UNIVERSITY OF SCIENCE & TECHNOLOGY

Korea Institute of Science and Technology Information

2021-1 딥러닝기술 및 응용 - Paper Review

Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift

Sergey Ioffe, Christian Szegedy
(Google Inc.)

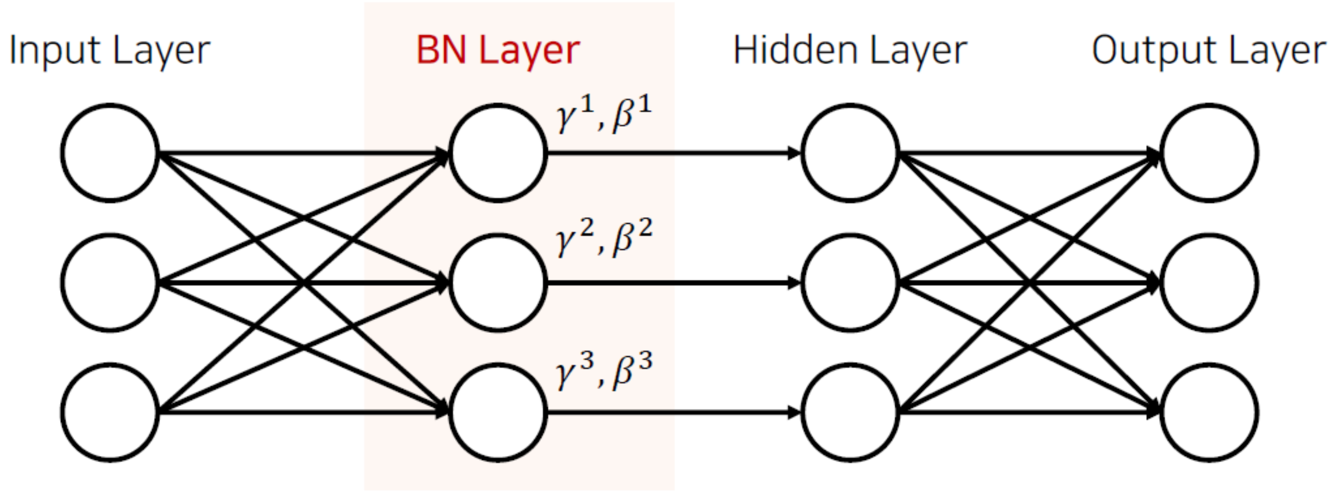
KISTI-UST
Gunho Lee

Some slides are borrowed from https://github.com/ndb796

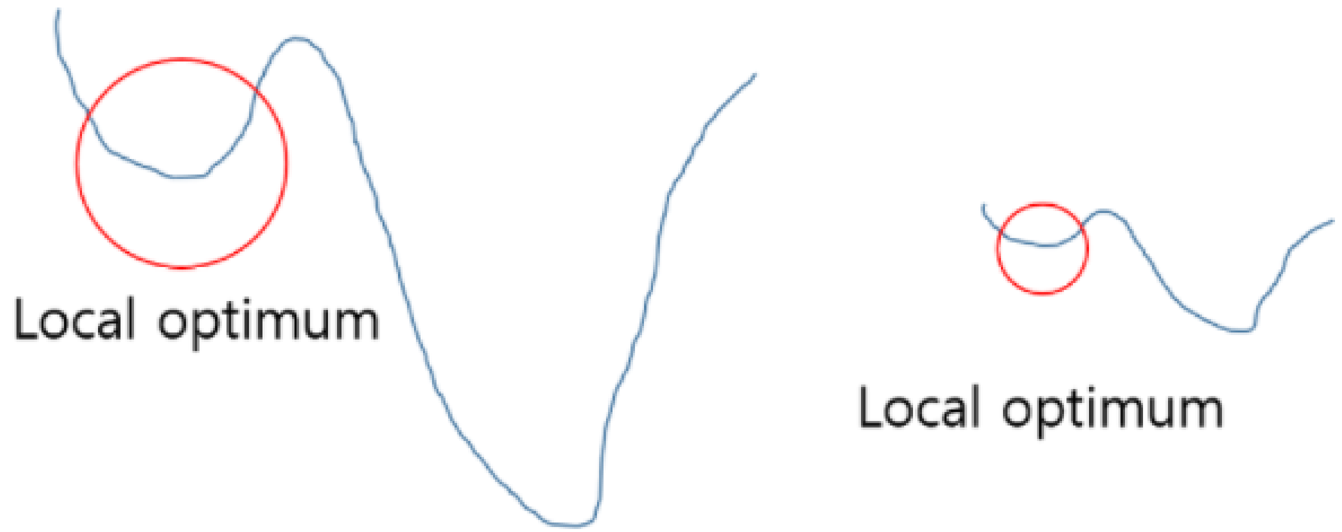
2021. 04. 02. FRI.

배치 정규화(BATCH NORMALIZATION)

- 배치 정규화의 잘 알려진 **장점**은 다음과 같습니다.
  ① **학습 속도(training speed)**를 빠르게 할 수 있습니다.
  ② **가중치 초기화(weight initialization)에 대한 민감도를 감소**시킵니다.
  ③ 모델의 **일반화(regularization)** 효과가 있습니다.



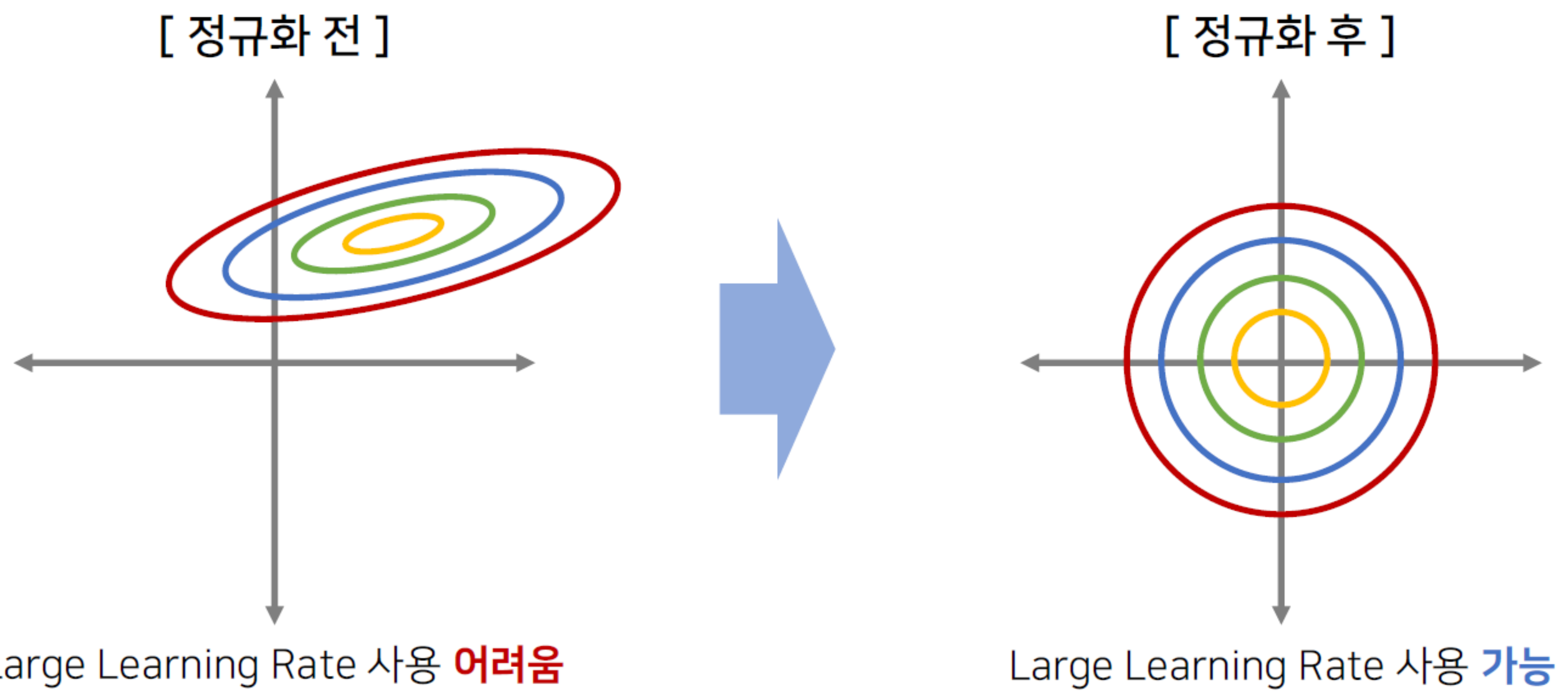
입력 정규화(NORMALIZATION)



(좌) Normalization 적용 전 / (우) Normalization 적용 후

입력 정규화(NORMALIZATION)

- 입력 데이터를 정규화하여 학습 속도(training speed)를 개선할 수 있습니다.



입력 표준화 (STANDARDIZATION)

- 입력 데이터를 N(0, 1) 분포를 따르도록 **표준화**하는 예제는 다음과 같습니다.

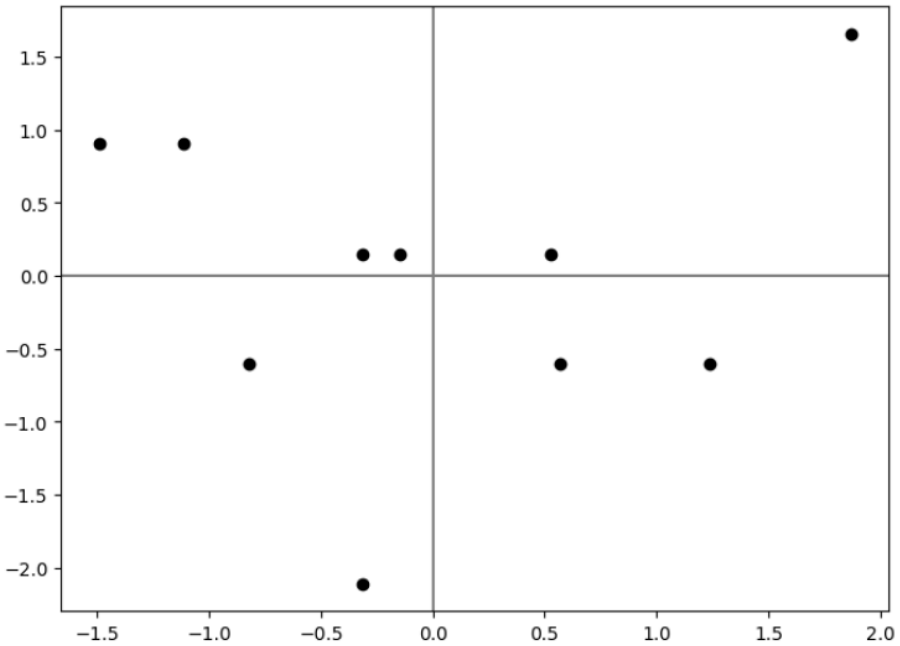
$$\hat{x} = \frac{x - E[x]}{\sqrt{Var[x]}}$$

```
x1 = np.asarray([33, 72, 40, 104, 52, 56, 89, 24, 52, 73])
x2 = np.asarray([9, 8, 7, 10, 5, 8, 7, 9, 8, 7])

normalized_x1 = (x1 - np.mean(x1)) / np.std(x1)
normalized_x2 = (x2 - np.mean(x2)) / np.std(x2)

plt.axvline(x=0, color='gray')
plt.axhline(y=0, color='gray')
plt.scatter(normalized_x1, normalized_x2, color='black')
plt.show()
```

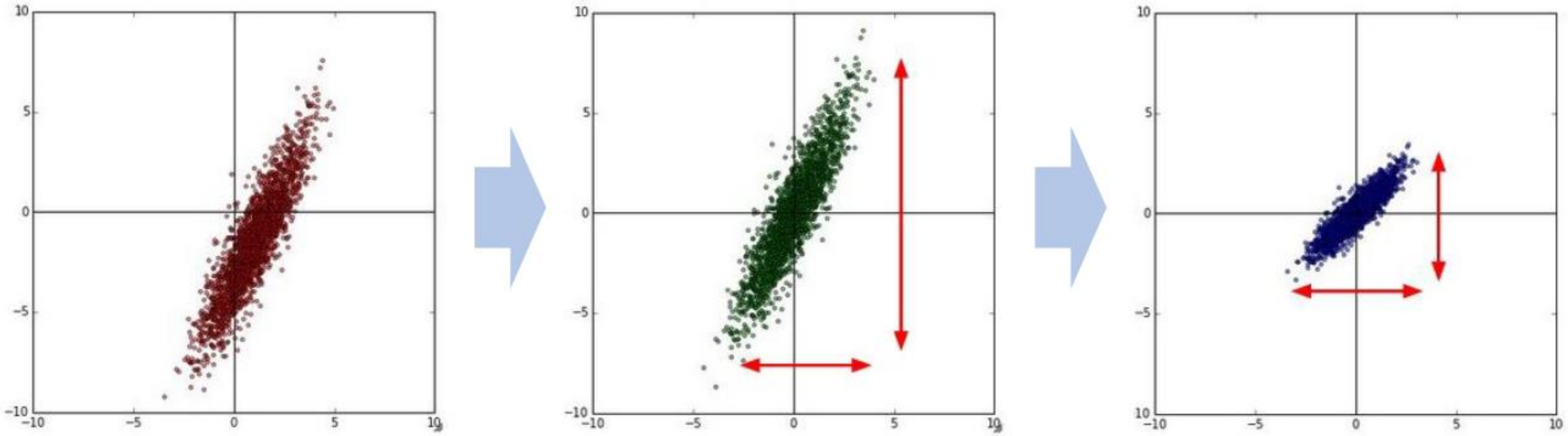
[평균(mean) = 0, 분산(variance) = 1]



입력 정규화(NORMALIZATION) VS 화이트닝(WHITENING)

- 입력 정규화를 이용해 **각 차원의 데이터가 동일한 범위 내의 값을 가지도록** 만들 수 있습니다.
  - 모든 특성(feature)에 대하여 각각 평균만큼 빼고 특정 범위의 값을 갖도록 조절할 수 있습니다.

[2개의 특성(feature)으로 구성된 데이터셋의 정규화 예시]



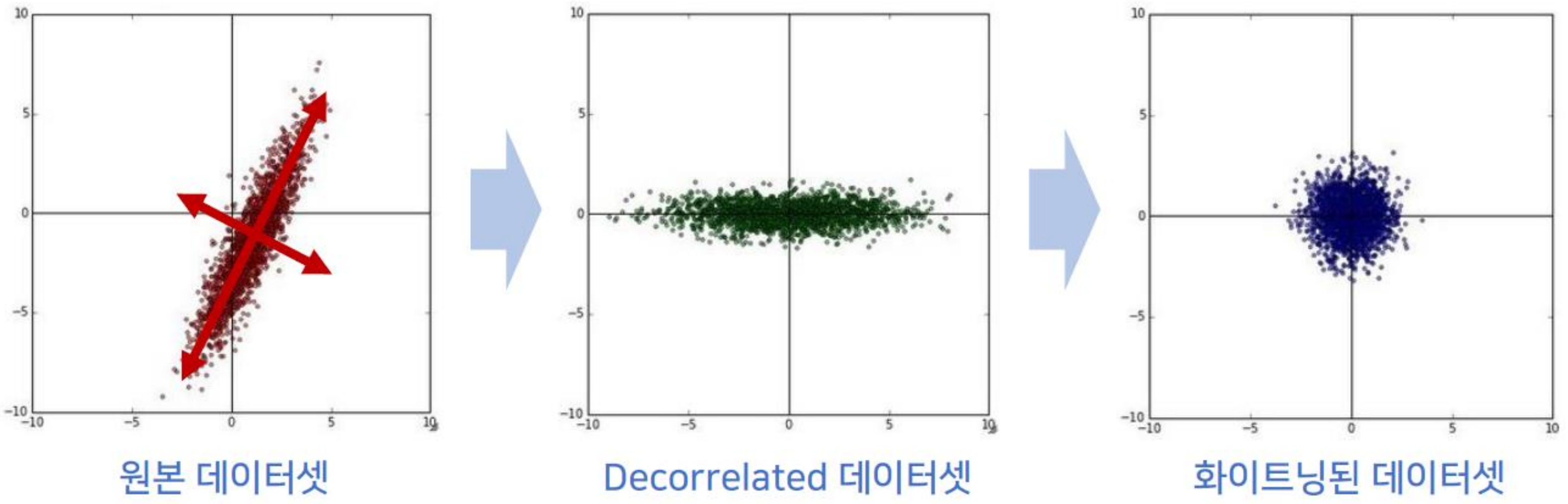
원본 데이터셋 | 평균이 0이 된 데이터셋 | 정규화된 데이터셋

http://cs231n.stanford.edu/

입력 정규화(NORMALIZATION) VS 화이트닝(WHITENING)

- **화이트닝**은 평균이 0이며 공분산이 단위행렬인 정규분포 형태의 데이터로 변환하는 기법입니다.
  - 일반적으로 PCA나 화이트닝보다는 정규화가 더 많이 사용됩니다.

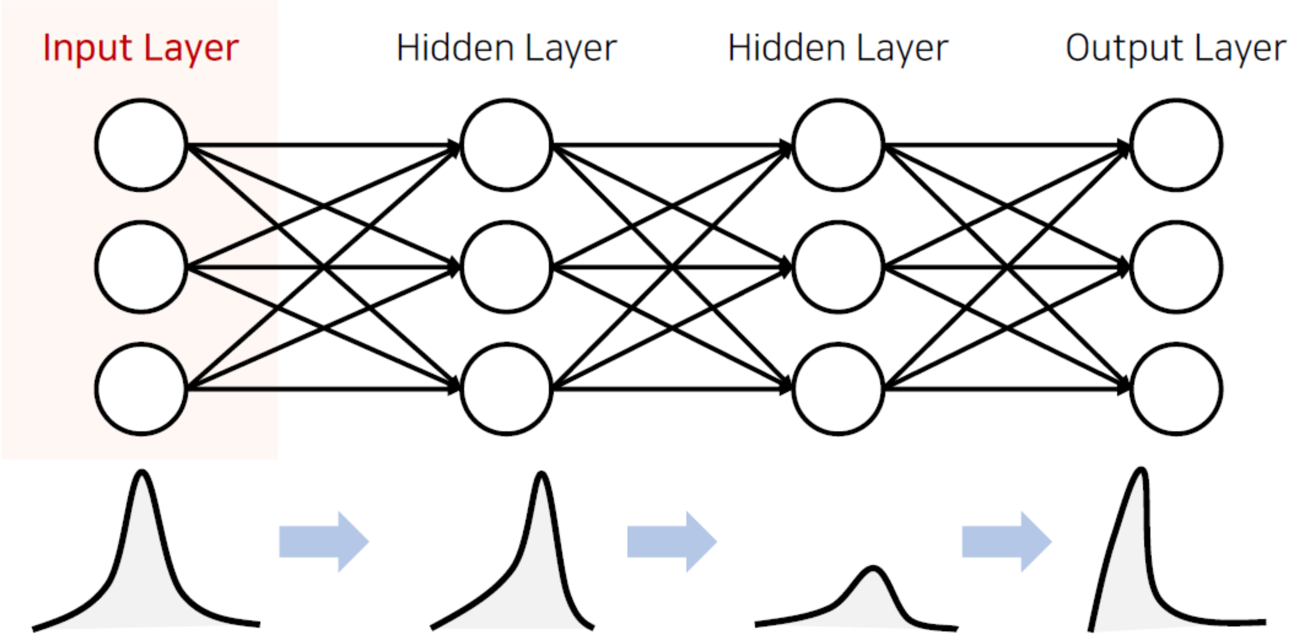
[2개의 특성(feature)으로 구성된 데이터셋의 화이트닝 예시]



http://cs231n.stanford.edu/

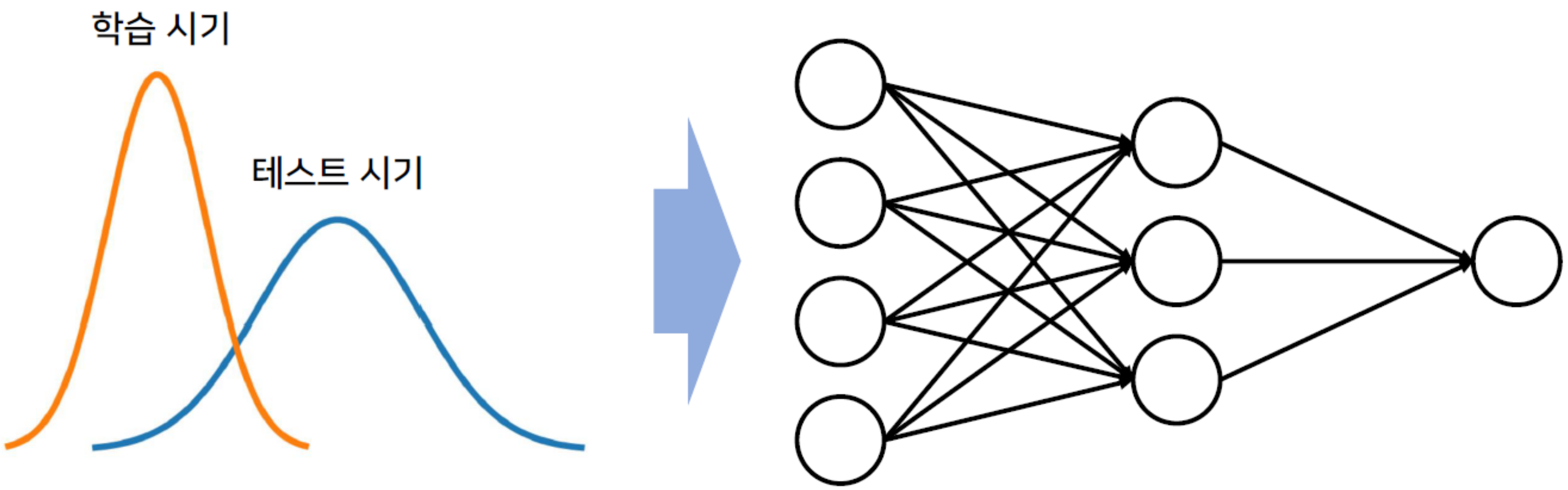
각 레이어에 대한 입력 분포

- 초기 입력 레이어의 데이터를 정규화하는 것은 상대적으로 간단합니다.
  - 하지만 히든 레이어의 입력은 어떻게 정규화할 수 있을까요?



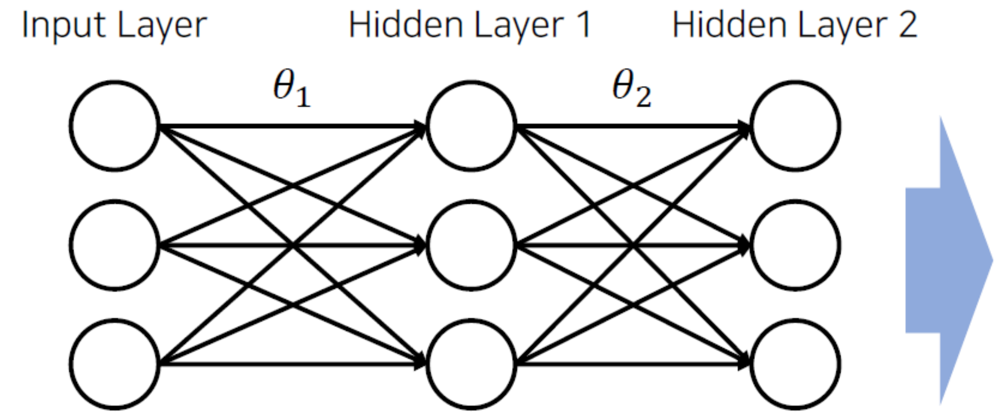
공변량 변화(COVARIATE SHIFT)

- **공변량 변화(covariate shift)**: 학습 시기와는 다르게 테스트 시기에 입력 데이터의 분포가 변경되는 현상
  - $P_{train}(y|x) = P_{test}(y|x)$ and $P_{train}(x) \neq P_{test}(x)$

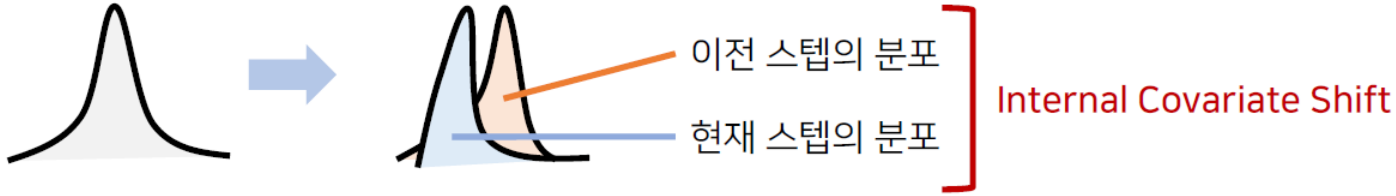


INTERNAL COVARIATE SHIFT (ICS) 가설

- 앞서 언급한 공변량 변화(covariate shift)가 **네트워크 내부에서 발생**하는 현상을 의미합니다.
  - 배치 정규화(batch normalization) 초창기 논문에서 해결하고자 했던 문제 상황입니다.



θ_1가 업데이트됨에 따라 뒤쪽에 있는 Hidden Layer들의 입력 분포가 변경됩니다. θ_2**의 입장**에서는 **매번 입력 분포가 바뀌는 것과 동일**하며 이는 레이어가 깊을수록 심화될 수 있습니다.



배치 정규화(BATCH NORMALIZATION)

- 입력(Input)
  - A mini-batch: $Batch = \{x_1, x_2, \ldots, x_m\}$
  - Parameters to be learned: γ, β
- 출력(Output)
  - $\{y_i = BN_{\gamma,\beta}(x_i)\}$

레이어의 입력 차원이 k일 때, 학습할 두 개의 파라미터 γ과 β 또한 k차원을 가집니다.

$$\mu_{Batch} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \quad \text{// 평균}$$

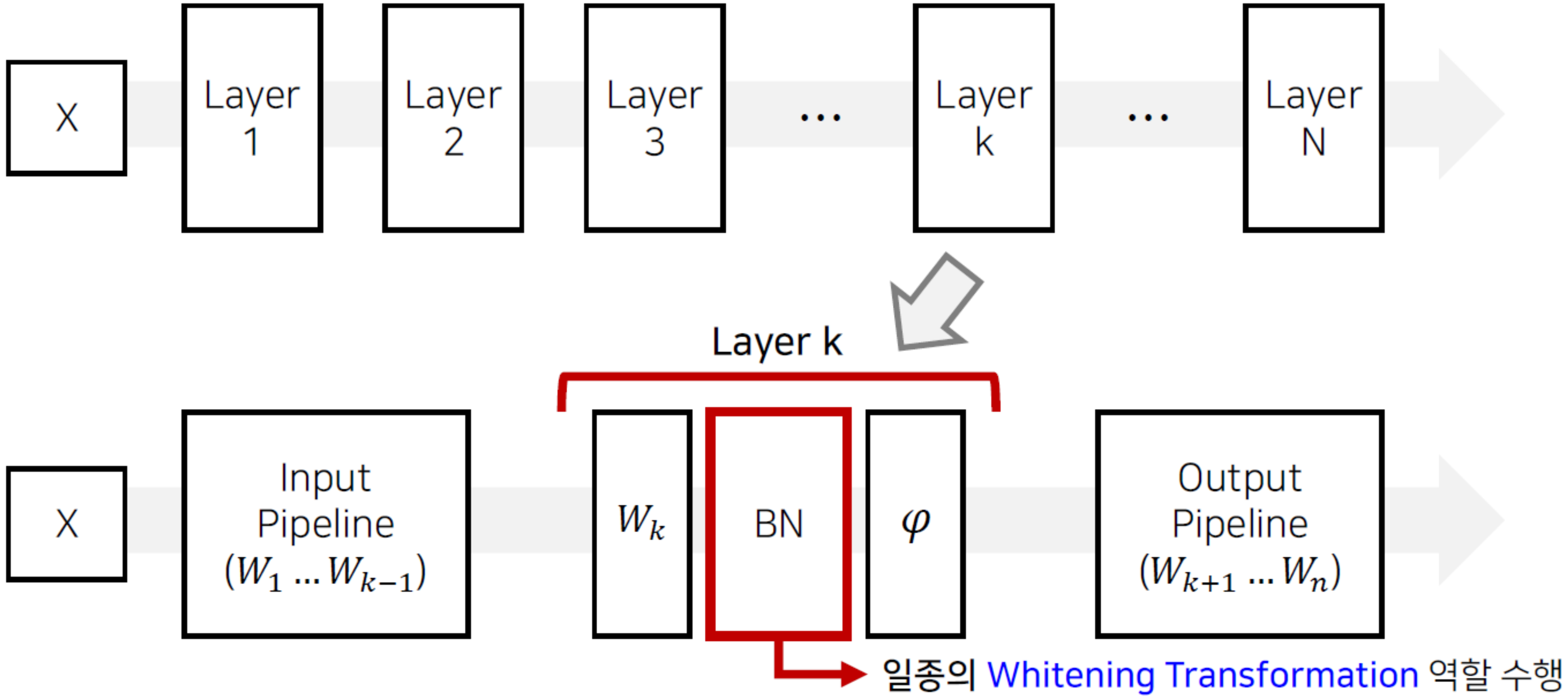
$$\sigma_{Batch}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{Batch})^2 \quad \text{// 분산}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_{Batch}}{\sqrt{\sigma_{Batch}^2 + \epsilon}} \quad \text{// 정규화}$$

$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv BN_{\gamma,\beta}(x_i)$$

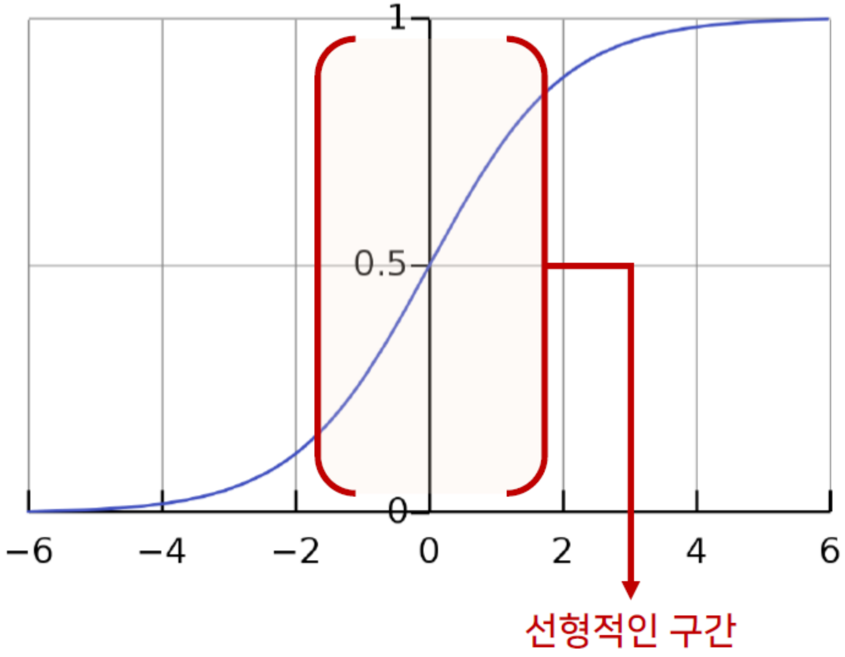
배치정규화의 적용

- 배치 정규화를 적용한 형태를 도식화하면 다음과 같습니다.



레이어 입력을 정규화 할 때 유의할 점

- 각 레이어를 단순히 N(0, 1)로 정규화하면 **비선형(non-linear) 활성화 함수의 영향력이 감소**할 수 있습니다.
  - Sigmoid 함수 예시

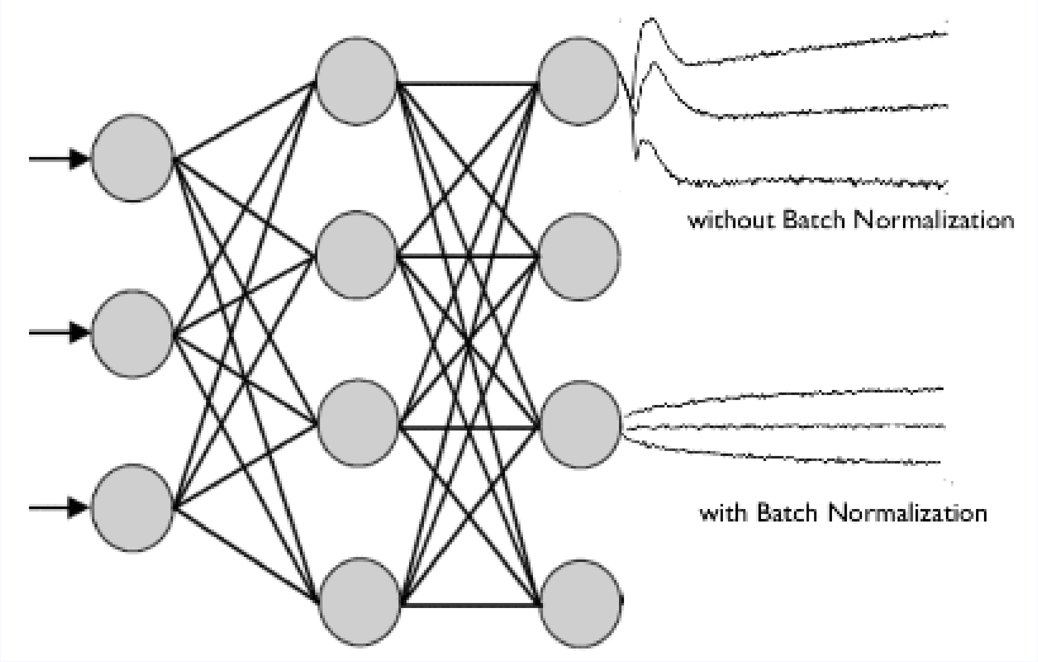


입력 데이터가 N(0, 1)로 정규화되므로 대부분의 입력에 대하여 매우 선형적으로 동작합니다.

정규화 이후에 사용하는 감마(γ)와 베타(β)는 non-linearity를 유지할 수 있도록 해줍니다.

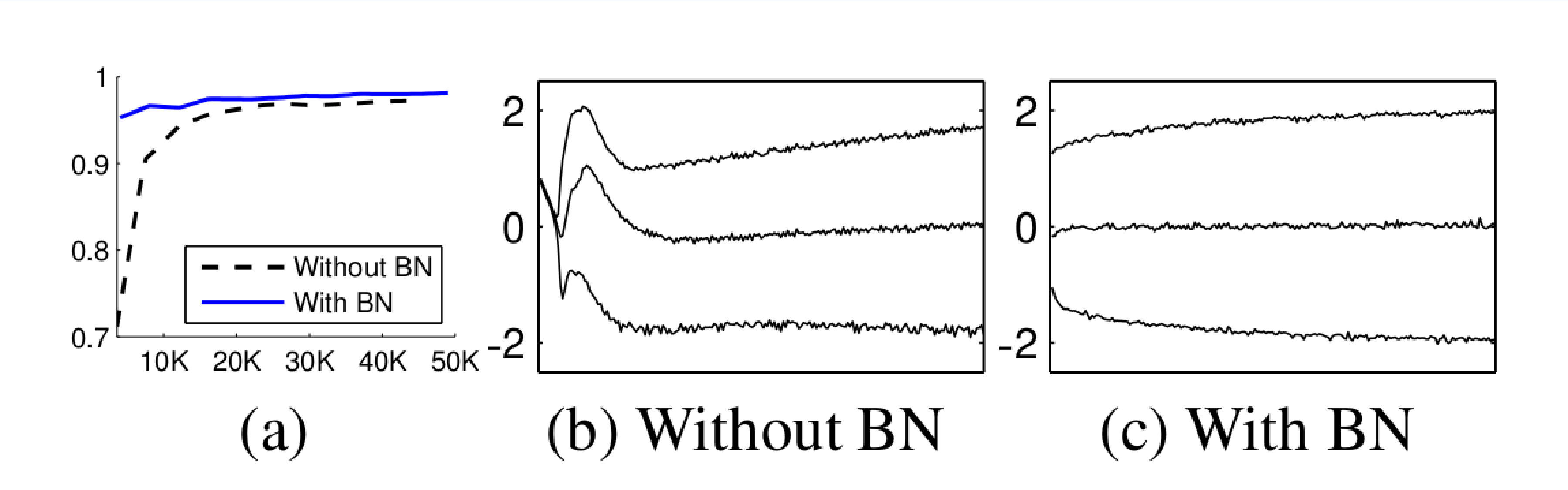
$$y_i \leftarrow \gamma \hat{x_i} + \beta \equiv BN_{\gamma,\beta}(x_i)$$

배치 정규화의 성능 향상



MNIST 에서의 배치 정규화 실험

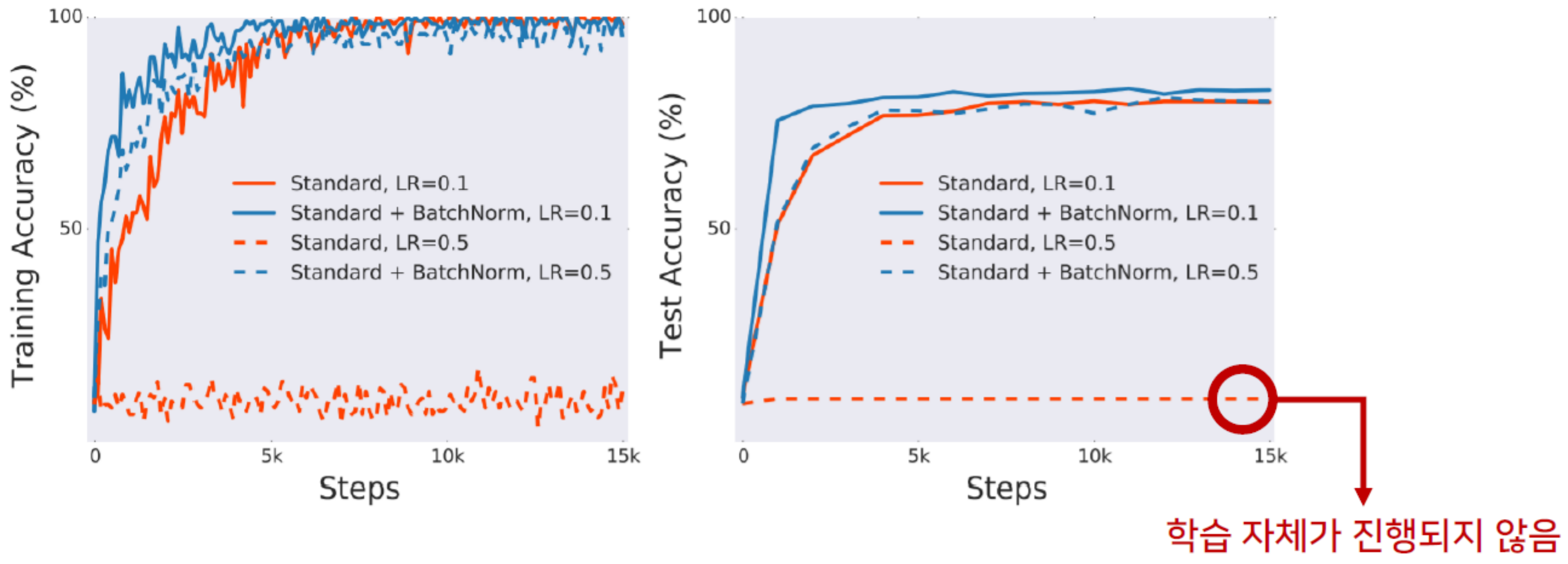
배치 정규화의 성능 향상



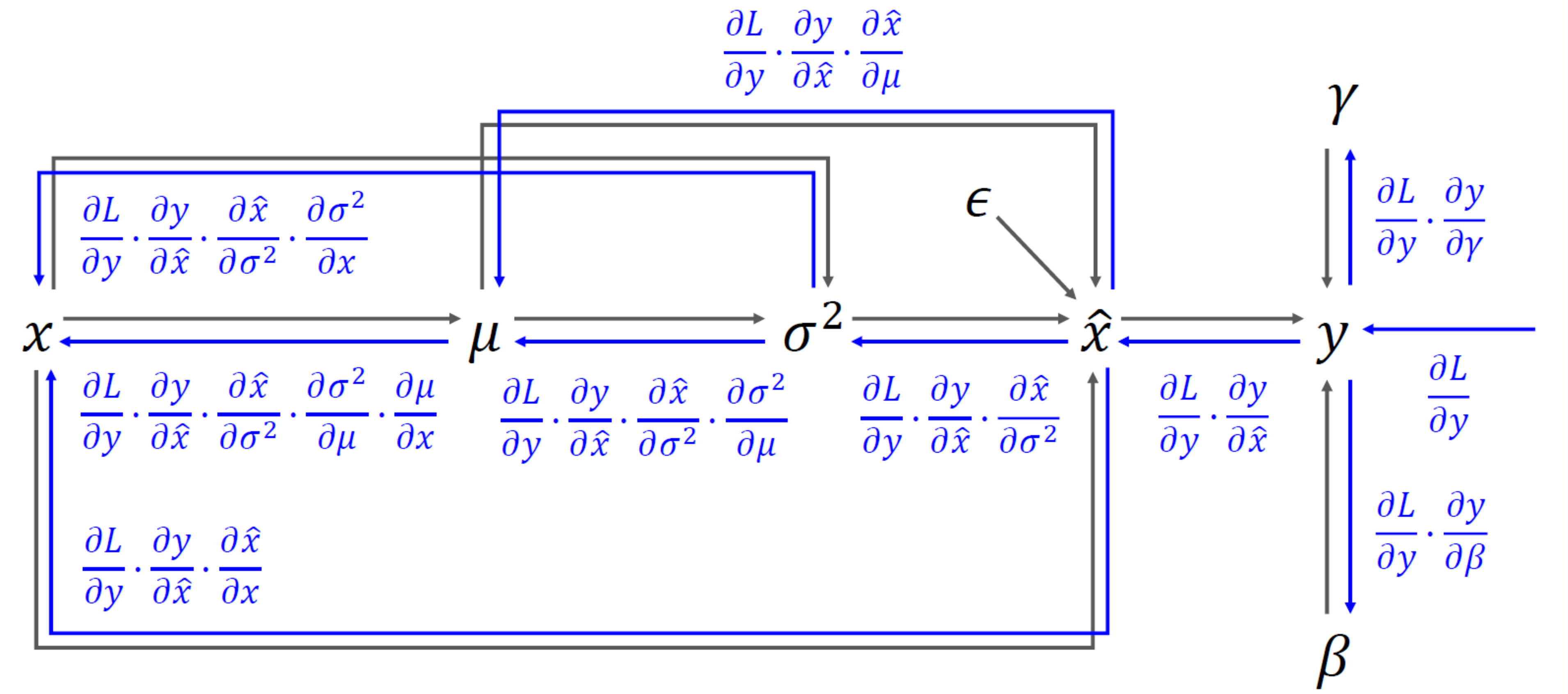
MNIST 에서의 배치 정규화 실험

배치 정규화의 성능 향상

- 배치 정규화를 이용함으로써 얻을 수 있는 성능 향상 효과는 반박의 여지가 없습니다.
  - 학습을 위한 **하이퍼 파라미터 설정**으로부터 더 자유로우며, **학습이 빠르게 수행**됩니다.



배치 정규화: 데이터 플로우 그래프



배치 정규화(BATCH NORMALIZATION)

- 입력(Input)
  - A mini-batch: $Batch = \{x_1, x_2, \dots, x_m\}$
  - Parameters to be learned: γ, β
- 출력(Output)
  - $\{y_i = BN_{\gamma,\beta}(x_i)\}$

레이어의 입력 차원이 k일 때, 학습할 두 개의 파라미터 γ과 β 또한 k차원을 가집니다.

$$\mu_{Batch} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// 평균}$$

$$\sigma_{Batch}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{Batch})^2 \qquad \text{// 분산}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_{Batch}}{\sqrt{\sigma_{Batch}^2 + \epsilon}} \qquad \text{// 정규화}$$

$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv BN_{\gamma,\beta}(x_i)$$

배치 정규화: 기울기(GRADIENT) 계산하기

$$\frac{\partial \ell}{\partial \widehat{x}_i} = \frac{\partial \ell}{\partial y_i} \cdot \gamma$$

$$\frac{\partial \ell}{\partial \sigma_{\mathcal{B}}^2} = \sum_{i=1}^{m} \frac{\partial \ell}{\partial \widehat{x}_i} \cdot (x_i - \mu_{\mathcal{B}}) \cdot \frac{-1}{2}(\sigma_{\mathcal{B}}^2 + \epsilon)^{-3/2}$$

$$\frac{\partial \ell}{\partial \mu_{\mathcal{B}}} = \left(\sum_{i=1}^{m} \frac{\partial \ell}{\partial \widehat{x}_i} \cdot \frac{-1}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \right) + \frac{\partial \ell}{\partial \sigma_{\mathcal{B}}^2} \cdot \frac{\sum_{i=1}^{m} -2(x_i - \mu_{\mathcal{B}})}{m}$$

$$\frac{\partial \ell}{\partial x_i} = \frac{\partial \ell}{\partial \widehat{x}_i} \cdot \frac{1}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} + \frac{\partial \ell}{\partial \sigma_{\mathcal{B}}^2} \cdot \frac{2(x_i - \mu_{\mathcal{B}})}{m} + \frac{\partial \ell}{\partial \mu_{\mathcal{B}}} \cdot \frac{1}{m}$$

$$\frac{\partial \ell}{\partial \gamma} = \sum_{i=1}^{m} \frac{\partial \ell}{\partial y_i} \cdot \widehat{x}_i$$

$$\frac{\partial \ell}{\partial \beta} = \sum_{i=1}^{m} \frac{\partial \ell}{\partial y_i}$$

학습시킬 파라미터

배치 정규화: 학습(TRAINING) 및 추론(INFERENCE)

Input: Network N with trainable parameters θ; subset of activations $\{x^{(k)}\}_{k=1}^{K}$

Output: Batch-normalized network for inference, N_{BN}^{infer}

$N_{BN}^{train} \leftarrow N$ // Training BN network

for $k = 1 \ldots K$ **do**

  Add transformation $y^{(k)} = BN_{\gamma^{(k)},\beta^{(k)}}(x^{(k)})$ to N_{BN}^{train}

  Modify each layer in N_{BN}^{train} with input $x^{(k)}$ to take $y^{(k)}$ instead

end for

Train N_{BN}^{train} to optimize the parameters $\theta \cup \{r^{(k)}, \beta^{(k)}\}_{k=1}^{K}$ 학습

학습(Training) 단계

$N_{BN}^{infer} = N_{BN}^{train}$ // Inference BN network with frozen parameters

for $k = 1 \ldots K$ **do**

  Process multiple training mini-batches $Batch$, each of size m, and average over them:

$$E[x] \leftarrow E_{Batch}[\mu_{Batch}]$$

$$Var[x] \leftarrow \frac{m}{m-1} E_{Batch}[\sigma_{Batch}^2]$$

  In N_{BN}^{infer}, replace the transform $y = BN_{\gamma,\beta}(x)$ with $y = \frac{\gamma}{\sqrt{Var[x]+\epsilon}} \cdot x + \left(\beta - \frac{\gamma E[x]}{\sqrt{Var[x]+\epsilon}}\right)$

end for

추론(Inference)을 위한
준비 단계

배치 정규화의 성능 향상

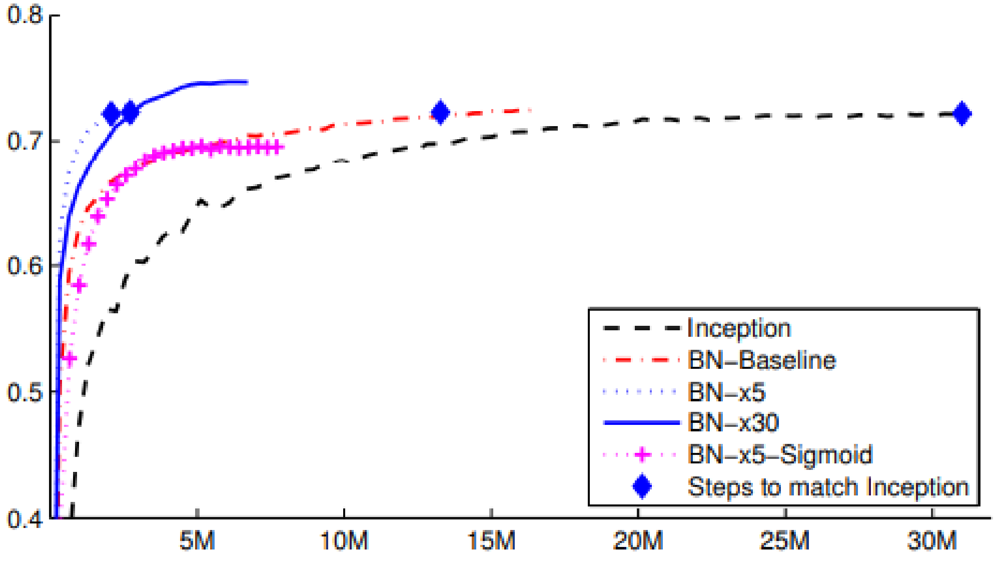


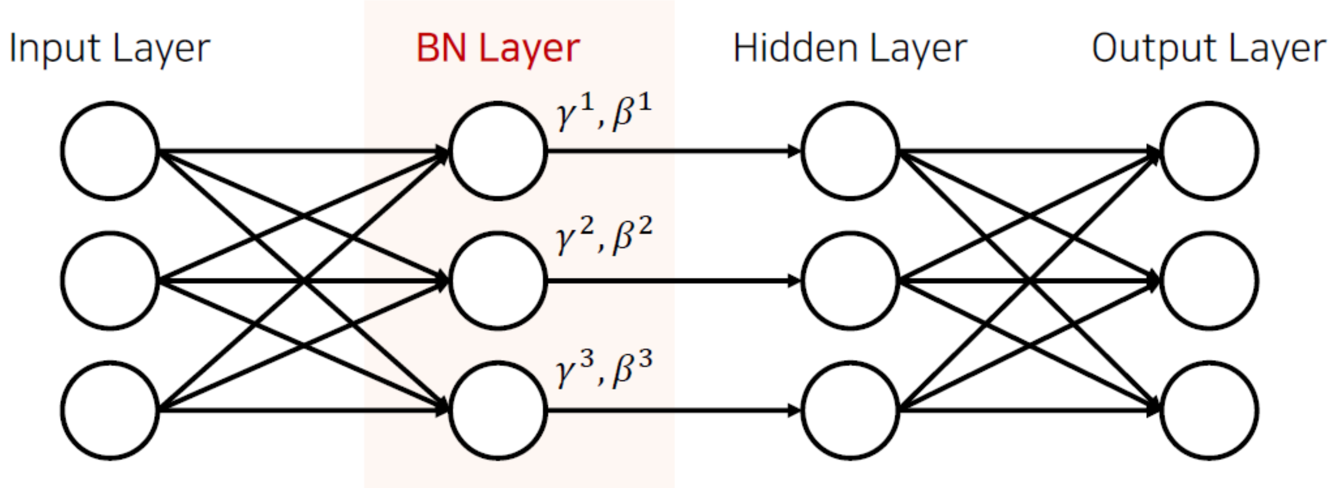
Figure 2. Single crop validation accuracy of Inception and its batch-normalized variants, vs. the number of training steps.

Model	Steps to 72.2%	Max accuracy
Inception	$31.0 \cdot 10^6$	72.2%
BN-Baseline	$13.3 \cdot 10^6$	72.7%
BN-x5	$2.1 \cdot 10^6$	73.0%
BN-x30	$2.7 \cdot 10^6$	74.8%
BN-x5-Sigmoid		69.8%

Figure 3. For Inception and the batch-normalized variants, the number of training steps required to reach the maximum accuracy of Inception (72.2%), and the maximum accuracy achieved by the network.

배치 정규화(BATCH NORMALIZATION)

- 배치 정규화의 잘 알려진 **장점**은 다음과 같습니다.
  - ① **학습 속도(training speed)**를 빠르게 할 수 있습니다.
  - ② **가중치 초기화(weight initialization)에 대한 민감도를 감소**시킵니다.
  - ③ 모델의 **일반화(regularization)** 효과가 있습니다.



REFERENCE

- 논문 리뷰 블로그 1. https://github.com/ndb796/Deep-Learning-Paper-Review-and-Practice
- 논문 리뷰 블로그 2. https://eehoeskrap.tistory.com/430

Thank You