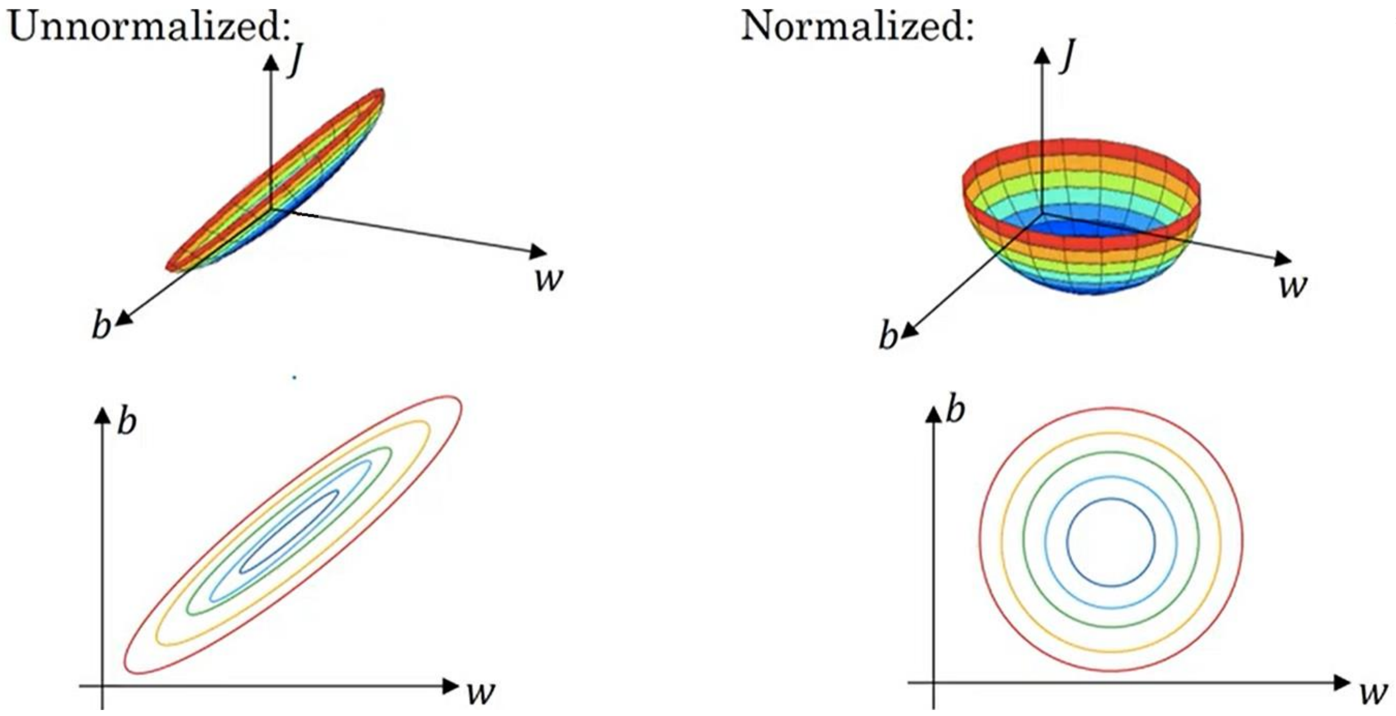
Batch Normalization: Accelerating Deep Neural Network Training by Reducing Internal Covariate Shift

12th Seminar in 2023, Paper Review

Samsung Software Developer Community
Korea Vision & Robotics
Eunjung Choi
2023.06.03

Background

- 입력 데이터 정규화 → 학습 속도(training speed) 개선
  - 비용 함수의 형태가 상대적으로 대칭적이기 때문에 큰 learning rate 사용 가능



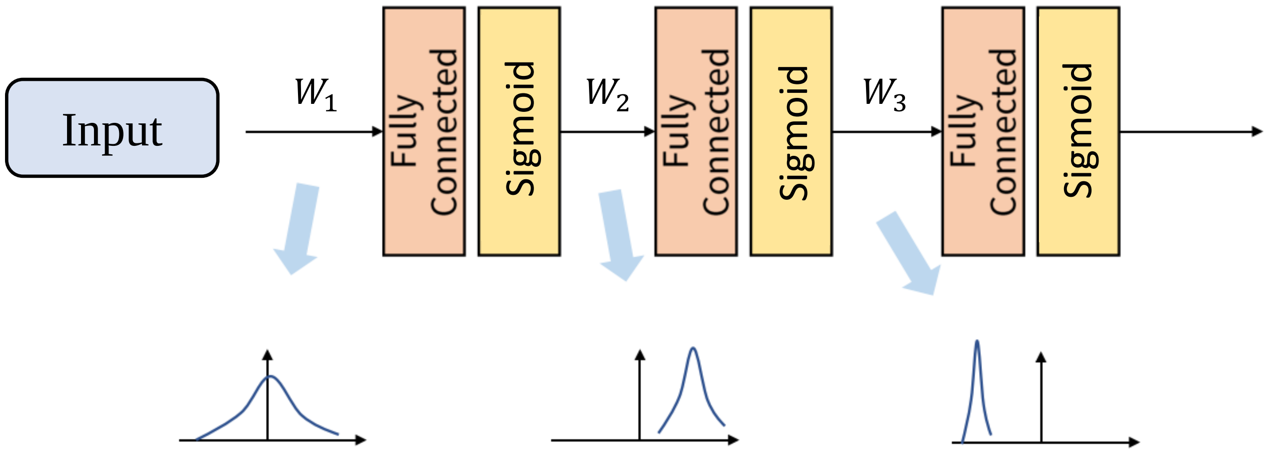
* Cost function

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m} L(\hat{y}_i, y_i)$$

Idea of Batch Normalization

https://gaussian37.github.io/dl-concept-batchnorm/

- Issue: **Internal Covariate Shift (내부적 공변량 변화)**
  - weight update에 따라 hidden layer의 입력 분포가 변하는 현상
  - 네트워크가 깊어질 수록 심화될 수 있음

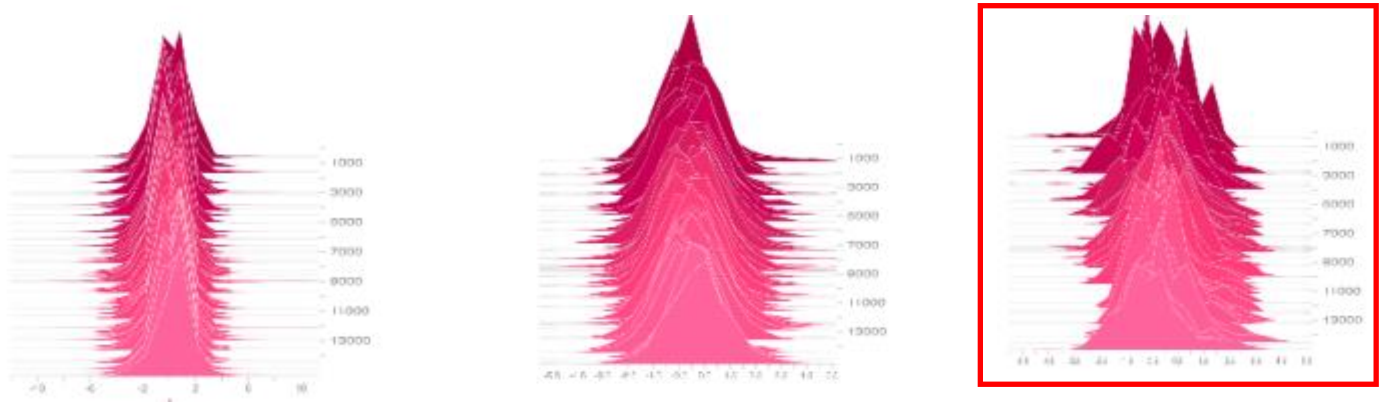


Idea of Batch Normalization

https://gaussian37.github.io/dl-concept-batchnorm/

- Issue: **Internal Covariate Shift (내부적 공변량 변화)**
  - weight update에 따라 hidden layer의 입력 분포가 변하는 현상
  - weight는 계속해서 새로운 분포에 맞춰 학습해야 함

Internal Covariate Shift



training steps

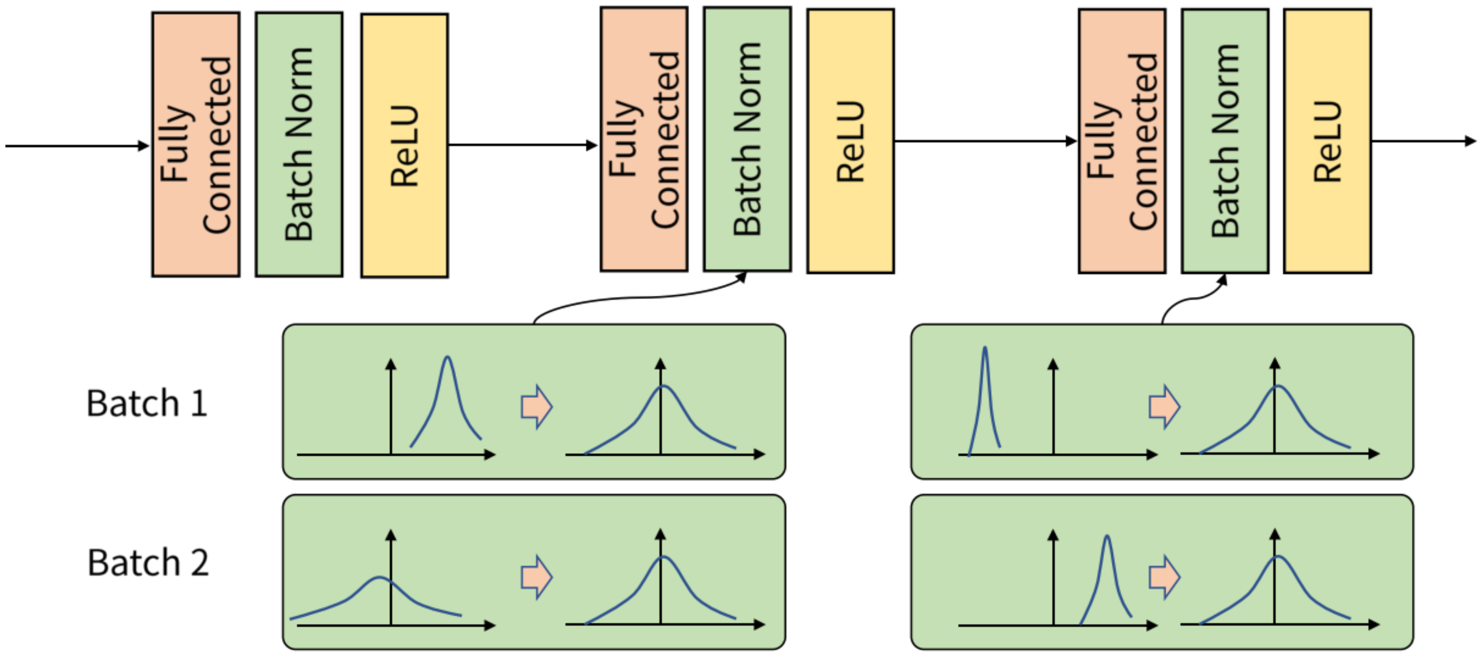
Layer #2 Layer #9 Layer #13

학습 스텝에 따른 각 레이어의 입력 분포 변화 예시

Idea of Batch Normalization

https://gaussian37.github.io/dl-concept-batchnorm/

- Batch Normalization
- **mini-batch 단위로 각 layer input을 정규화**



Batch Normalization (Overall Training)

- **Input**

  A mini-batch: $B = \{x_1, \dots x_m\}$

  Parameters to be learned: γ, β

- **Output**

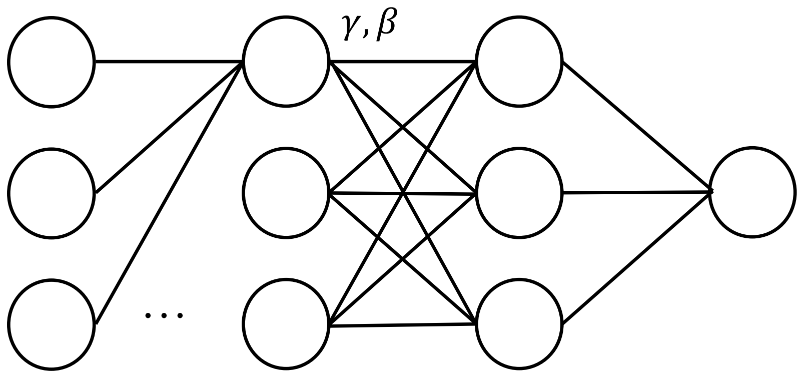
  $\{y_i = \mathbf{BN}_{\gamma,\beta}(x_i)\}$

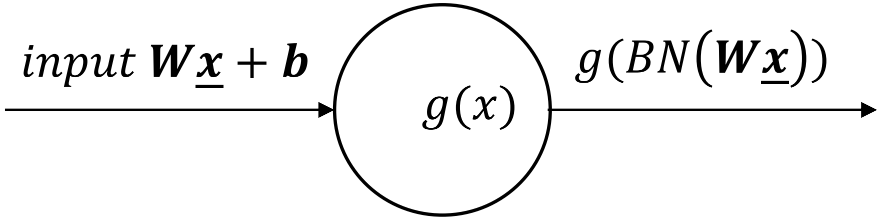
$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

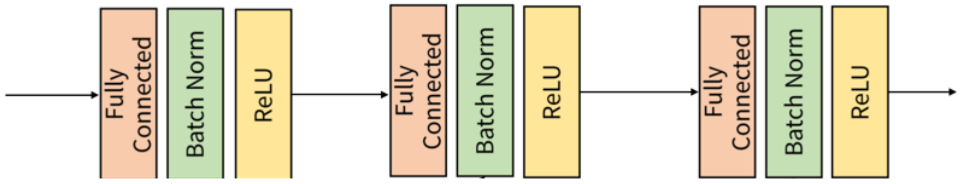
$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

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

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







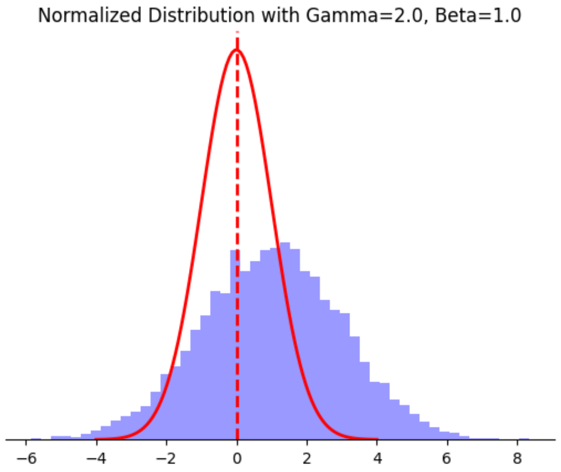
Batch Normalization (Training) : Parameters

- Normalization (Standardization)

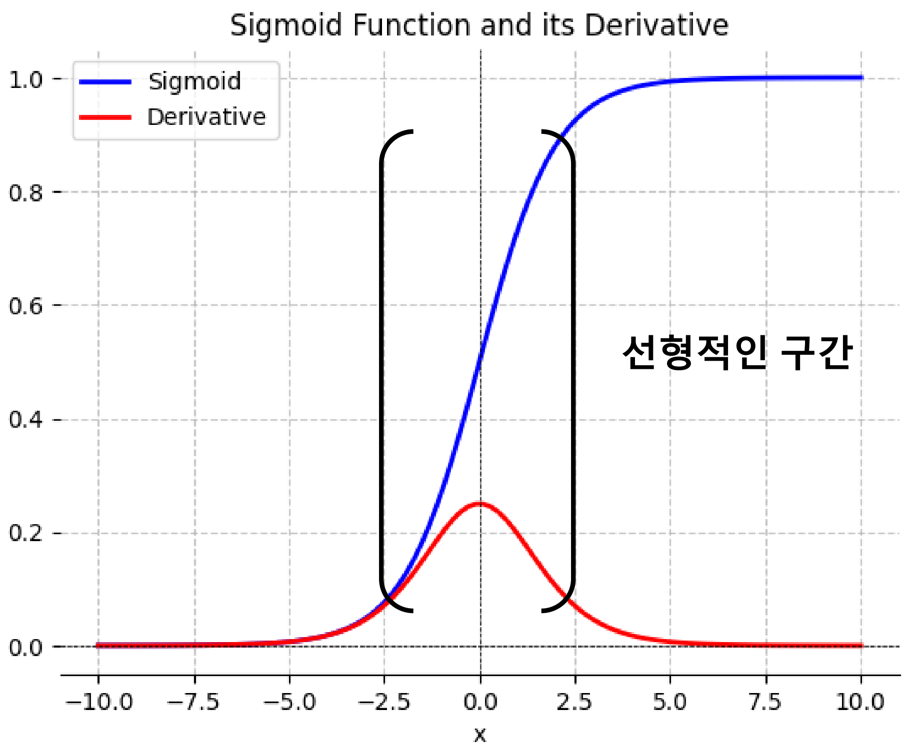
$$\widehat{x_i} = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \sim (0, 1)$$

- Scale and Shift

$$y_i = \gamma \widehat{x_i} + \beta \equiv BN_{\gamma,\beta}(x_i)$$



- 정규화만 진행할 경우, 비선형 함수의 입력을 선형 영역으로 제한함



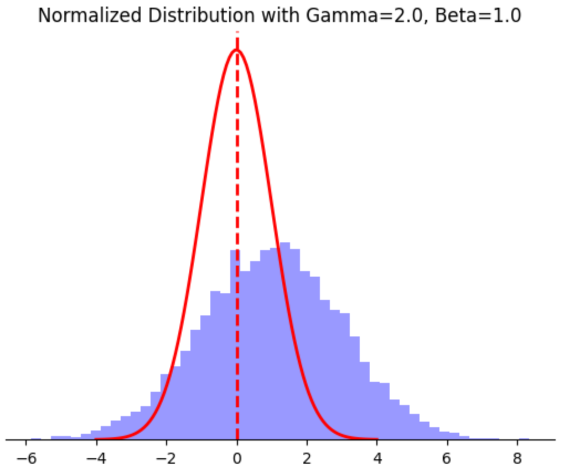
Batch Normalization (Training) : Parameters

- Normalization (Standardization)

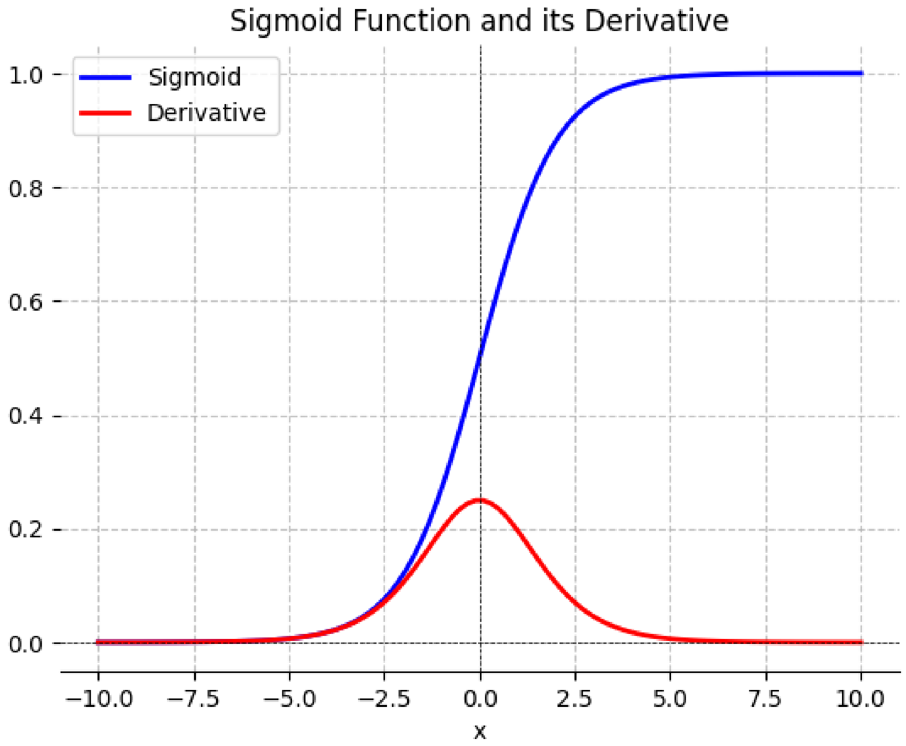
$$\widehat{x_i} = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \sim (0, 1)$$

- Scale and Shift

$$y_i = \gamma \widehat{x_i} + \beta \equiv BN_{\gamma,\beta}(x_i)$$



- 정규화 이후 γ, β 를 통해서 non-linearity 유지
- Vanishing Gradient 완화



Batch Normalization (Test)

- 학습된 parameter 고정된 Inference Network
- 사용된 mini-batch 통계량에 대한 평균과 분산 사용

Train

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

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

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

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

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

Output: Batch-normalized network for inference, $N_{\mathrm{BN}}^{\mathrm{inf}}$

```
N^tr_BN ← N    // Training BN network
for k = 1 ... K do
    Add transformation y^(k) = BN_{γ^(k),β^(k)}(x^(k)) to N^tr_BN (Alg. 1)
    Modify each layer in N^tr_BN with input x^(k) to take y^(k) instead
end for
Train N^tr_BN to optimize the parameters Θ ∪ {γ^(k), β^(k)}_{k=1}^K
N^inf_BN ← N^tr_BN    // Inference BN network with frozen
                      // parameters
for k = 1 ... K do
    // For clarity, x ≡ x^(k), γ ≡ γ^(k), μ_B ≡ μ_B^(k), etc.
    Process multiple training mini-batches B, each of size m, and average over them:
```

$$\boxed{\begin{aligned} \mathrm{E}[x] &\leftarrow \mathrm{E}_{\mathcal{B}}[\mu_{\mathcal{B}}] \\ \mathrm{Var}[x] &\leftarrow \tfrac{m}{m-1}\mathrm{E}_{\mathcal{B}}[\sigma_{\mathcal{B}}^2] \end{aligned}}$$

```
    In N^inf_BN, replace the transform y = BN_{γ,β}(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
```

Algorithm 2: Training a Batch-Normalized Network

Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift (PMLR 2015)

Batch Normalization 후속 연구

Batch Normalization:

Accelerating Deep Network by Reducing Internal Covariate Shift (2015)

- Issue: Internal Covariate Shift(ICS)
- Contribution
  - 같은 성능을 내면서 14배 학습 단계 감소
  - 혹은 성능이 더 좋음

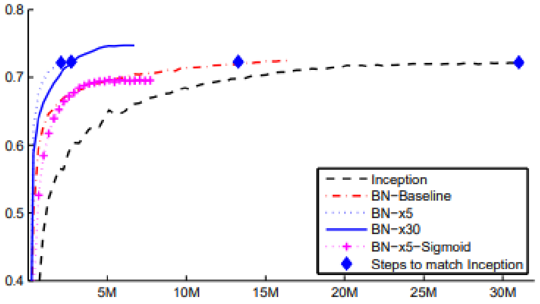


Figure 2: *Single crop validation accuracy of Inception and its batch-normalized variants, vs. the number of training steps.*

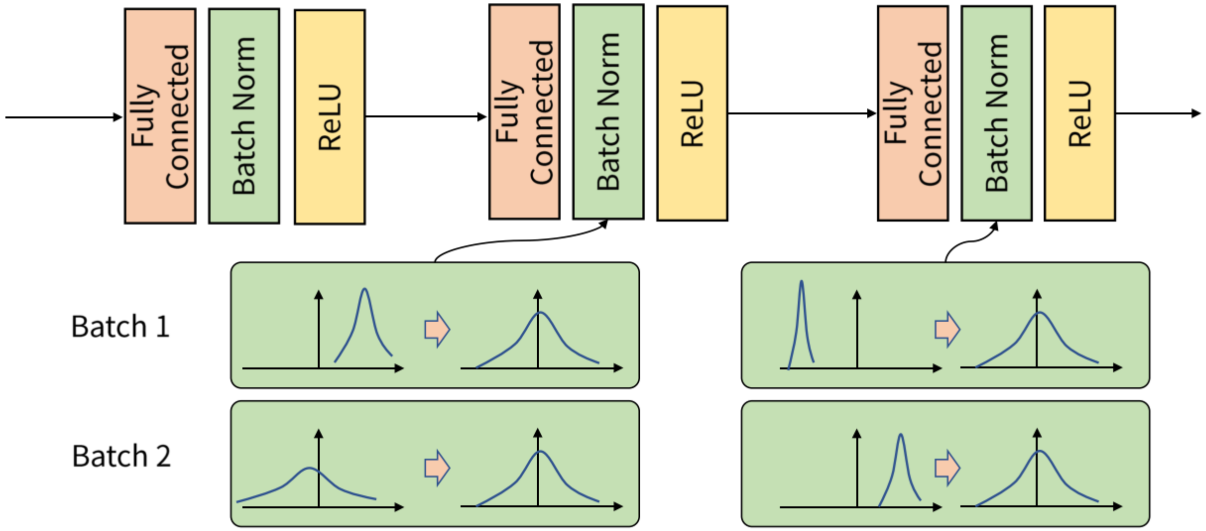
How Does Batch Normalization Help Optimization?(2018)

- Batch Normalization의 효과는 ICS 감소와 큰 상관이 없다

Batch Normalization

https://gaussian37.github.io/dl-concept-batchnorm/

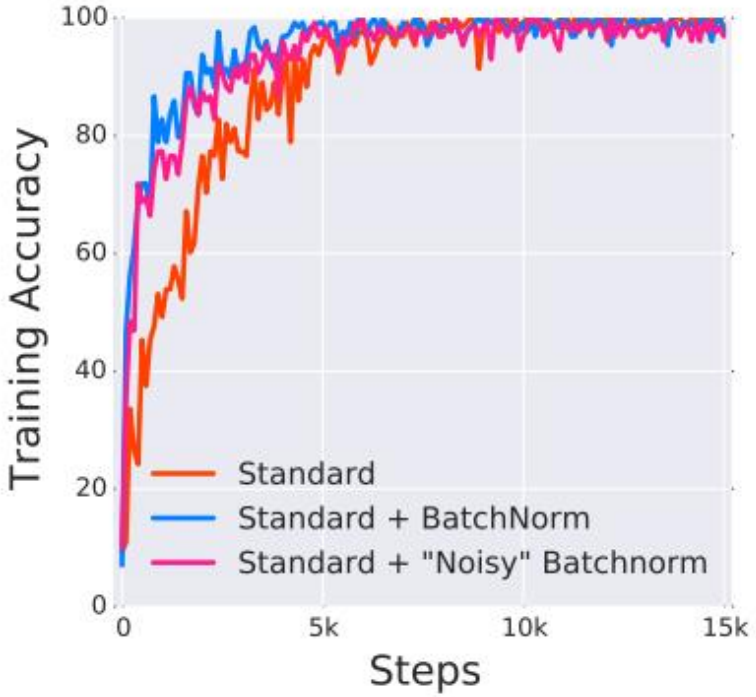
- **Batch Normalization의 장점**
  1. 학습 속도(training speed) 개선
  2. 가중치 초기화(weight initialization)에 대한 민감도 감소
  3. 모델의 일반화(regularization) 효과 – (dropout 필요성 감소)

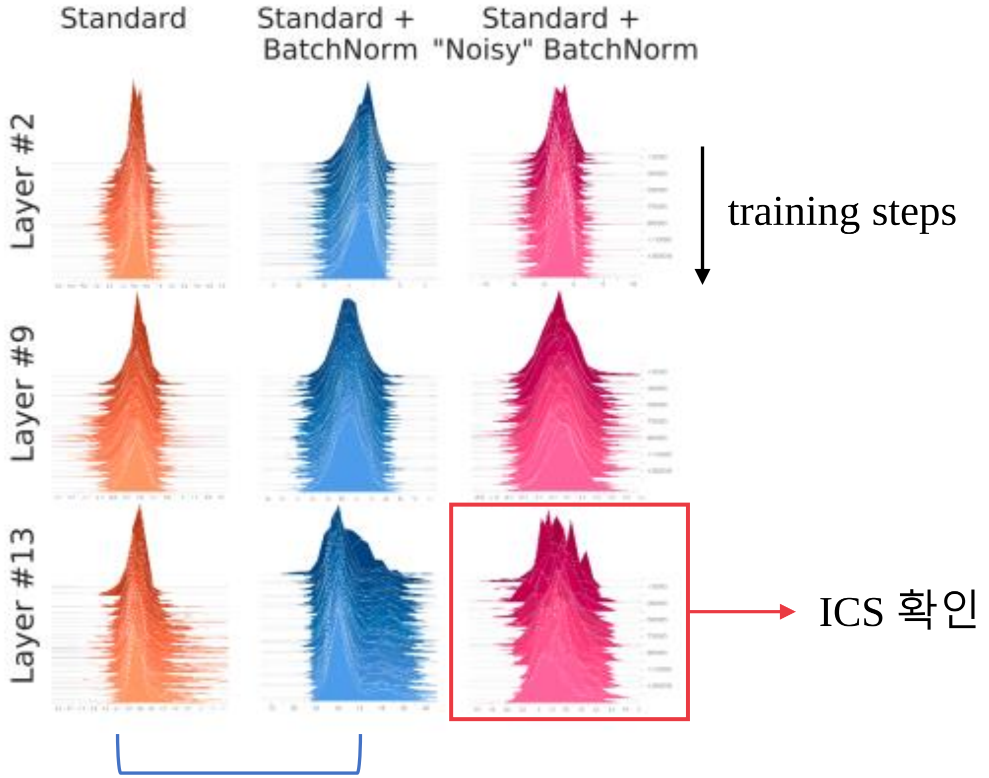


Batch Normalization and ICS

https://www.youtube.com/watch?v=58fuWVu5DVU

- Batch Normalization 직후에 랜덤 노이즈를 넣어 ICS를 발생시켰을 때에도 일반 네트워크보다 성능 우수





Smoothing Effect

- Batch Normalization이 Loss Landscape를 **부드럽게 만드는(Smoothing) 효과**가 있음
  - Loss Landscape: Loss function을 다차원 공간에 시각화한 것
  - Smoothing method: Batch Normalization, Residual Connection

How Does Batch Normalization Help Optimization? (NIPS 2018)

Normalization methods