exam 개인학습

Lec 6

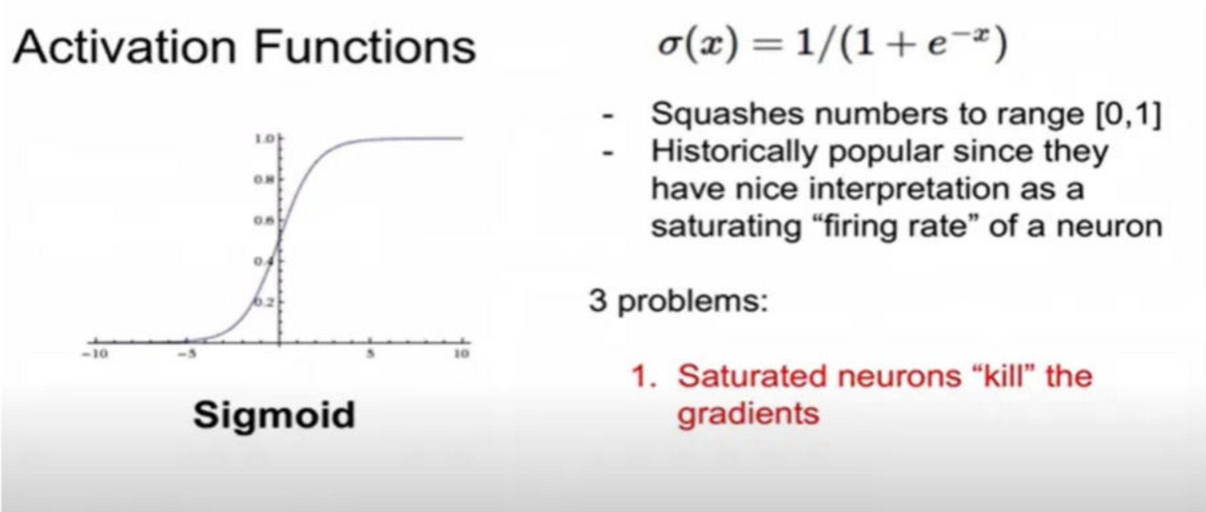
AI summary ✨ Try on this page

Type something...

01) Training Neural Networks 1

01_Activation Functions

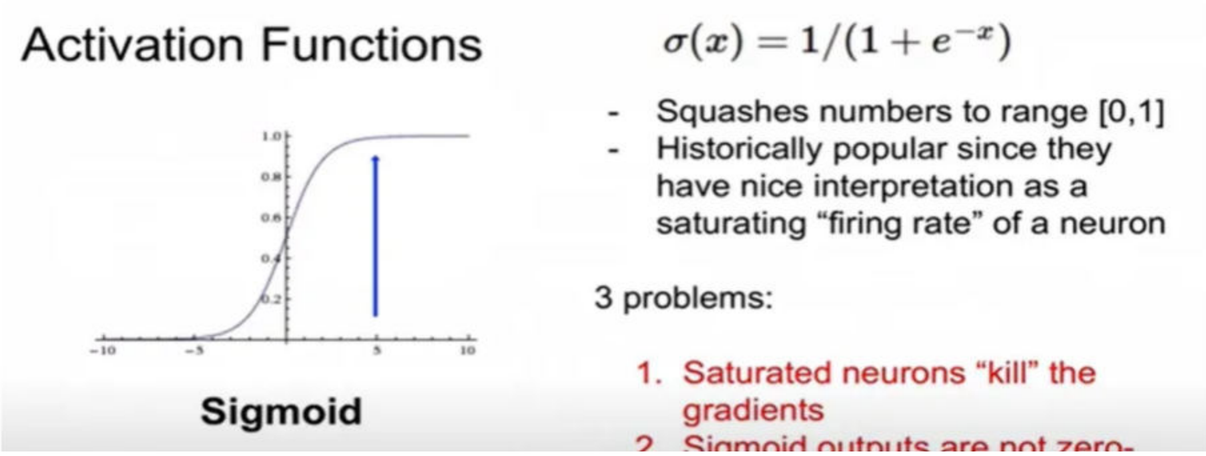
1. Sigmoid functions



- Sigmoid function은 지금은 잘 사용되지 않는 함수이다.

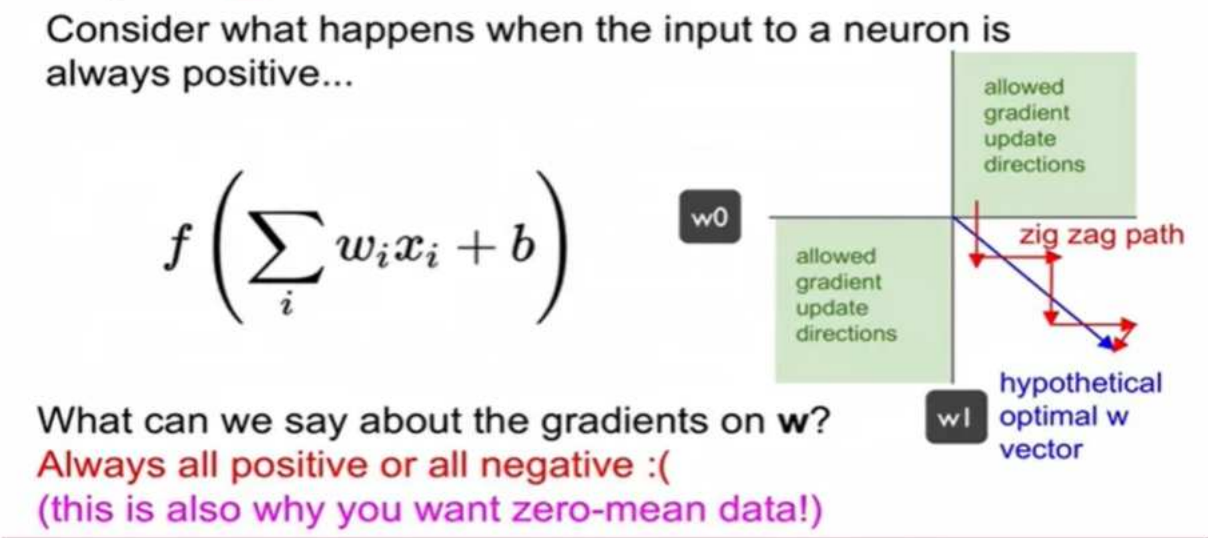
1. 그 이유는 gradient vanish 문제가 있기 때문이다.

- x값이 꽤 크거나 꽤 작으면 local gradient가 0에 가깝게 된다.

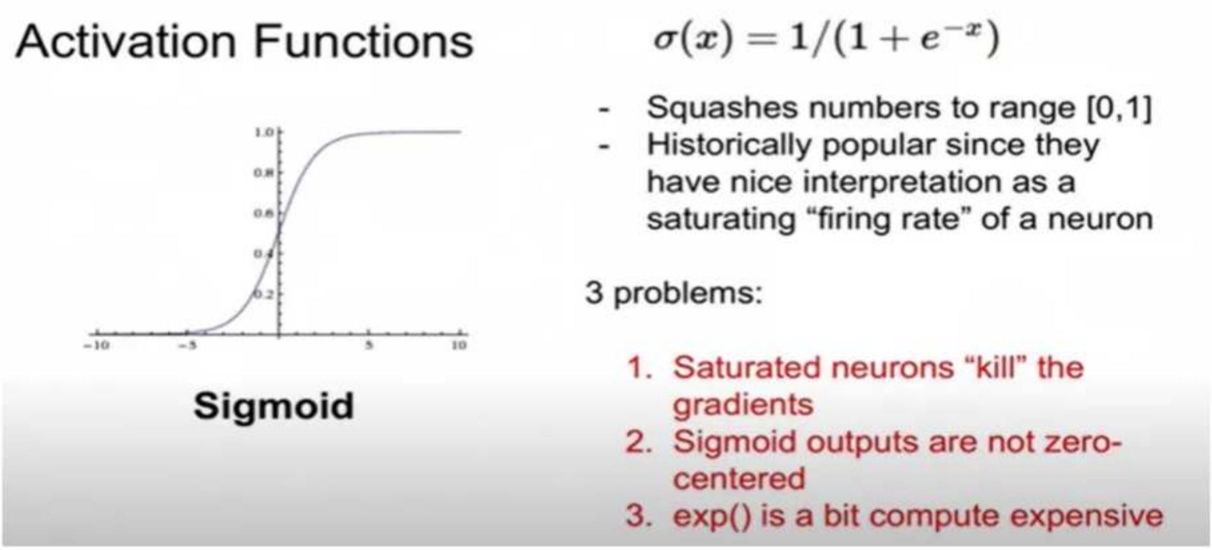


2. 두번째 문제는 output이 0이 중심이지 않아 수렴하는데 오래 걸린다.

- 만약 x가 양수라면 w에 대한 gradient가 모두 `dL/df` 의 부호가 같아지기 때문에 모두 양수이거나 모두 음수이다.

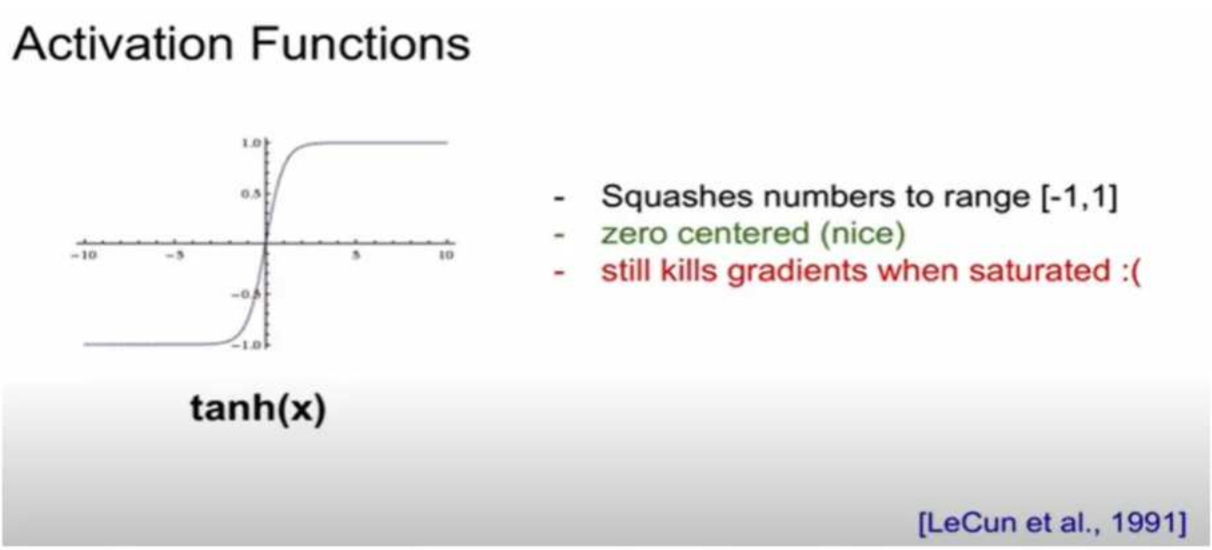


- zero-centered가 아니기 때문에 부호가 하나로 정해져서 수렴 속도가 느려지게 된다.



3. 마지막 문제는 exp() 연산이 매우 비싼 연산이라는 점이다.

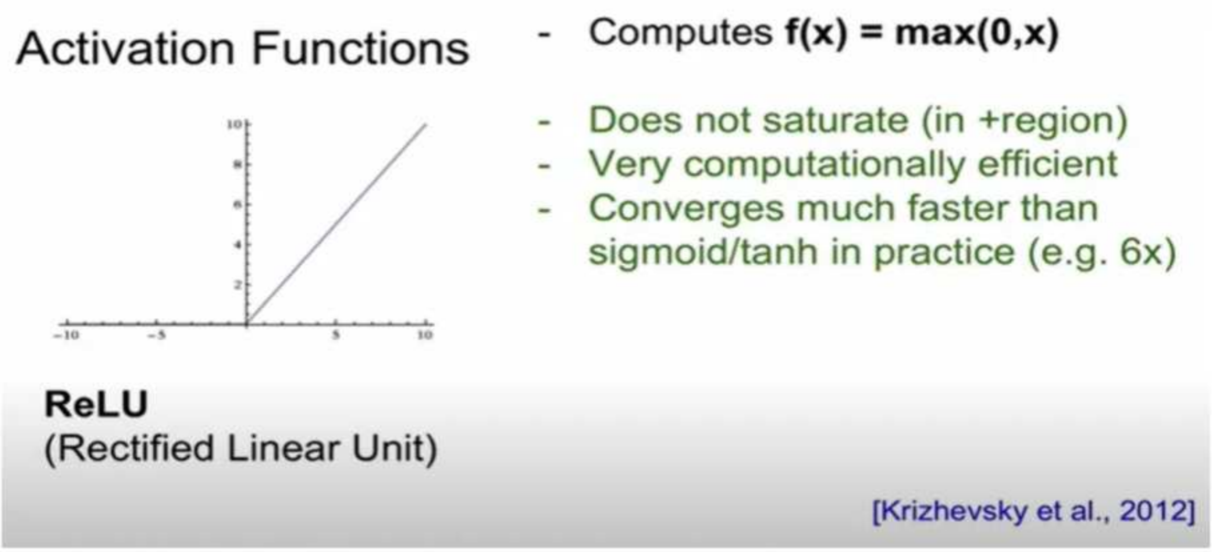
2. tanh(x)



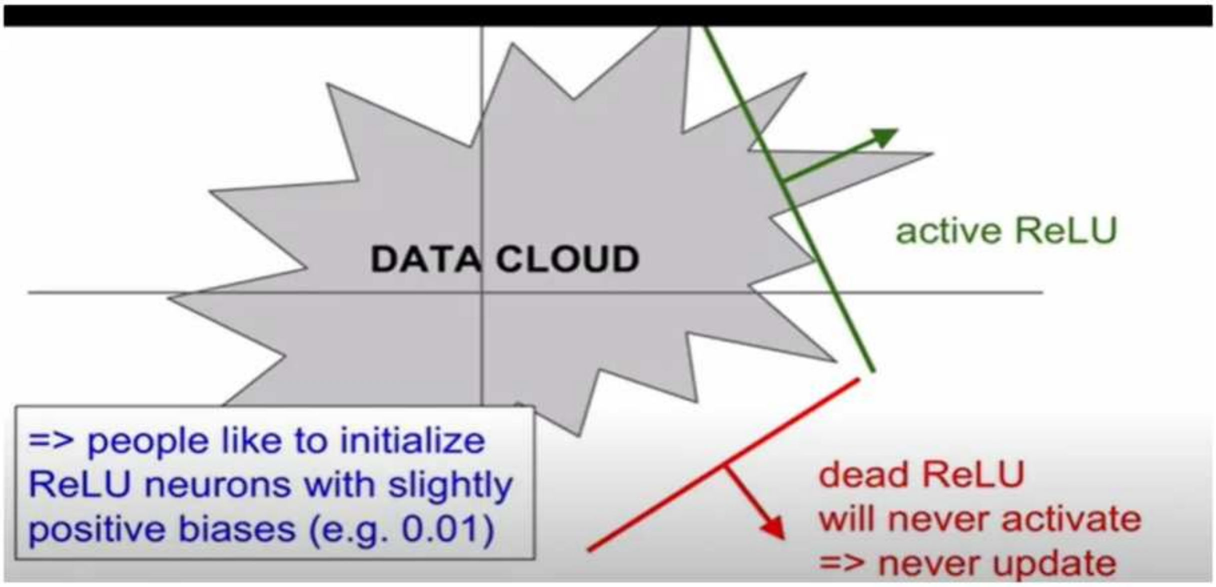
- sigmoid의 문제점을 해결하기 위해 zero-centered된 함수를 만들었다.

- 하지만 이것 또한 gradient vanish 문제를 피할 수 없었다.

3. ReLU = max(0, x)

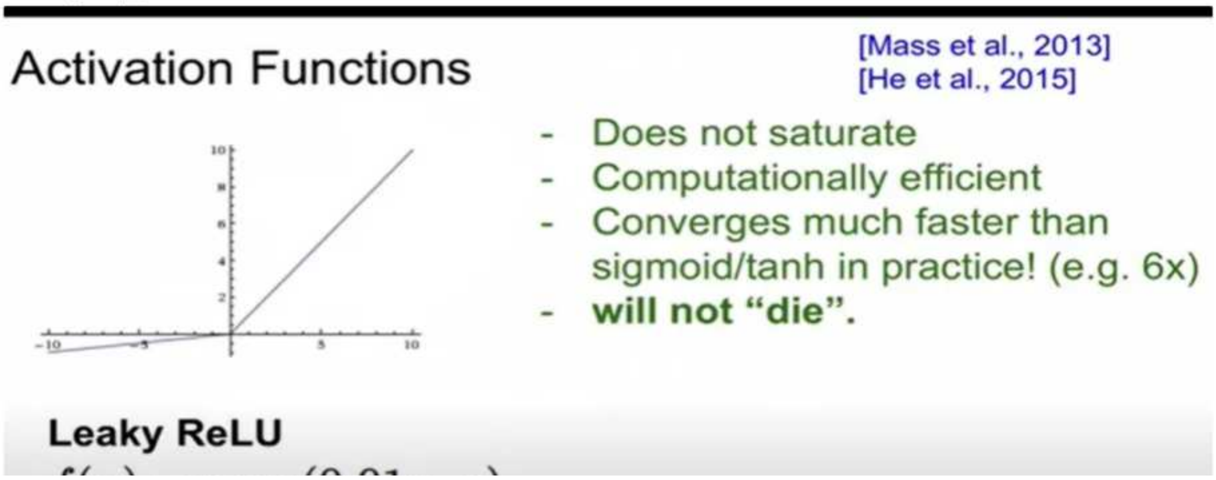


- 연산이 매우 효율적이고 수렴 속도가 빠르다.
- 현재 기본적인 activation function으로 사용되고 있다.



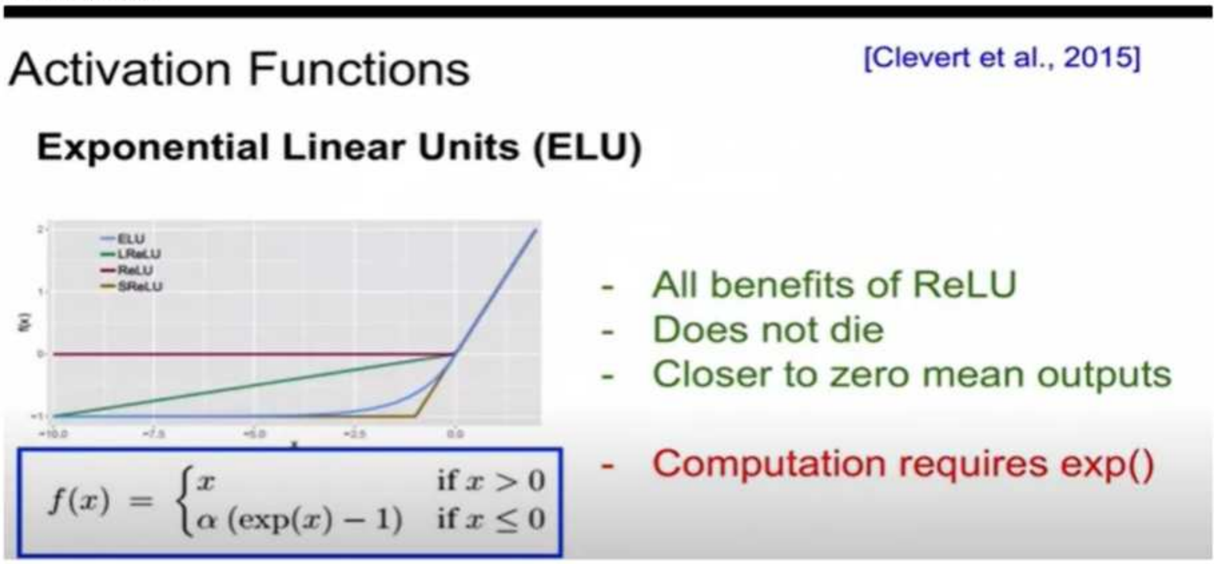
- 음수인 부분은 gradient가 0이기 때문에 update가 되지 않는 문제를 가지고 있다.
- learning_rate가 너무 큰 경우에도 발생한다.
- 그래서 사람들이 약간 작은 양수로 초기화 하는 것을 좋아한다.

4. Leaky ReLU = max(0.01x, x)



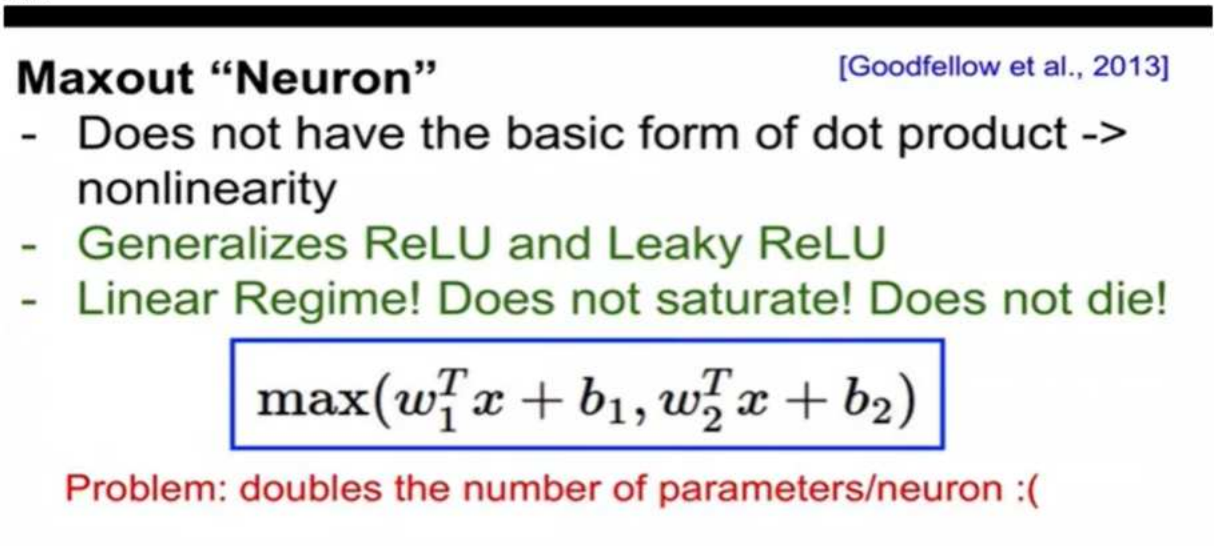
- 기울기가 죽지도 않는 좋은 함수로 보인다.
- 하지만 무조건 ReLU보다 낫다고는 말할 수 없다.

5. Exponential Linear Units(ELU)

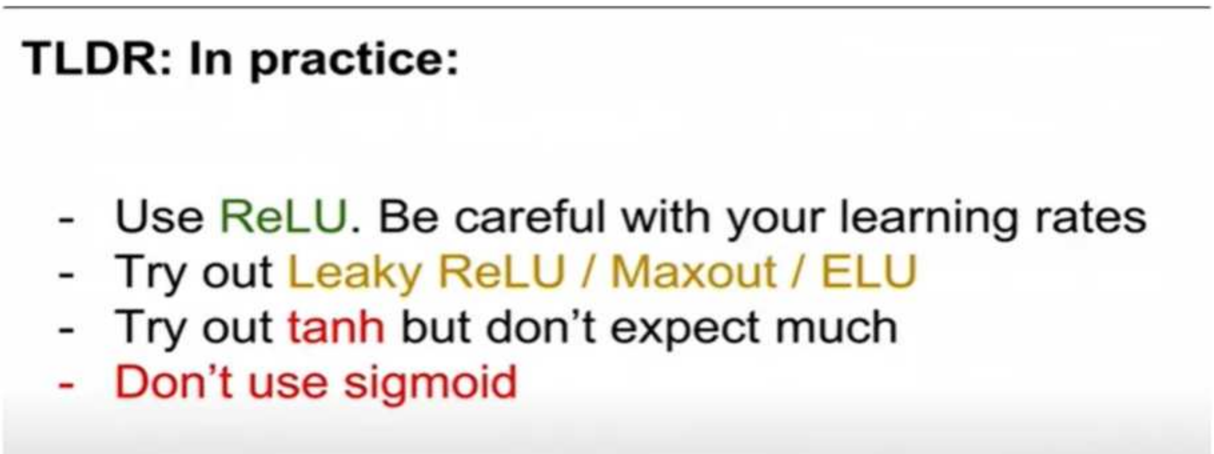


- ReLU의 장점을 가지면서 zero mean을 가지지만 exp 연산이 비싸다는 단점이 있다.

6. Maxout Neuron

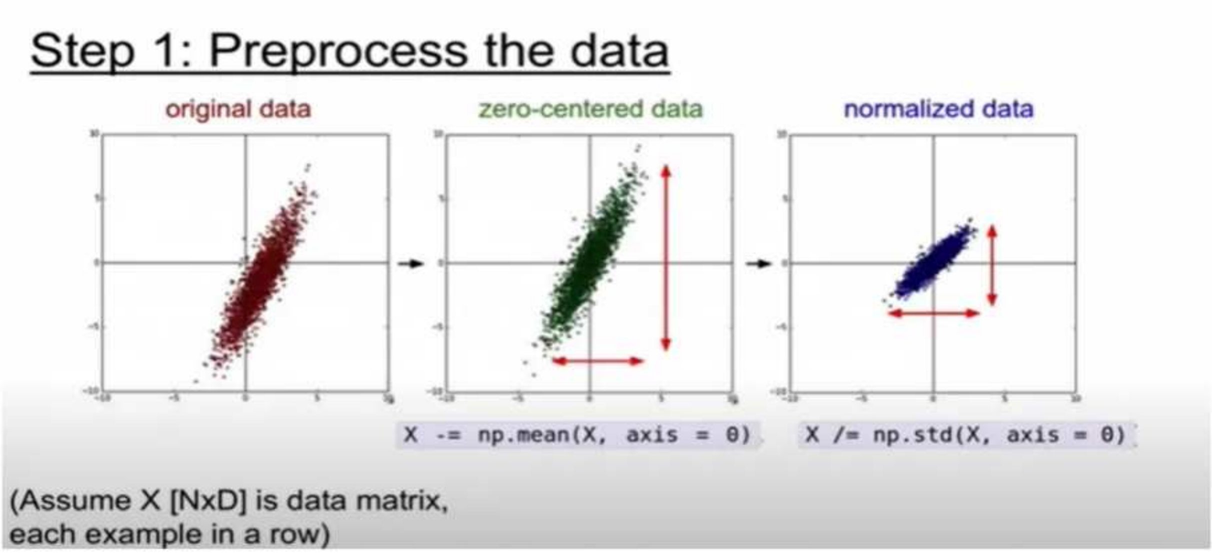


- ReLu와 Leaky ReLu를 일반화한 함수라고 볼 수 있지만 parameter와 neuron이 2배로 든다는 단점이 있다.

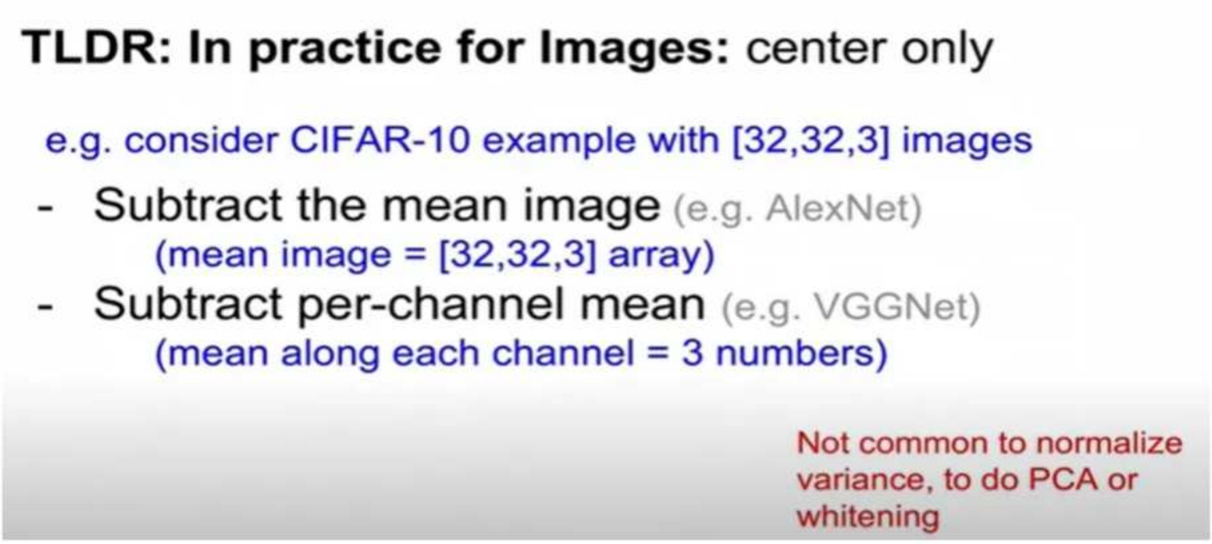


- 따라서 일반적으로는 ReLU를 사용하지만 Leaky ReLU/Maxout/ELU를 시도해 볼 수 있다.
- 하지만 sigmoid는 사용하지 마라.

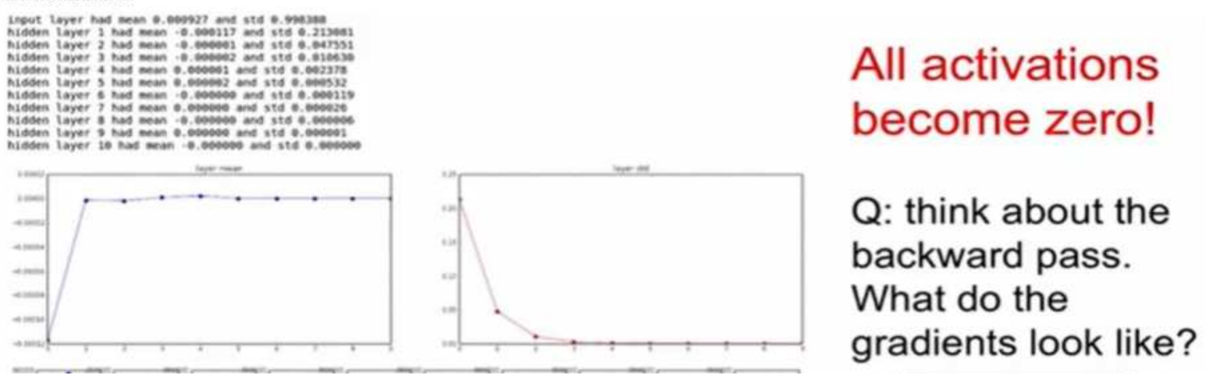
02_Data Preprocess

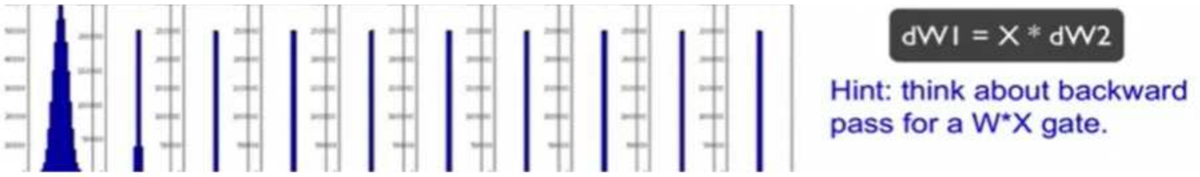


- zero-centered + normalized
- 이미지에서는 값이 [0~255] 사이의 값을 가지기 때문에 굳이 normalized를 하지 않는다.

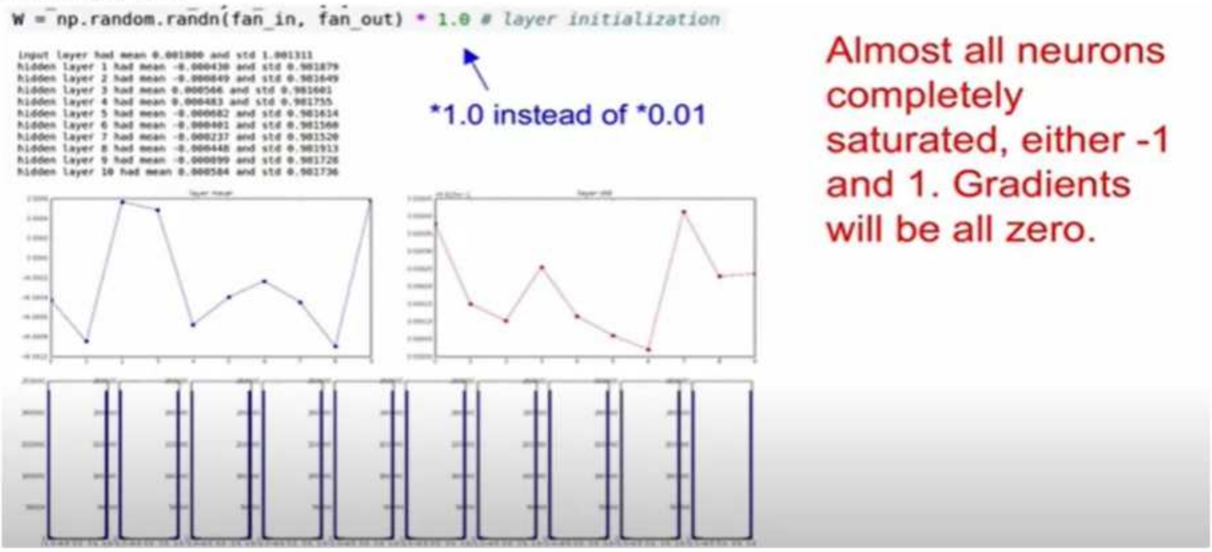


03_Weight Initialization



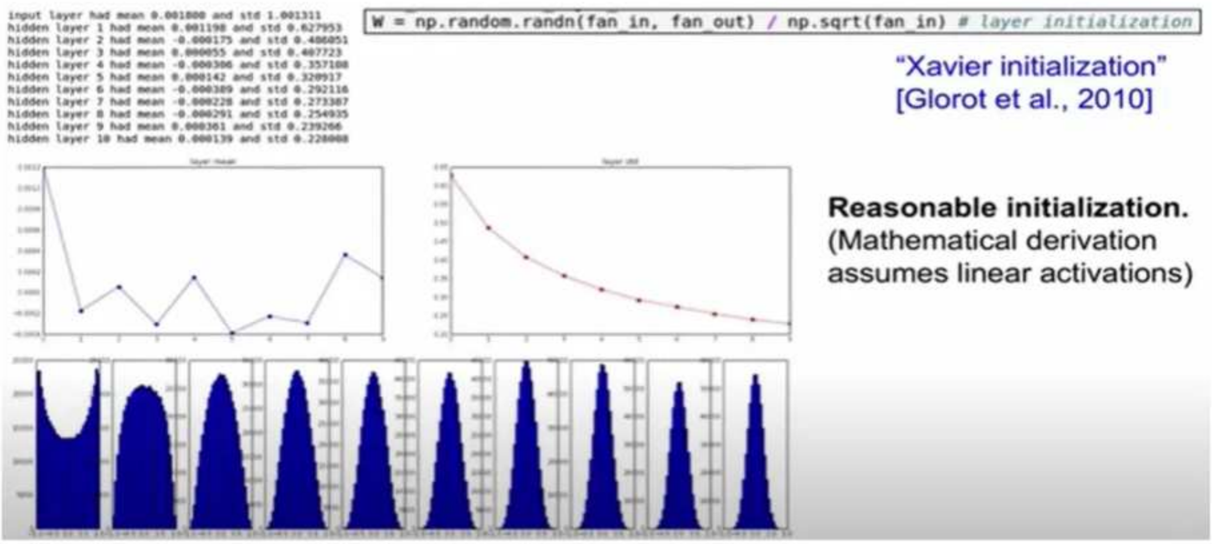


- 랜덤으로 값을 초기화 한 후 학습을 시켜보니 activation이 0으로 가는 문제가 발생한다.
- 그 결과 gradient vanishing이 발생한다.

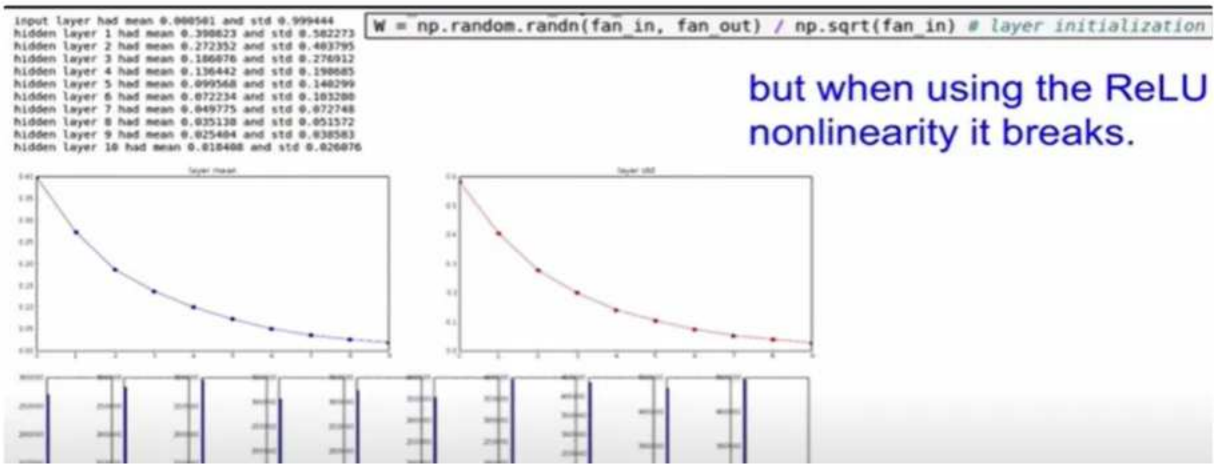


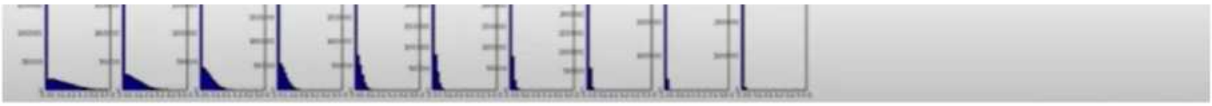
- 1로 초기화 하게 되면 값이 튀어서 -1 또는 1로 포화된다.

1. Xavier Initalization



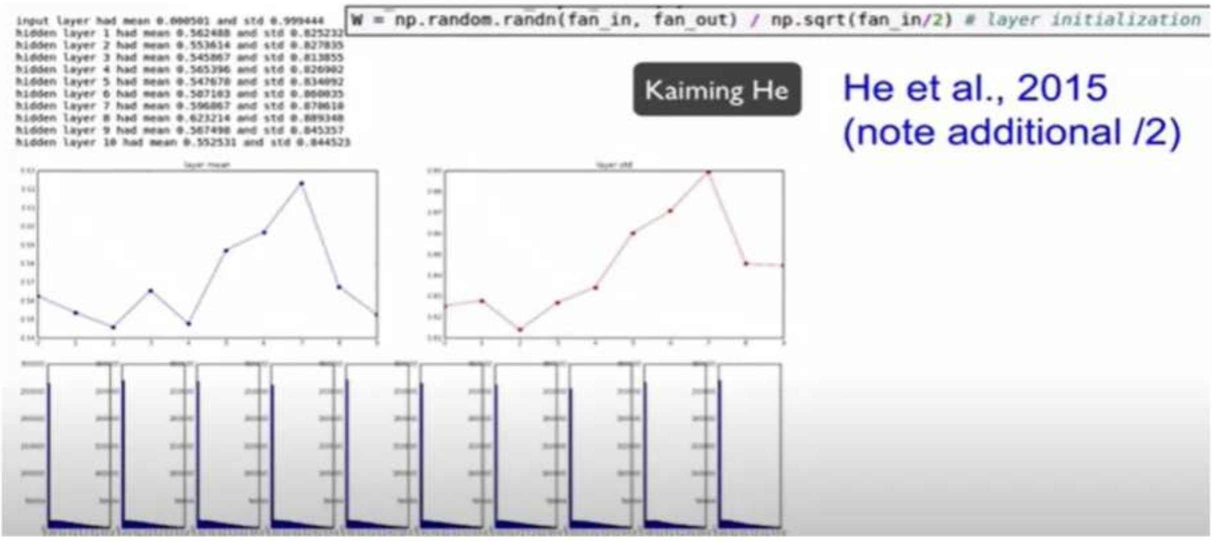
- Xavier initialization 방식은 매우 합리적인 방식으로 linear activation에 많이 쓰는 초기화 방식이다.





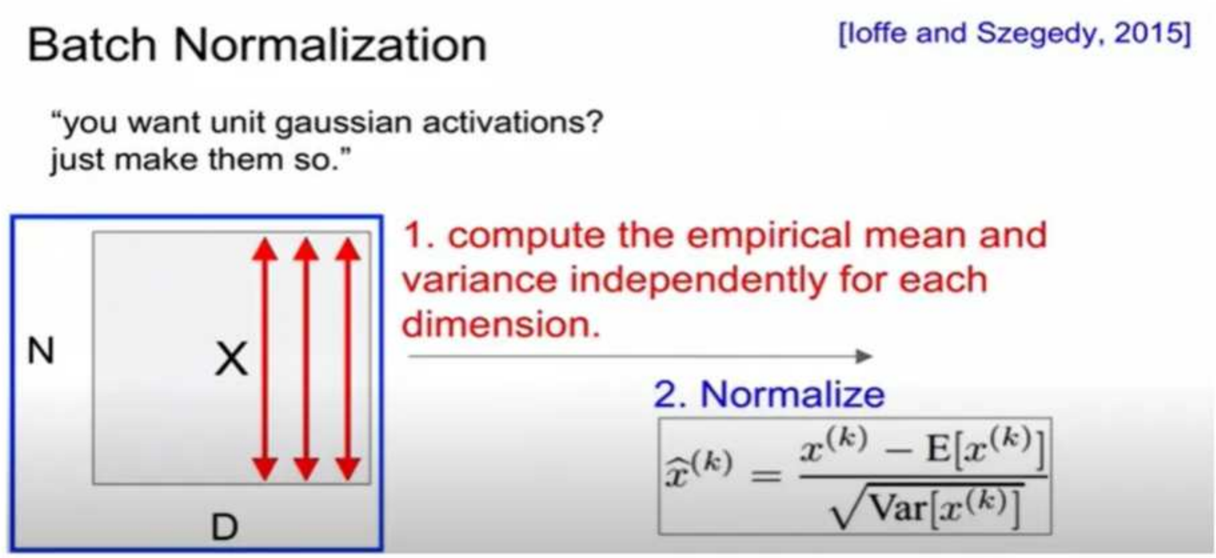
- 하지만 ReLU같은 nonlinearity 함수에는 잘 적용되지 못했다.

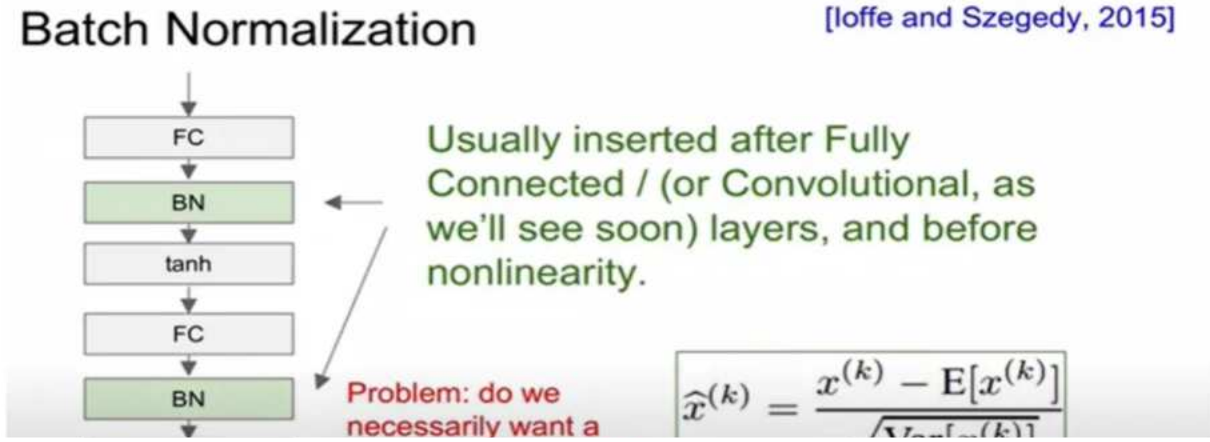
2. He Initialization



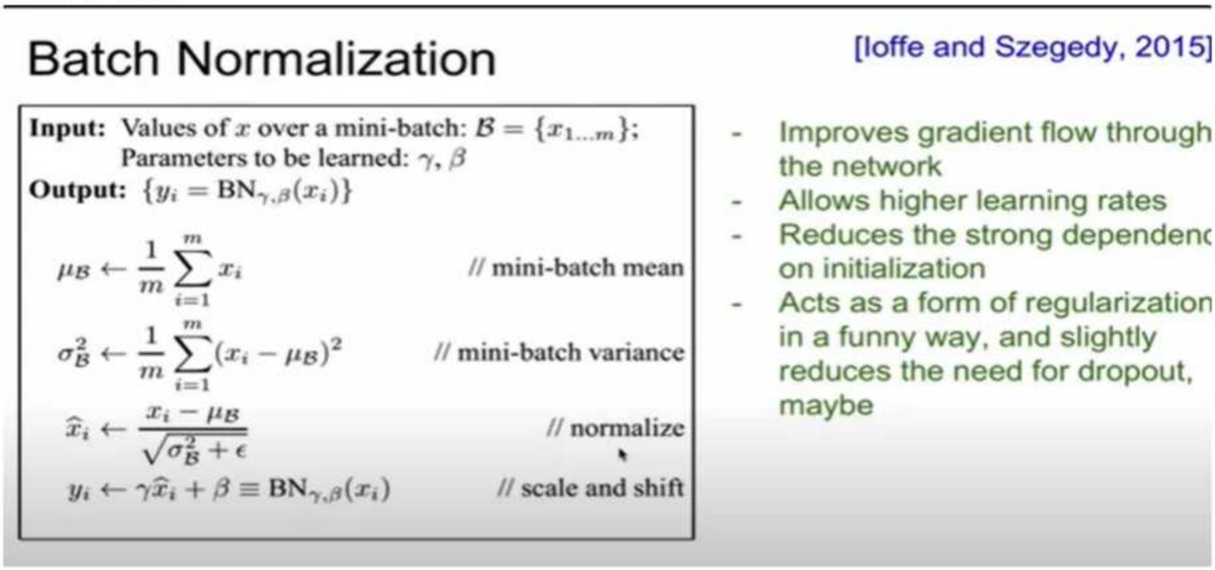
- ReLU에는 He initialization을 적용하면 잘 동작하는 것을 확인할 수 있다.

04_Batch Normalization

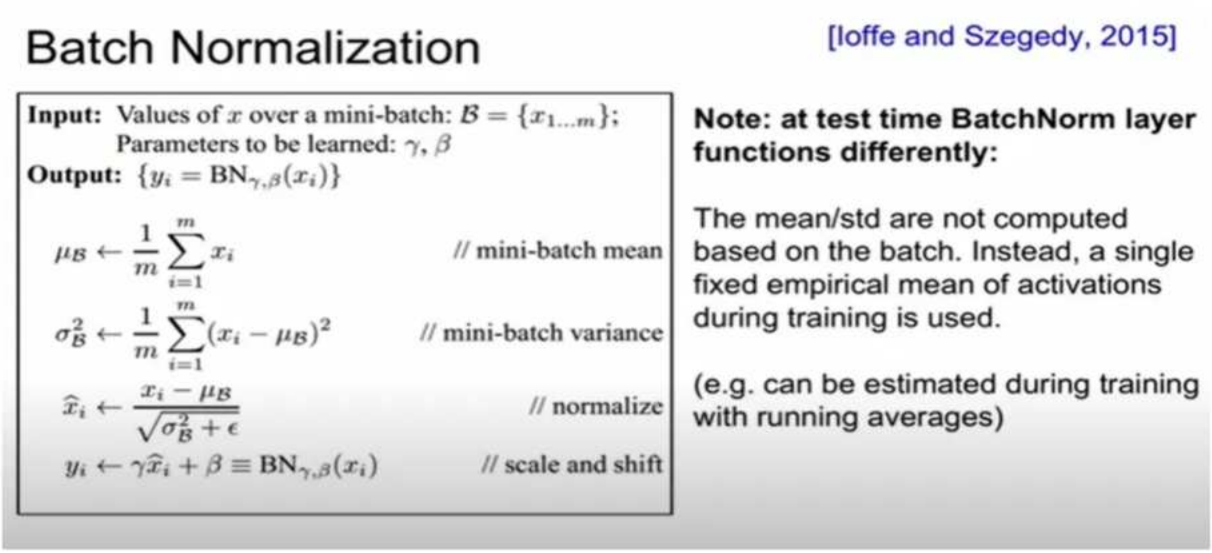




- FC - BN - activation 순서로 적용한다.

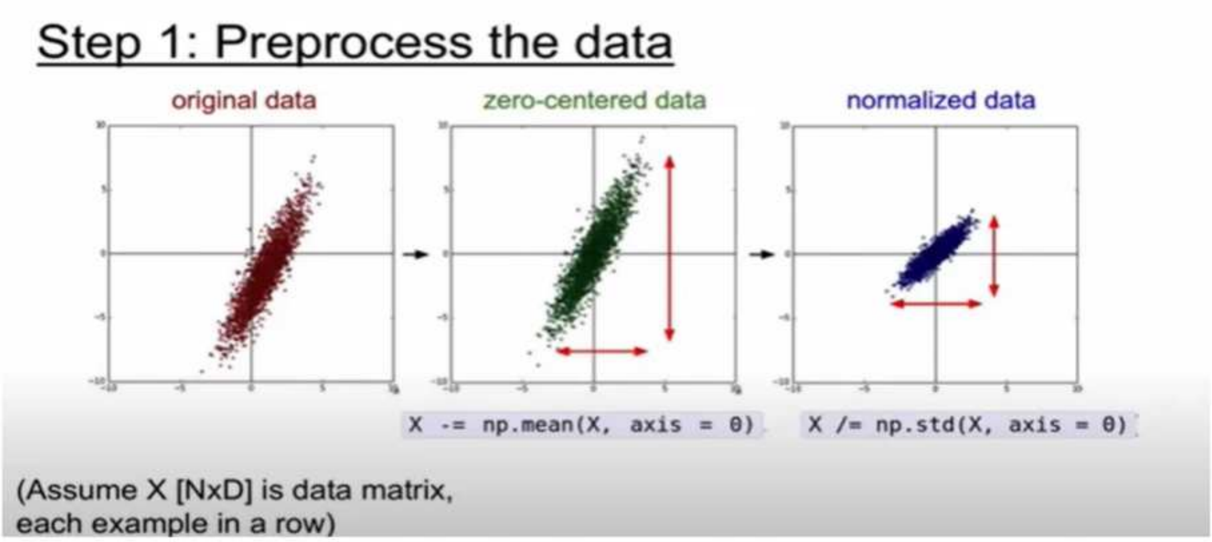


- 자세한 과정은 과제를 참고하도록 하자.

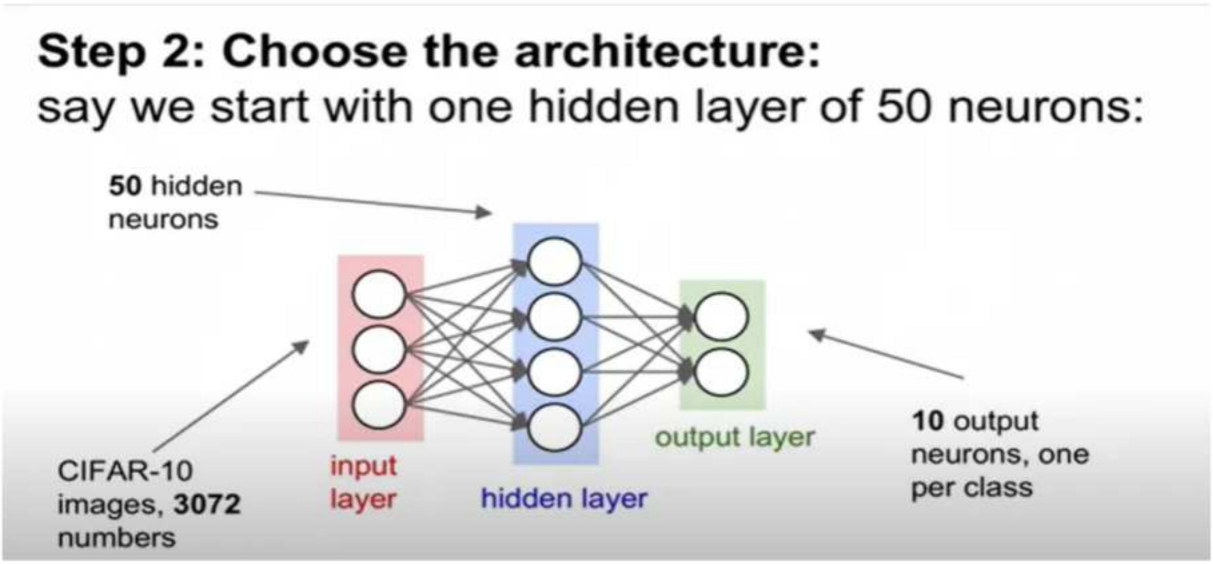


- test를 할 때는 미리 mean과 variance를 저장해둔 것을 활용한다.

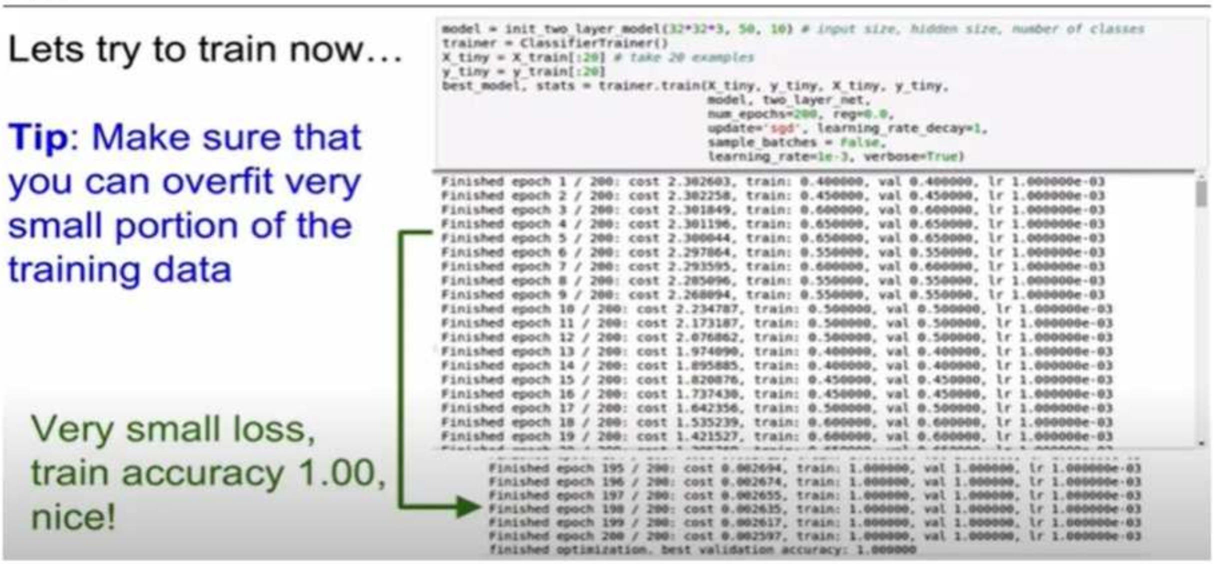
05_Babysitting the learning process



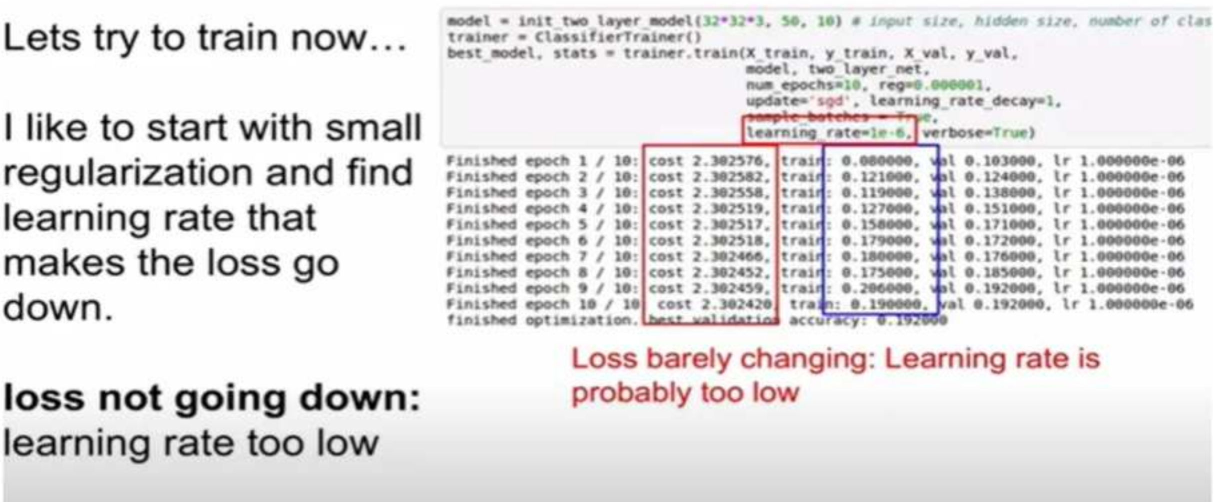
1. zero centered



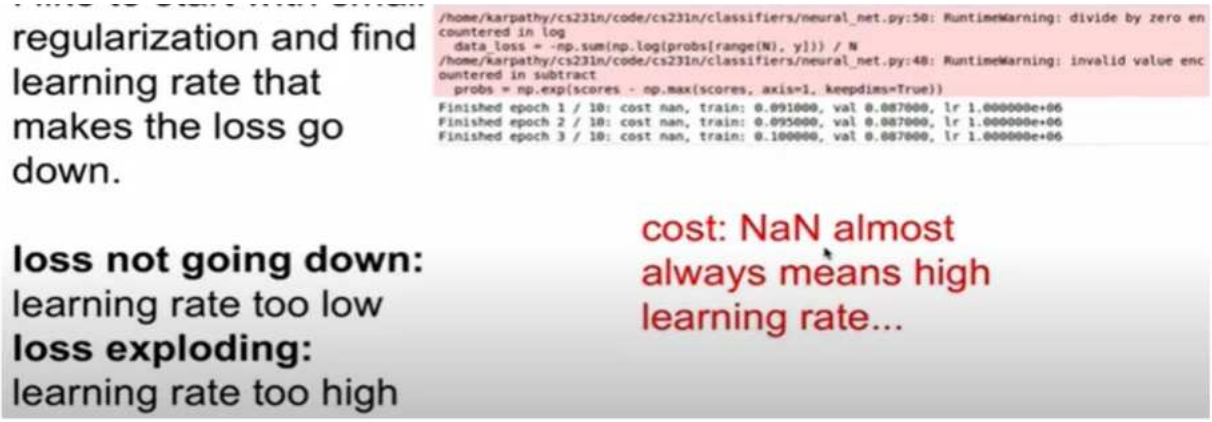
2. Choose the architecture



- 작은 데이터를 취한 후에 학습을 시키면 무조건 overfitting이 나와야 한다.
- 이것이 모델이 잘 동작한다는 의미이다.

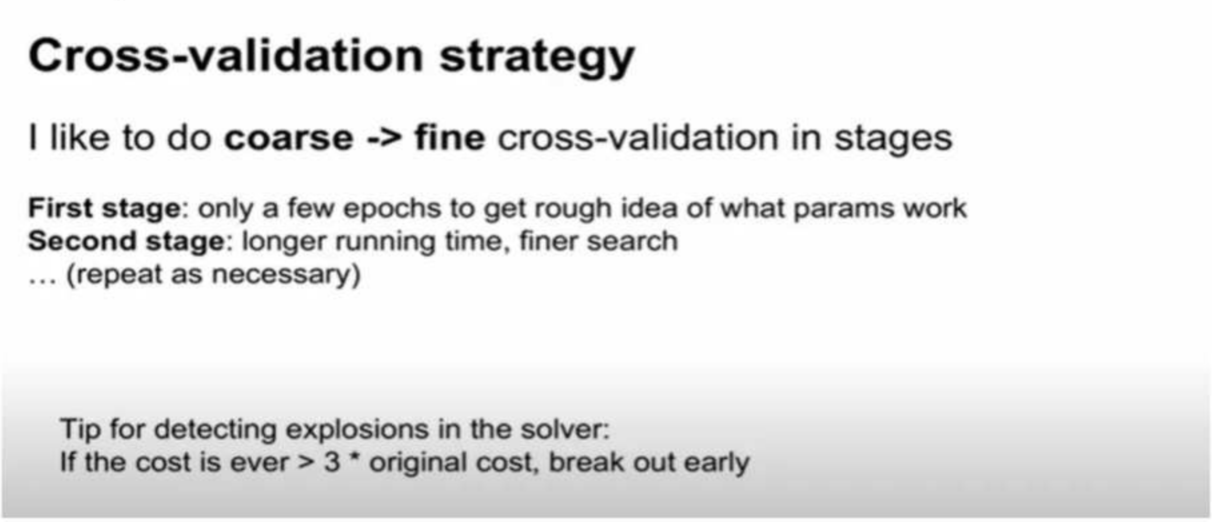


- Learning_rate가 너무 작으면 loss가 줄어들지 않는다.

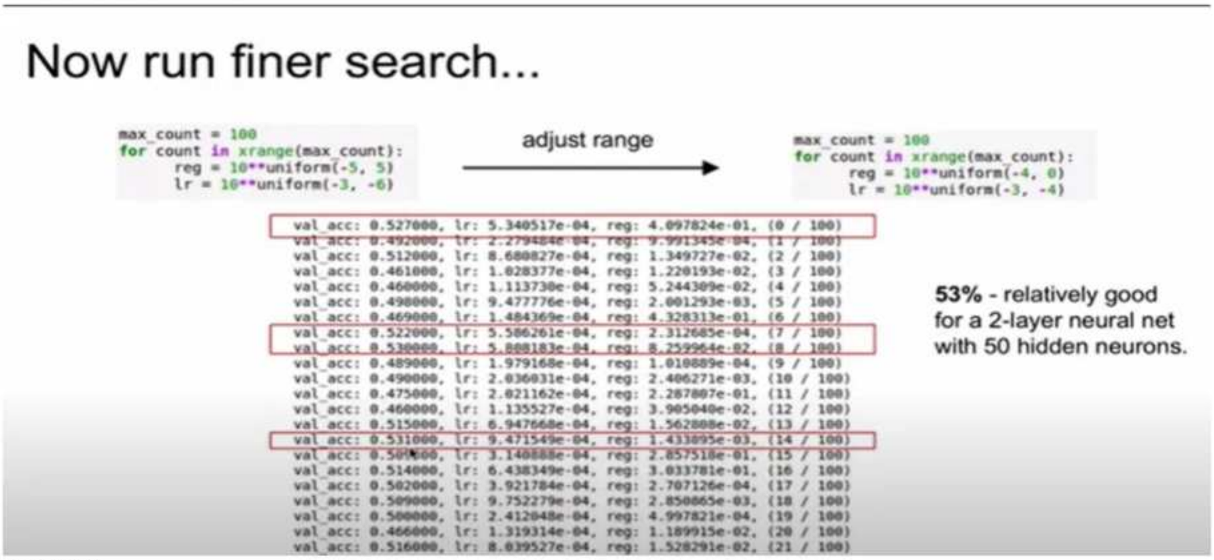


- Learning_rate가 너무 크면 loss가 exploding 해버린다.

06_Hyperparameter Optimization

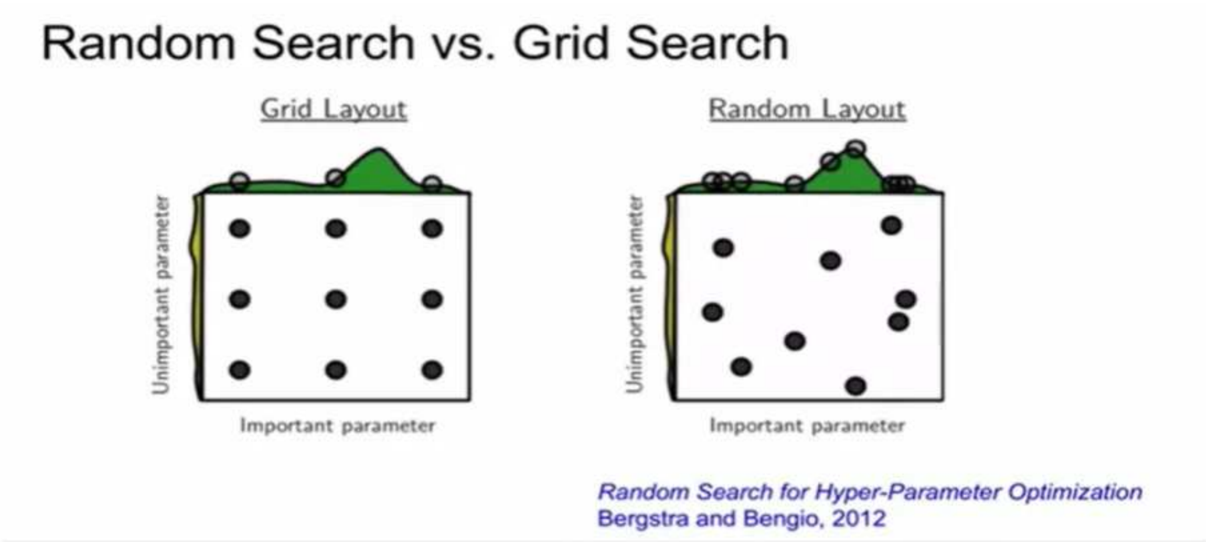


- epoch을 작게 한 후 나중에 세부적으로 조정한다.

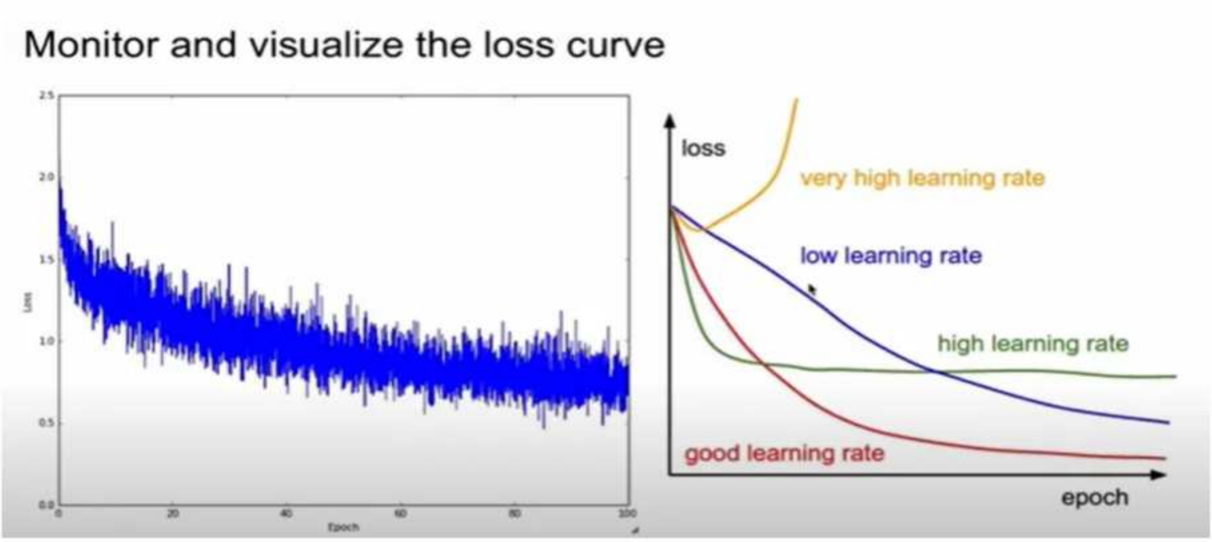


- 값을 점점 좁혀가면서 진행.
- boundary에 가까운 값이 최적이라면 범위를 다시 넓혀서 진행.

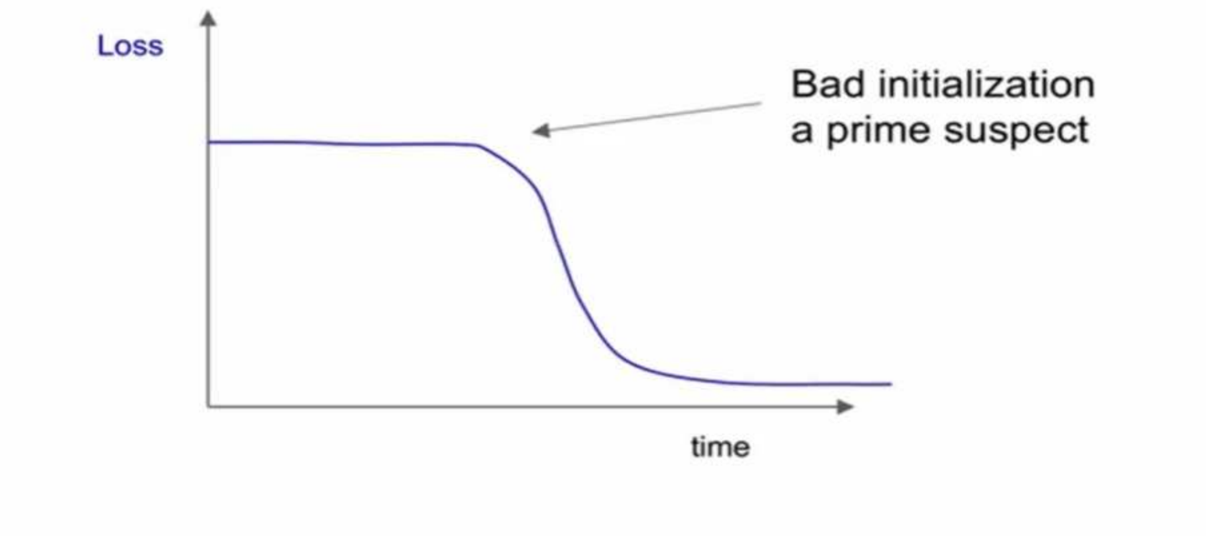
Generated by Notion to PDF



- grid 등간격으로 진행하면 우리가 찾고 싶은 최적의 parameter를 찾지 못할 수도 있다.
- 결국 최적화를 하기 위해서는 random하게 접근해야 한다.



- Loss curve를 통해서 좋은 learning_rate를 찾는다.



- Loss가 정체되는 것은 initialization이 잘못된 것을 의미한다.