

Principles and Practice of Problem Solving: Lecture 17-Intro to Deep Learning (2)

Lecturer: Haiming Jin

Part II:

Why Deep?

Deeper is Better?

Layer X Size	Word Error Rate (%)
1 X 2k	24.2
2 X 2k	20.4
3 X 2k	18.4
4 X 2k	17.8
5 X 2k	17.2
7 X 2k	17.1

Not surprised, more parameters, better performance

Seide, Frank, Gang Li, and Dong Yu. "Conversational Speech Transcription Using Context-Dependent Deep Neural Networks." *Interspeech*. 2011.

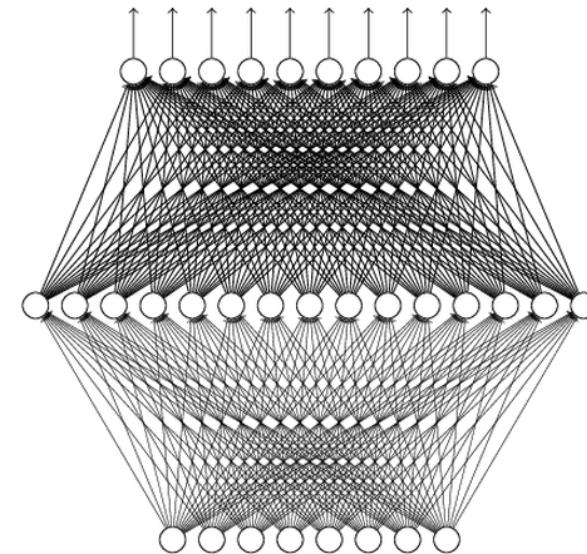
Universality Theorem

Any continuous function f

$$f : R^N \rightarrow R^M$$

Can be realized by a network
with one hidden layer

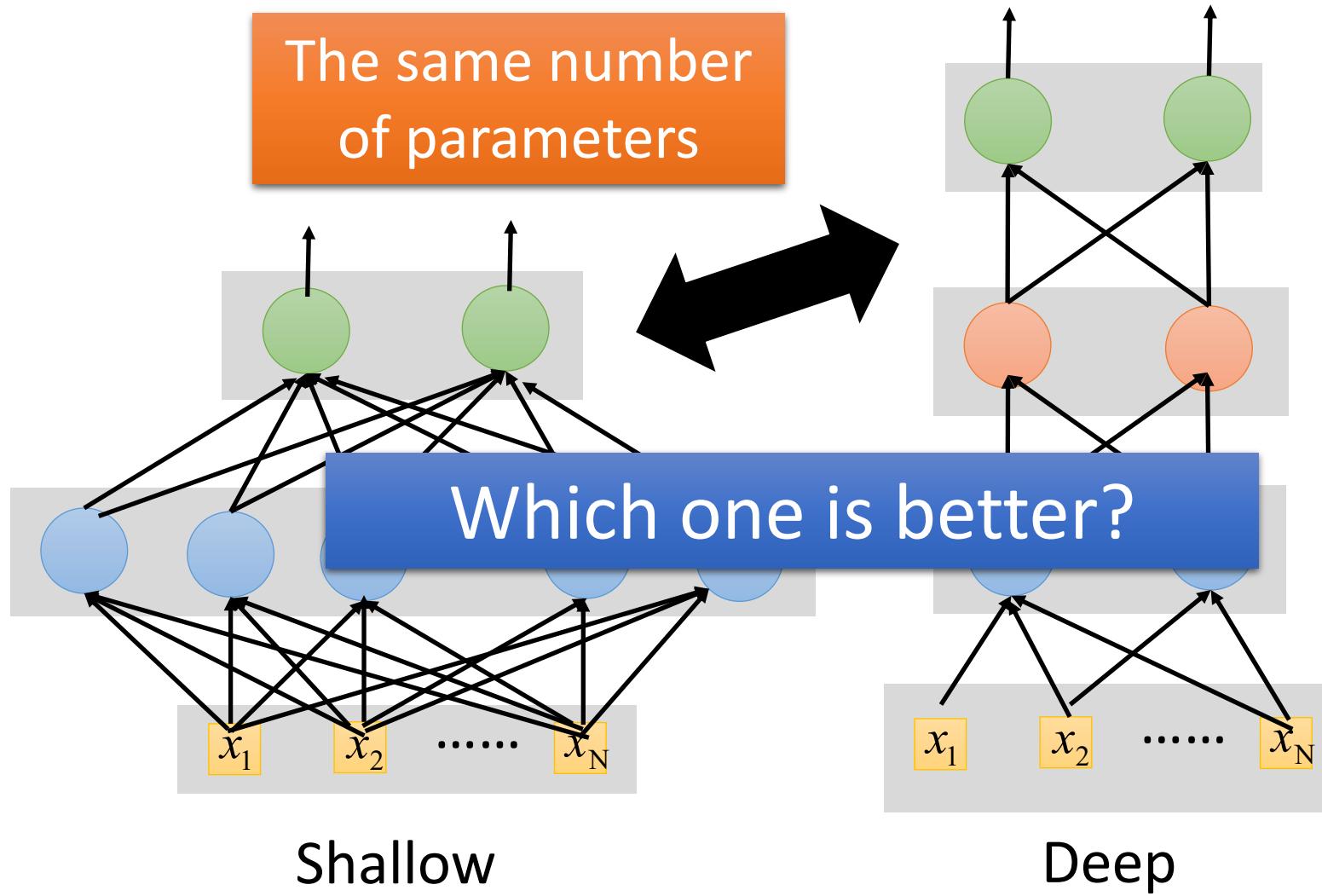
(given **enough** hidden
neurons)



Reference for the reason:
<http://neuralnetworksanddeeplearning.com/chap4.html>

Why “Deep” neural network not “Fat” neural network?

Fat + Short v.s. Thin + Tall



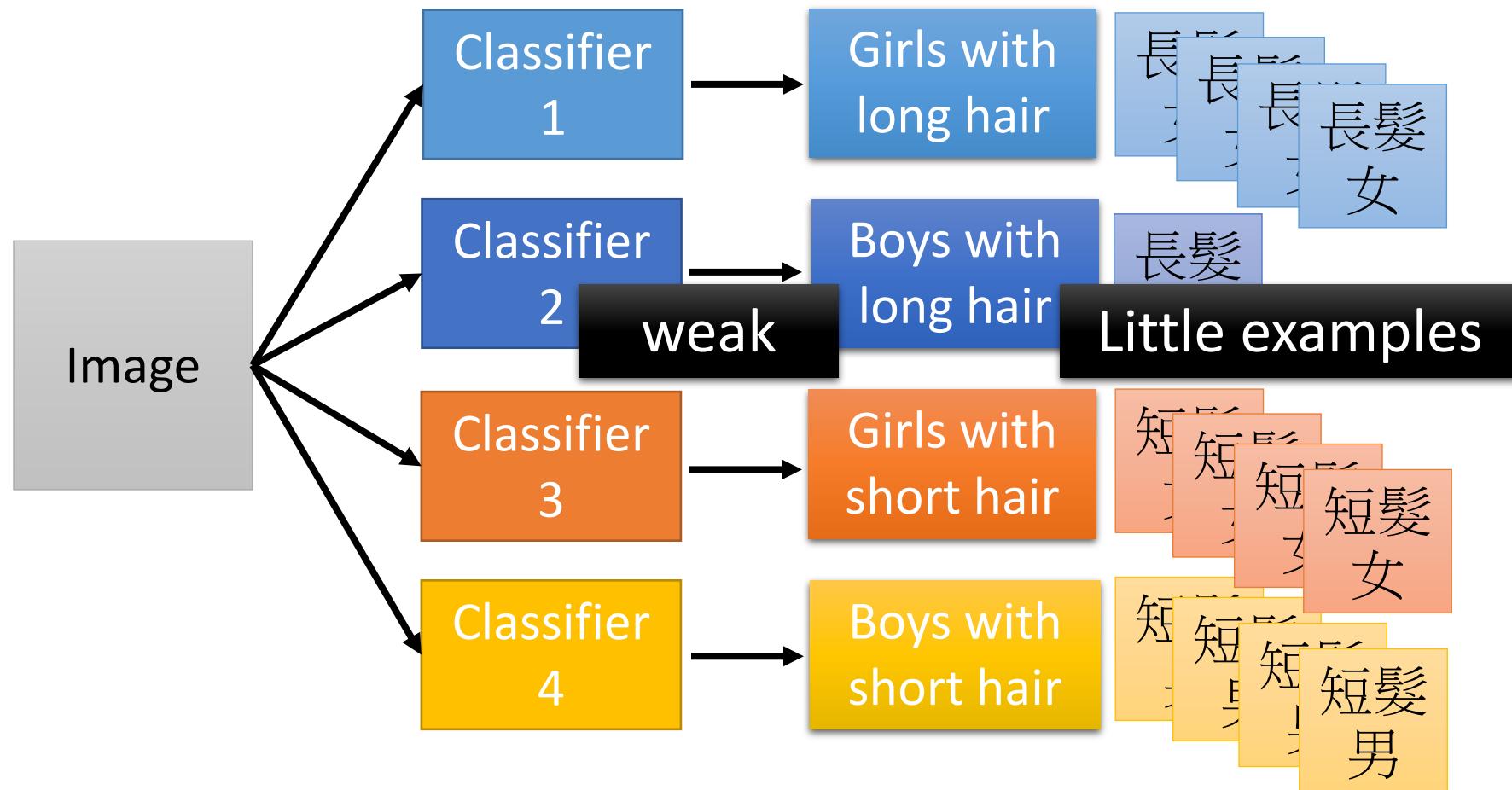
Fat + Short v.s. Thin + Tall

Layer X Size	Word Error Rate (%)	Layer X Size	Word Error Rate (%)
1 X 2k	24.2		
2 X 2k	20.4		
3 X 2k	18.4		
4 X 2k	17.8		
5 X 2k	17.2	↔ 1 X 3772	22.5
7 X 2k	17.1	↔ 1 X 4634	22.6
		1 X 16k	22.1

Seide, Frank, Gang Li, and Dong Yu. "Conversational Speech Transcription Using Context-Dependent Deep Neural Networks." *Interspeech*. 2011.

Why Deep?

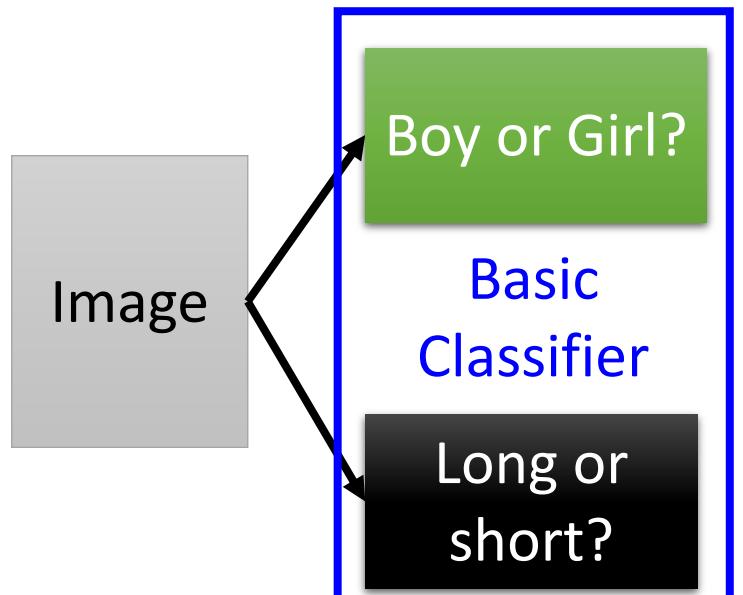
- Deep → Modularization



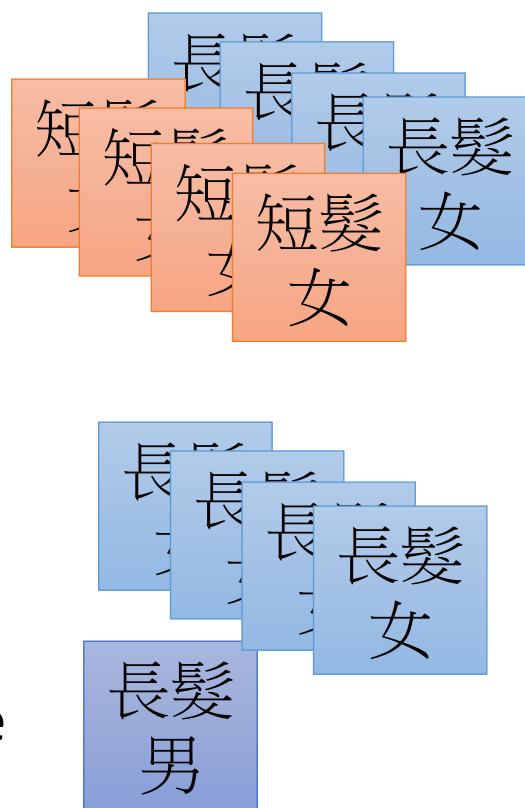
Why Deep?

Each basic classifier can have sufficient training examples.

- Deep → Modularization



Classifiers for the attributes



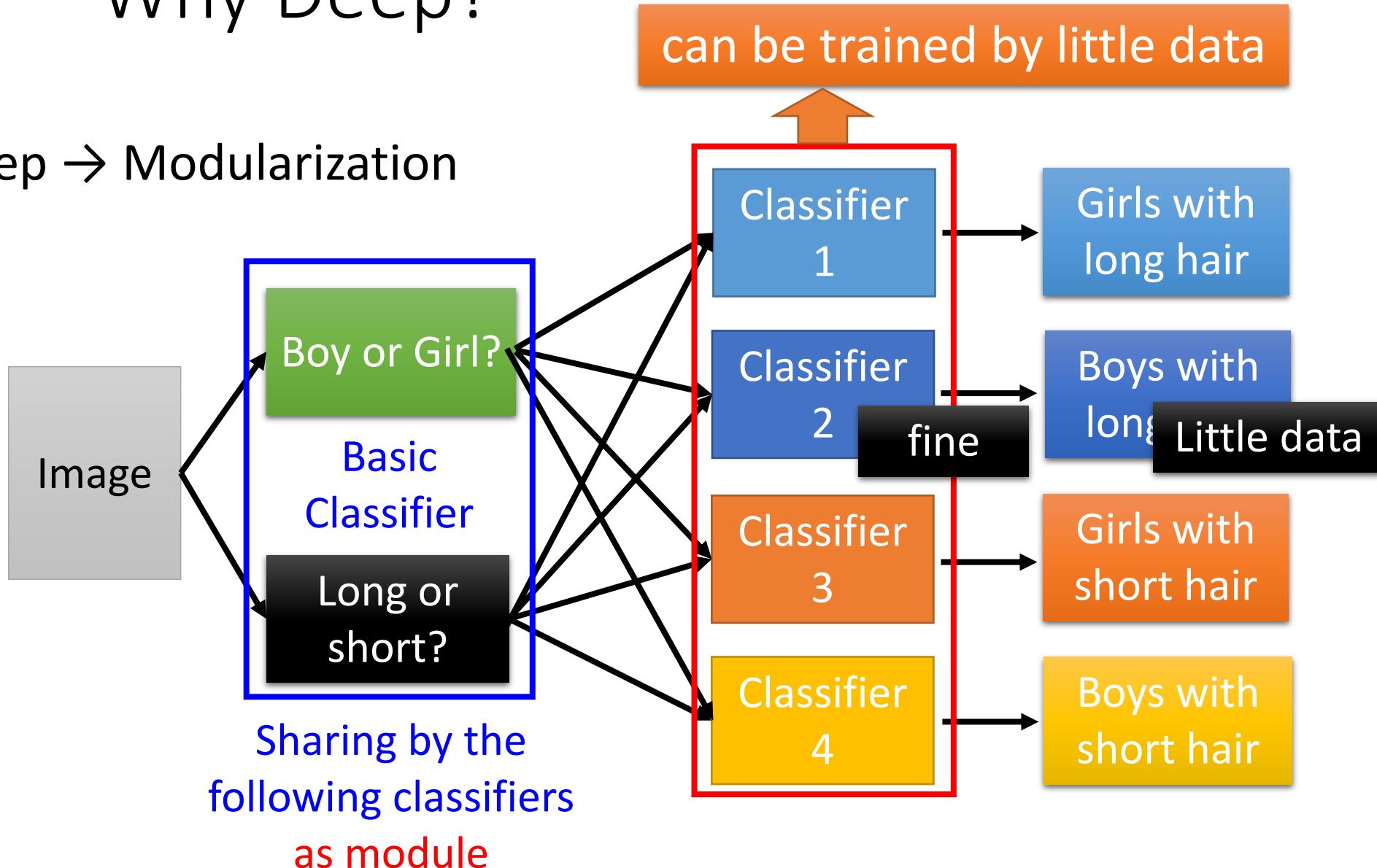
v.s.



v.s.

Why Deep?

- Deep → Modularization

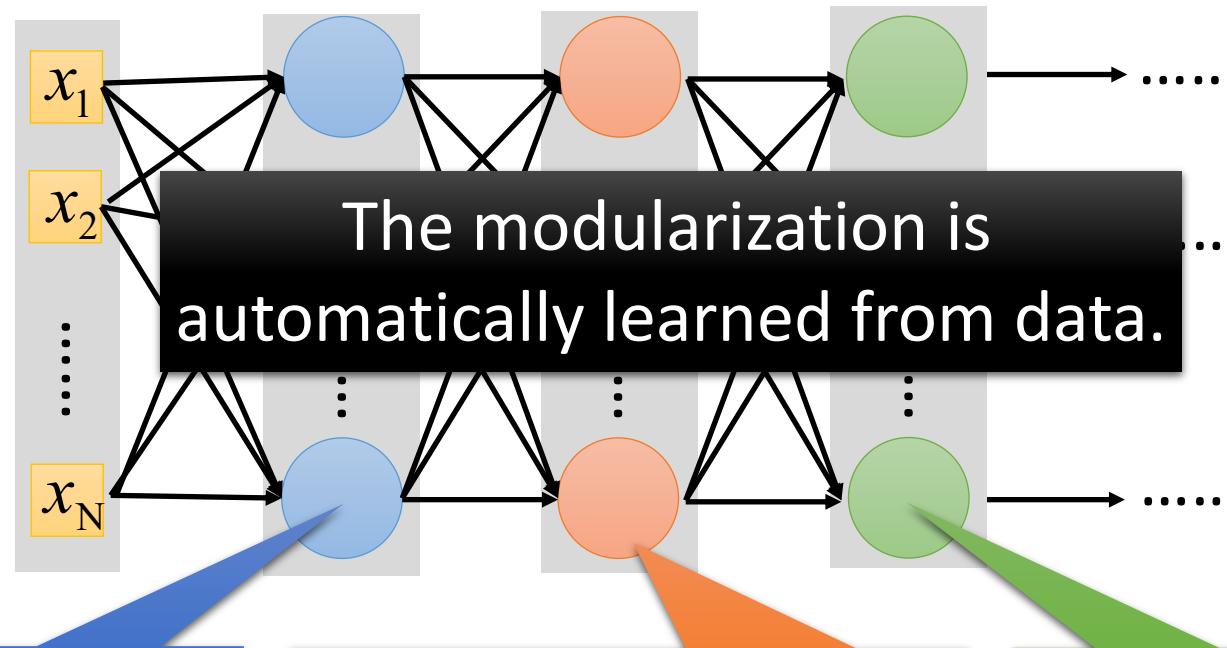


Why Deep?

Deep Learning also works on small data set like TIMIT.

- Deep → Modularization

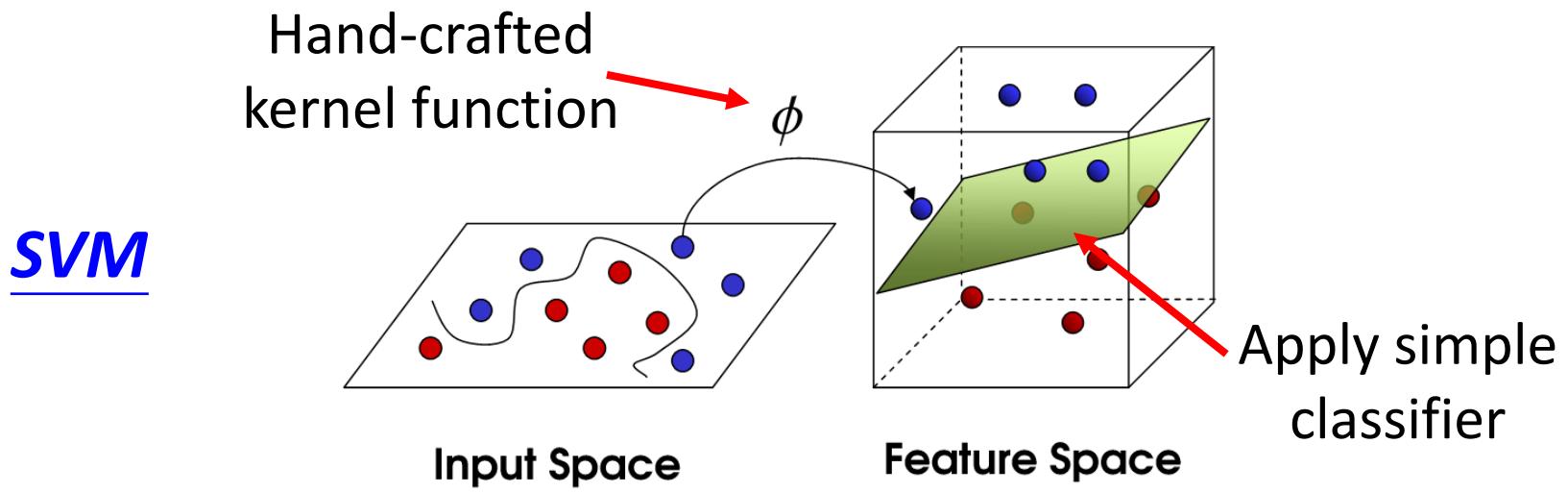
→ Less training data?



The most basic classifiers

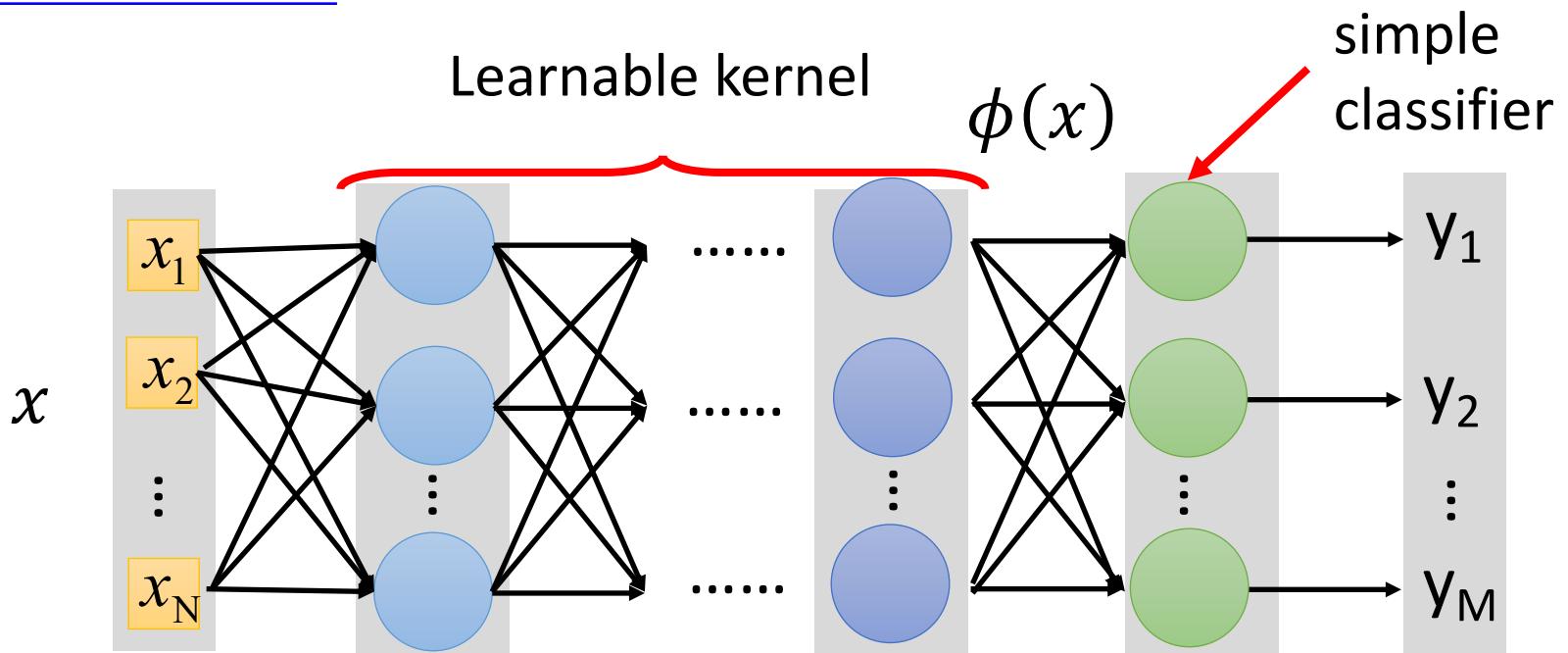
Use 1st layer as module to build classifiers

Use 2nd layer as module

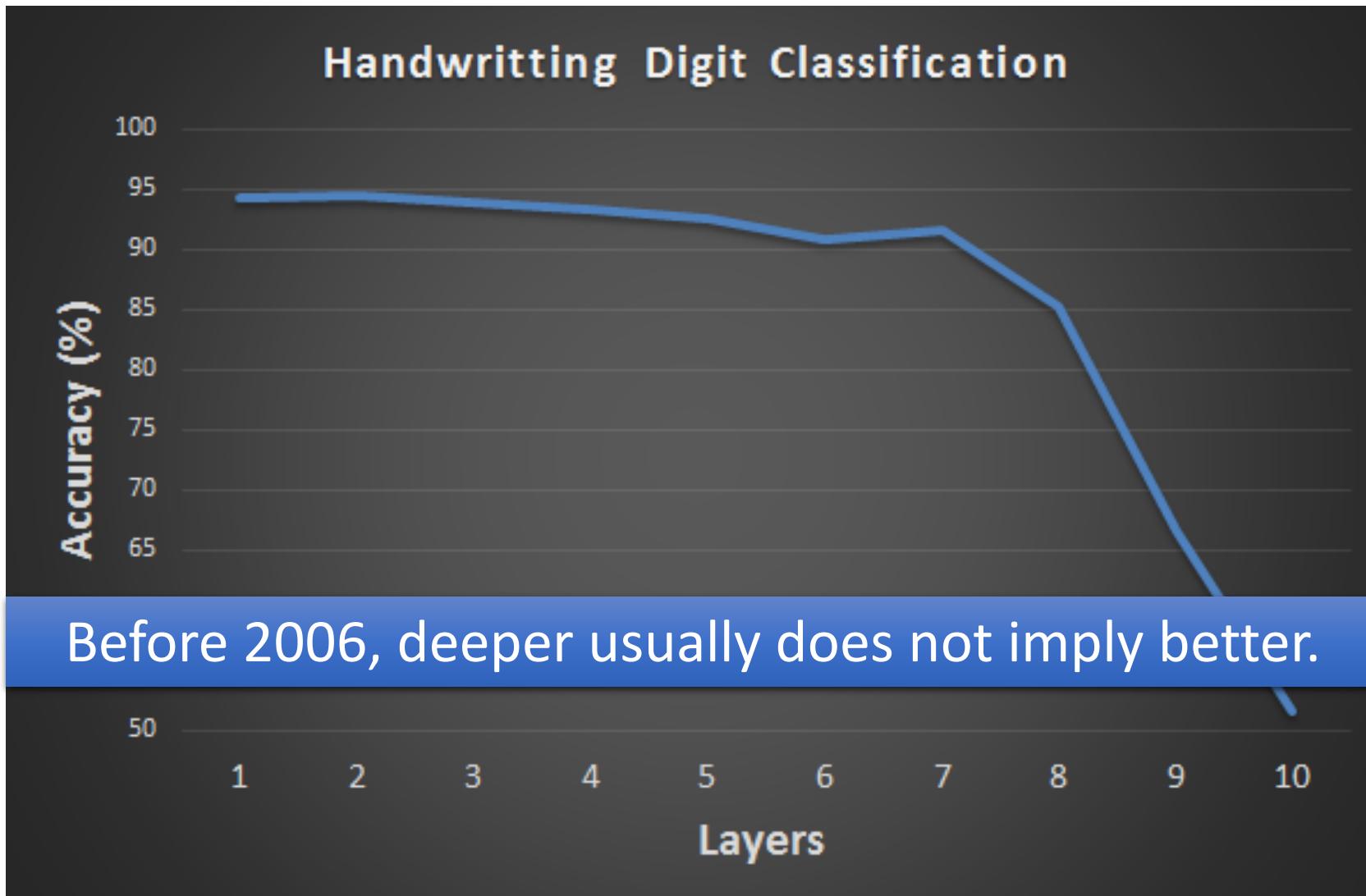


Source of image: http://www.gipsa-lab.grenoble-inp.fr/transfert/seminaire/455_Kadri2013Gipsa-lab.pdf

Deep Learning



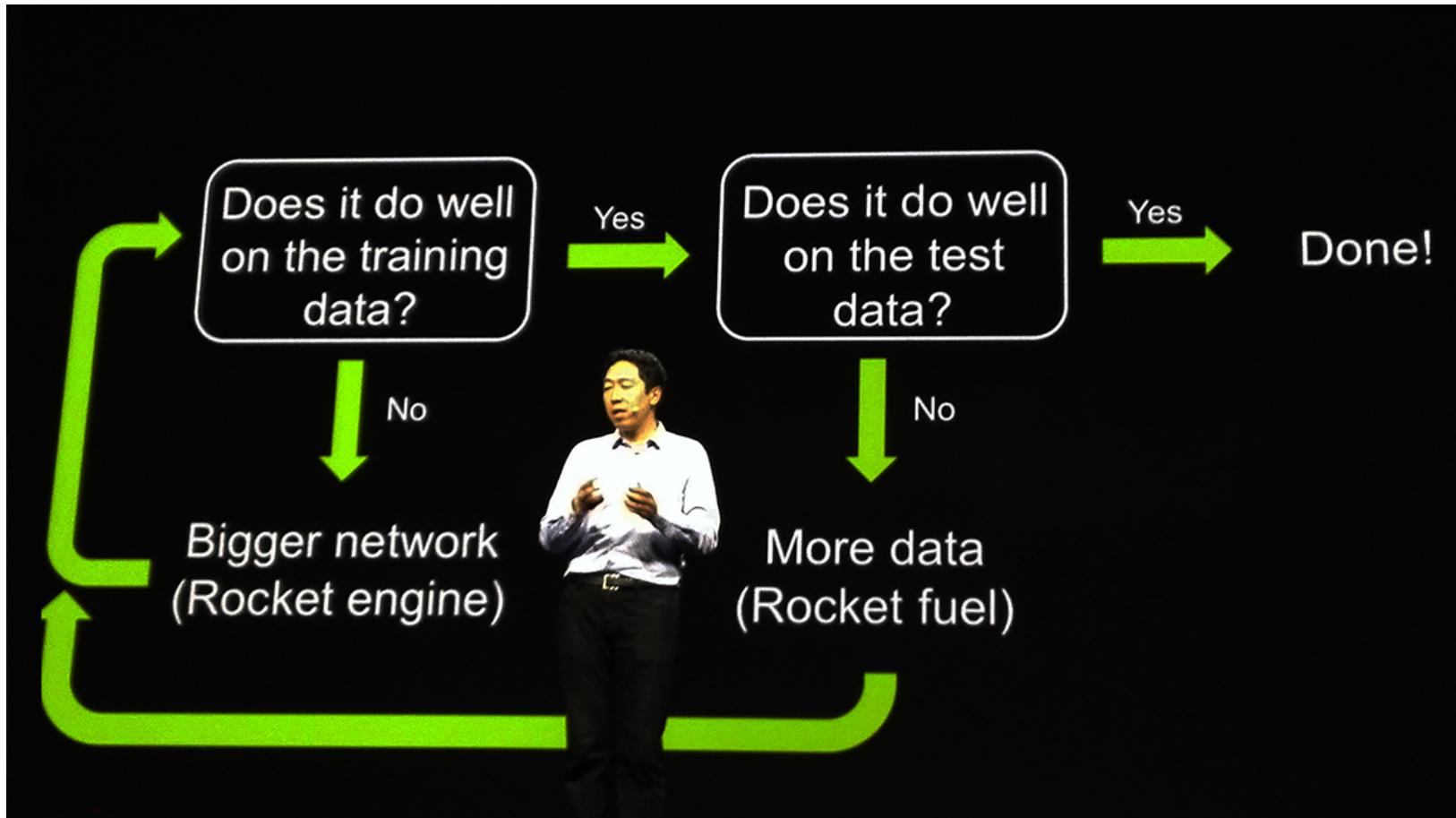
Hard to get the power of Deep ...



Part III:

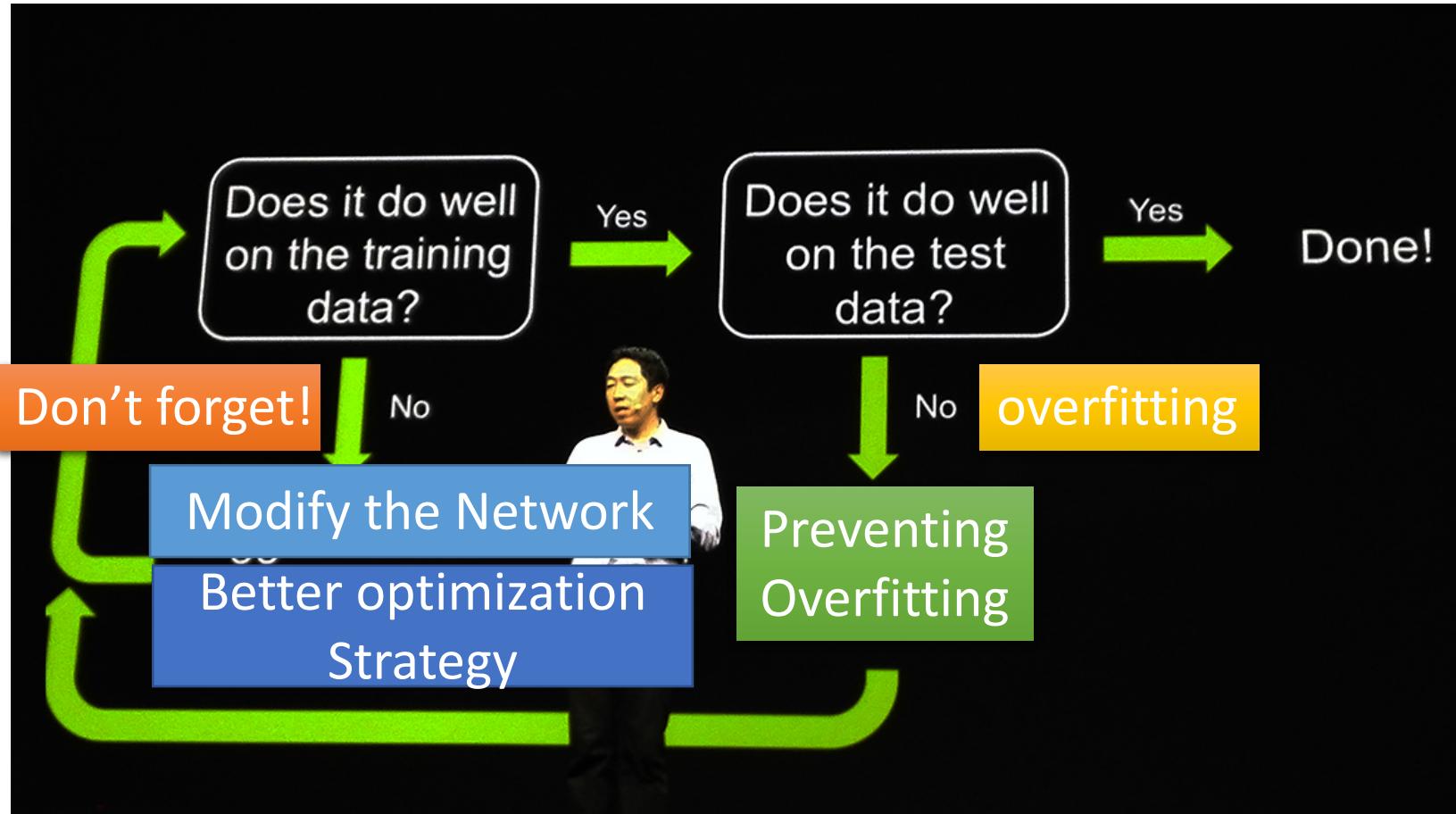
Tips for Training DNN

Recipe for Learning



<http://www.gizmodo.com.au/2015/04/the-basic-recipe-for-machine-learning-explained-in-a-single-powerpoint-slide/>

Recipe for Learning



<http://www.gizmodo.com.au/2015/04/the-basic-recipe-for-machine-learning-explained-in-a-single-powerpoint-slide/>

Recipe for Learning

Modify the Network

- New activation functions, for example, ReLU or Maxout

Better optimization Strategy

- Adaptive learning rates

Prevent Overfitting

- Dropout

Only use this approach when you already obtained good results on the training data.

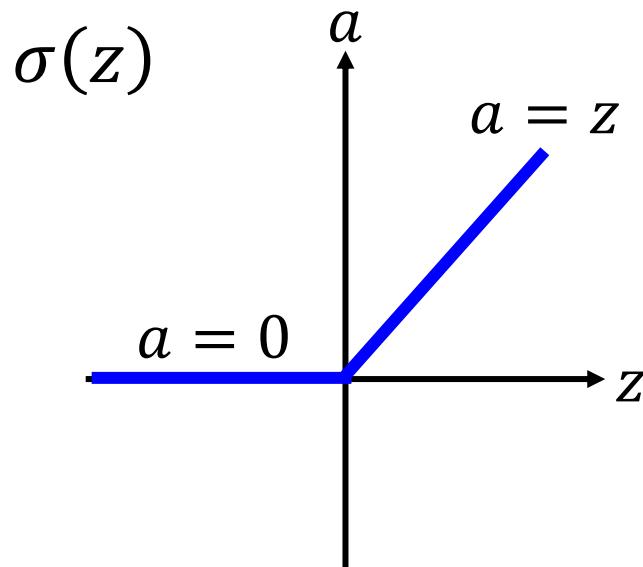
Part III:

Tips for Training DNN

New Activation Function

ReLU

- Rectified Linear Unit (ReLU)

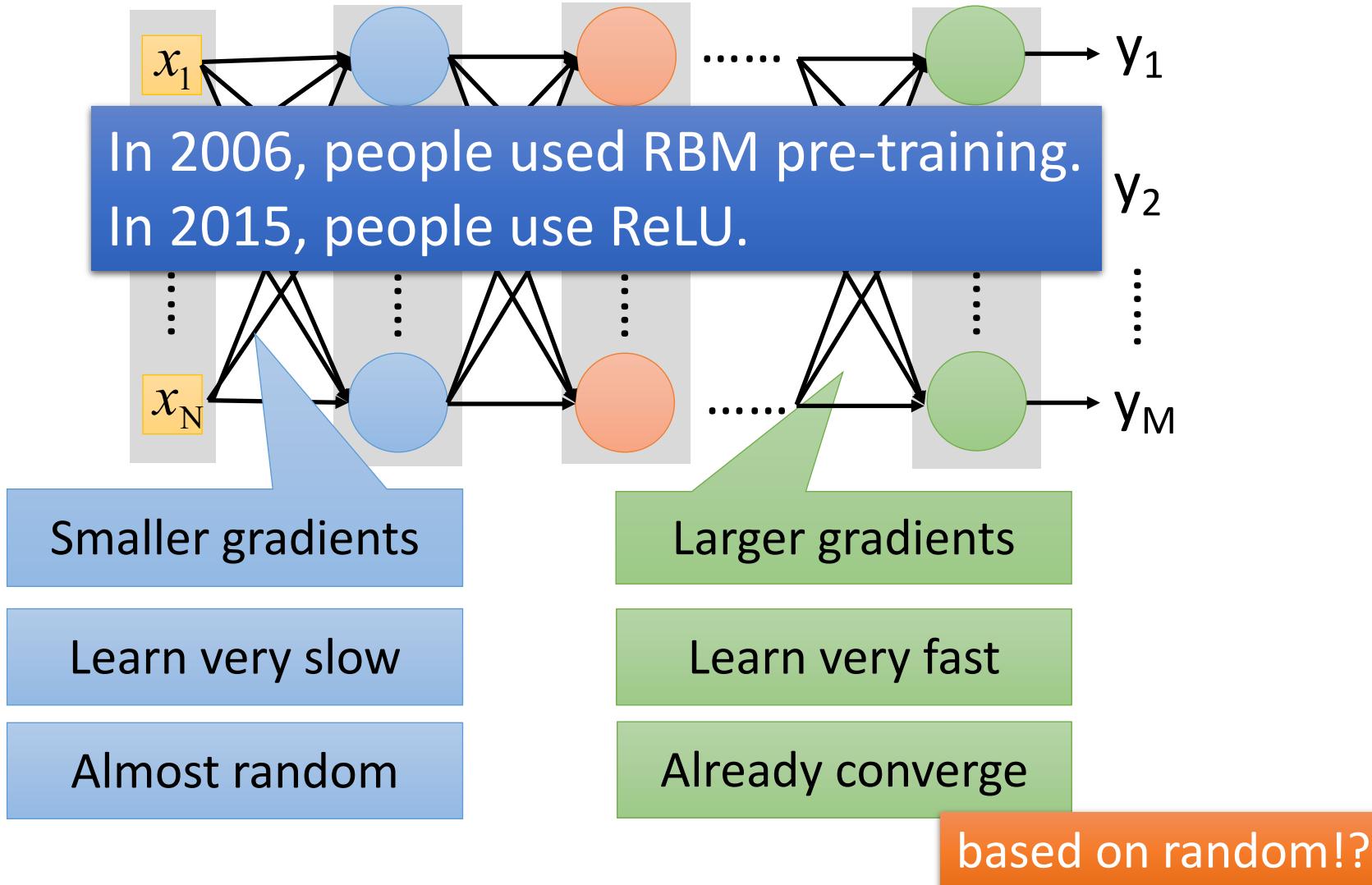


[Xavier Glorot, AISTATS'11]
[Andrew L. Maas, ICML'13]
[Kaiming He, arXiv'15]

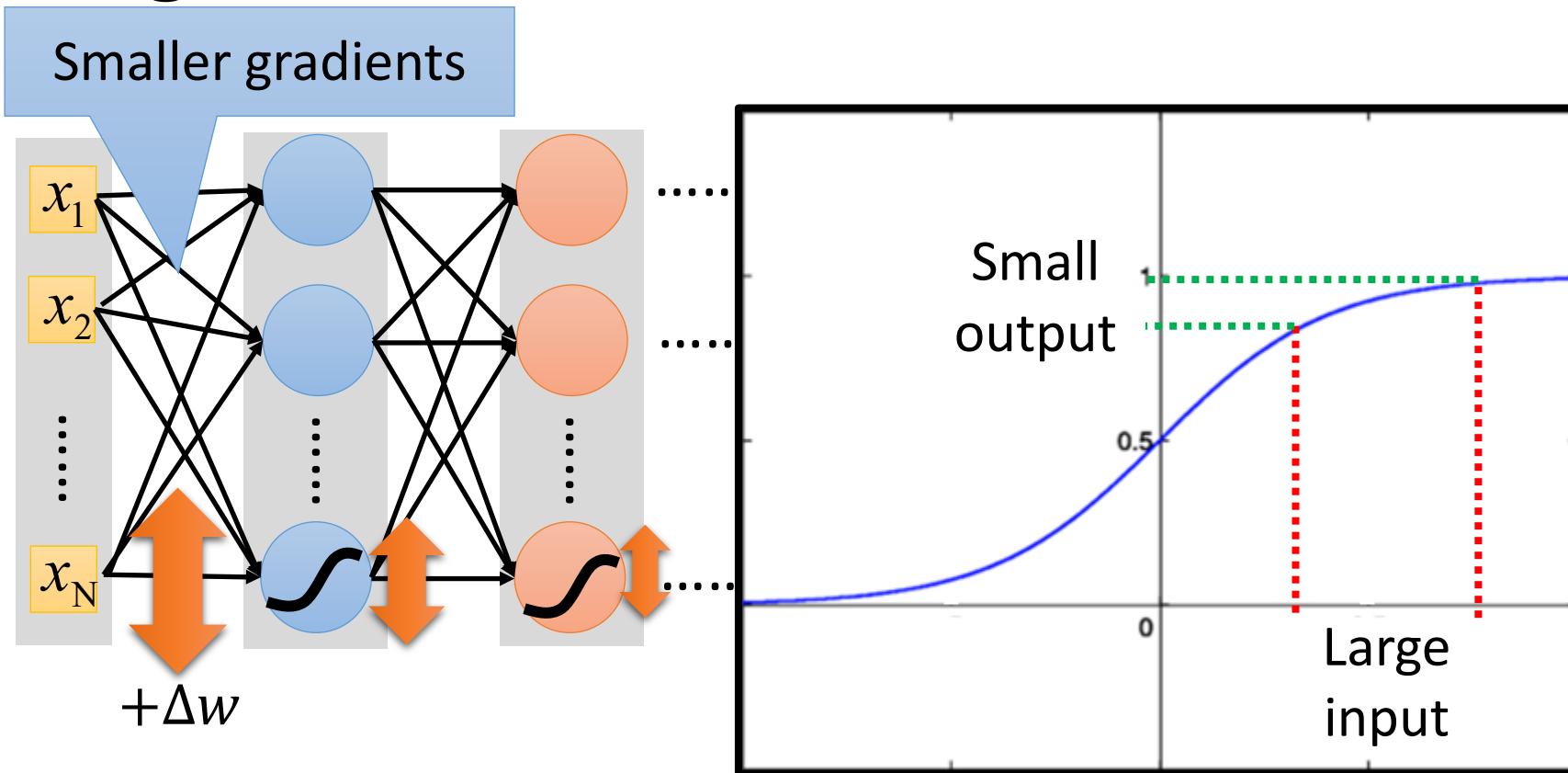
Reason:

1. Fast to compute
2. Biological reason
3. Infinite sigmoid with different biases
4. Vanishing gradient problem

Vanishing Gradient Problem



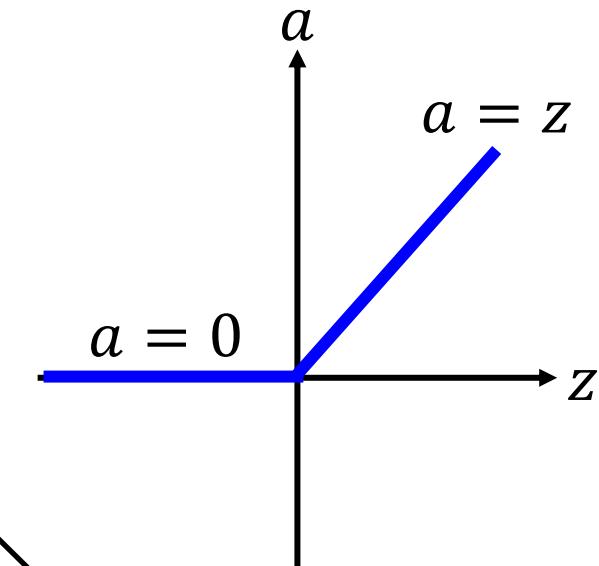
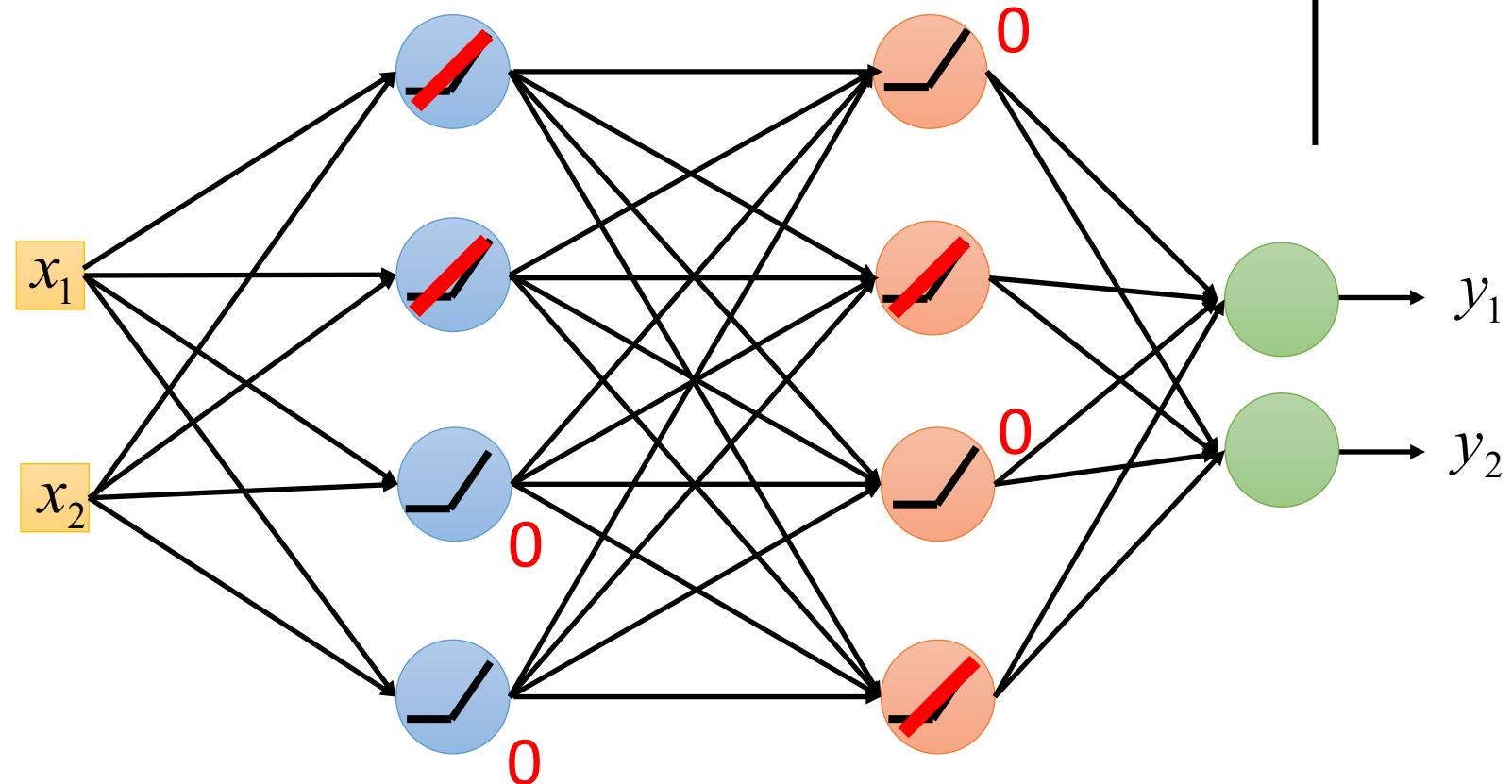
Vanishing Gradient Problem



Intuitive way to compute the gradient ...

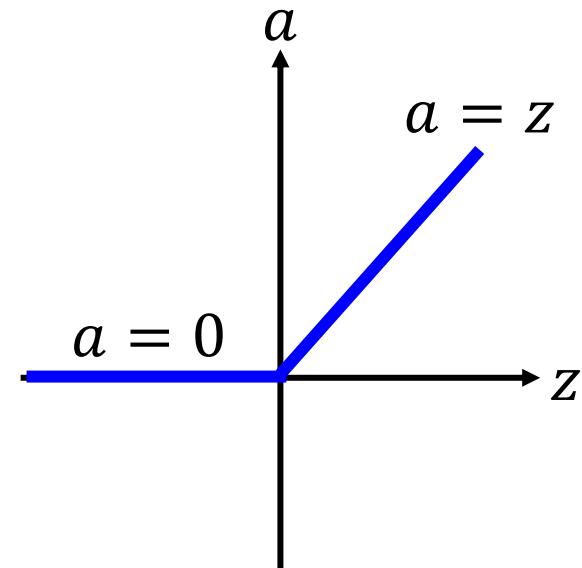
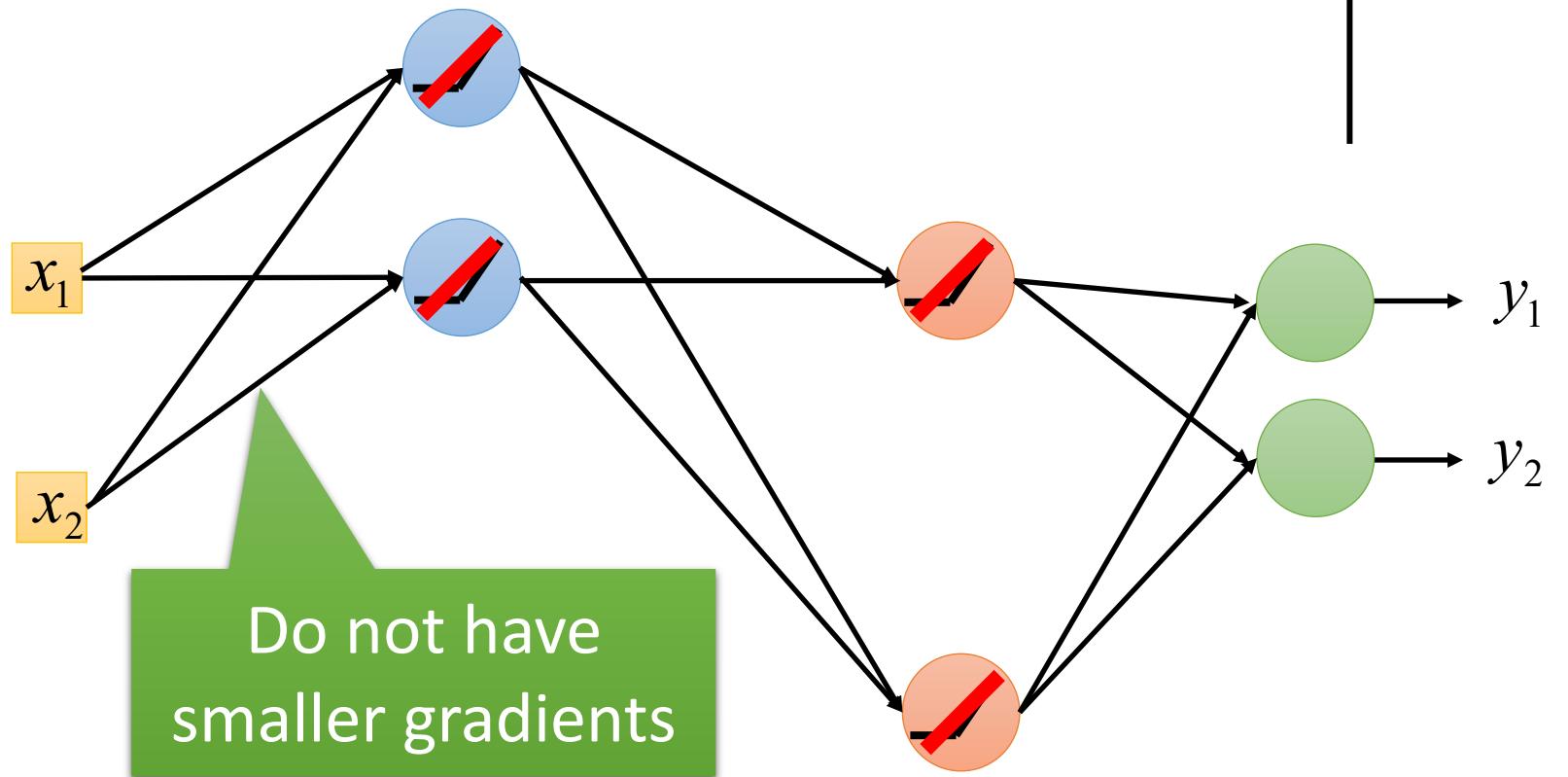
$$\frac{\partial C}{\partial w} = ? \frac{\Delta C}{\Delta w}$$

ReLU



ReLU

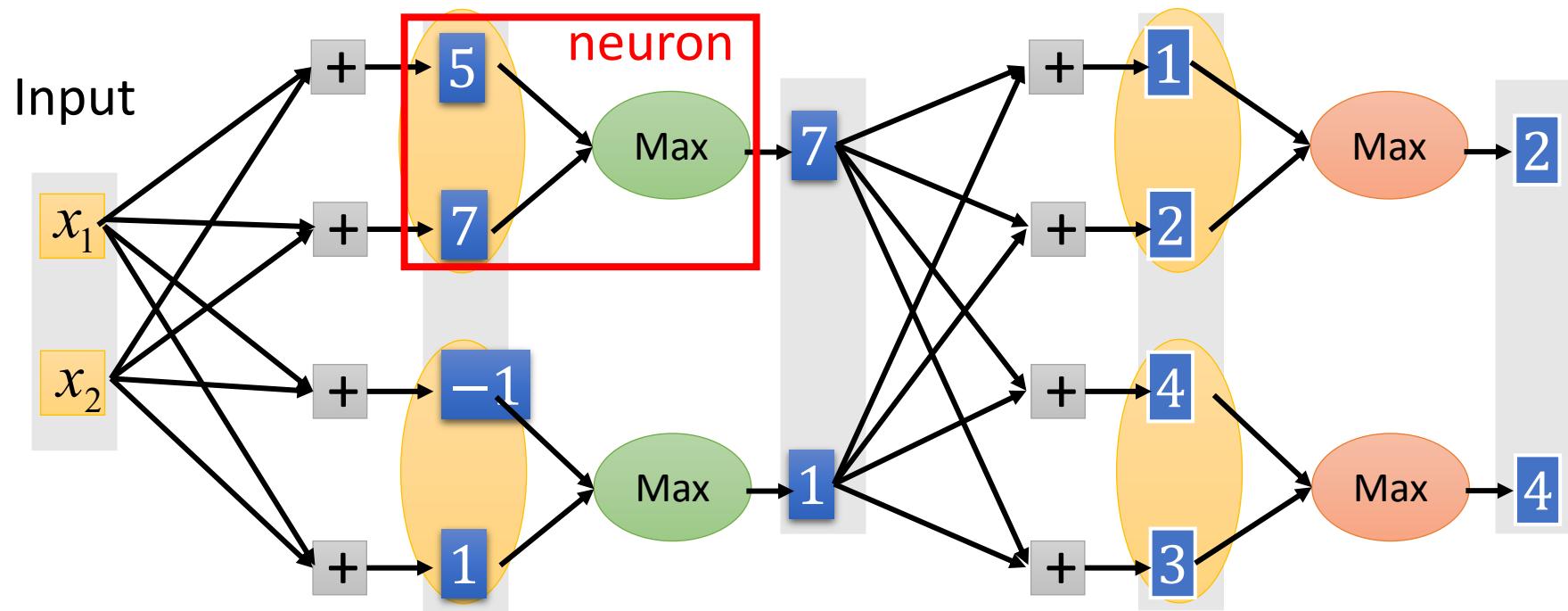
A Thinner linear network



Maxout

ReLU is a special cases of Maxout

- Learnable activation function [Ian J. Goodfellow, ICML'13]



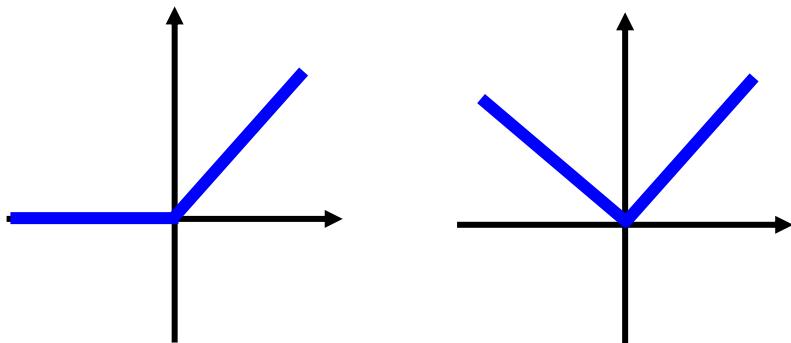
You can have more than 2 elements in a group.

Maxout

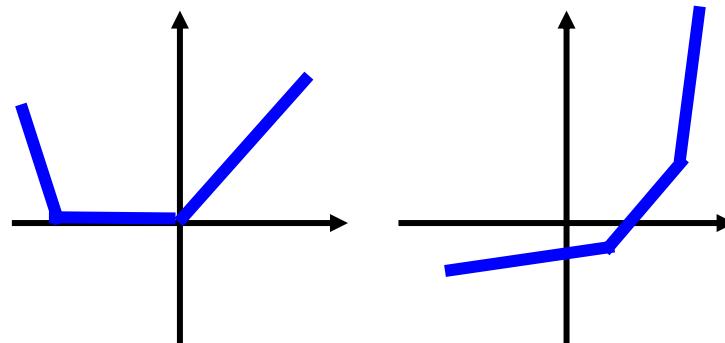
ReLU is a special cases of Maxout

- Learnable activation function [Ian J. Goodfellow, ICML'13]
 - Activation function in maxout network can be any piecewise linear convex function
 - How many pieces depending on how many elements in a group

2 elements in a group



3 elements in a group



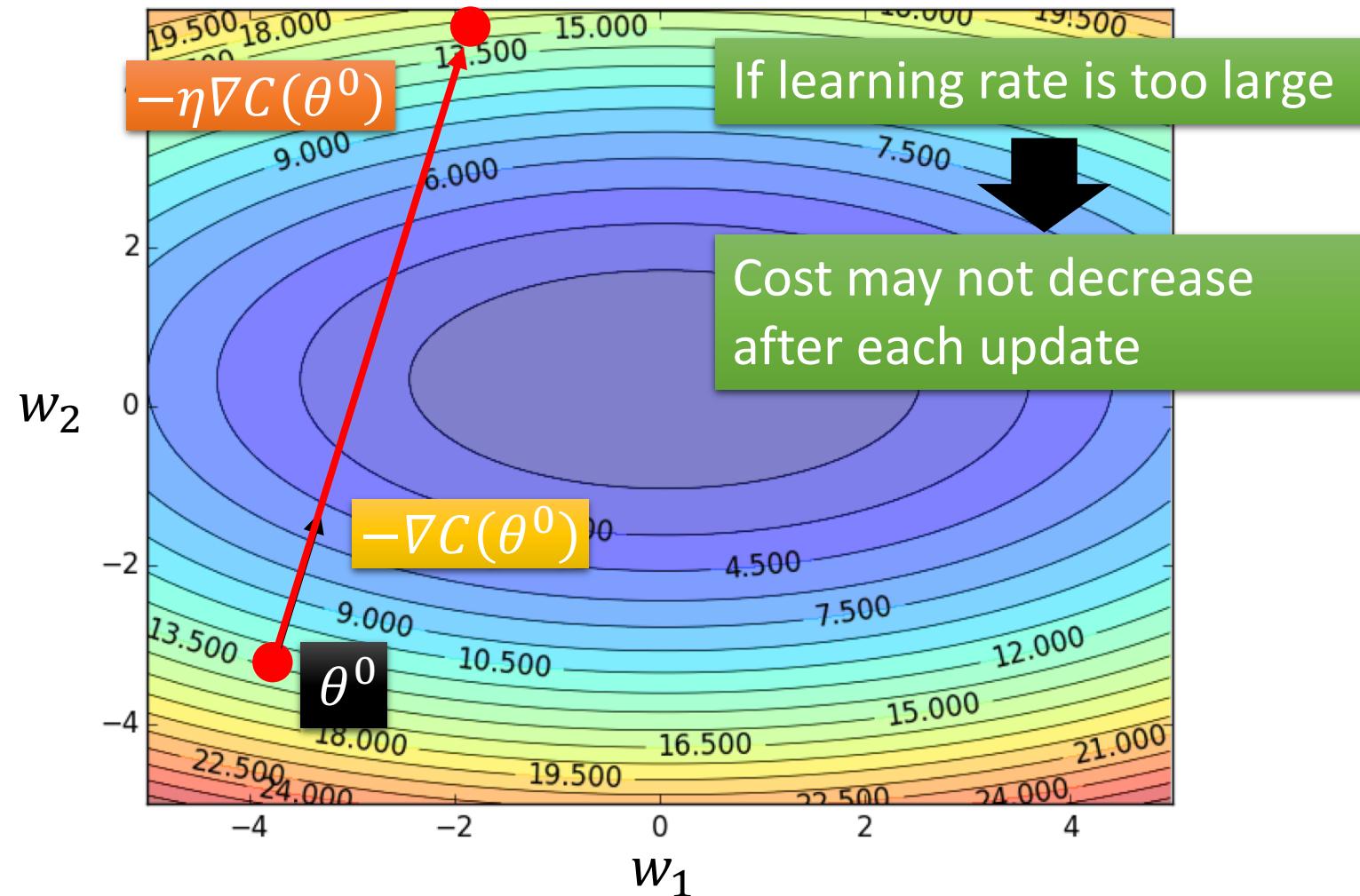
Part III:

Tips for Training DNN

Adaptive Learning Rate

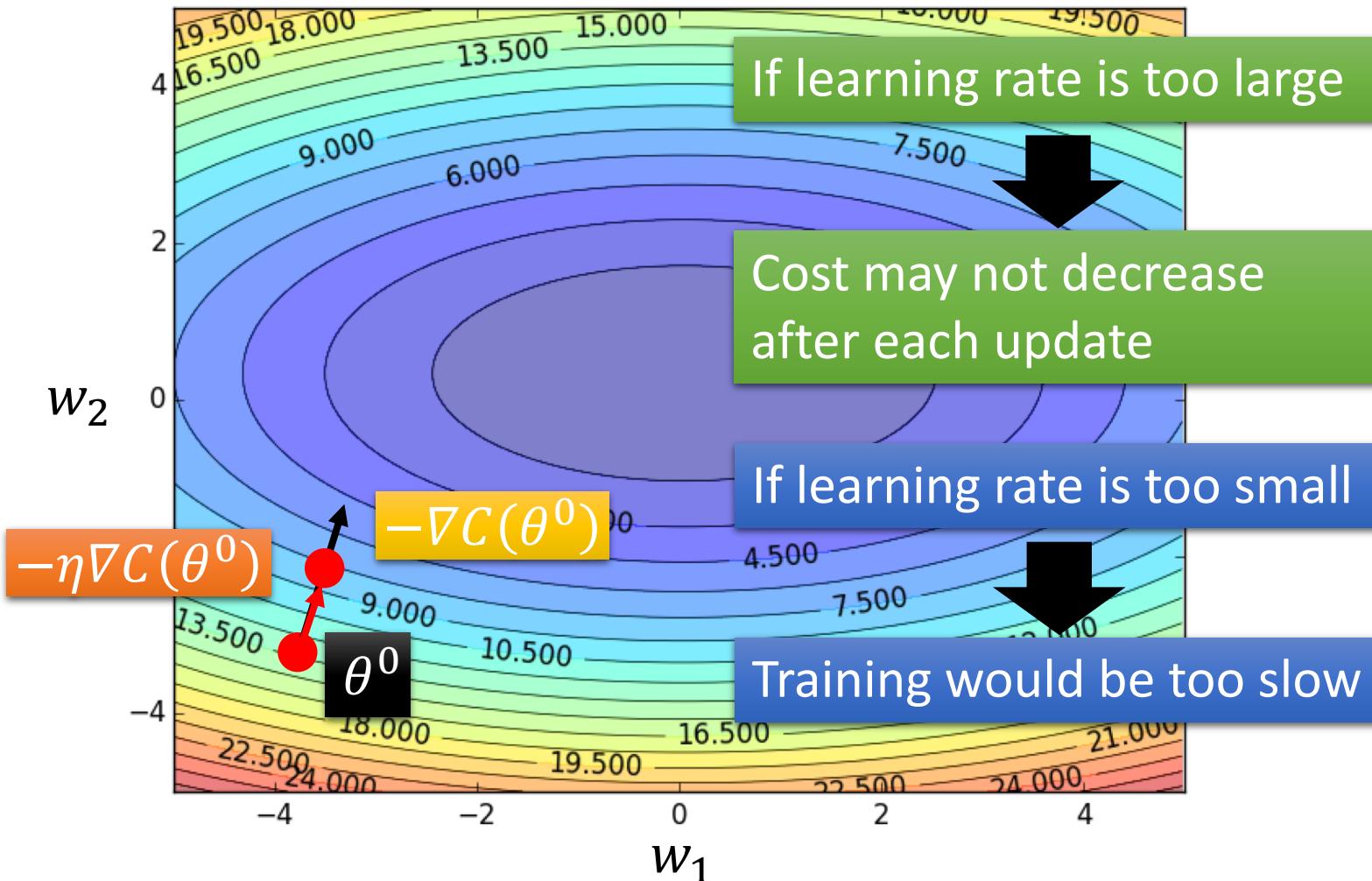
Learning Rate

Set the learning rate η carefully



Learning Rate

Can we give different parameters different learning rates?



Adagrad

Original Gradient Descent

$$\theta^t \leftarrow \theta^{t-1} - \eta \nabla C(\theta^{t-1})$$

Each parameter w are considered separately

$$w^{t+1} \leftarrow w^t - \eta_w g^t \quad g^t = \frac{\partial C(\theta^t)}{\partial w}$$

Parameter dependent learning rate

$$\eta_w = \frac{\eta}{\sqrt{\sum_{i=0}^t (g^i)^2}}$$

constant

Summation of the square of the previous derivatives

Adagrad

$$\eta_w = \frac{\eta}{\sqrt{\sum_{i=0}^t (g^i)^2}}$$

w_1	$\begin{matrix} g^0 \\ 0.1 \end{matrix}$
-------	--

w_2	$\begin{matrix} g^0 \\ 20.0 \end{matrix}$
-------	---

Learning rate:

$$\frac{\eta}{\sqrt{0.1^2}}$$

$$\frac{\eta}{\sqrt{0.1^2 + 0.2^2}}$$

$$= \frac{\eta}{0.1}$$

$$= \frac{\eta}{0.22}$$



Learning rate:

$$\frac{\eta}{\sqrt{20^2}}$$

$$\frac{\eta}{\sqrt{20^2 + 10^2}}$$

$$= \frac{\eta}{20}$$

$$= \frac{\eta}{22}$$

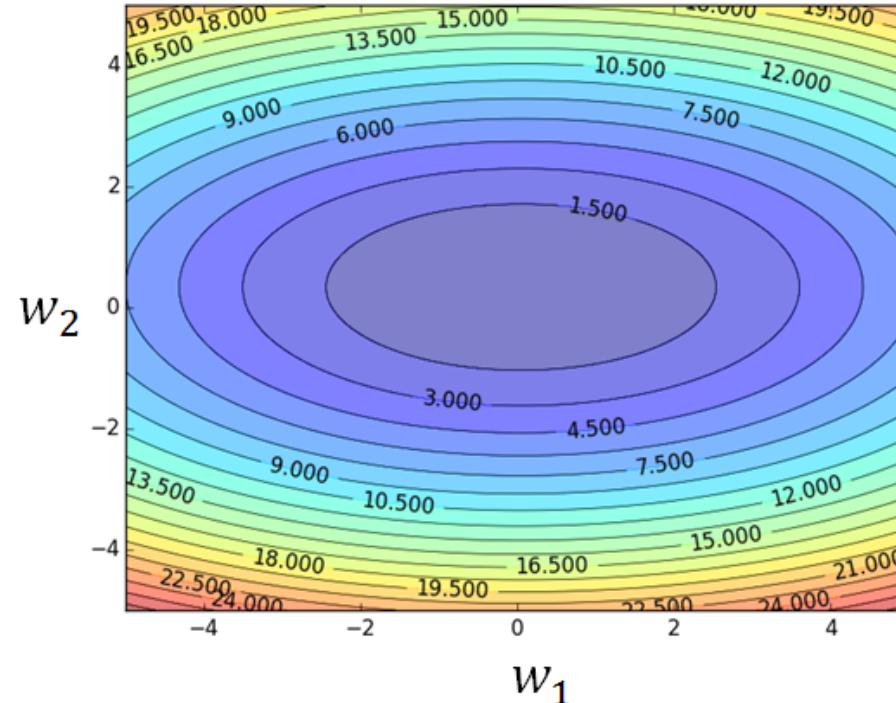


- Observation:**
1. Learning rate is smaller and smaller for all parameters
 2. Smaller derivatives, larger learning rate, and vice versa

Why?

Larger derivatives

Smaller Learning Rate



Smaller Derivatives

Larger Learning Rate

2. Smaller derivatives, larger learning rate, and vice versa

Why?

Not the whole story

- Adagrad [John Duchi, JMLR'11]
- RMSprop
 - <https://www.youtube.com/watch?v=O3sxAc4hxZU>
- Adadelta [Matthew D. Zeiler, arXiv'12]
- Adam [Diederik P. Kingma, ICLR'15]
- AdaSecant [Caglar Gulcehre, arXiv'14]
- “No more pesky learning rates” [Tom Schaul, arXiv'12]

Part III:

Tips for Training DNN

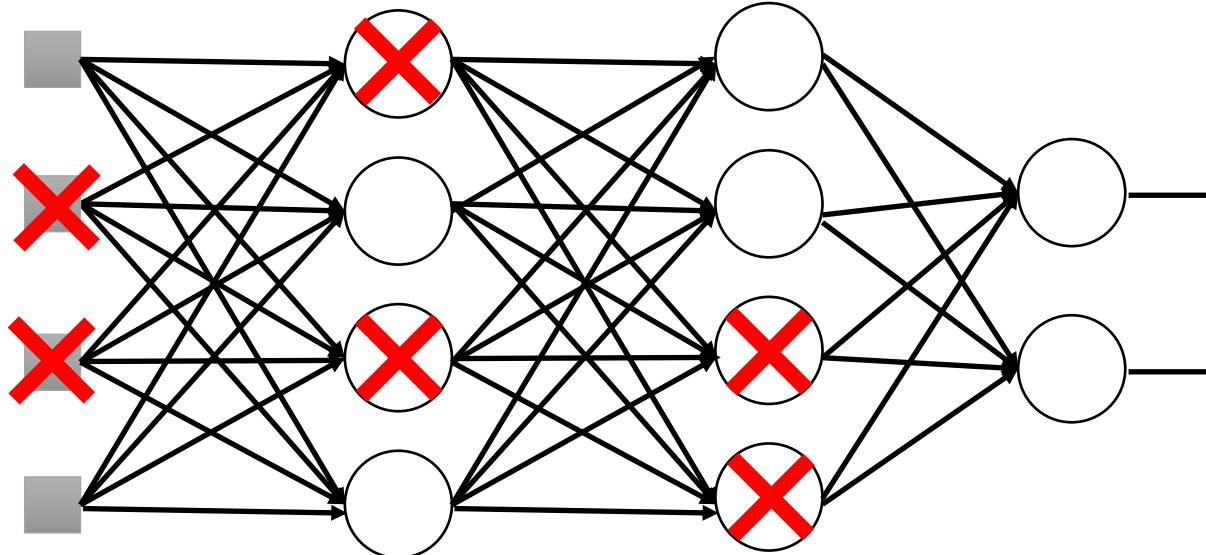
Dropout

Dropout

Pick a mini-batch

$$\theta^t \leftarrow \theta^{t-1} - \eta \nabla C(\theta^{t-1})$$

Training:



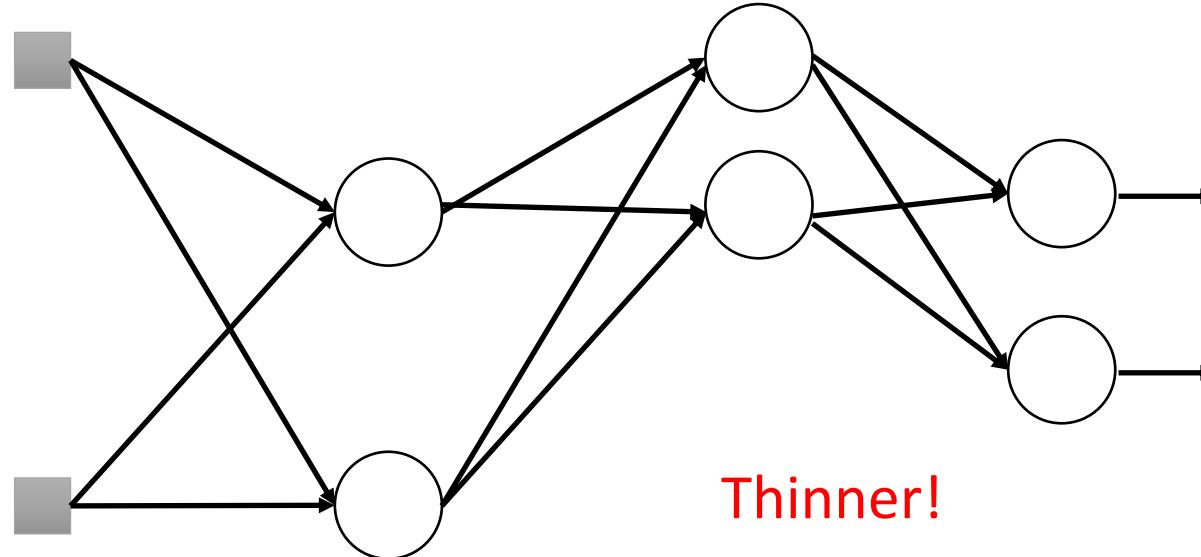
- **Each time before computing the gradients**
 - Each neuron has p% to dropout

Dropout

Pick a mini-batch

$$\theta^t \leftarrow \theta^{t-1} - \eta \nabla C(\theta^{t-1})$$

Training:

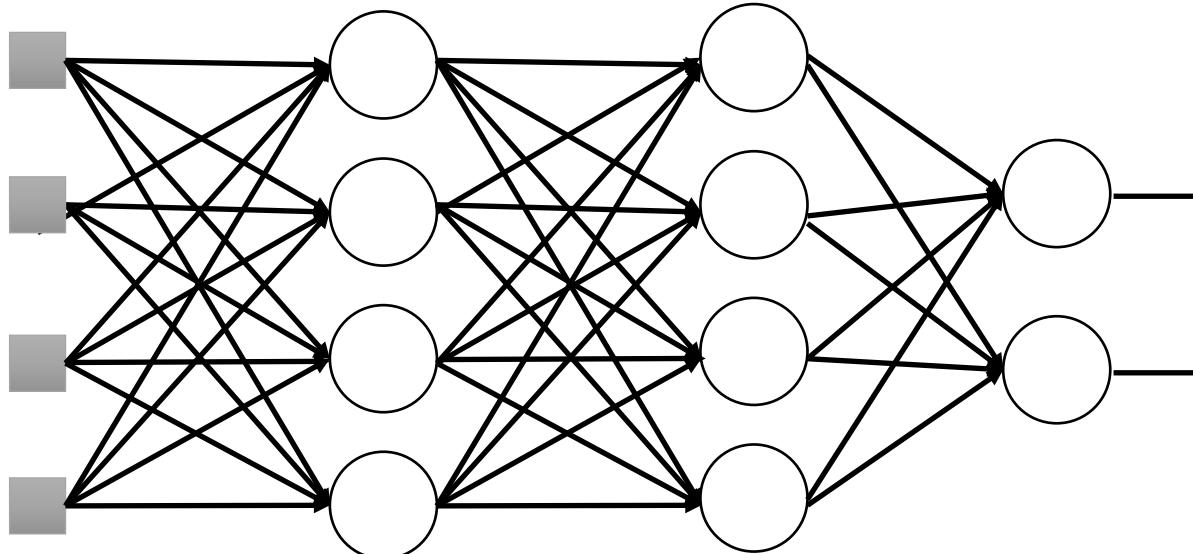


- **Each time before computing the gradients**
 - Each neuron has p% to dropout
 - ➡ **The structure of the network is changed.**
 - Using the new network for training

For each mini-batch, we resample the dropout neurons

Dropout

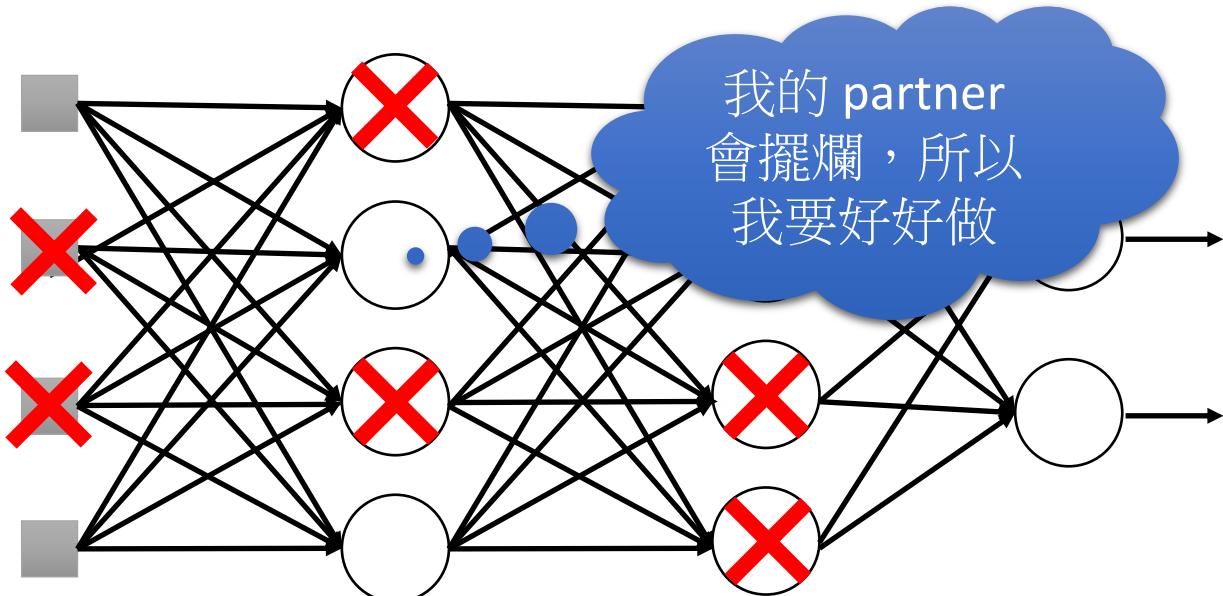
Testing:



➤ No dropout

- If the dropout rate at training is $p\%$,
all the weights times $(1-p)\%$
- Assume that the dropout rate is 50%.
If a weight $w = 1$ by training, set $w = 0.5$ for testing.

Dropout - Intuitive Reason



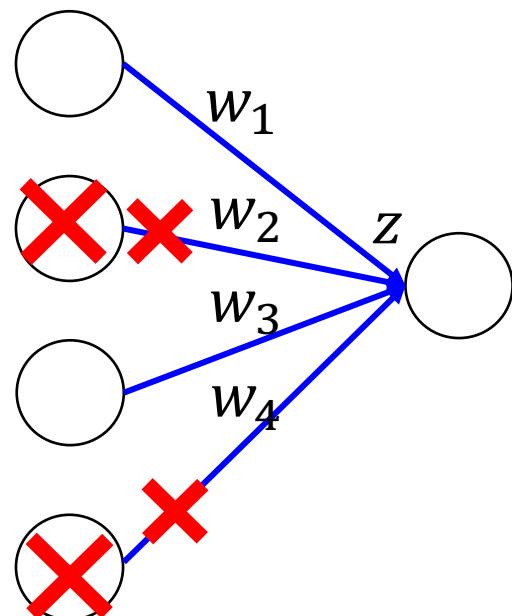
- When teams up, if everyone expect the partner will do the work, nothing will be done finally.
- However, if you know your partner will dropout, you will do better.
- When testing, no one dropout actually, so obtaining good results eventually.

Dropout - Intuitive Reason

- Why the weights should multiply $(1-p)\%$ (dropout rate) when testing?

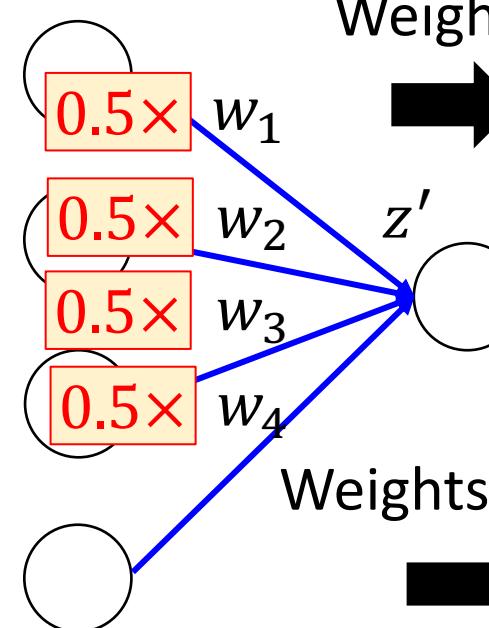
Training of Dropout

Assume dropout rate is 50%



Testing of Dropout

No dropout



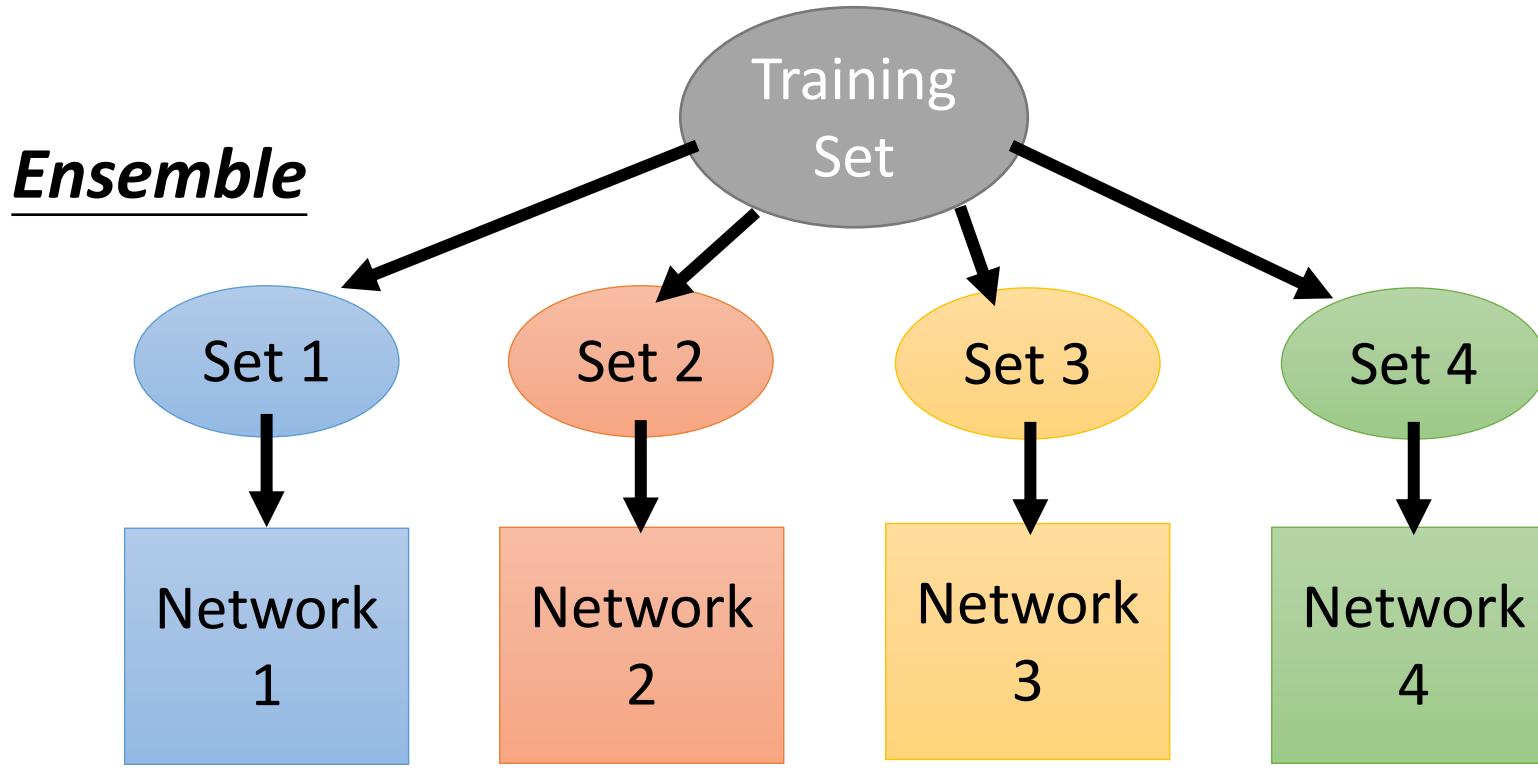
Weights from training

$$\rightarrow z' \approx 2z$$

Weights multiply $(1-p)\%$

$$\rightarrow z' \approx z$$

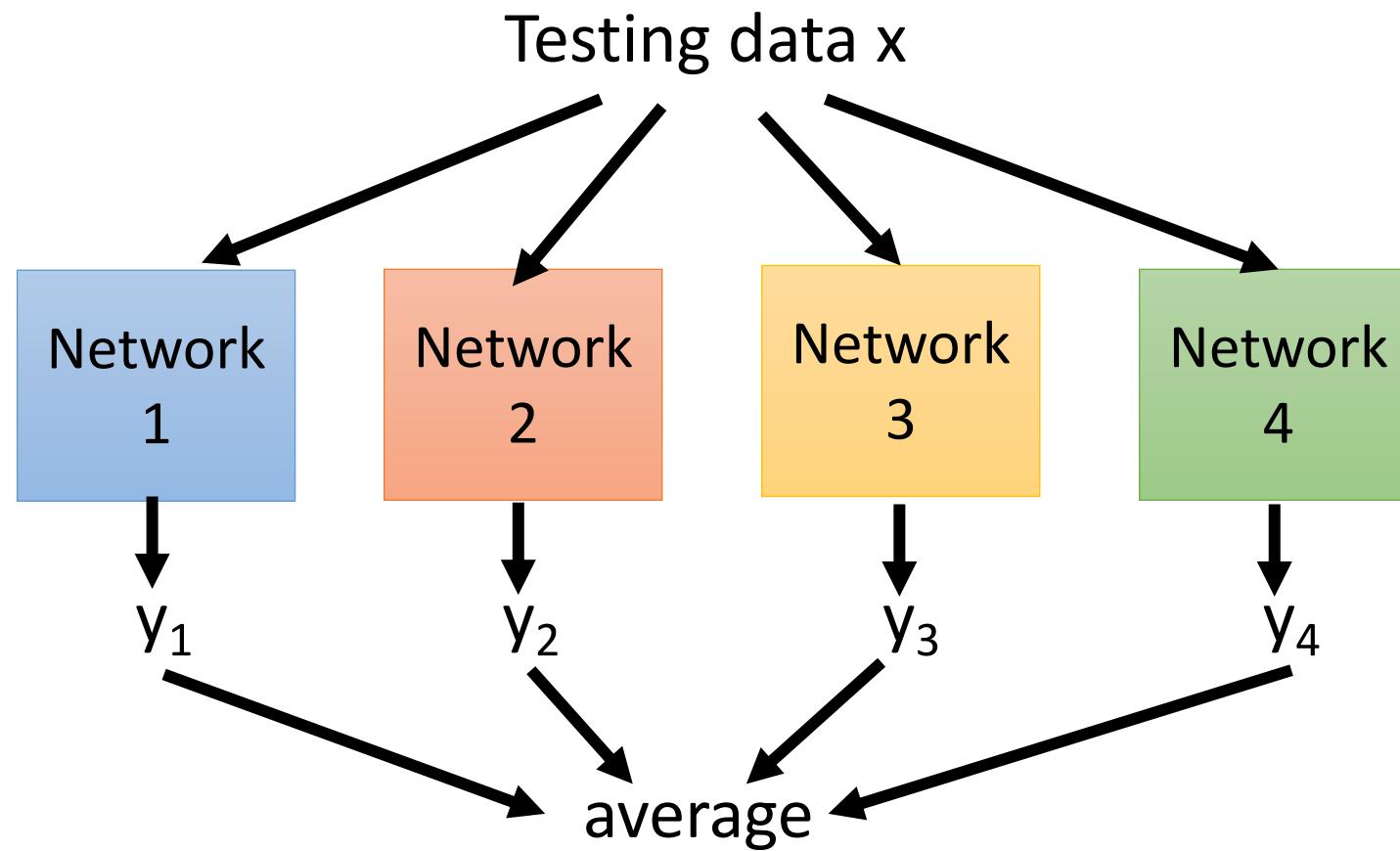
Dropout is a kind of ensemble.



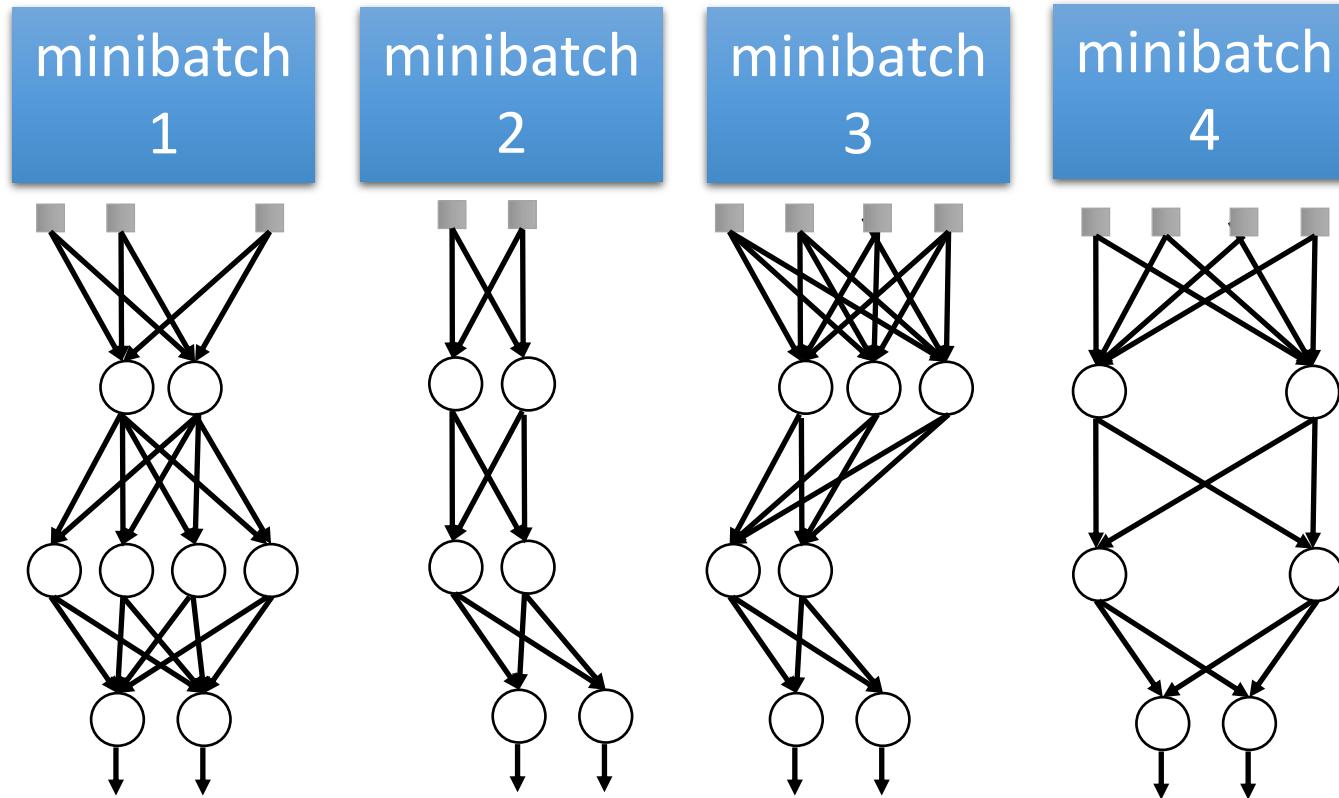
Train a bunch of networks with different structures

Dropout is a kind of ensemble.

Ensemble

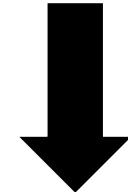


Dropout is a kind of ensemble.



Training of Dropout

M neurons

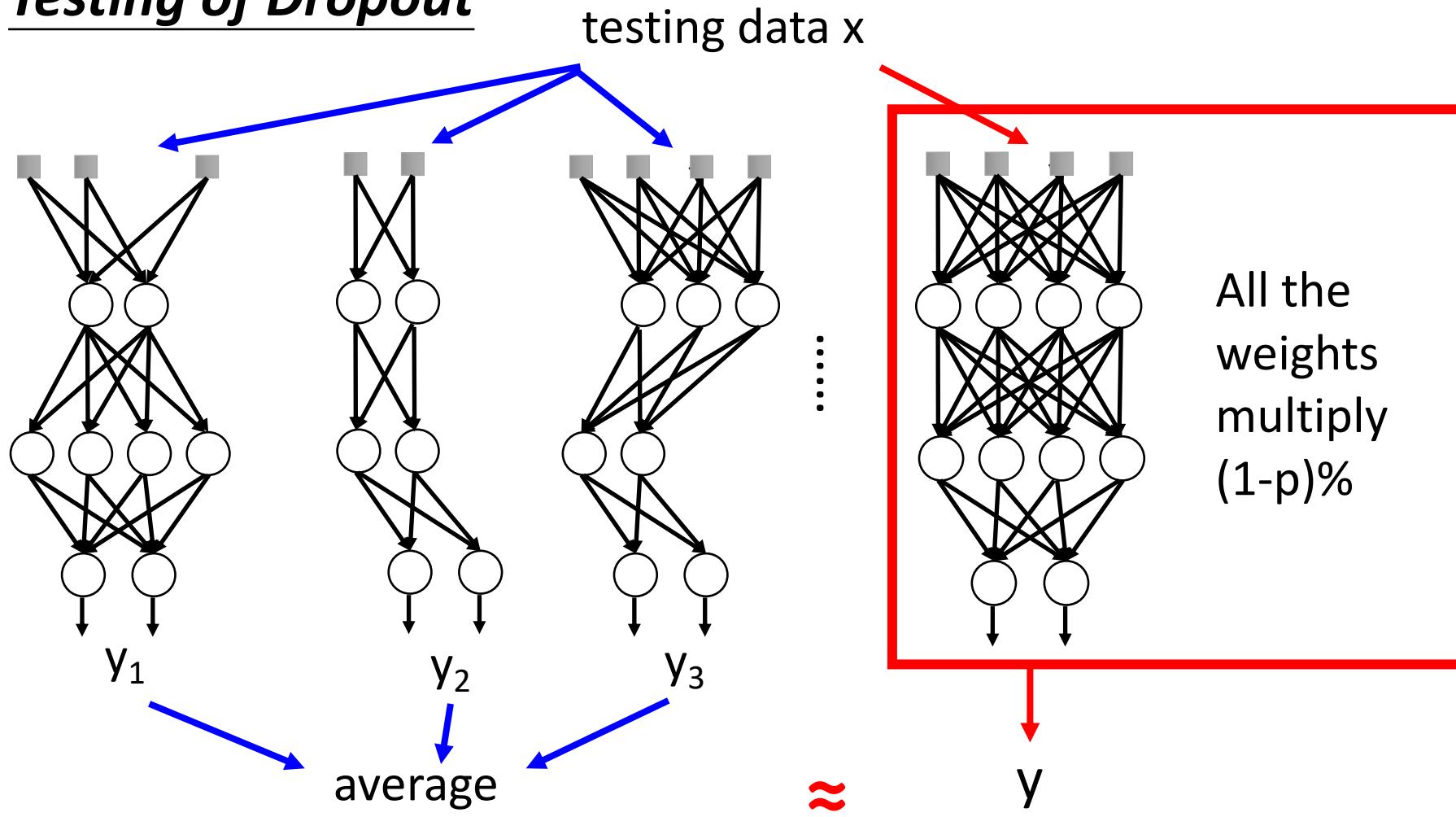


2^M possible
networks

- Using one mini-batch to train one network
- Some parameters in the network are shared

Dropout is a kind of ensemble.

Testing of Dropout



More about dropout

- More reference for dropout [Nitish Srivastava, JMLR'14] [Pierre Baldi, NIPS'13][Geoffrey E. Hinton, arXiv'12]
- Dropout works better with Maxout [Ian J. Goodfellow, ICML'13]
- Dropconnect [Li Wan, ICML'13]
 - Dropout delete neurons
 - Dropconnect deletes the connection between neurons
- Annealed dropout [S.J. Rennie, SLT'14]
 - Dropout rate decreases by epochs
- Standout [J. Ba, NISP'13]
 - Each neural has different dropout rate