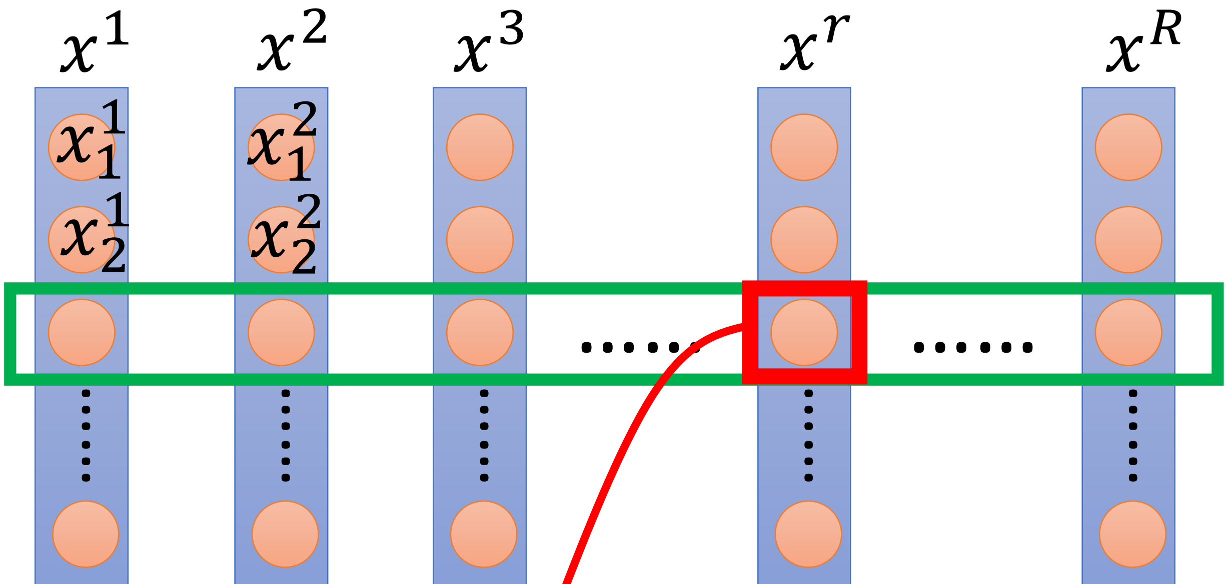
Batch Normalization

Feature Scaling

對每一個dimension做一次normalization

如果說沒有做normalization，則有些input較大所影響到的weight其gradient較大
造成loss surface不平均，這樣optimizer很難做optimization，做完normalization後每一個維度所影響的weight才能平均



For each dimension i:

mean: m_i

standard deviation: σ_i

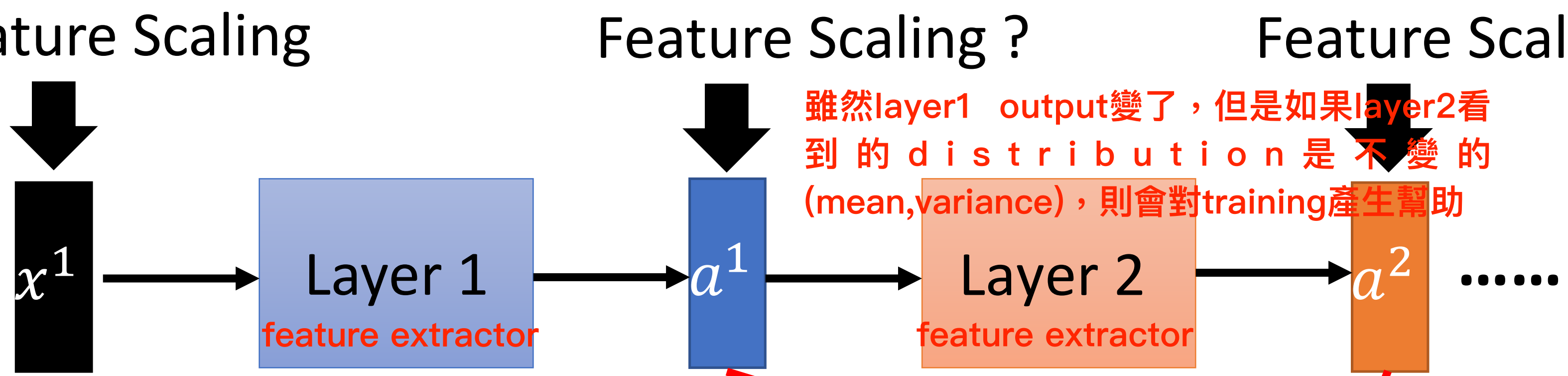
$$x_i^r \leftarrow \frac{x_i^r - m_i}{\sigma_i}$$

The means of all dimensions are 0, and the variances are all 1

In general, gradient descent converges much faster with feature scaling than without it.

How about Hidden Layer?

但是如果每一層都算一次normalization似乎不切實際，運算太大

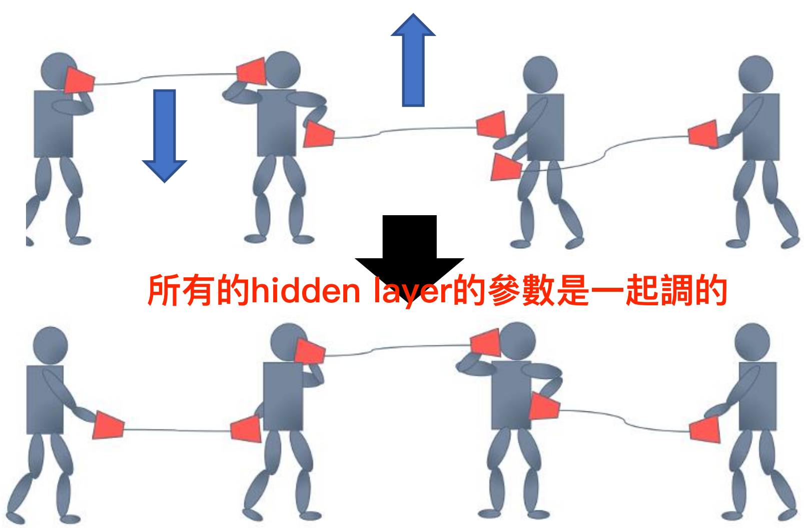


雖然layer1 output變了，但是如果layer2看到的distribution是不變的(mean,variance)，則會對training產生幫助

Difficulty: their statistics change during the training …

Batch normalization

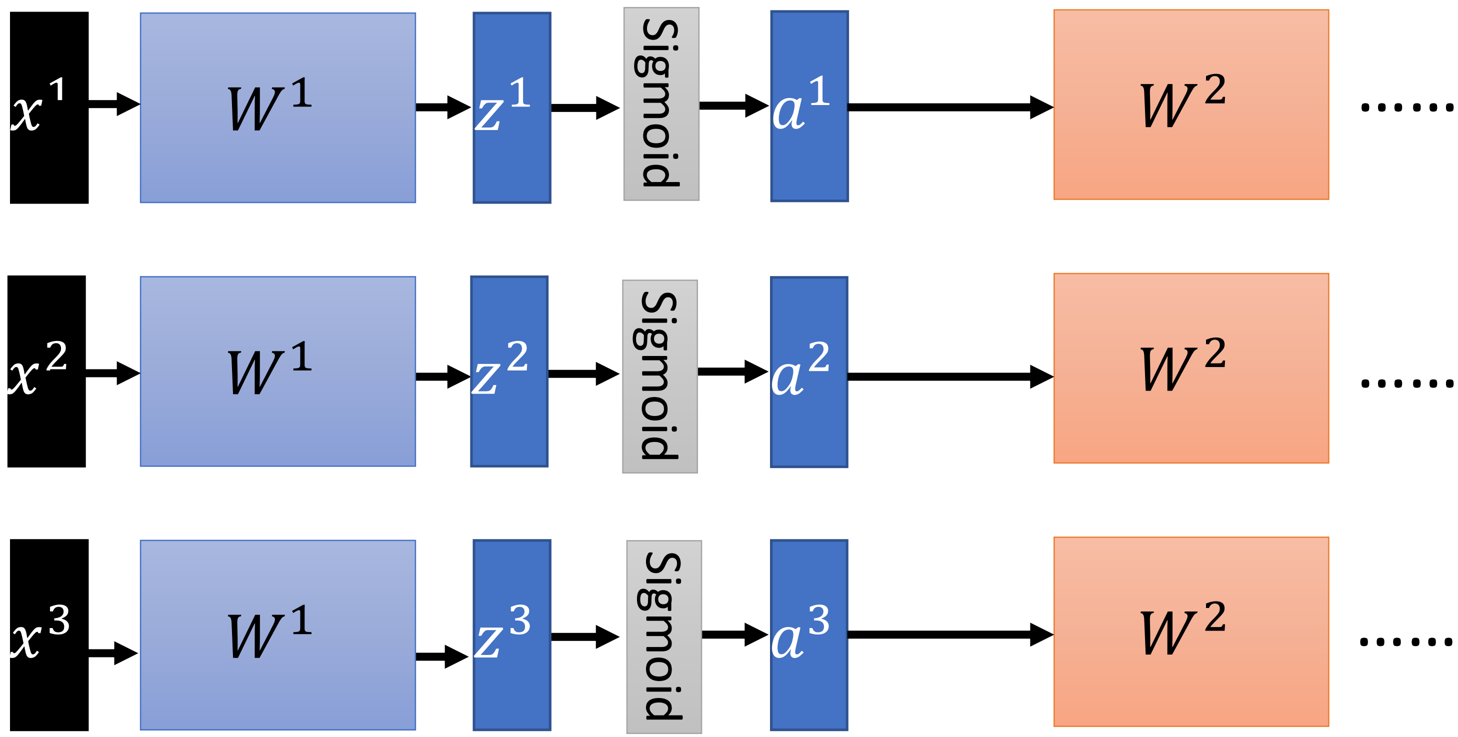
Smaller learning rate can be helpful, but the training would be slower.



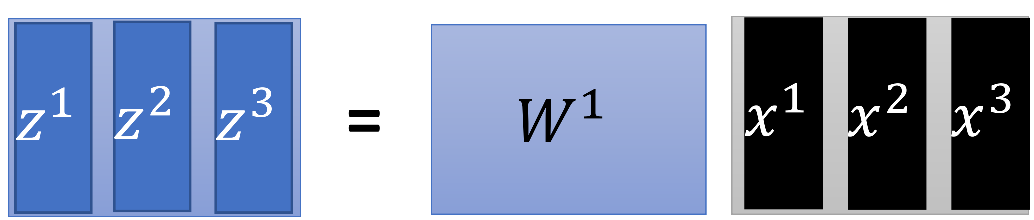
所有的hidden layer的參數是一起調的

Internal Covariate Shift

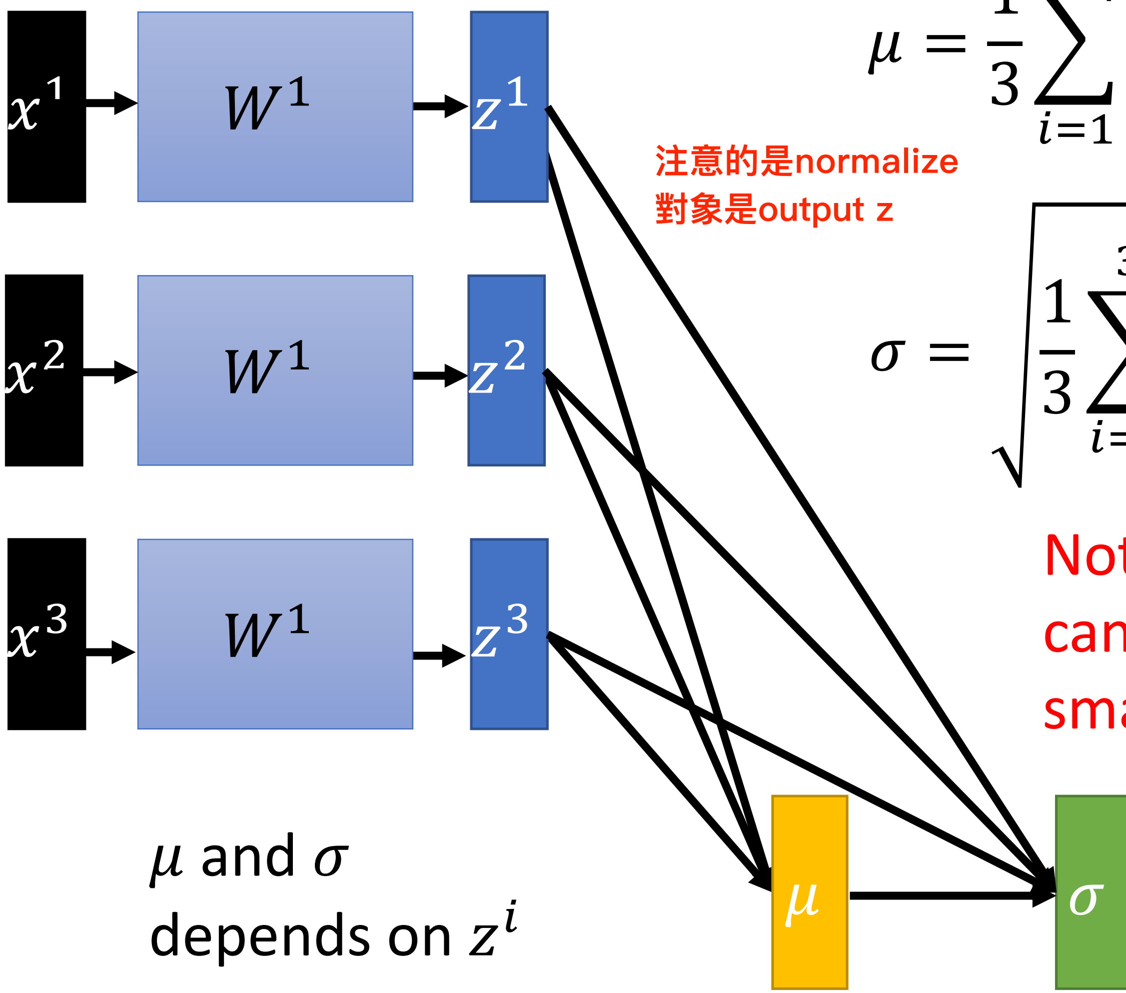
Batch



Batch 加快運算的速度



Batch normalization



$$\mu = \frac{1}{3}\sum_{i=1}^{3} z^i$$

注意的是normalize
對象是output z

$$\sigma = \sqrt{\frac{1}{3}\sum_{i=1}^{3}(z^i - \mu)^2}$$

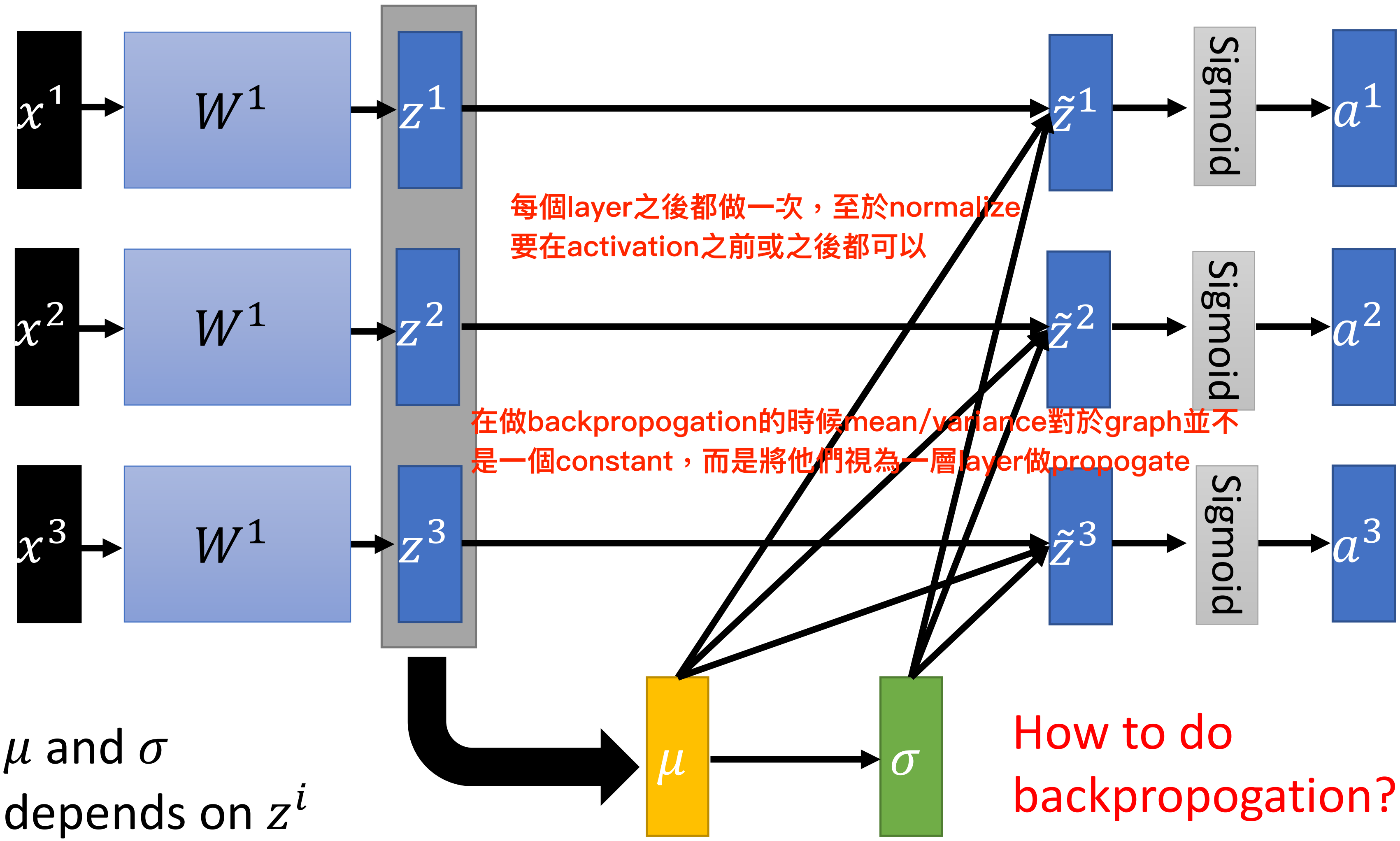
Note: Batch normalization cannot be applied on small batch.

因為mean, variance會算不準
找不到他真正的distribution

μ and σ
depends on z^i

Batch normalization

$$\tilde{z}^i = \frac{z^i - \mu}{\sigma}$$



μ and σ
depends on z^i

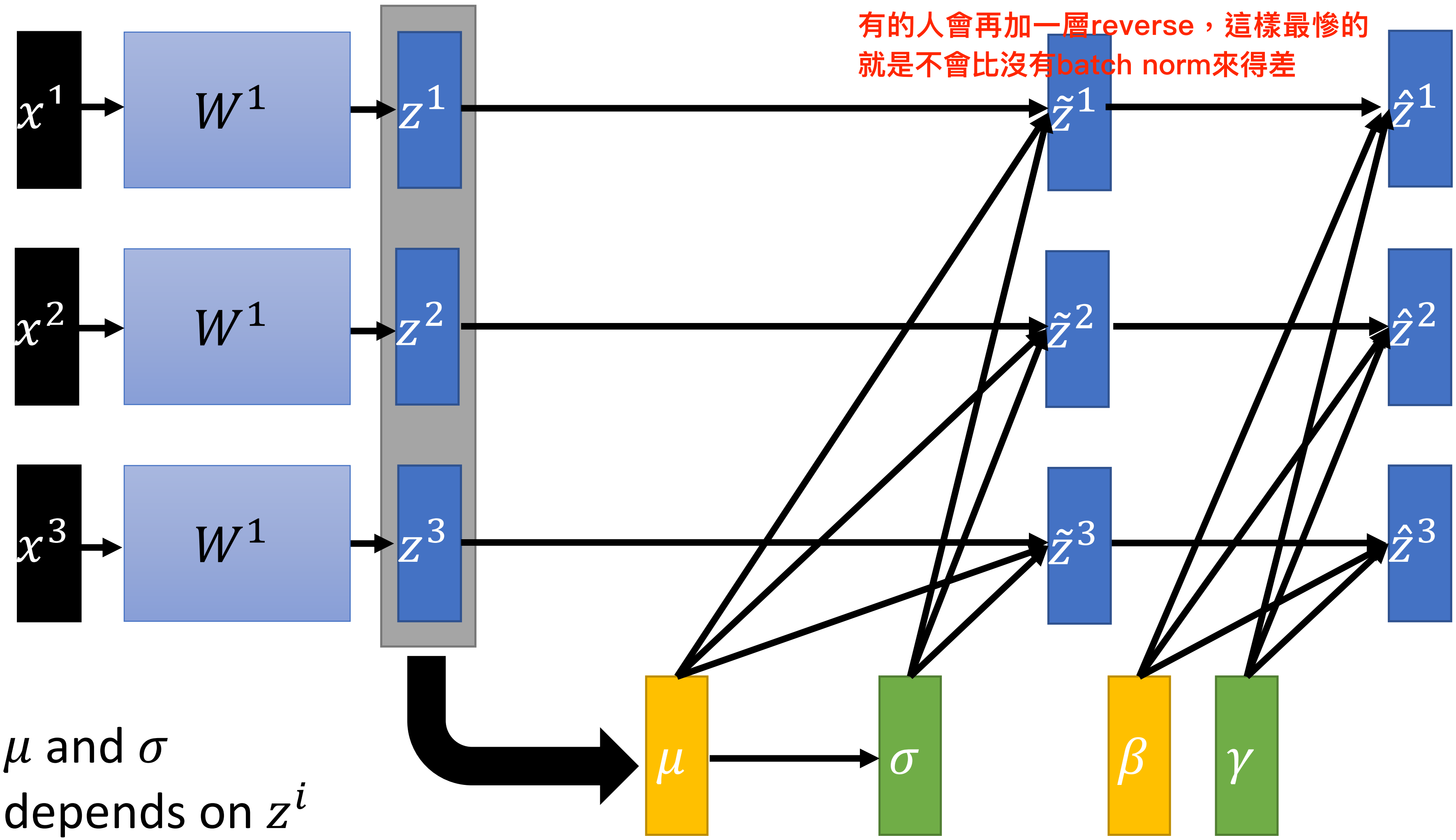
How to do
backpropogation?

Batch normalization

$$\tilde{z}^i = \frac{z^i - \mu}{\sigma}$$

$$\hat{z}^i = \gamma \odot \tilde{z}^i + \beta$$

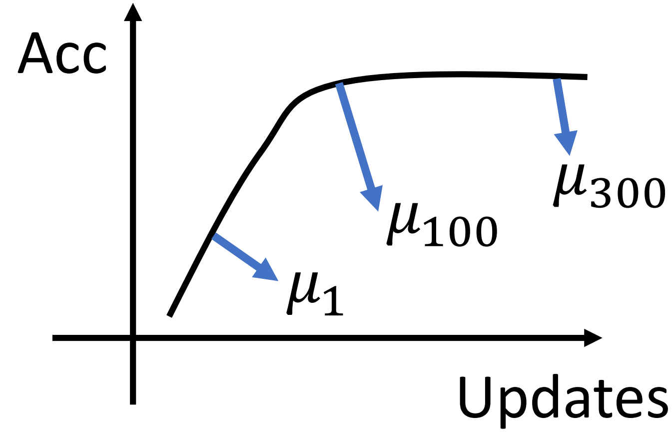
有的人會再加一層reverse，這樣最慘的
就是不會比沒有batch norm來得差



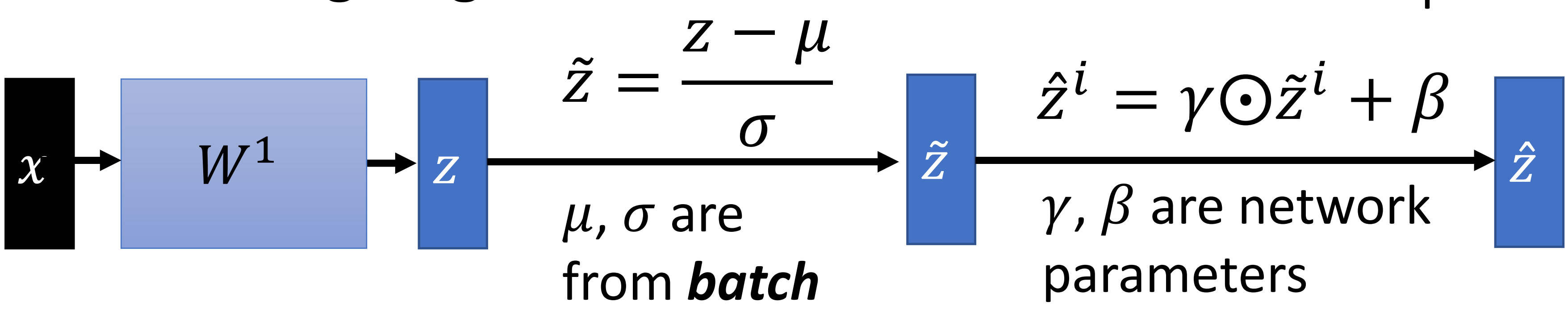
μ and σ
depends on z^i

initial= 0 initial= 1
machine自己決定要不要做reverse

Batch normalization



- At testing stage:



We do not have ***batch*** at testing stage.

Ideal solution:

將整個training set算mean/variance並對testing data norm，但實作會有問題(mem不夠之類的)

Computing μ and σ using the whole training dataset.

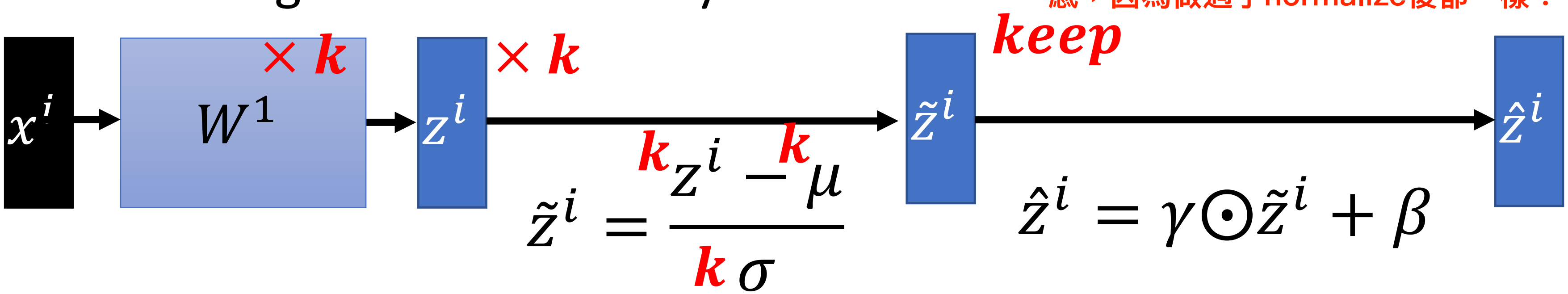
Practical solution:

但會造成一開始的mean跟最後幾個epoch的mean差太多，造成noise
可能的做法是去掉前幾個跟最後幾個epoch的mean/variace，取中間的

Computing the moving average of μ and σ of the batches during training.

Batch normalization - Benefit

- BN reduces training times, and make very deep net trainable.
  - Because of less Covariate Shift, we can use larger learning rates.
  - Less exploding/vanishing gradients 如果對activation的input就先做batch norm，可以對抗gradient vanishing
    - Especially effective for sigmoid, tanh, etc.
- Learning is less affected by initialization. network對於weight的initial沒有那麼敏感，因為做過了normalize後都一樣！



$$\tilde{z}^i = \frac{k z^i - k \mu}{k \sigma}$$

$$\hat{z}^i = \gamma \odot \tilde{z}^i + \beta$$

- BN reduces the demand for regularization.
  有一些regularization的功能，可以對抗overfitting，但是可以採用drop out即可

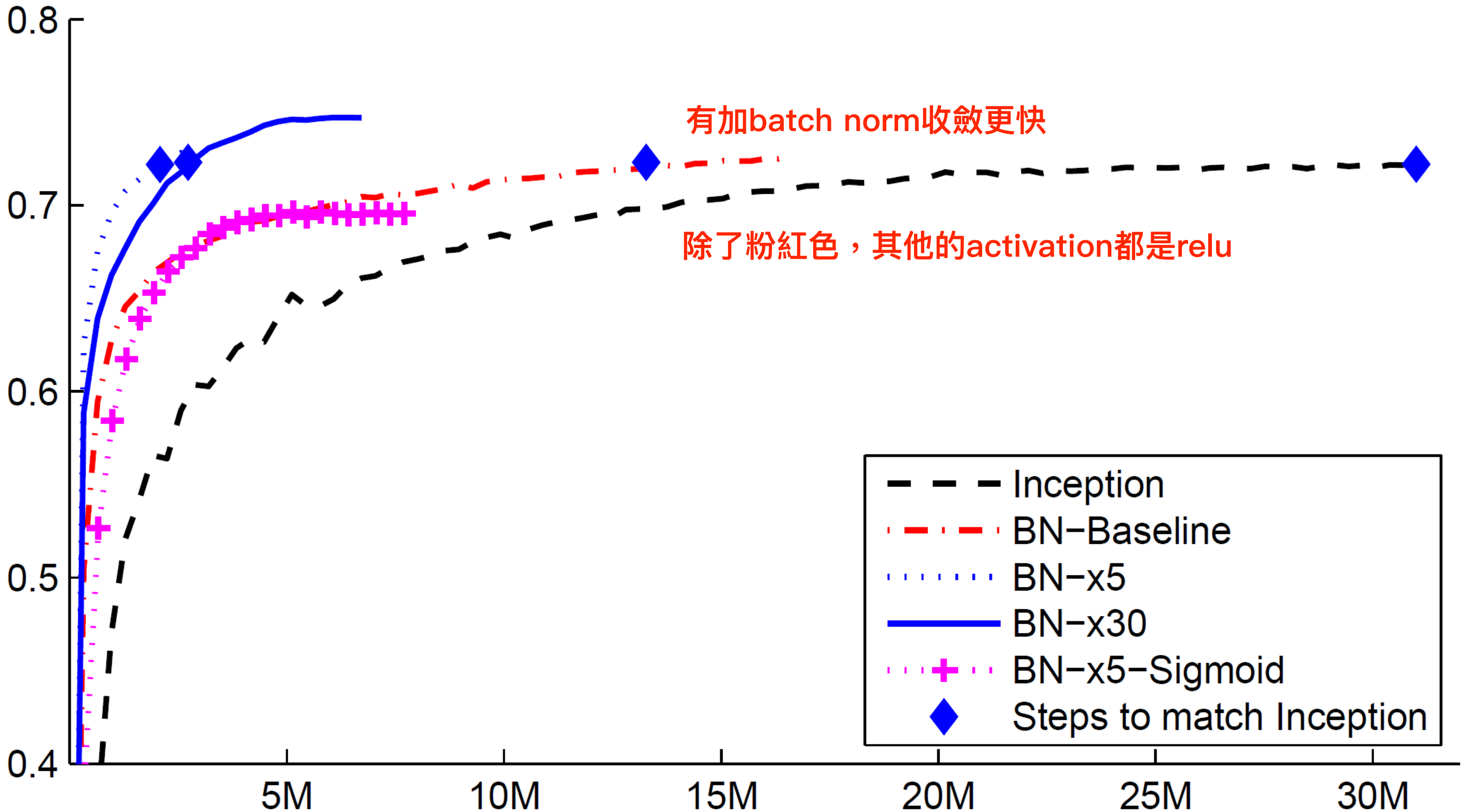


Figure 2: *Single crop validation accuracy of Inception and its batch-normalized variants, vs. the number of training steps.*

RNN: layer normalization
CNN: instance/group normalization
GAN: spectrum normalization