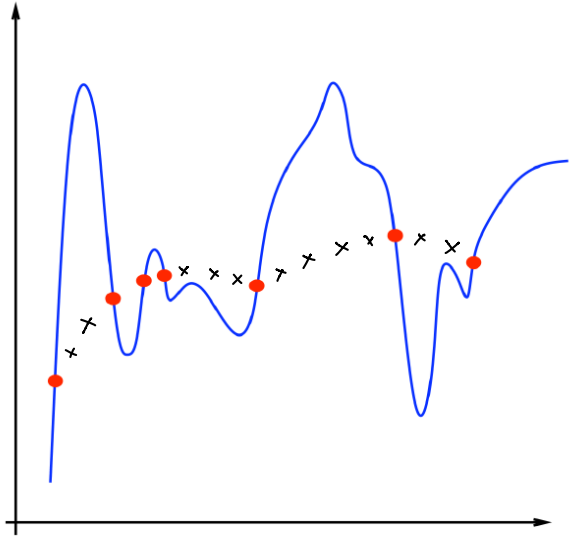


Overfitting



[T. Poggio]

Overfitting



Empirical risk : ⇊

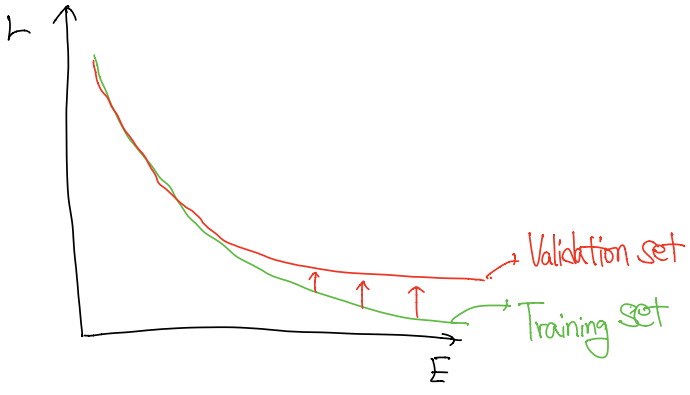
True risk : ⇈

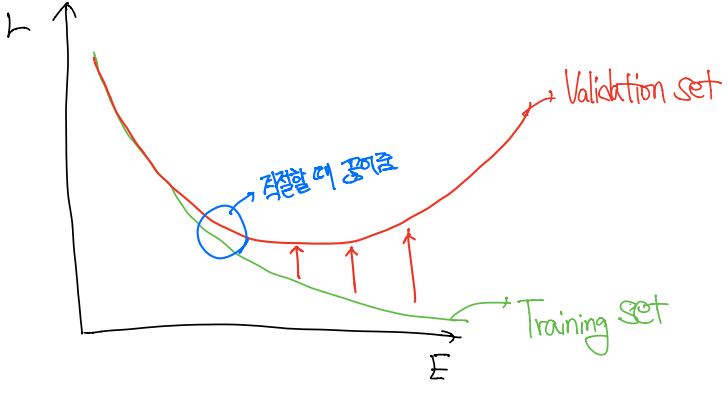
↳ Training set에서 Validation set을 빼서 중간 중간 overfitting이 너무 심하지 않도록 만들어줌

X만 같은 데이터 셋이 들어오면 Error가 튀겠지

[T. Poggio]

Overfitting





Summary

Overfitting occurs when model memorizes the train dataset thus cannot be applied to general dataset such as test set.

As model capacity increases, model become available to represent complicated systems but also be able to memorize the specific dataset.

We can know whether overfitting occurs or not by comparing training loss (empirical risk) and validation loss (approximated true risk)

Regularizations

General data set에도 적합 하도록 만드는 것

L2 Regularization and Dropout

L2 Regularization

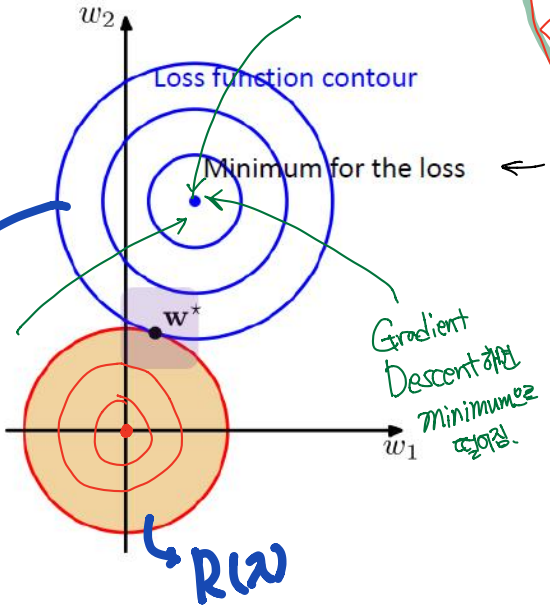
Loss = MSE + R(λ)

↳ λ‖w‖²

↳ constant

w (가중치)가 발산하지 않도록 항을 추가하여 컨트롤 해줌.

위에서 본 모습

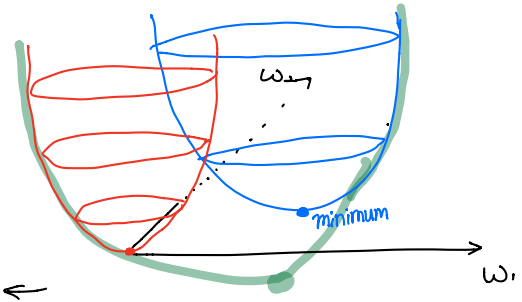


MSE

Gradient Descent해서 minimum으로 떨어짐.

↳ R(w)

옆에서 본 모습



어느정도 minimum을 균일하게 만들어줌

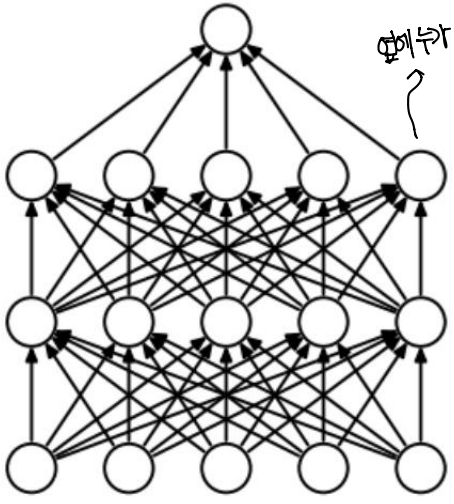
Dropout

↳ Training 할때만 사용.

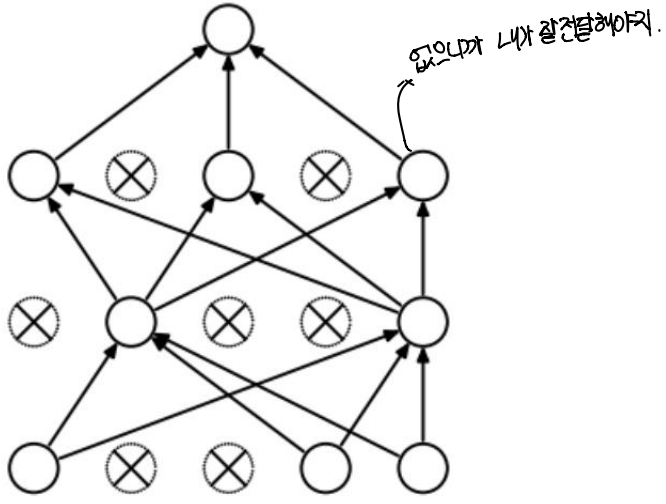
① 파라미터 줄어듦.

② 간단한 여러모델로 예측 후 합침.

⇒ Overfitting 줄어듦.

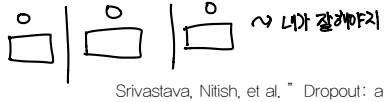


(a) Standard Neural Net



(b) After applying dropout.

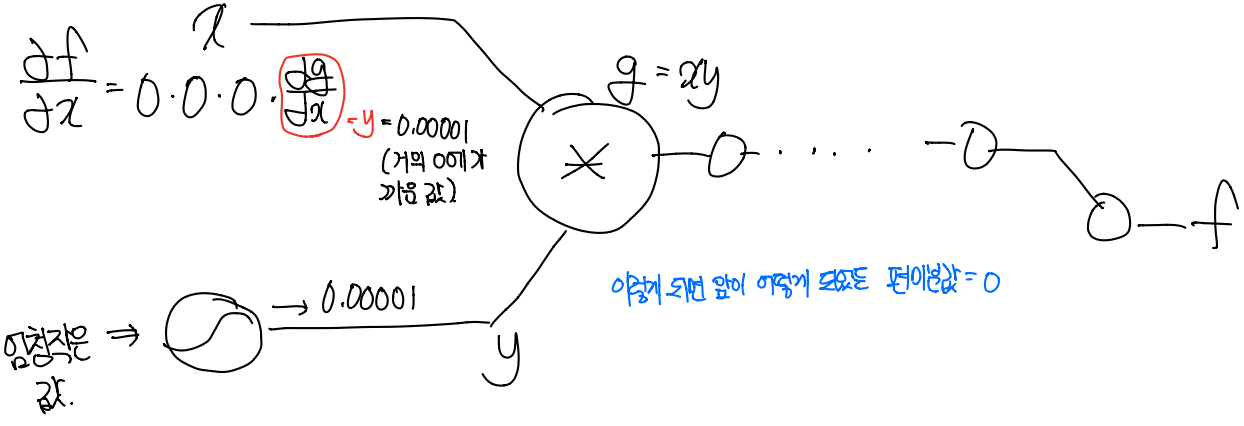
ex) 품질관리에서 → 누가 잘했겠지~



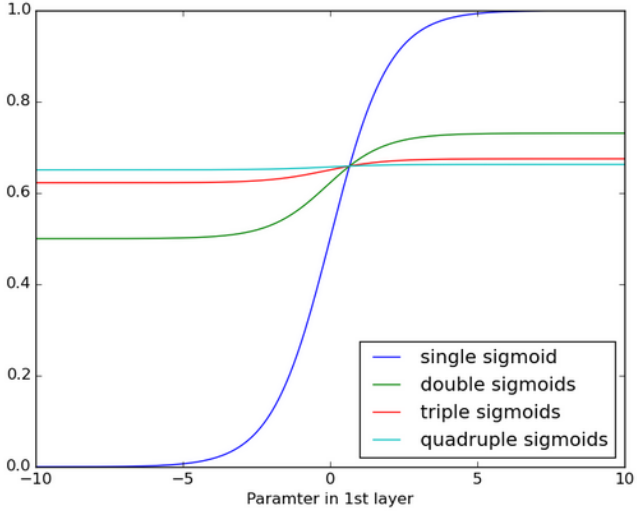
Srivastava, Nitish, et al. " Dropout: a simple way to prevent neural networks from overfitting", JMLR 2014

Problems of MLP
Gradient Vanishing

Gradient Vanishing



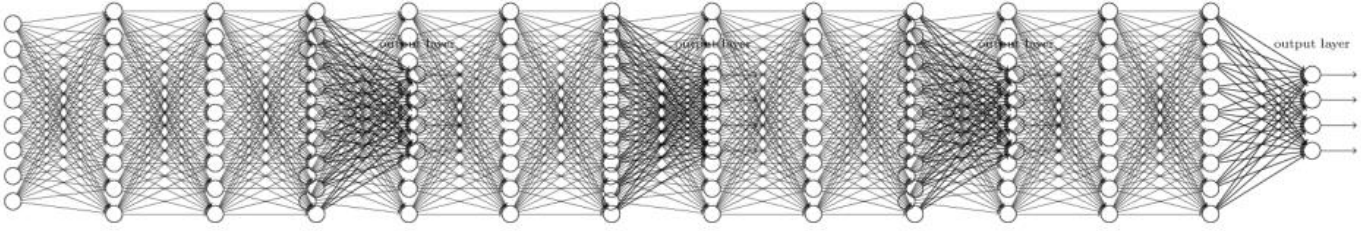
Gradient Vanishing



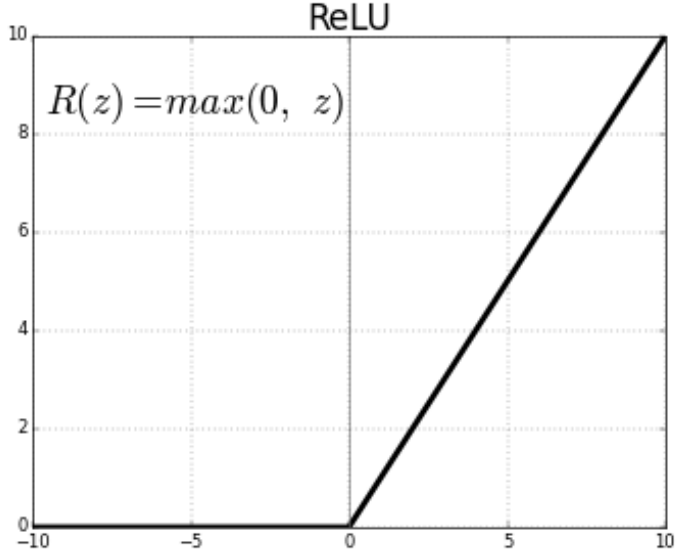
sigmoid 를 곱해 질수록
"0"에 가까운 값이 나옴

https://skymind.ai/wiki/lstm

Gradient Vanishing



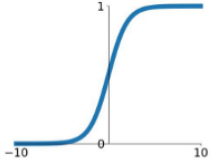
Rectified Linear Unit (ReLU) Activation



Activation Functions

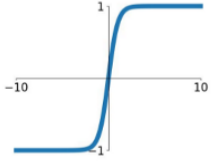
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



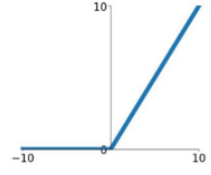
tanh

$$\tanh(x)$$



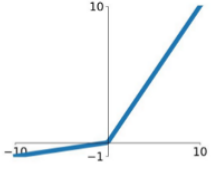
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

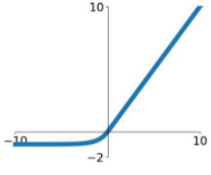


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$

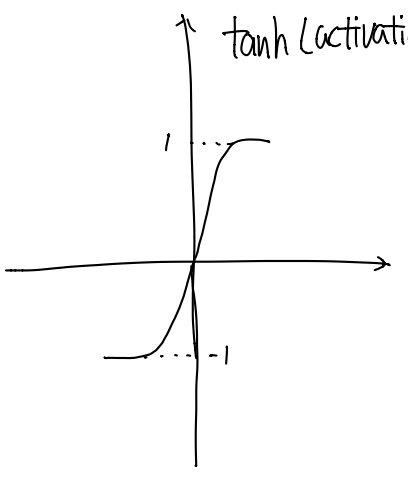


Other Tenchniques
Xavier Initialization and Batch Normalization

Xavier Initialization

모든 W가 0으로 초기화 되었다면?

⇒ 각 Layer를 거쳐도 값이 0만 나와서 Loss 측정 불가능.



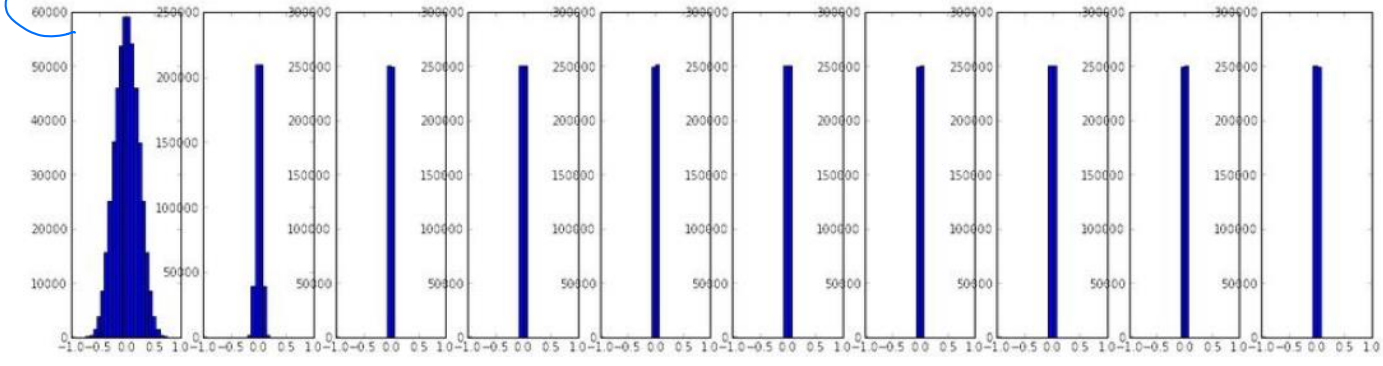
① 작은 값 : W가 0으로 수렴

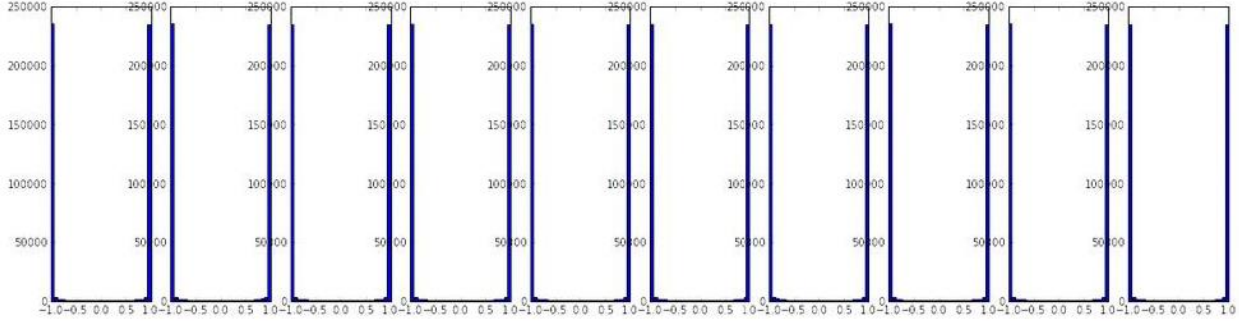
② 큰 값 : -1 , 1로 극악 수렴.

Xavier Initialization

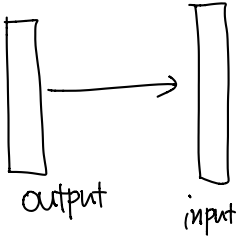
Standard gaussian N(0.1)

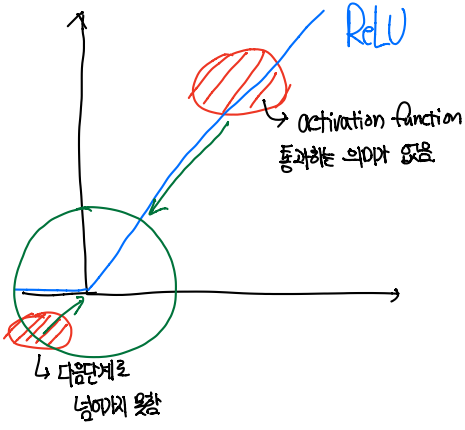
tanh를 거칠 수록 작은 값이 되어 W가 사라짐.(0으로 수렴함)





Batch Normalization





Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

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

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

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

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

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

Batch Normalization: Accelerating Deep Network Training b y Reducing Internal Covariate Shift