모두가 알지만 모르는

딥러닝 기초

20191582 소프트웨어학부 김혜성

익숙하고 당연하겠지만..

은닉층에선 무슨 일이 일어날까

- 깊은 모델을 위해 활성화 함수는 필수
- 어째서?

정규화는 왜 할까

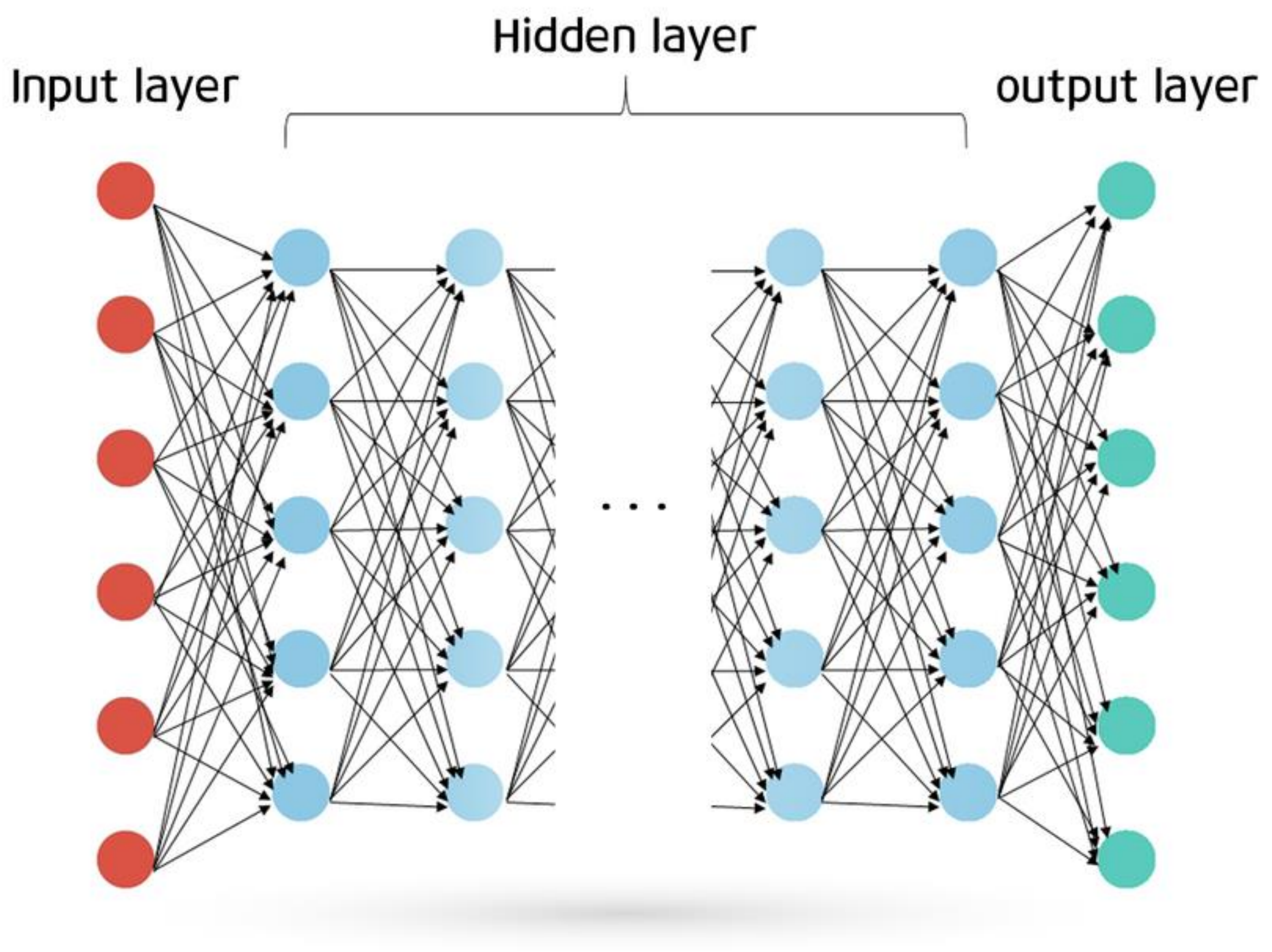
- 배치정규화는 무슨 차이가 있을까?
- 장점은?

여러가지 최적화 함수

변수 초기화 왜 랜덤으로 할까

- 랜덤으로 초기화 하는 이유는?
- 그냥 0이나 1로 초기화 하면 안되나?

1. 은닉층 (Hidden Layer)



은닉층에선 무슨 일이 일어날까?

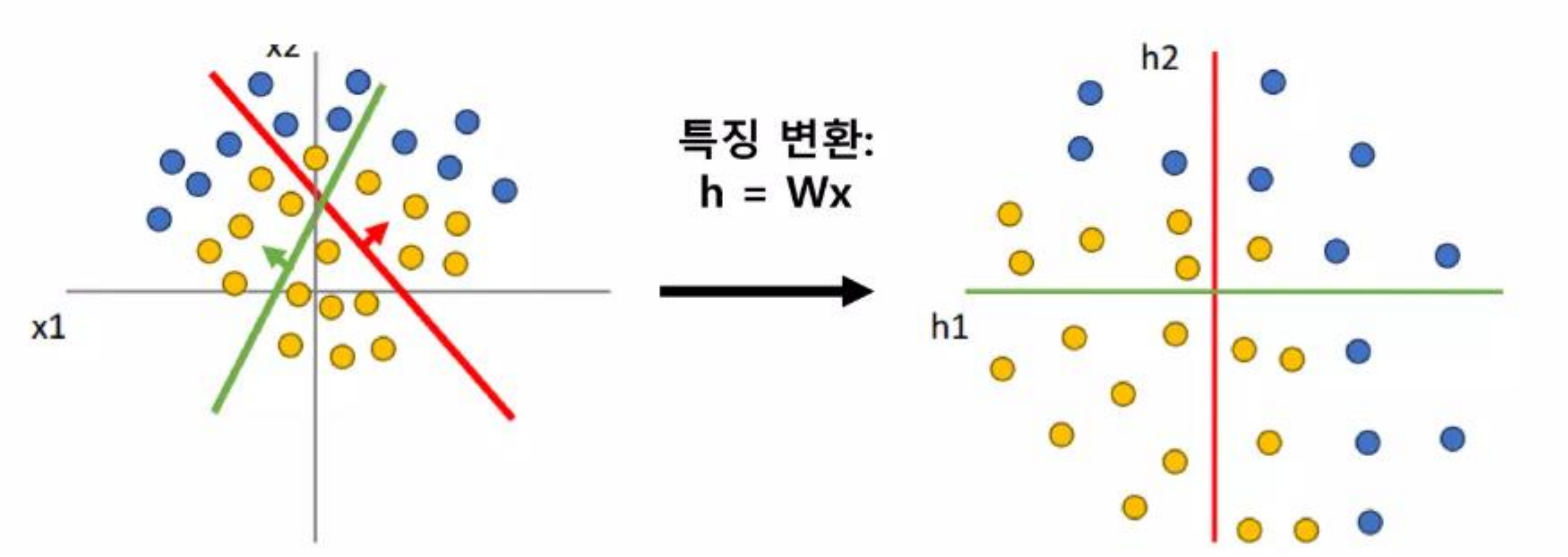
활성화 함수는 왜 필요할까?

선형성(Linearity)과 선형변환

Linear 하다는 것은 다음 두 가지 식을 만족하는 것

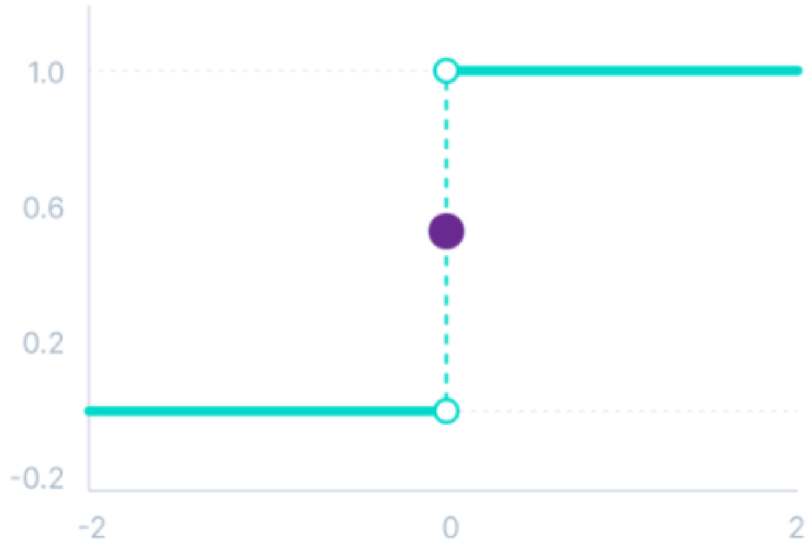
$$f(x_1+x_2)=f(x_1)+f(x_2)$$

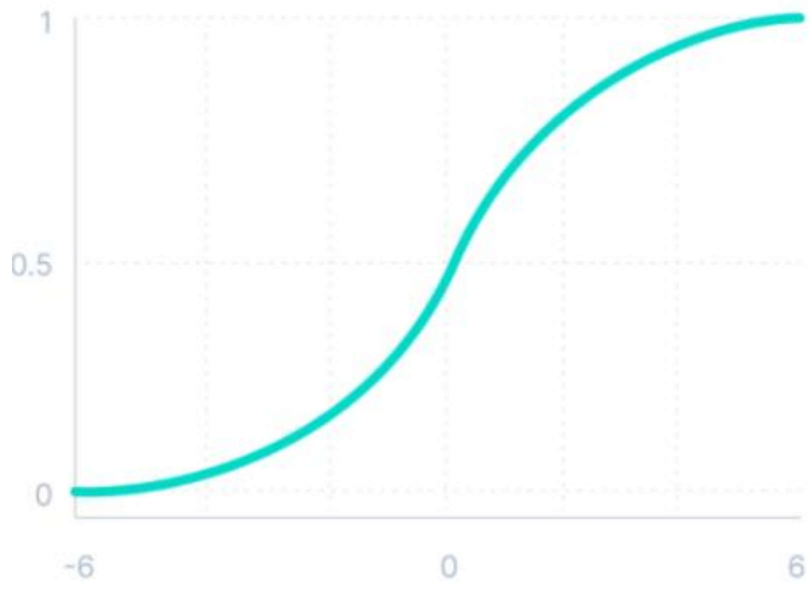
$$f(ax_1)=af(x_1)$$

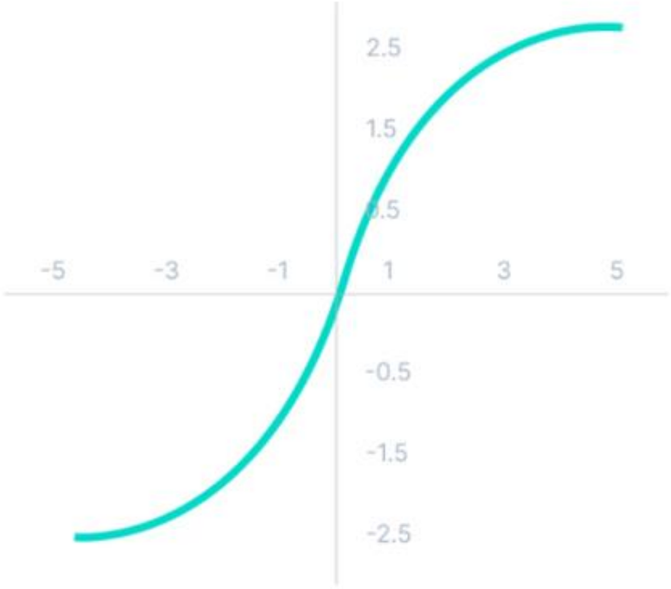


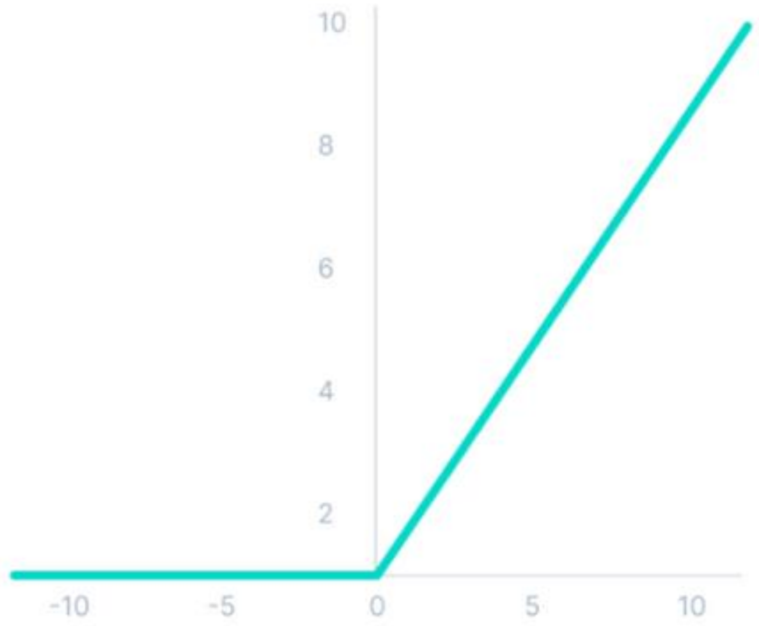
- 선형변환의 결과는 반드시 선형성을 만족한다.
- 이 선형성을 깨기 위해 <활성화 함수>가 사용되는 것!

Activation Function

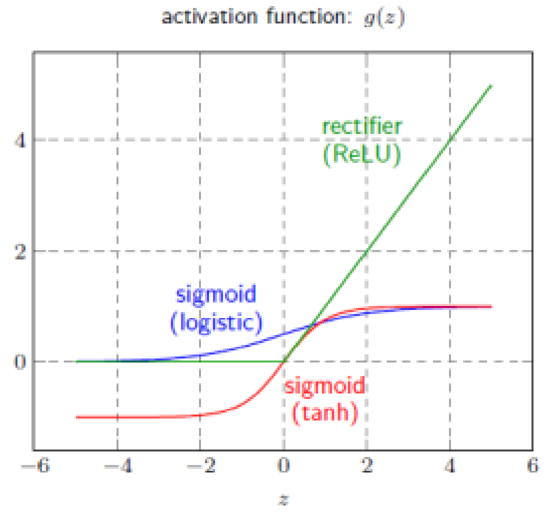


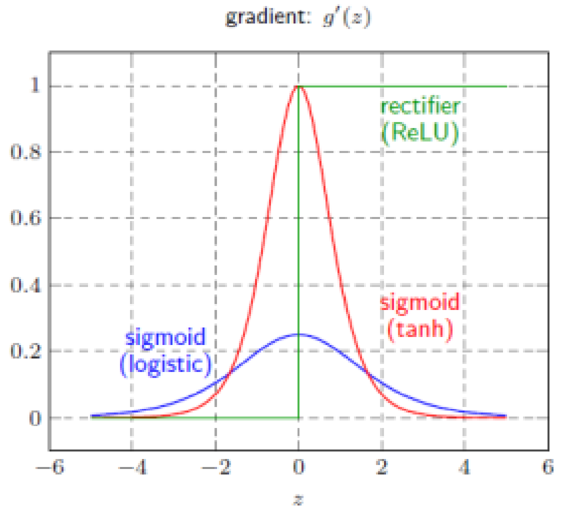


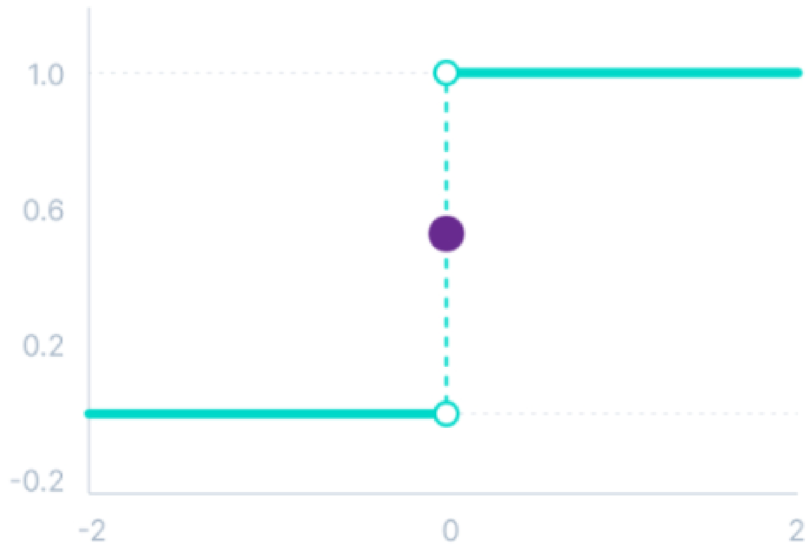




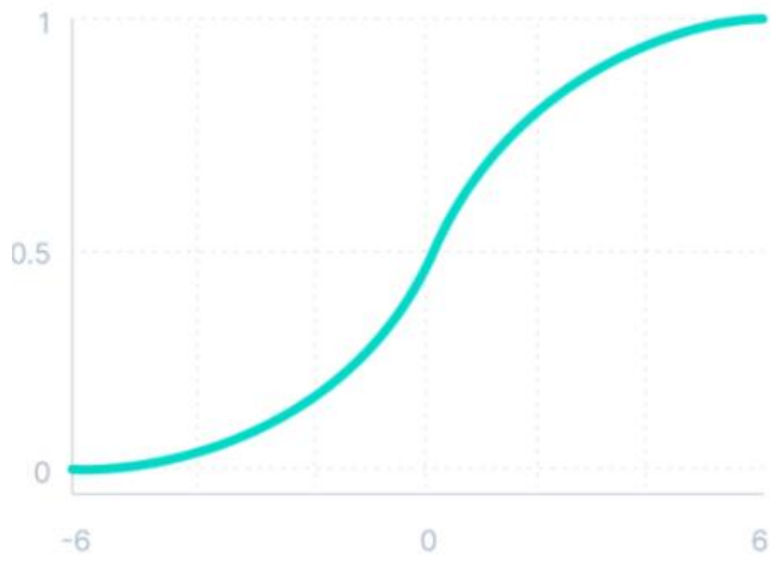
Non-Linear Function



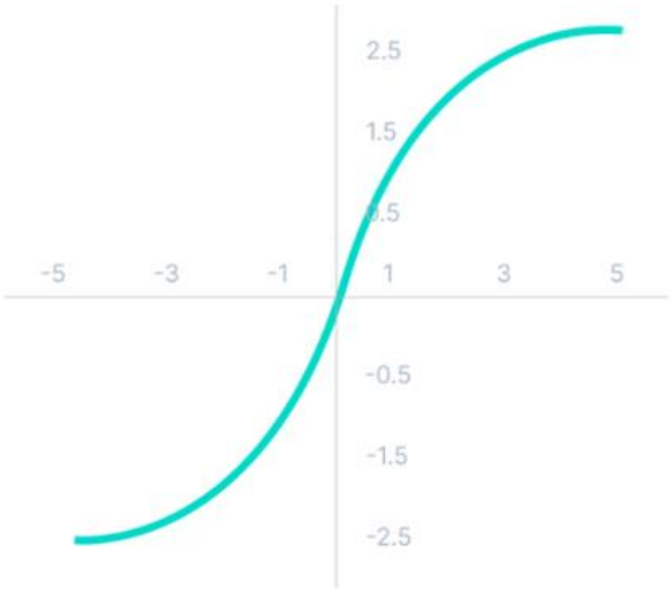




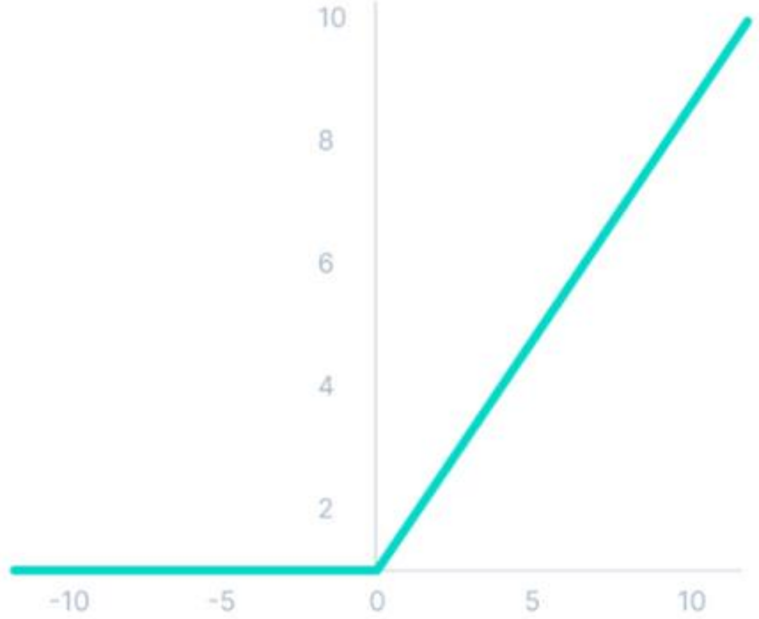
STEP



SIGMOID

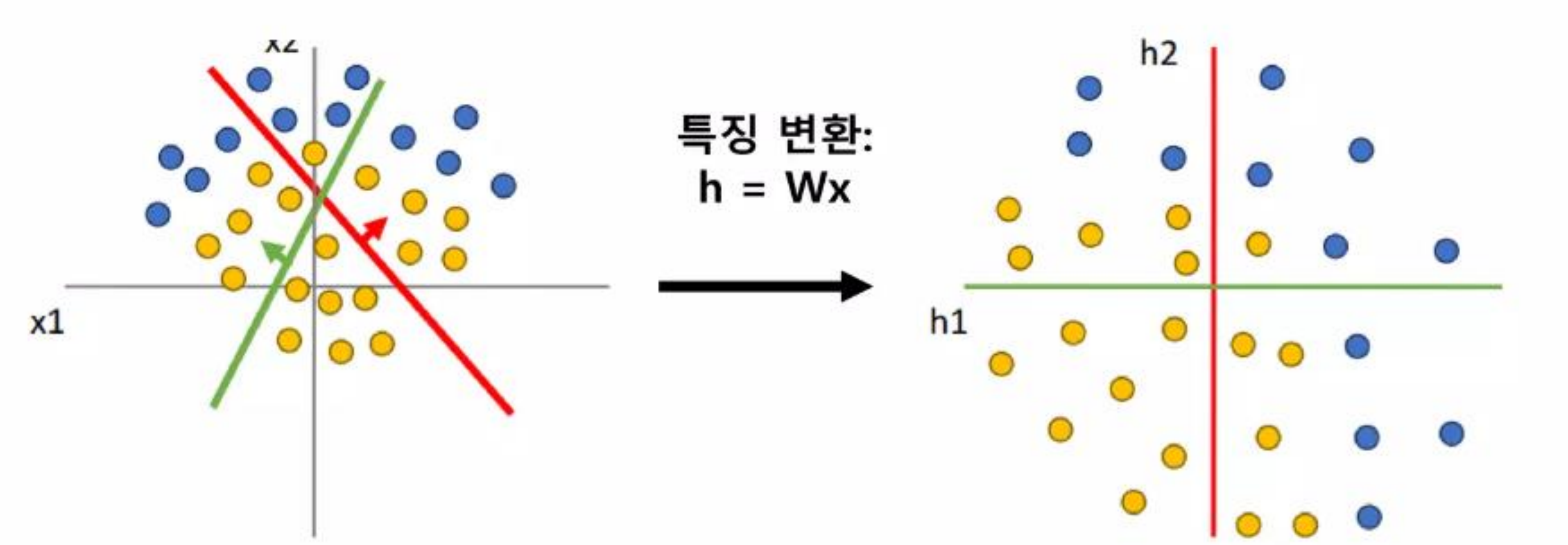


TANH



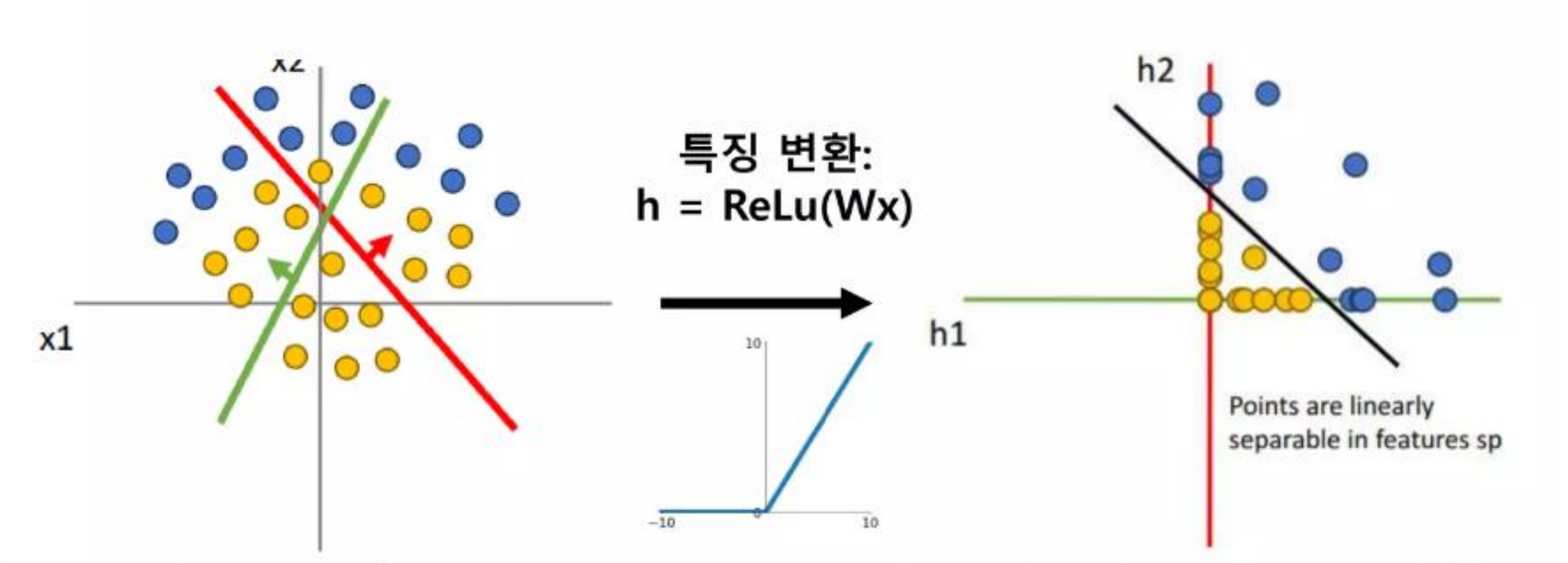
RELU

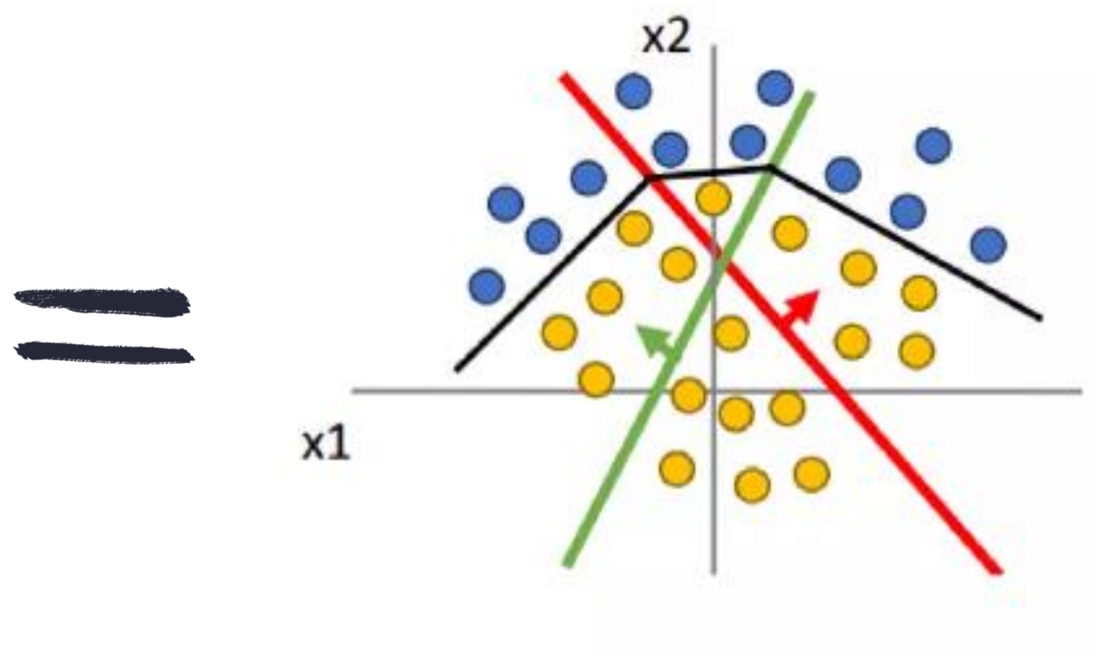
1) 선형 변환



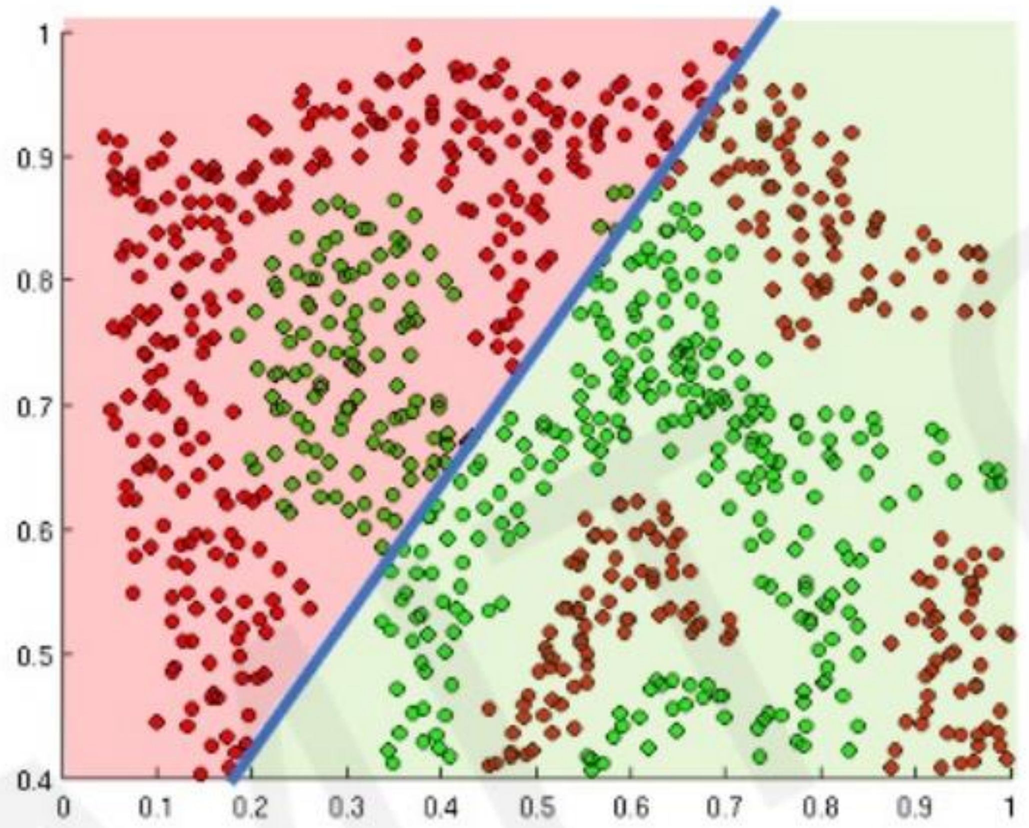
2) 선형 변환 + 비선형 함수

비선형성이 추가됨!

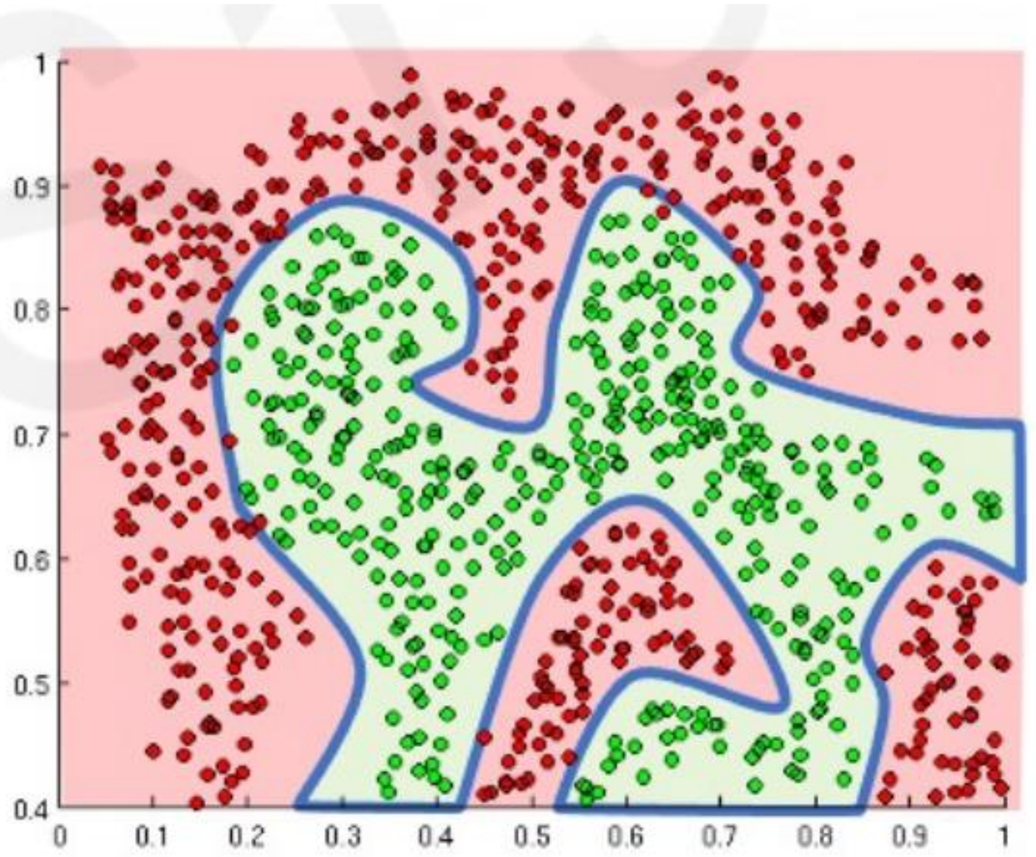




비선형 함수가 없다면



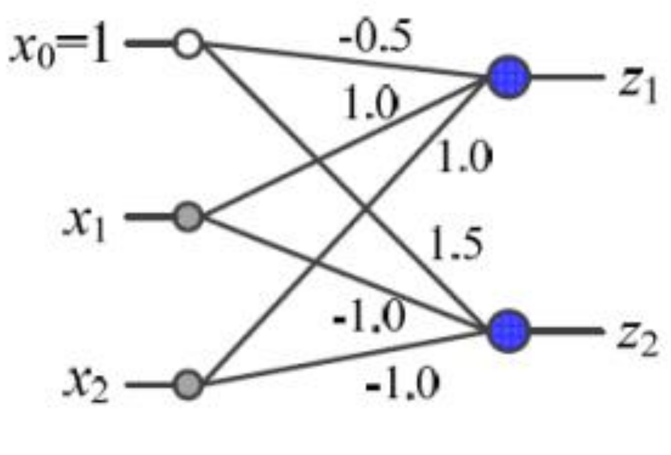
Linear activation functions produce linear decisions no matter the network size



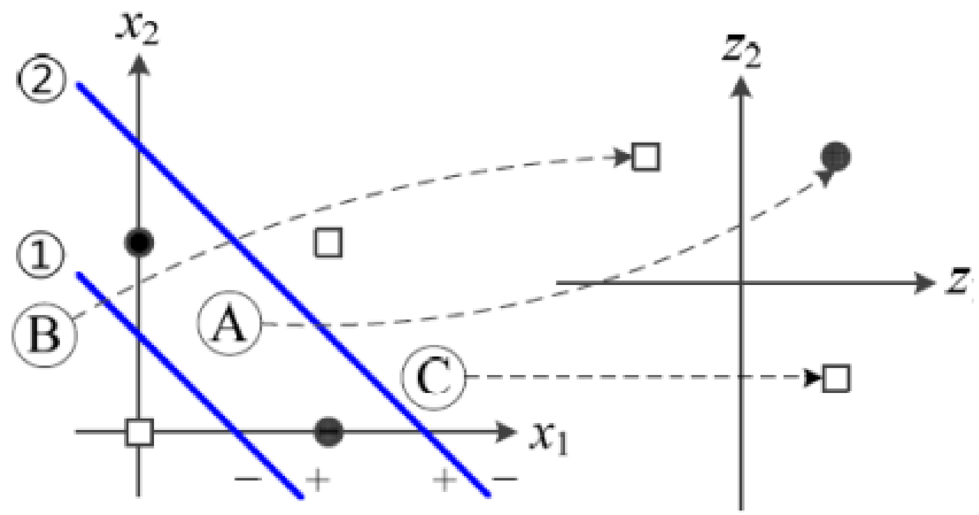
Non-linearities allow us to approximate arbitrarily complex functions

은닉층은 특징추출기!

예시 1) 2차원 공간을 3개로 분할하여 2차원에 매핑

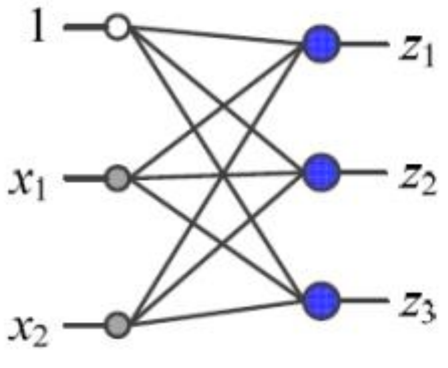


(a) 두 퍼셉트론을 병렬로 결합

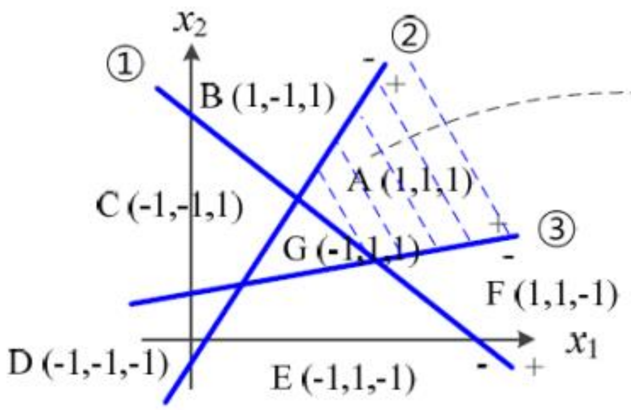


(b) 원래 특징 공간 x를 새로운 특징 공간 z로 변환

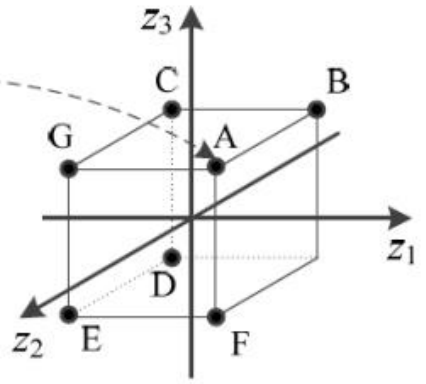
예시 2) 2차원 공간을 7개로 분할하여 3차원에 매핑



(a) 퍼셉트론 3개를 결합



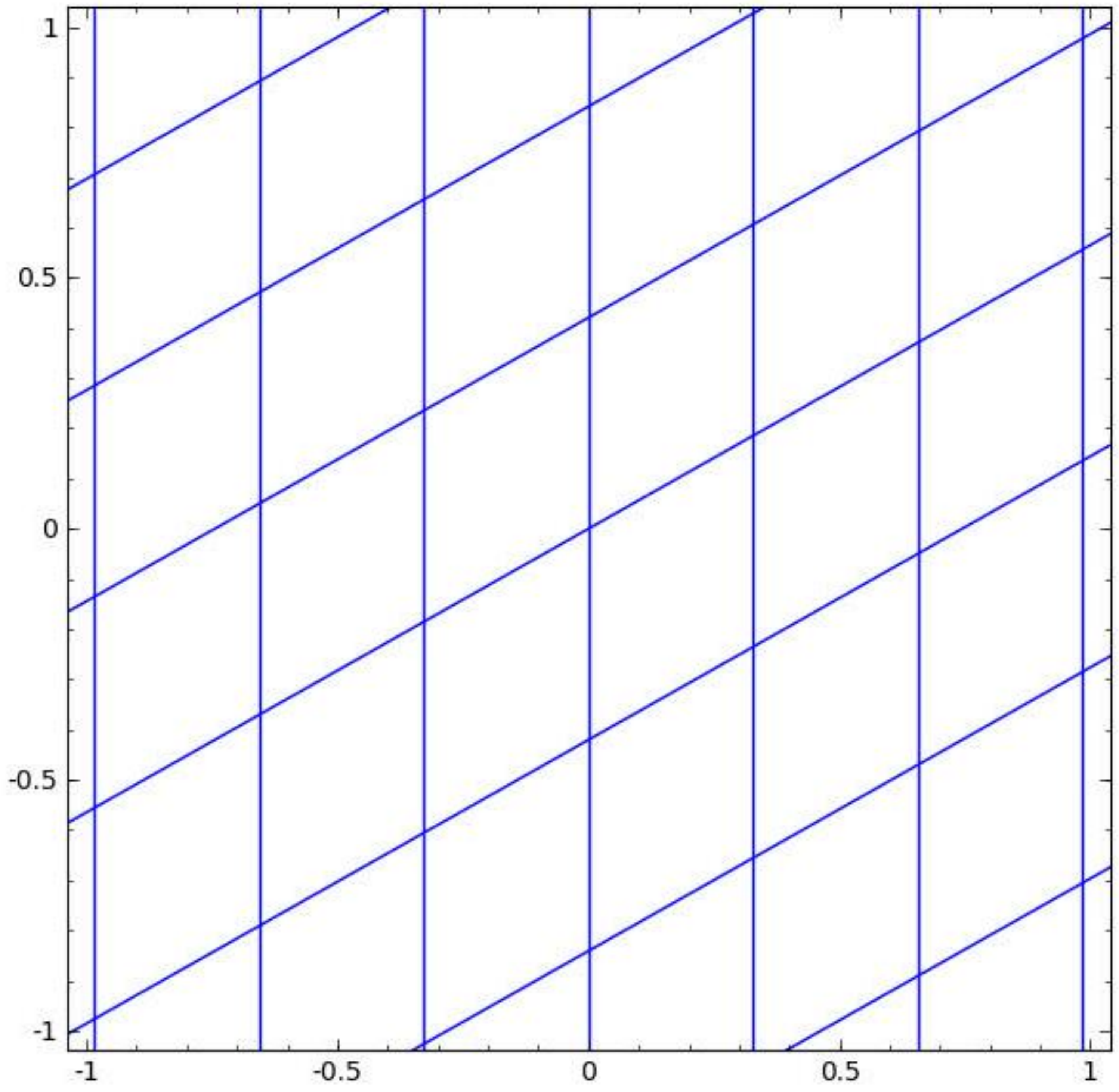
(b) 7개 부분공간으로 나눔



(c) 3차원 공간의 점으로 매핑

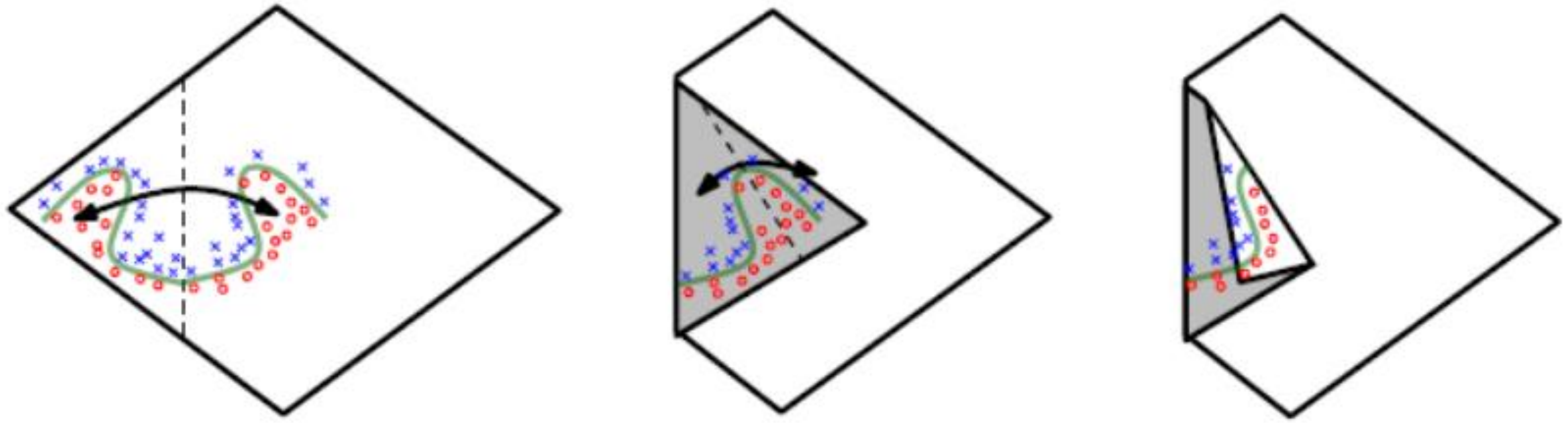
은닉층은 특징 벡터를 분류에 더 유리한 새로운 특징 공간으로 변환

Affine 변환 = 선형 변환 + 비선형 함수



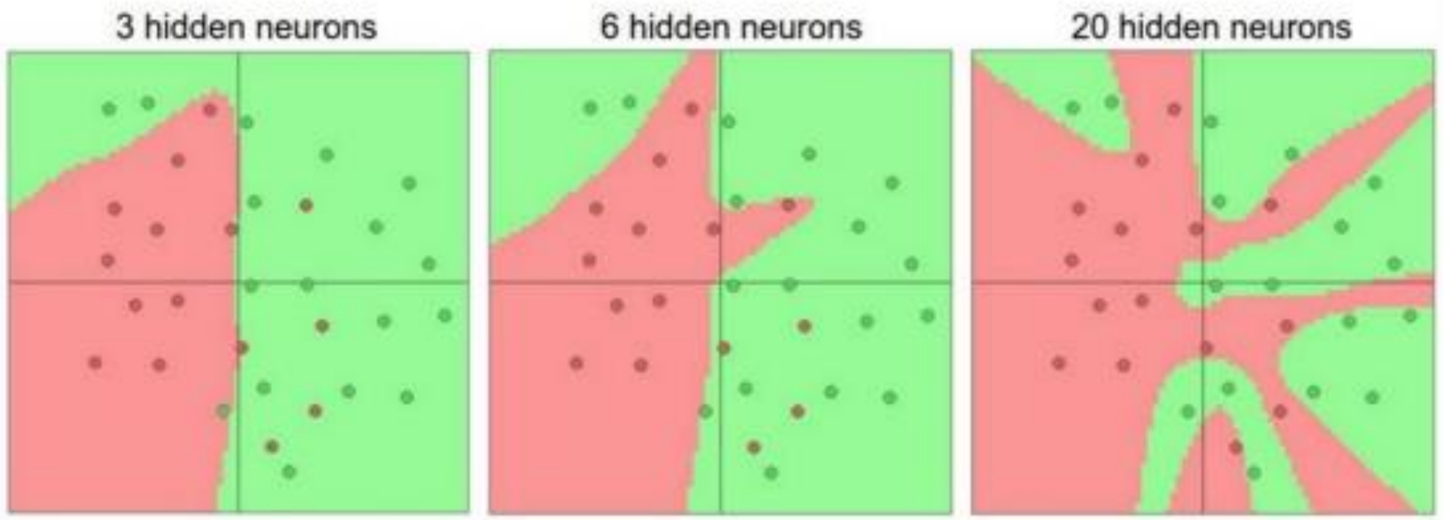
은닉층을 통한 특징공간의 변환
- 행렬 곱: 회전
- 편향: 이동
- 비선형 함수: 왜곡

층(Layer)을 쌓는 의미

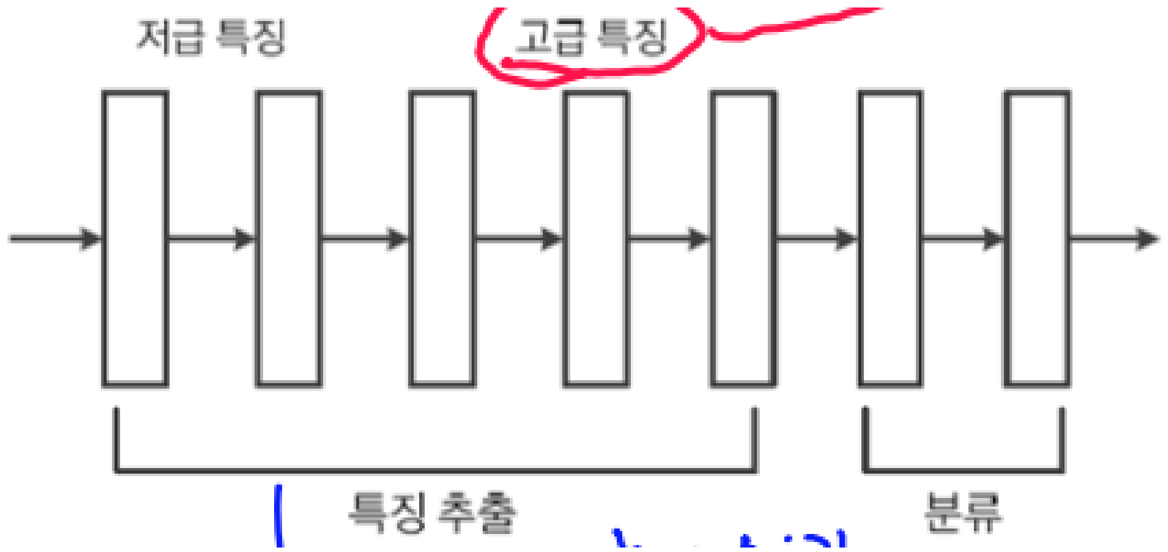


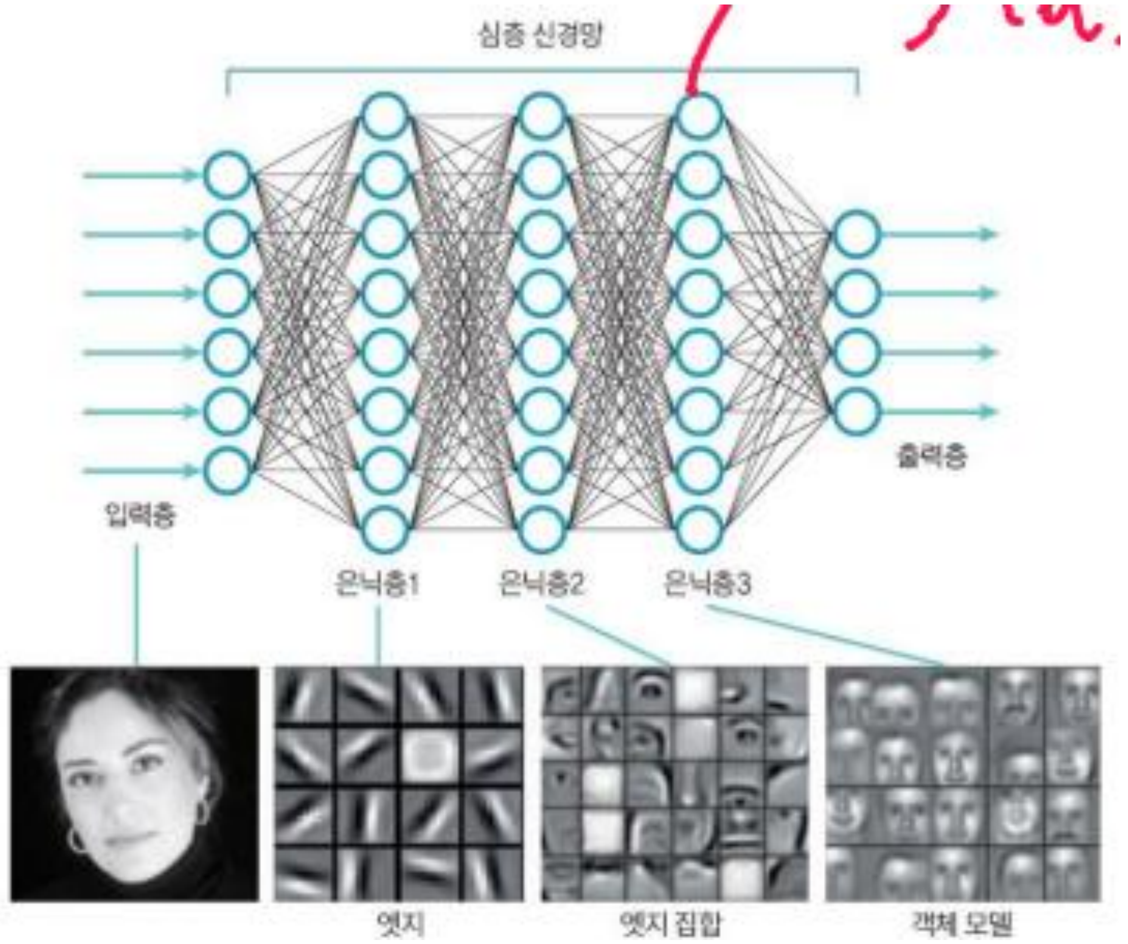
깊은 은닉층은 공간을 더 많이 접어 Task에 유리한 특징공간으로 변환시킨다.

- # of layers



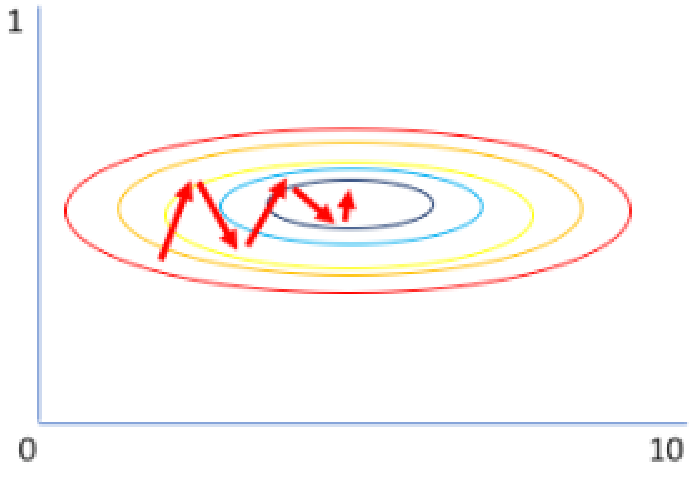
층(Layer)에 따른 점진적 추상화



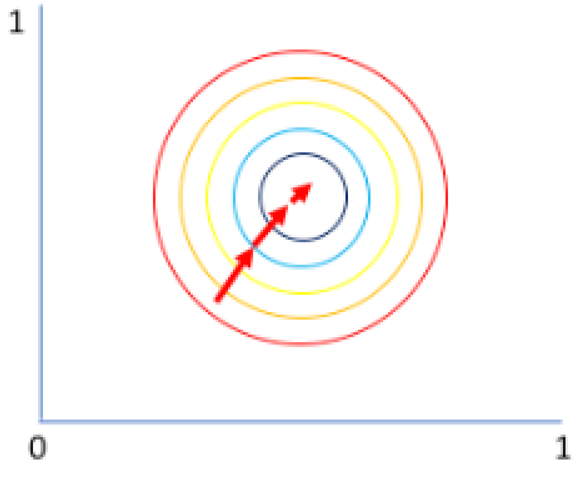


- 낮은 단계 은닉층 (입력쪽): 선이나 모서리와 같은 간단한 (저급) 특징 추출
- 높은 단계 은닉층 (출력쪽): 추상적인 형태의 복잡한 (고급) 특징을 추출

2. 데이터 전처리 및 초기화



Gradient of larger parameter dominates the update



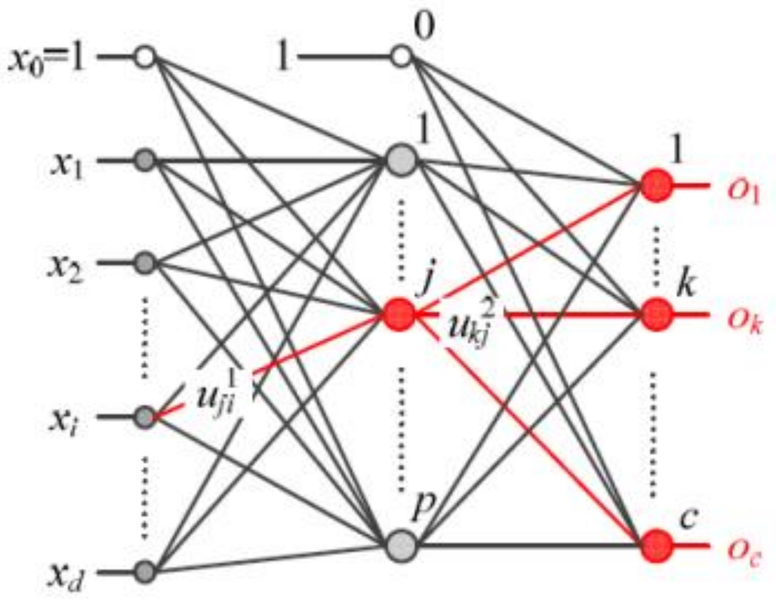
Both parameters can be updated in equal proportions

데이터를 정규화 하는 이유는?

가중치는 왜 랜덤으로 초기화 할까?

데이터 규모(Scale)가 맞지 않는 문제

- 데이터 전처리
  - 규모scale 문제
    - \+ 예) 건강에 관련된 데이터 (키(m), 몸무게(kg), 혈압)^T
      - \+ 1.885m와 1.525m는 33cm나 차이가 나지만 특징 값 차이는 불과 0.33
      - \+ 65.5kg과 45.0kg은 20.5라는 차이
    - \+ 첫 번째와 두 번째 특징은 양수이며, 대략 100배의 규모 차이



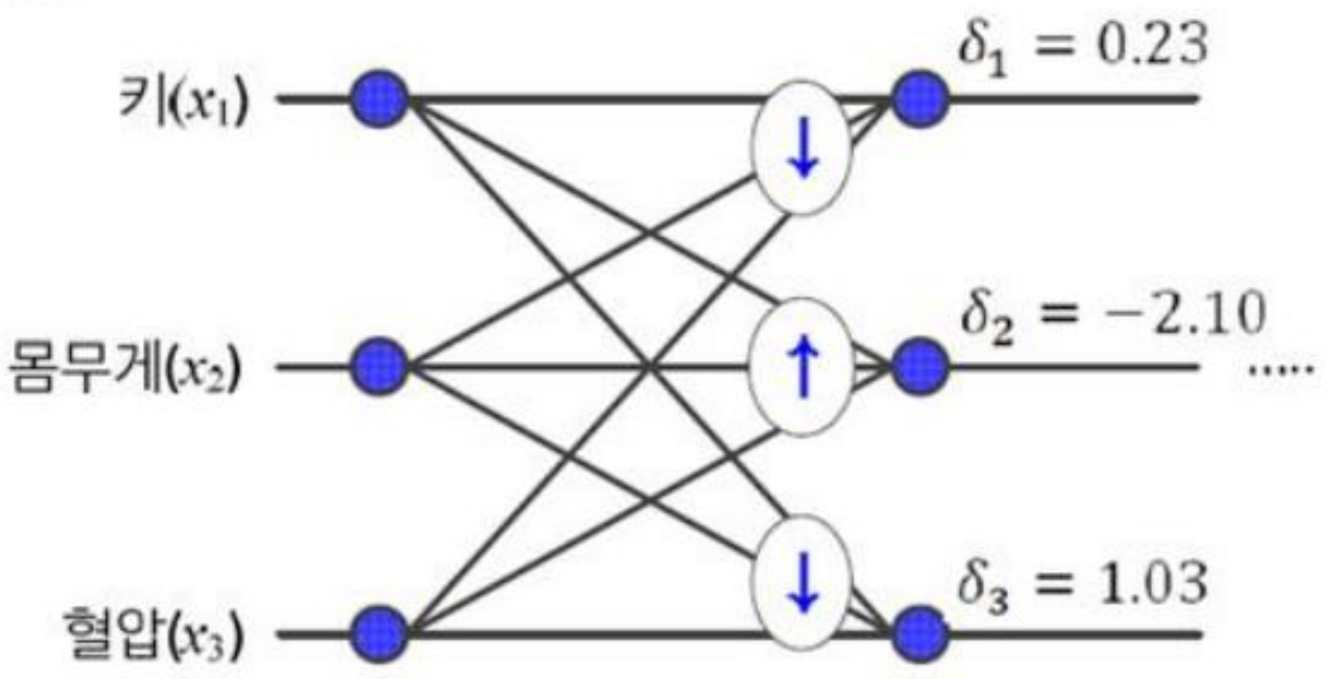
$$\delta_k = (y_k - o_k)\tau'(osum_k), \qquad 1 \le k \le c$$

$$\frac{\partial J}{\partial u_{kj}^2} = \Delta u_{kj}^2 = -\delta_k z_j, \qquad 0 \le j \le p, 1 \le k \le c$$

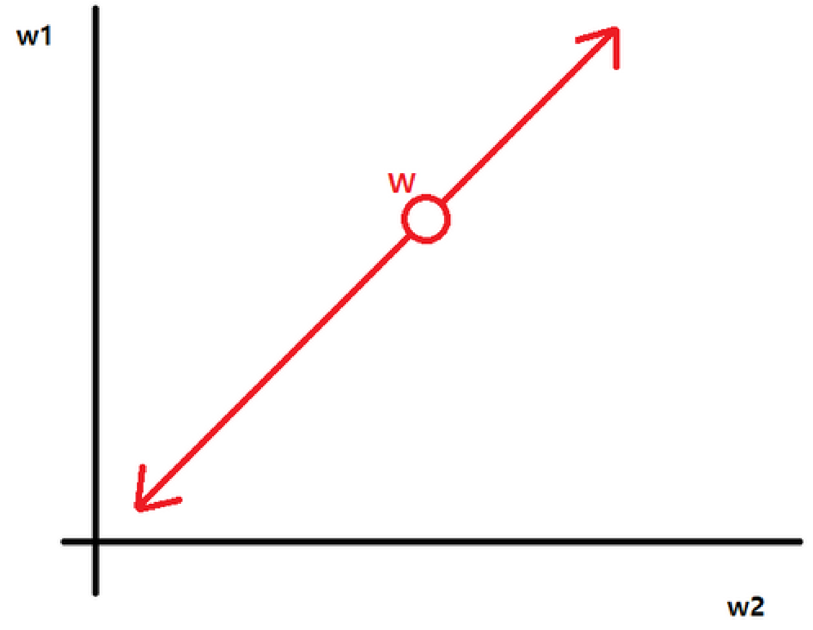
- \+ $-\delta_j z_i$가 경사도
  - → 첫 번째 특징에 연결된 가중치는 두 번째 특징에 연결된 가중치에 비해 100여 배 느리게 학습됨

모든 입력의 부호가 동일할 때

- 모든 입력이 양수인 경우의 문제



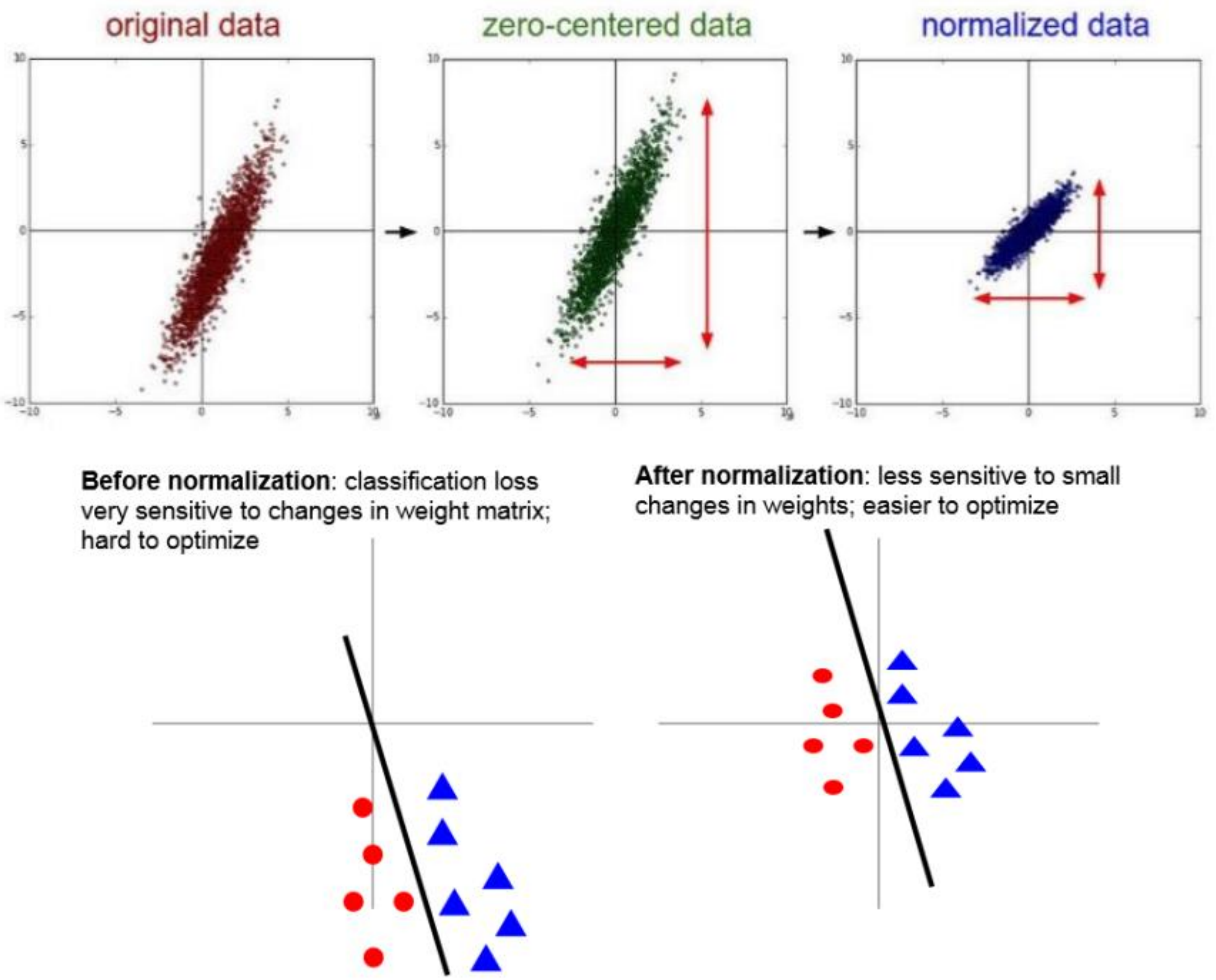
특징이 모두 양수일 때 가중치가 뭉치로 갱신되는 효과



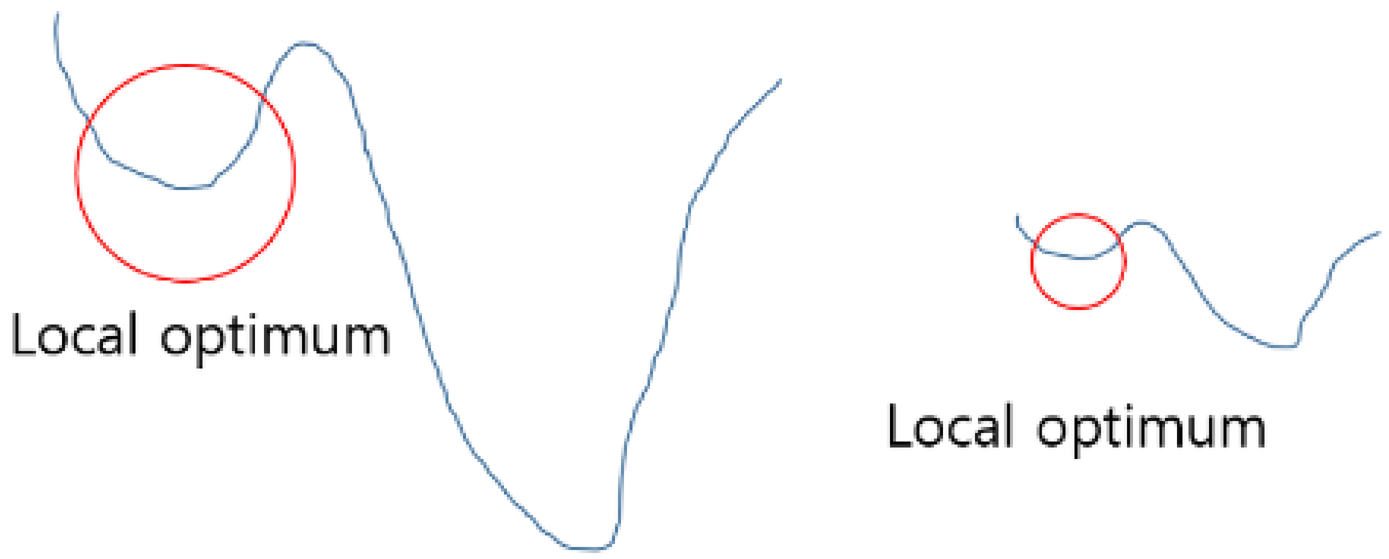
+ $-\delta_j z_i$가 경사도이므로 [위 그림]의 경우 ↑ 표시된 가중치는 모두 증가, ↓ 표시된 가중치는 모두 감소

→ 위처럼 가중치가 뭉치로 증가 또는 감소하면 최저점을 찾아가는 경로가 지그재그zigzag하여 느린 수

정규화가 최적화에 도움이 되는 이유



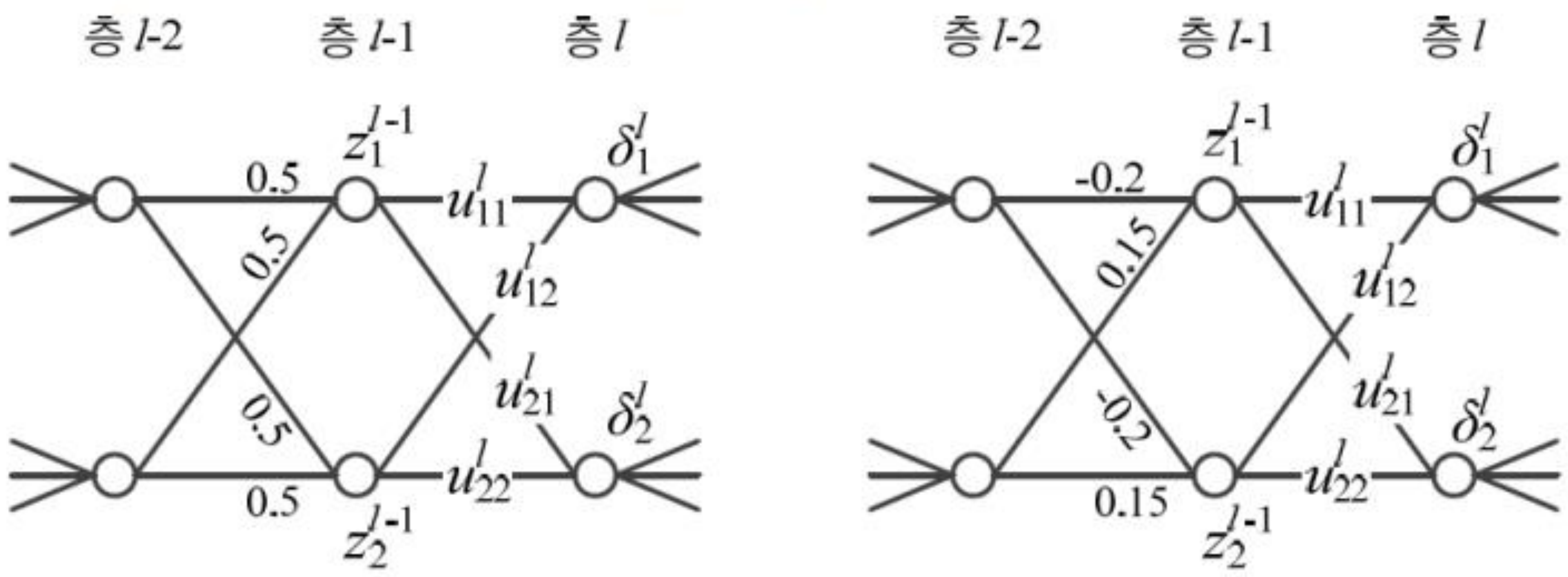
더 빠른 수렴이 가능!



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

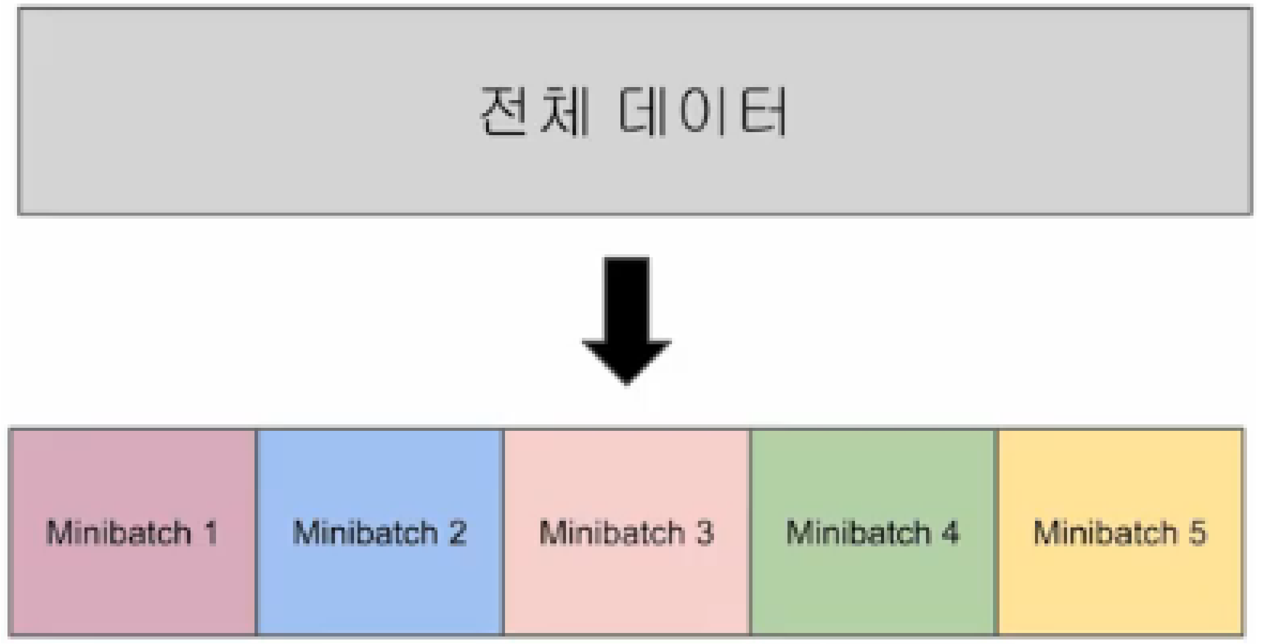
난수(Random) 가중치 초기화

- 가중치 초기화
  - 대칭적 가중치 문제
    - [그림 아래]의 대칭적 가중치에서는 z_1^{l-1}과 z_2^{l-1}가 같은 값이 됨



    - $-\delta_j z_i$가 경사도이기 때문에 u_{11}^l과 u_{12}^l이 같은 값으로 갱신됨 ← 두 노드가 같은 일을 하는 중복 발생
  - 난수로 초기화를 통한 대칭 파괴symmetry break로 문제 해결

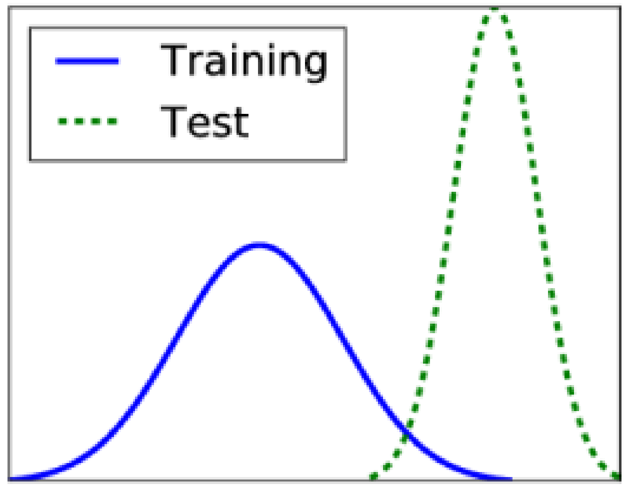
배치 정규화(Batch Normalization)



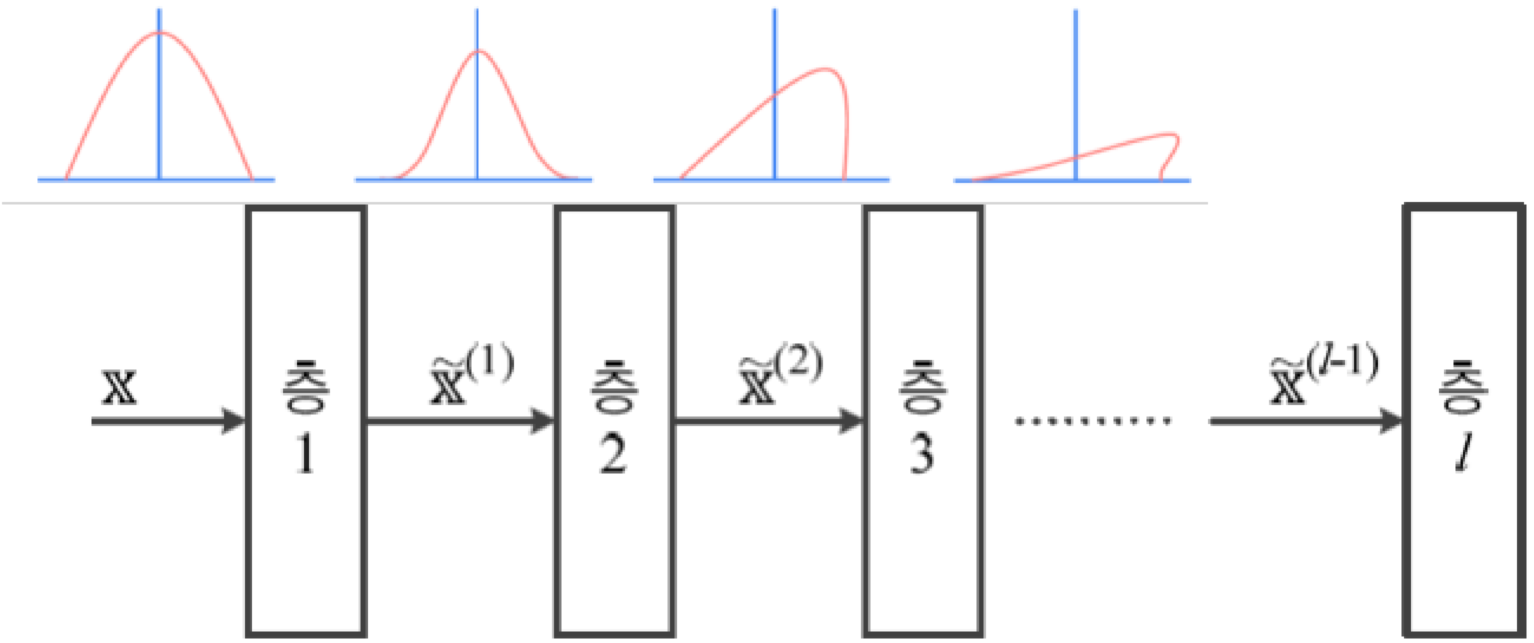
- 전에 설명했던 정규화와 ‘목적’에서의 차이점은?

공변량 변화 (Convariate shift)

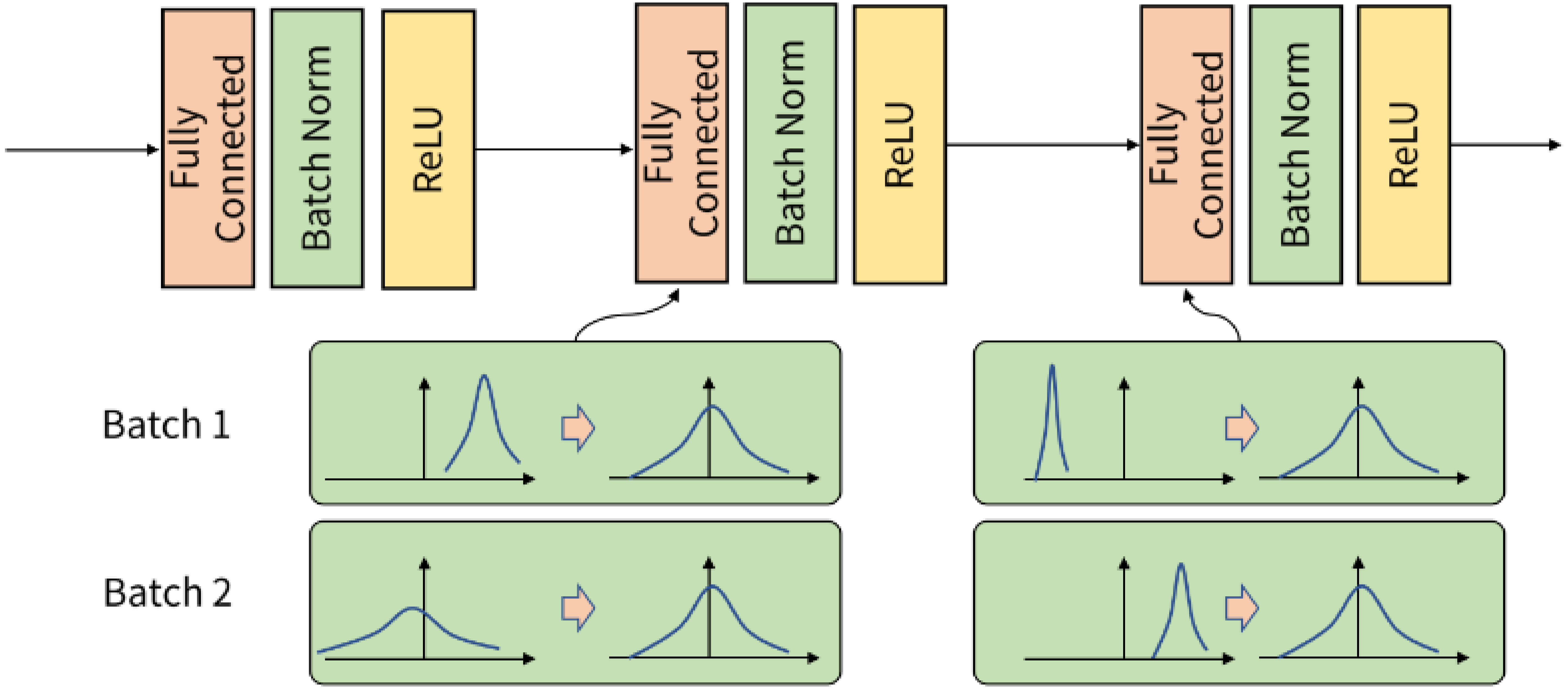
- 일반적인 공변량 변화covariate shift 현상: 훈련 데이터집합과 시험 데이터집합의 분포가 다름



- 내부의 공변량 변화internal covariate shift
  - 학습이 진행되면서 첫번째 층의 매개변수가 바뀜에 따라 $\tilde{\mathbb{x}}^{(1)}$이 따라 바뀜
    → 두번째 층 입장에서 보면 자신에게 입력되는 데이터의 분포가 수시로 바뀌는 셈
  - 층2, 층3, …으로 깊어짐에 따라 더욱 심각→ 학습을 방해하는 요인으로 작용



배치 정규화 (Batch Normalization)



feature들이 layer를 지나갈수록 서로 다른 분포가 생기는 것을 방지

배치 정규화의 과정

학습 단계

코드 1:

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} z_i \qquad \text{\# 미니배치 평균}$$

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(z_i - \mu_B)^2 \qquad \text{\# 미니배치 분산}$$

$$\tilde{z}_i = \frac{z_i - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}}, \qquad i = 1,2,\cdots,m \qquad \text{\# 정규화}$$

$$z_i' = \gamma \tilde{z}_i + \beta, \qquad i = 1,2,\cdots,m \qquad \text{\# 비례}^{\text{scale}}\text{와 이동}^{\text{shift}}$$

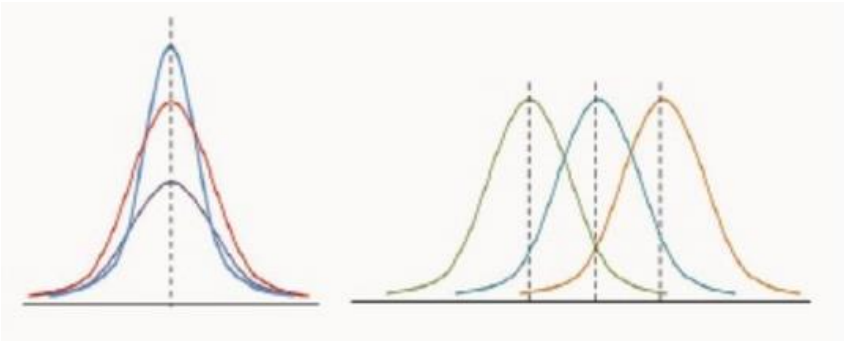
추론 단계

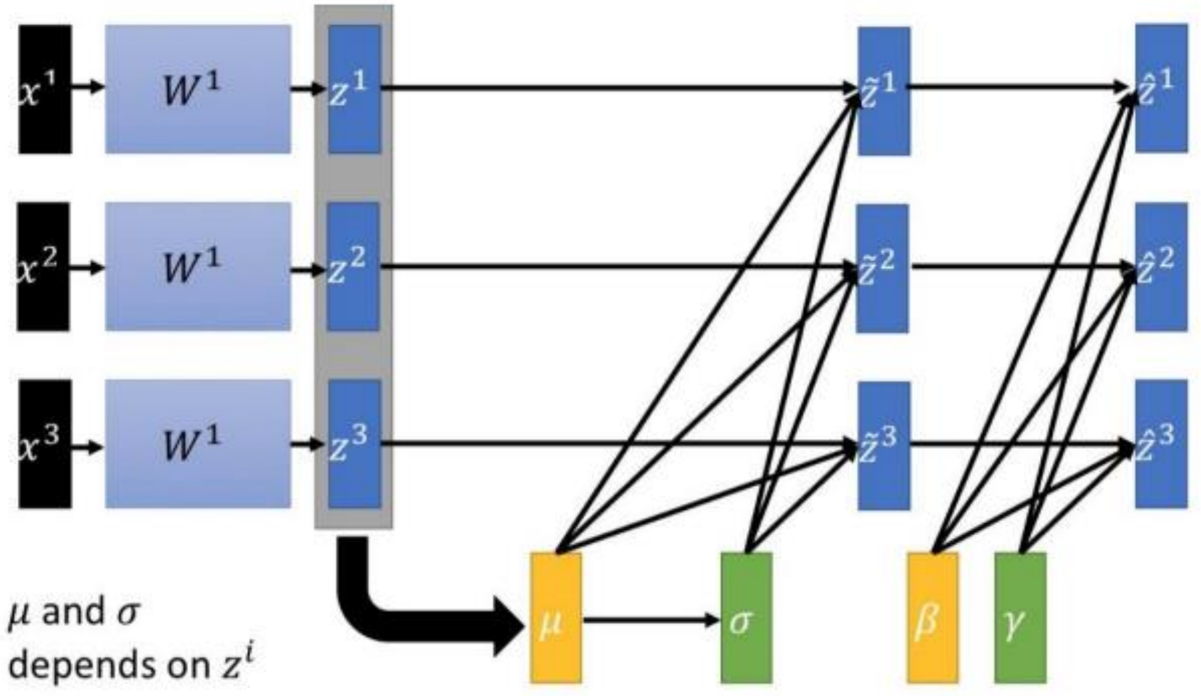
코드 2:

$$\mu = \frac{1}{n}\sum_{i=1}^{n} z_i$$

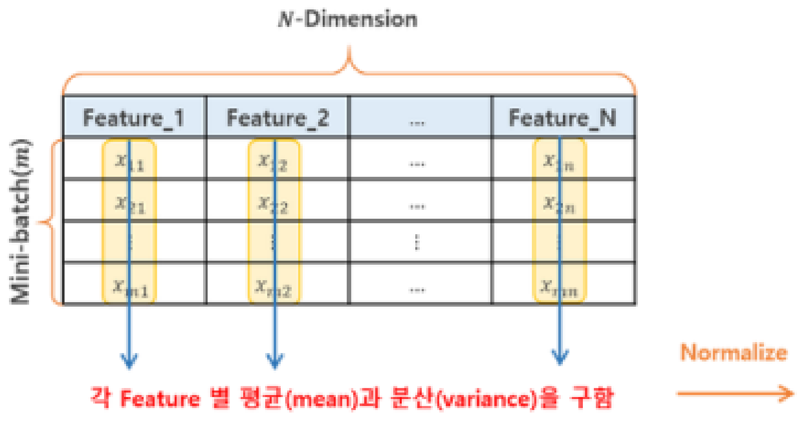
$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(z_i - \mu)^2$$

노드에 μ, σ^2, γ, β를 저장한다.





1. 미니배치 단위로 평균(μ)과 분산(σ) 계산
2. 구한 평균과 분산을 통해 정규화



$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}}$$

3. 비례(γ)와 이동(β) 세부 조정

배치 정규화의 장점

초기화에 대한 의존성 감소

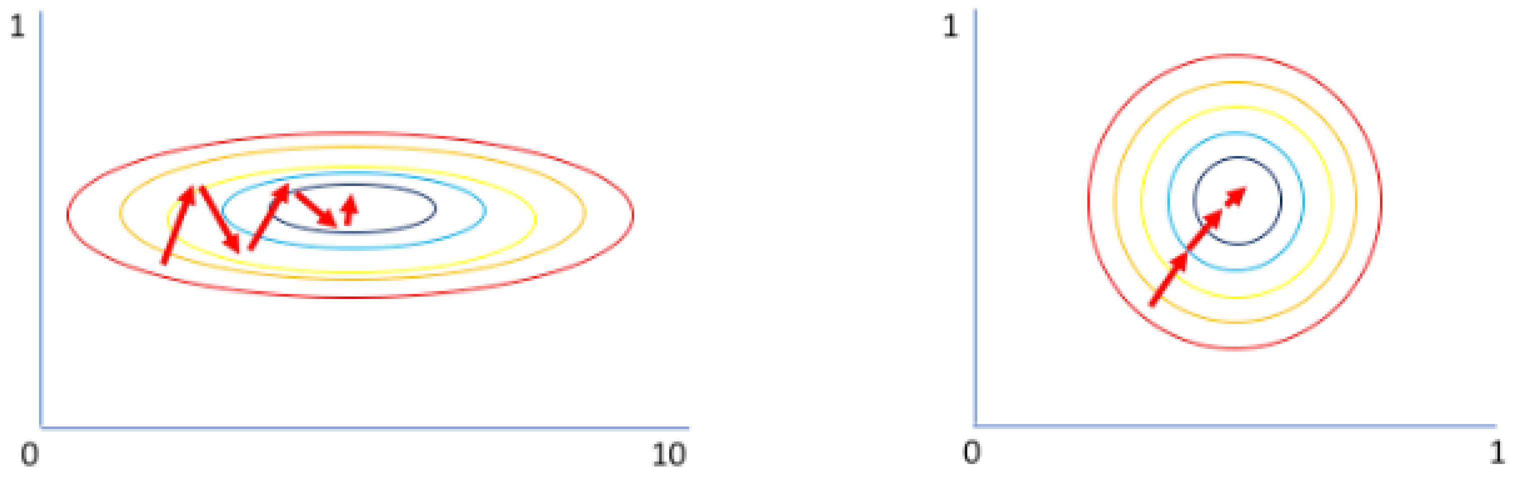
- 초기값에 상관없이 분포를 맞춰주기 때문

드롭아웃 필요성 감소

- 정규화 과정은 노이즈를 추가하는 과정

학습률에 대한 의존성 감소

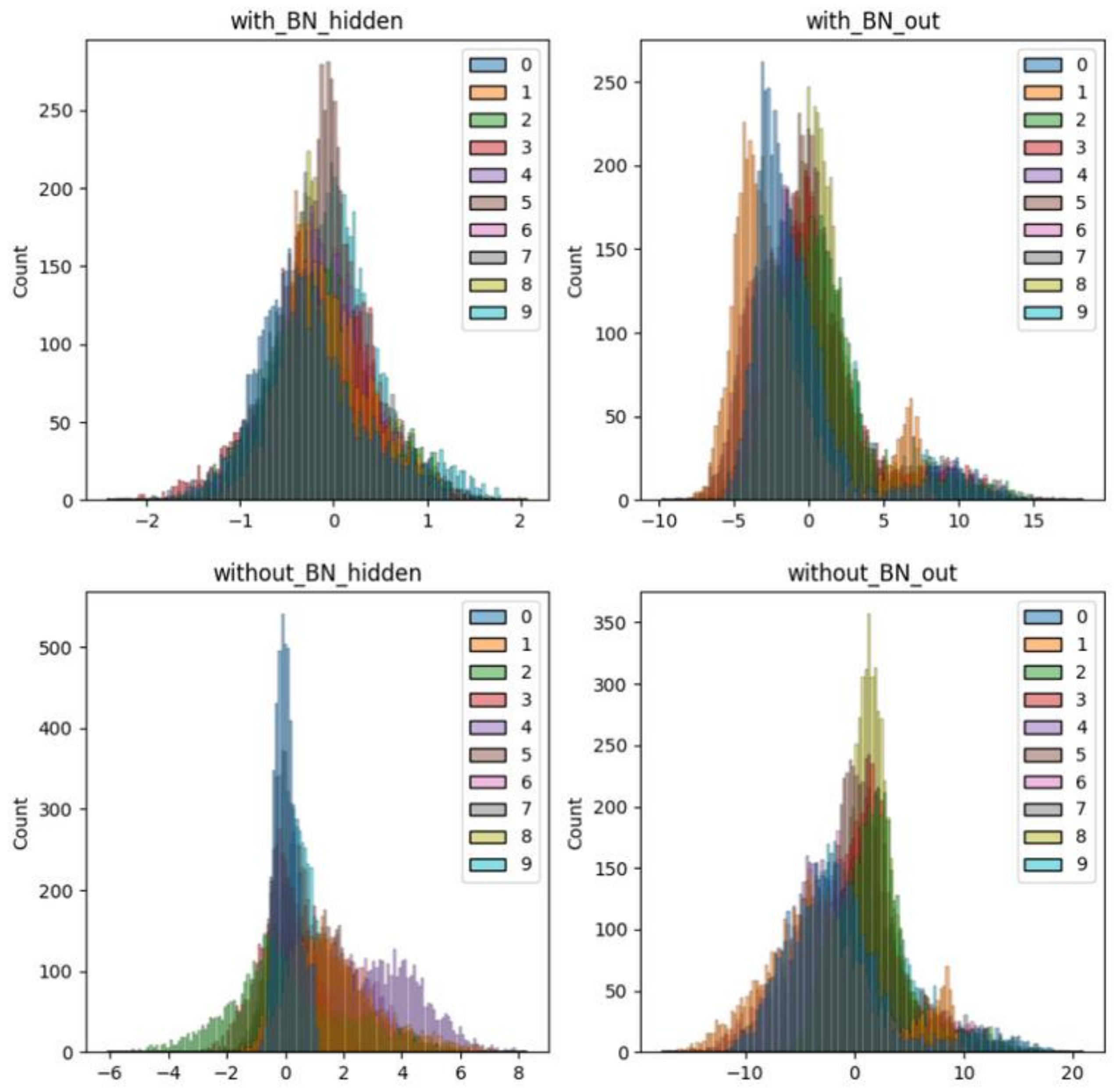
- 파라미터의 scaling 되었기 때문

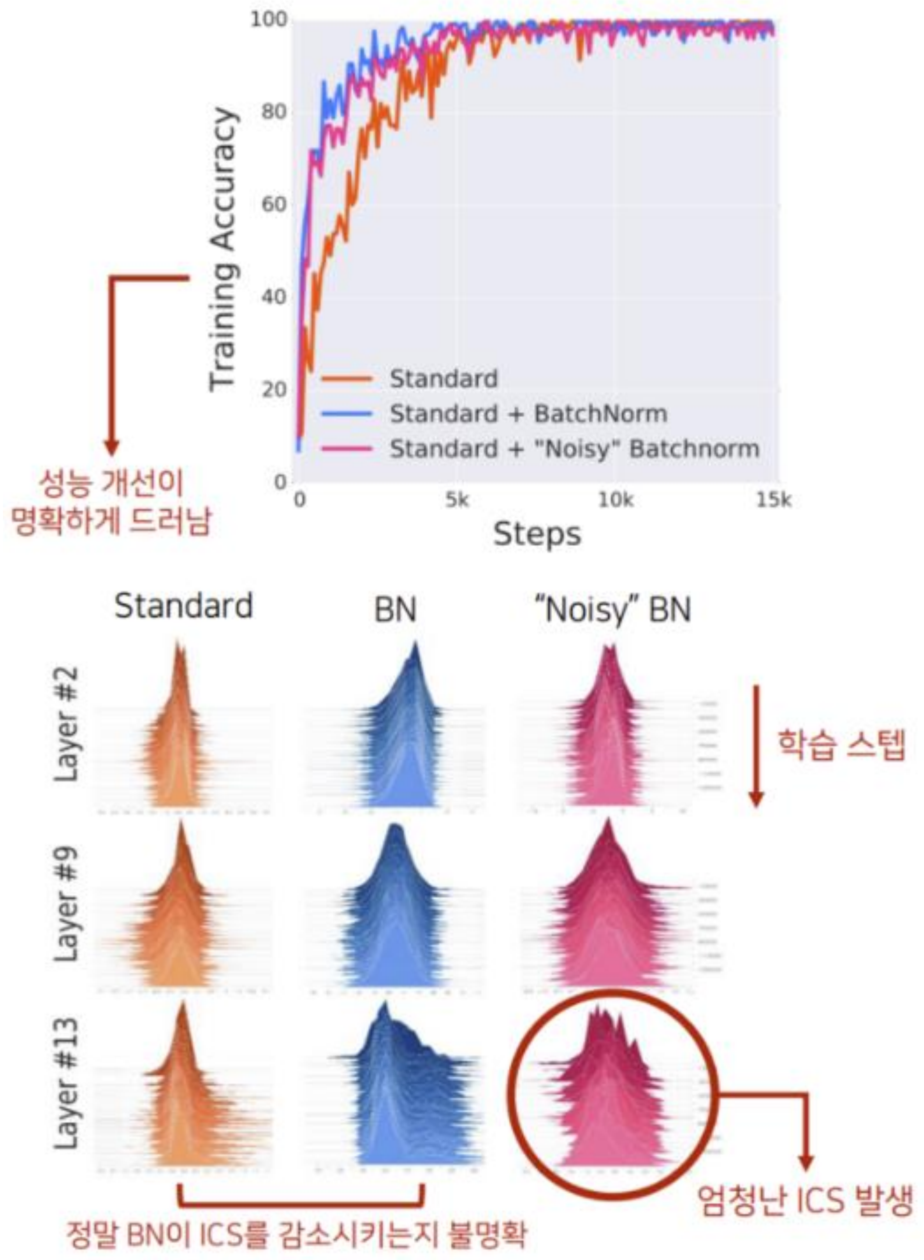


Gradient of larger parameter dominates the update

Both parameters can be updated in equal proportions

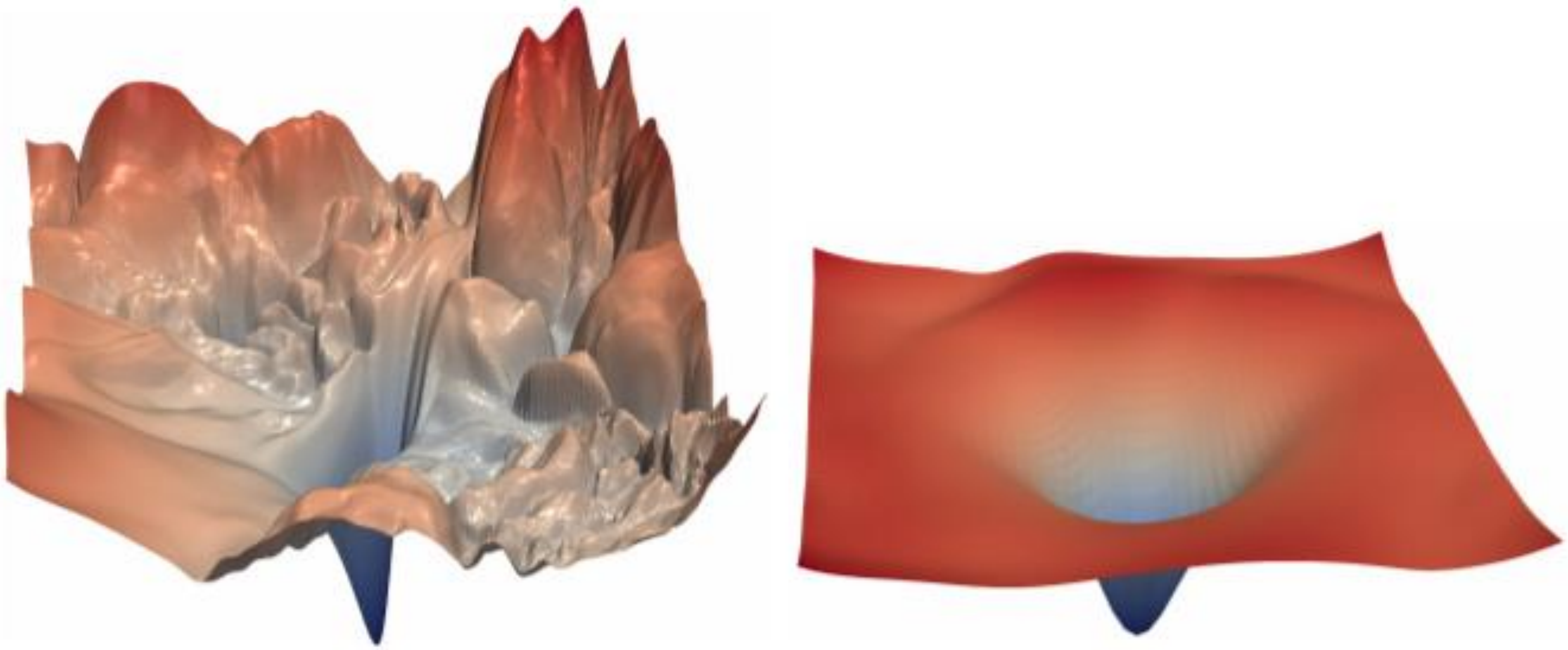
BN의 효과는 굉장했다! 그러나 ..





결과적으로 BN이 ICS를 해결하지는 못했음

사실은 Smoothing 효과 때문임



- Optimization Landscape를 부드럽게 만드는 효과가 있다.
- 위 그림과 같이 Smoothing효과로 global optim에 도달하기 쉬워진 것

감사합니다 !