CSC4007 Advanced Machine Learning

Lesson 08: Deep Learning

by Vien Ngo
EEECS / ECIT / DSSC

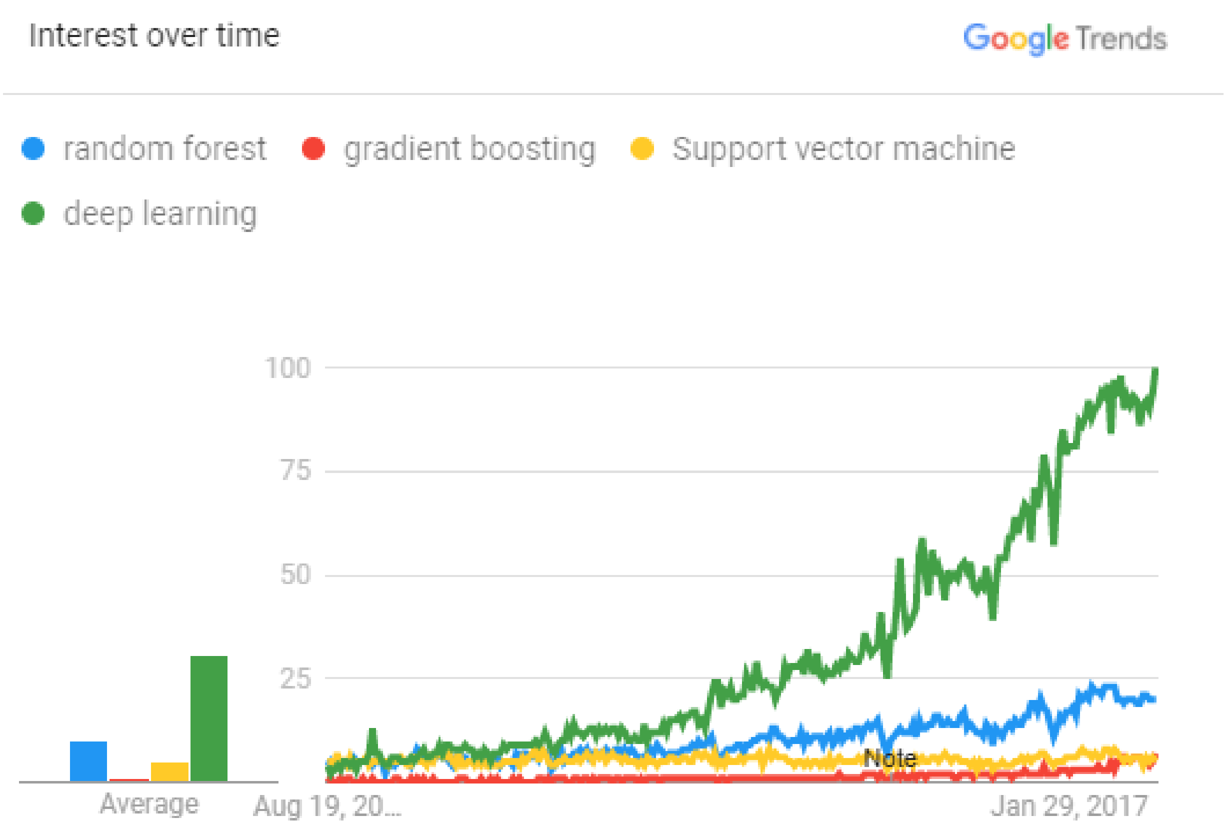
Slides adapted from Hung-yi Lee

Outline

- Neural network basics and representation
- Perceptron learning, multi-layer perceptron
- Neural network training: Backpropagation
- **Modern neural network architecture (a.k.a Deep learning):**
  - **Convolutional neural network (CNN)**
  - **Recurrent neural network (RNN), long-short term memory network (LSTM)**

Deep learning

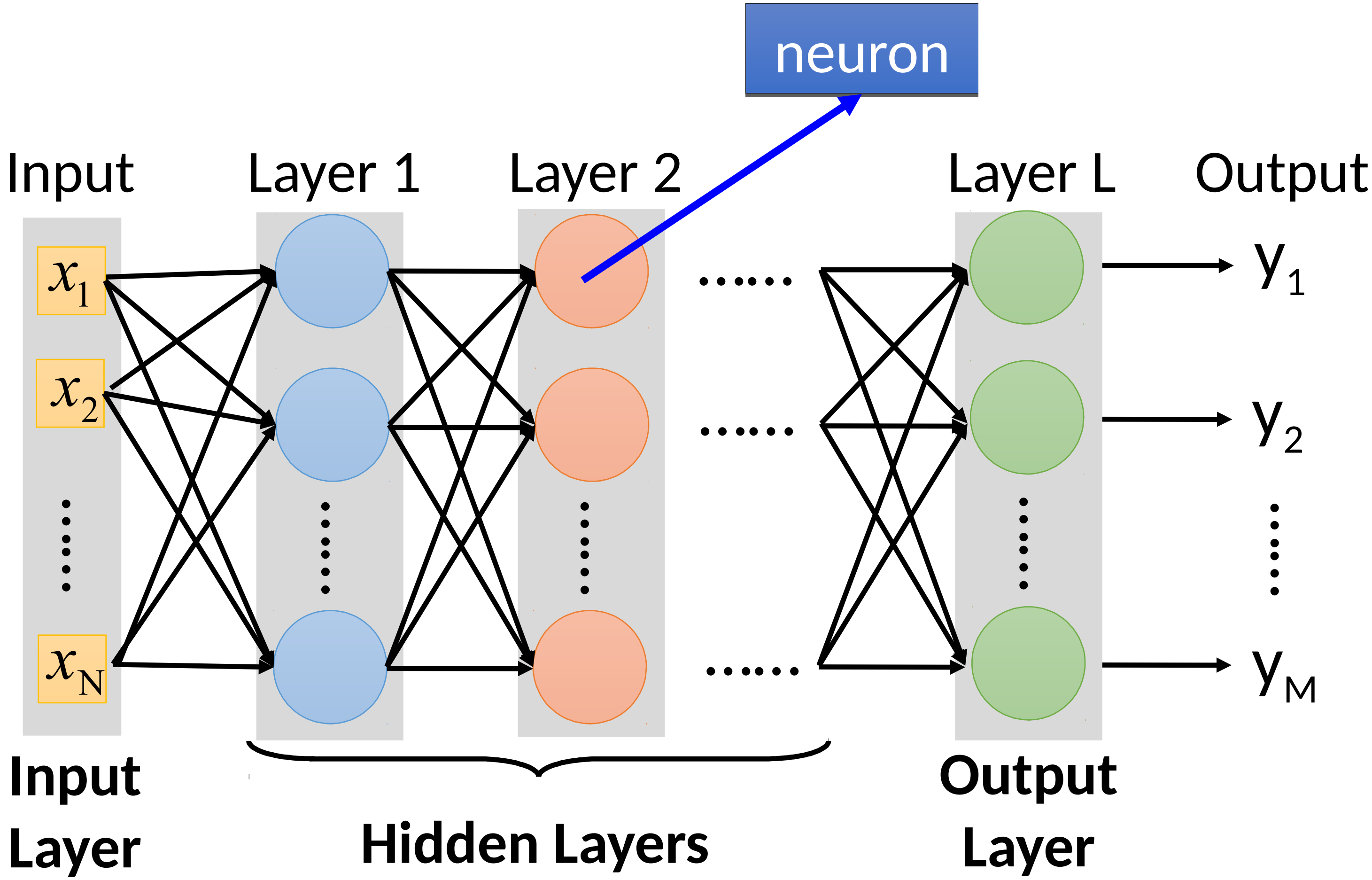
- Google Trends: attracts lots of attention.
  - Deep learning obtains many exciting results.



SAS Blogs

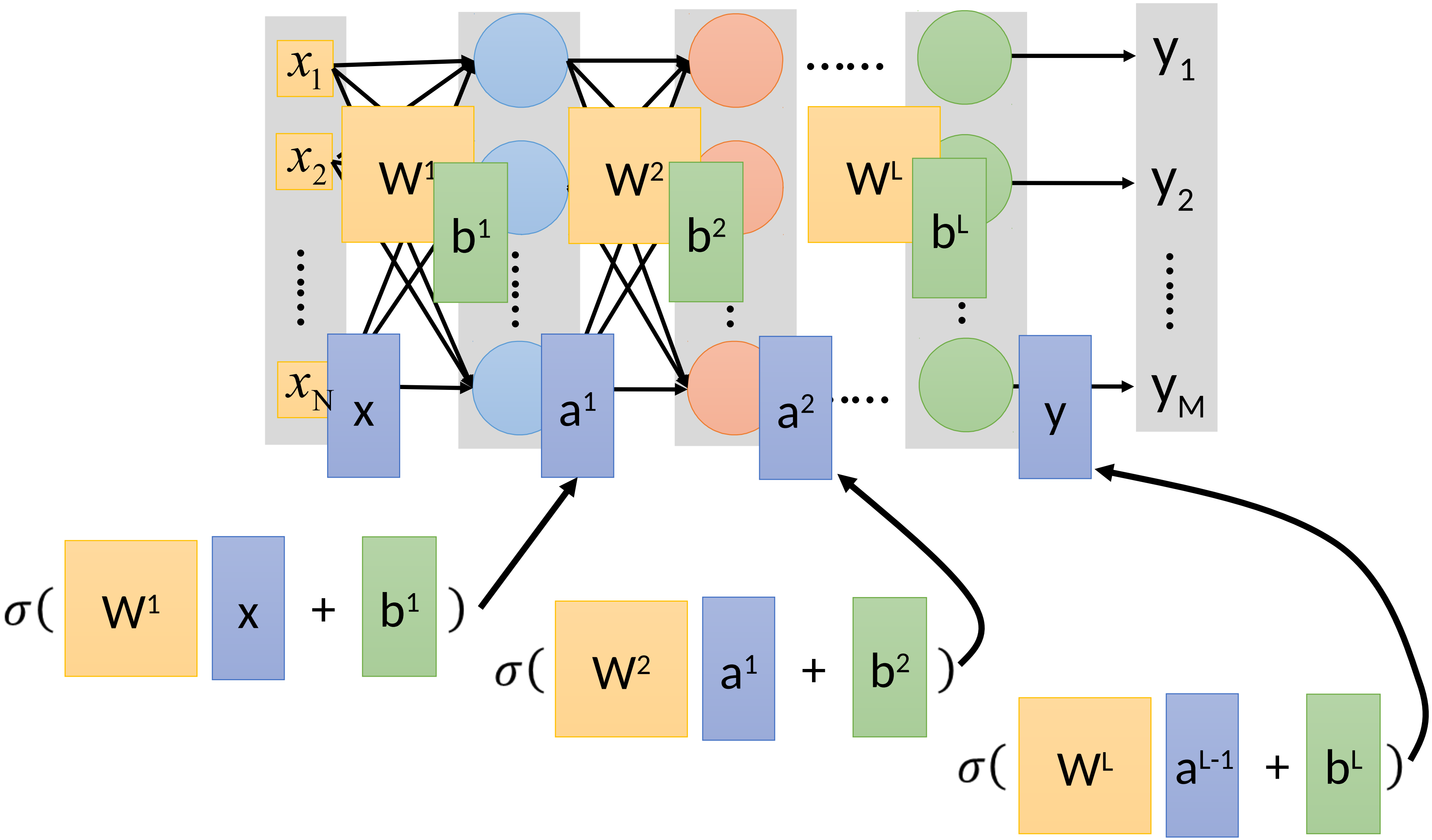
Deep Neural Network

Example: a fully connected deep NN

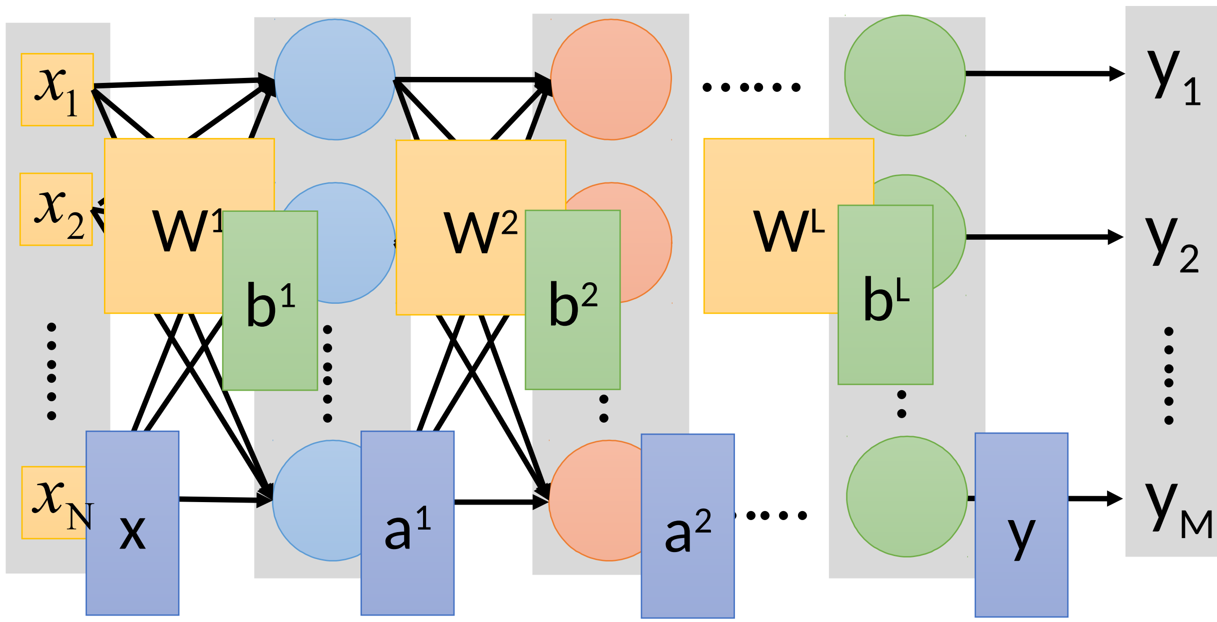


Deep means many hidden layers

Deep Neural Network



Deep Neural Network: Forward propagation



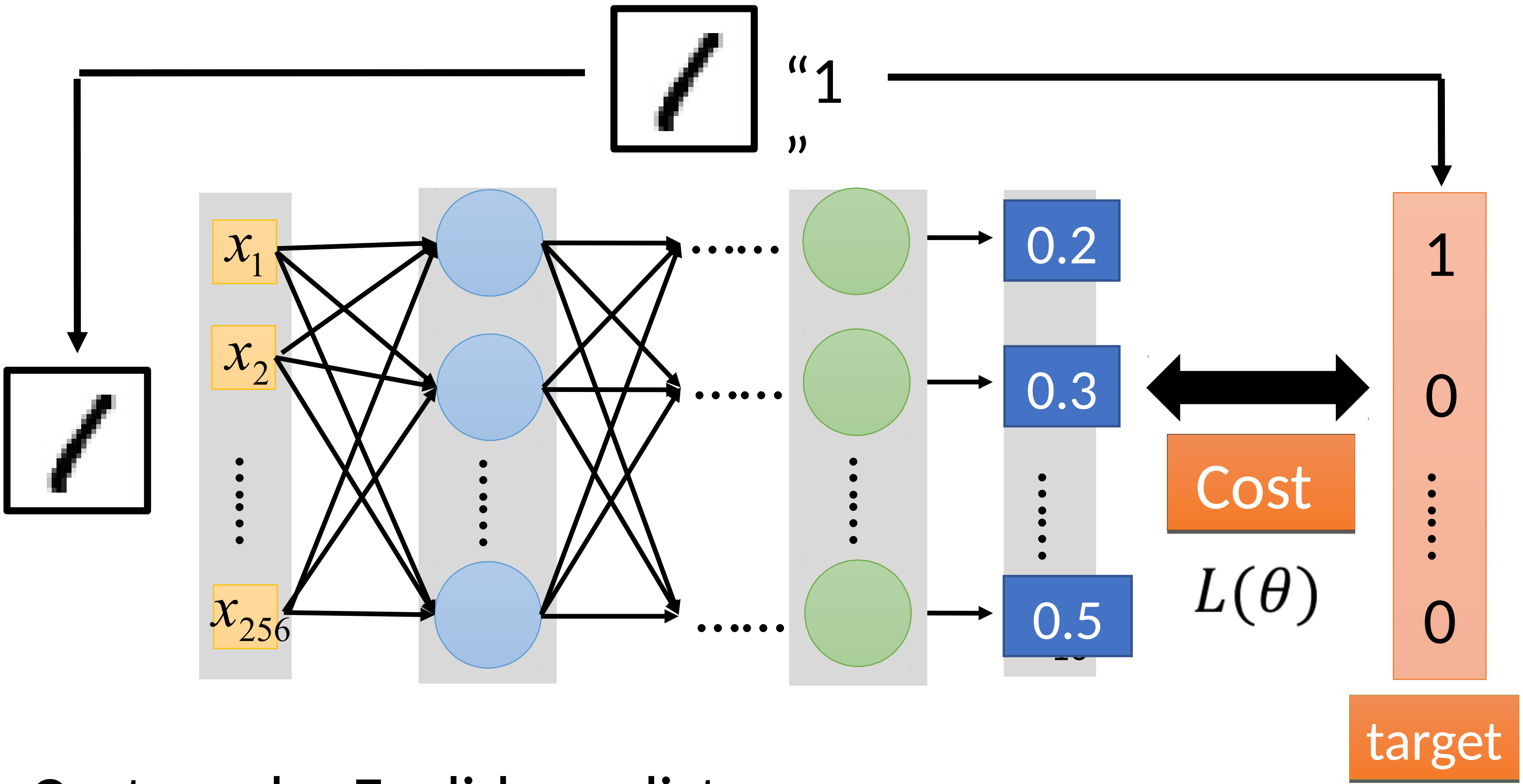
$$y = f(x)$$

Using parallel computing techniques to speed up matrix operation

$$= \sigma(W^L \cdots \sigma(W^2 \, \sigma(W^1 x + b^1) + b^2) \cdots + b^L)$$

Cost

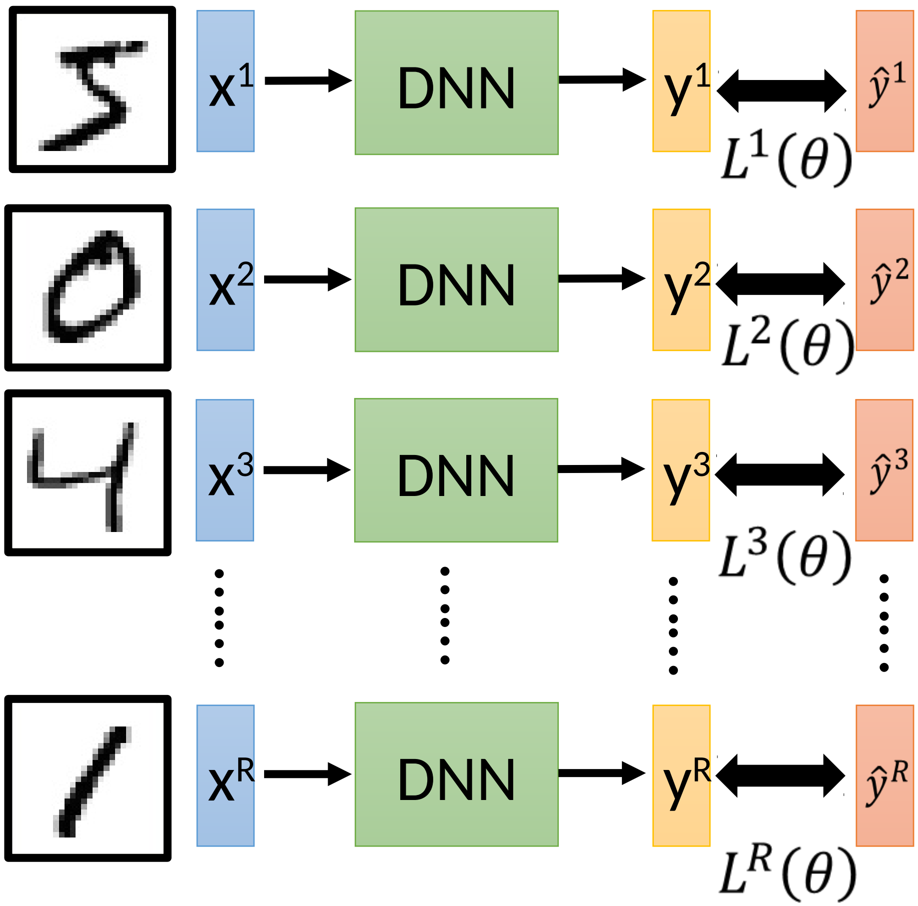
Given a set of network parameters θ, each example has a cost value.



Cost can be Euclidean distance or cross entropy of the network output and target

Total Cost

For all training data …



Total Cost:

$$C(\theta) = \sum_{r=1}^{R} L^r(\theta)$$

How bad the network parameters θ is on this task

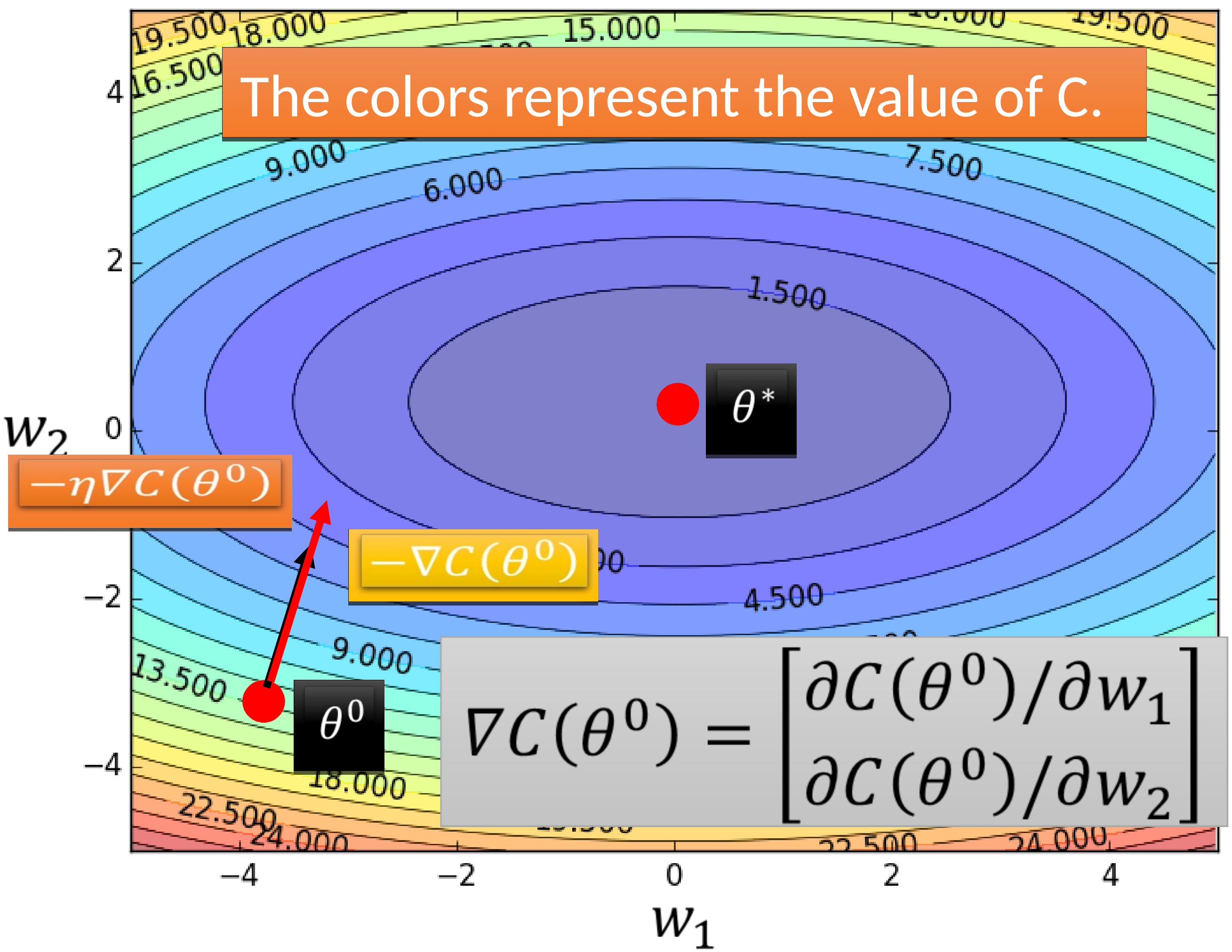
Find the network parameters θ^* that minimize this value

Gradient Descent

Assume there are only two parameters w_1 and w_2 in a network.

$$\theta = \{w_1, w_2\}$$

Error Surface



Randomly pick a starting point θ^0

Compute the negative gradient at θ^0

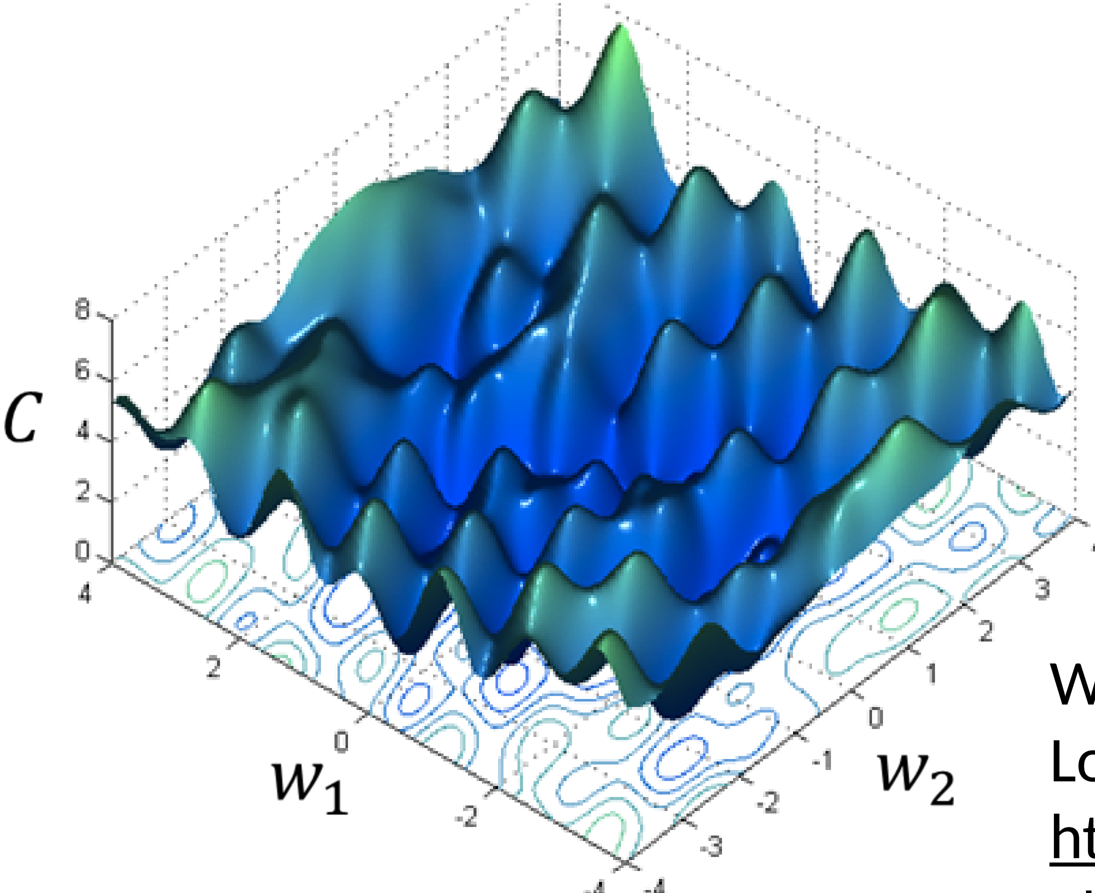
$\rightarrow -\nabla C(\theta^0)$

Times the learning rate η

$\rightarrow -\eta \nabla C(\theta^0)$

Training DNN via Backpropagation

- Gradient descent never guarantee global minima

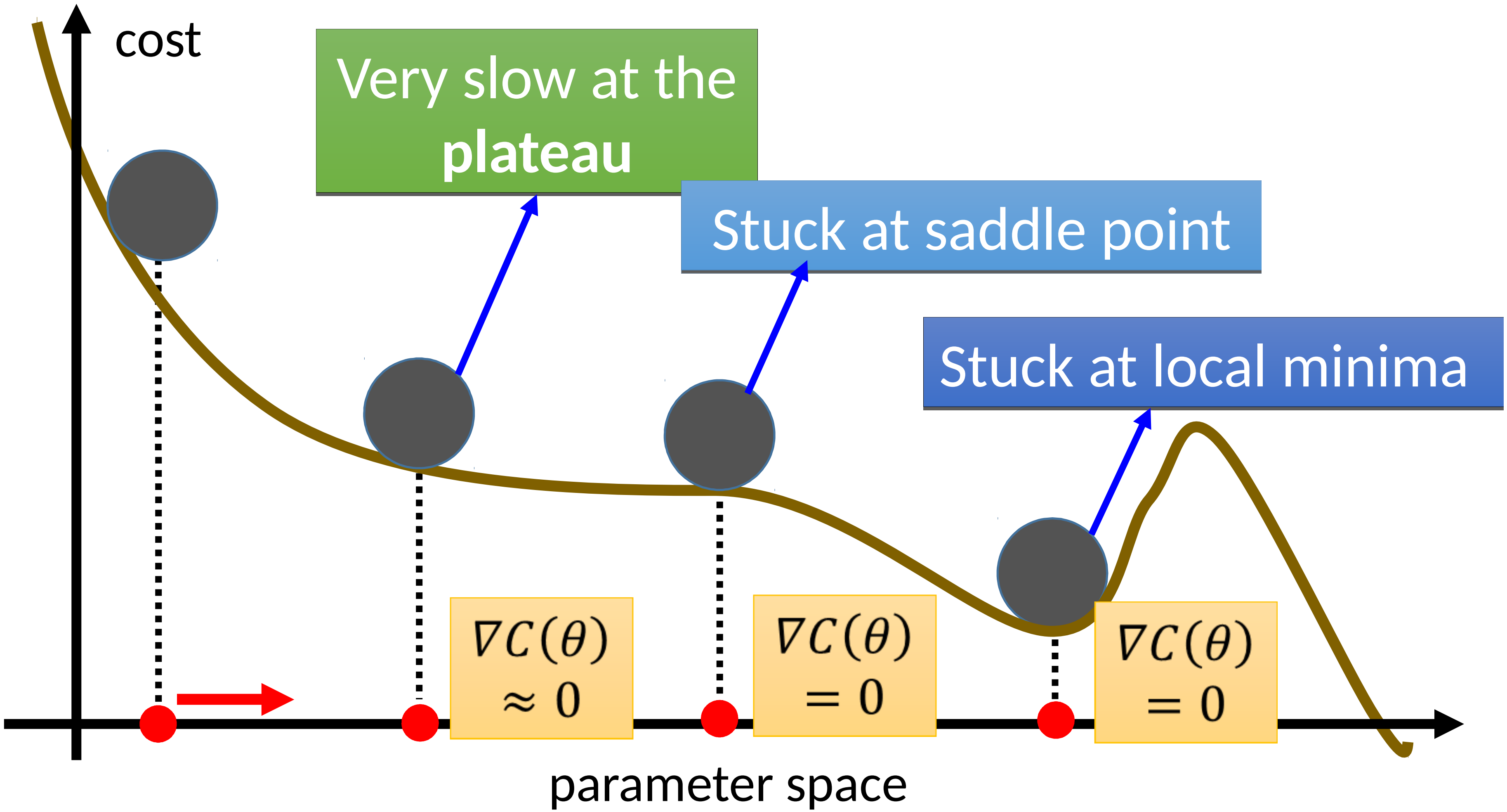


Different initial point θ^0

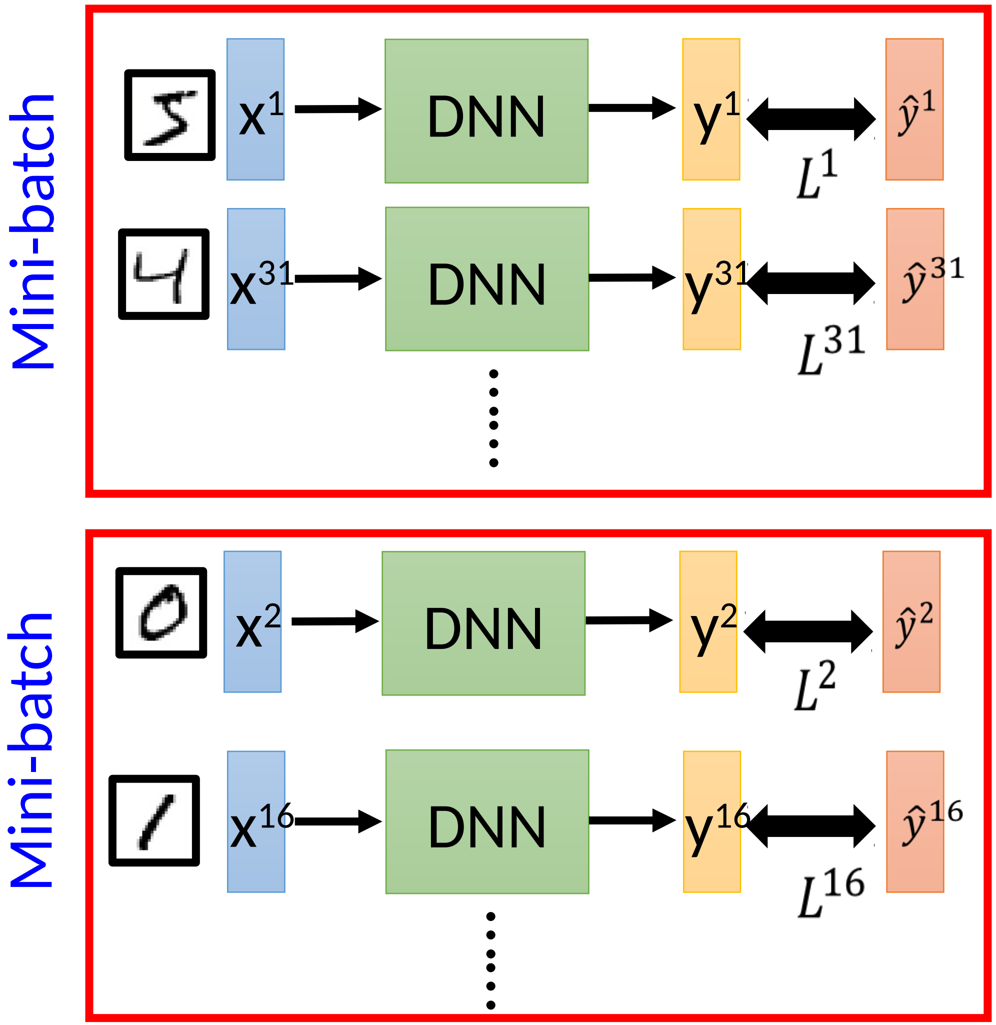
Reach different minima, so different results

Who is Afraid of Non-Convex Loss Functions?
http://videolectures.net/eml07_lecun_wia/

Besides local minima ……



Training with Mini-batch



- Randomly initialize θ^0
- Pick the 1st batch

$$C = L^1 + L^{31} + \cdots$$

$$\theta^1 \leftarrow \theta^0 - \eta \nabla C(\theta^0)$$

- Pick the 2nd batch

$$C = L^2 + L^{16} + \cdots$$

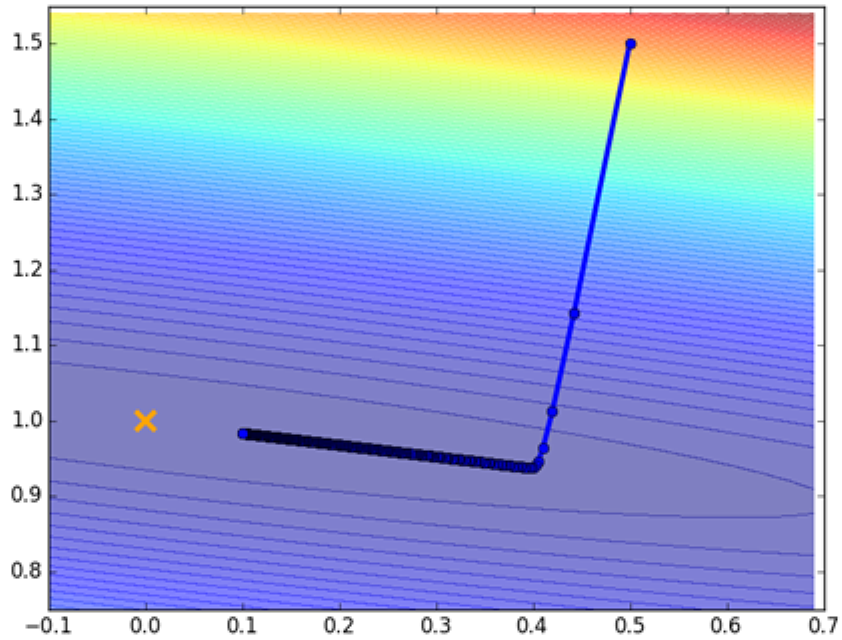
$$\theta^2 \leftarrow \theta^1 - \eta \nabla C(\theta^1)$$

$$\vdots$$

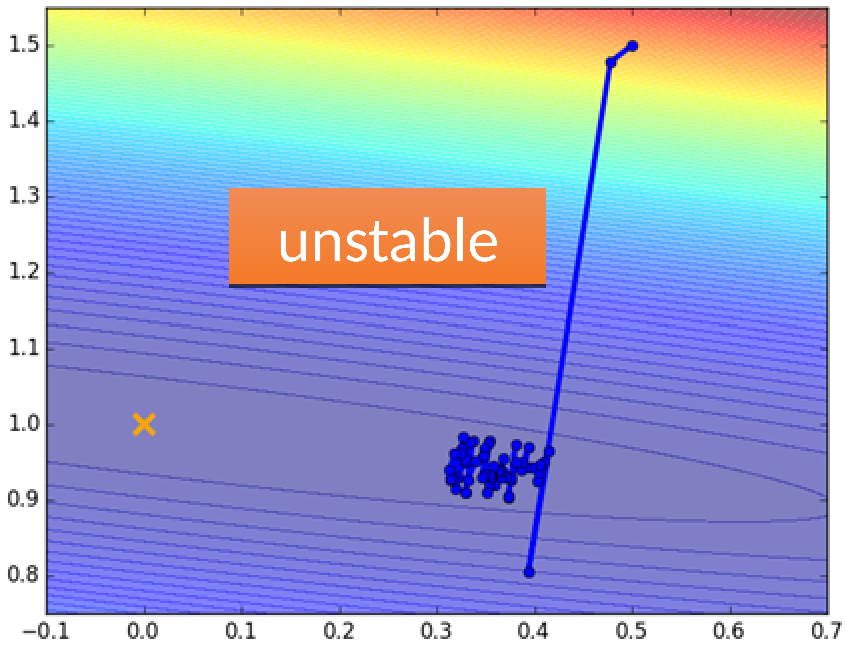
C is different each time when we update parameters!

Training with Mini-batch

Original Gradient Descent



With Mini-batch



The colors represent the total C on all training data.

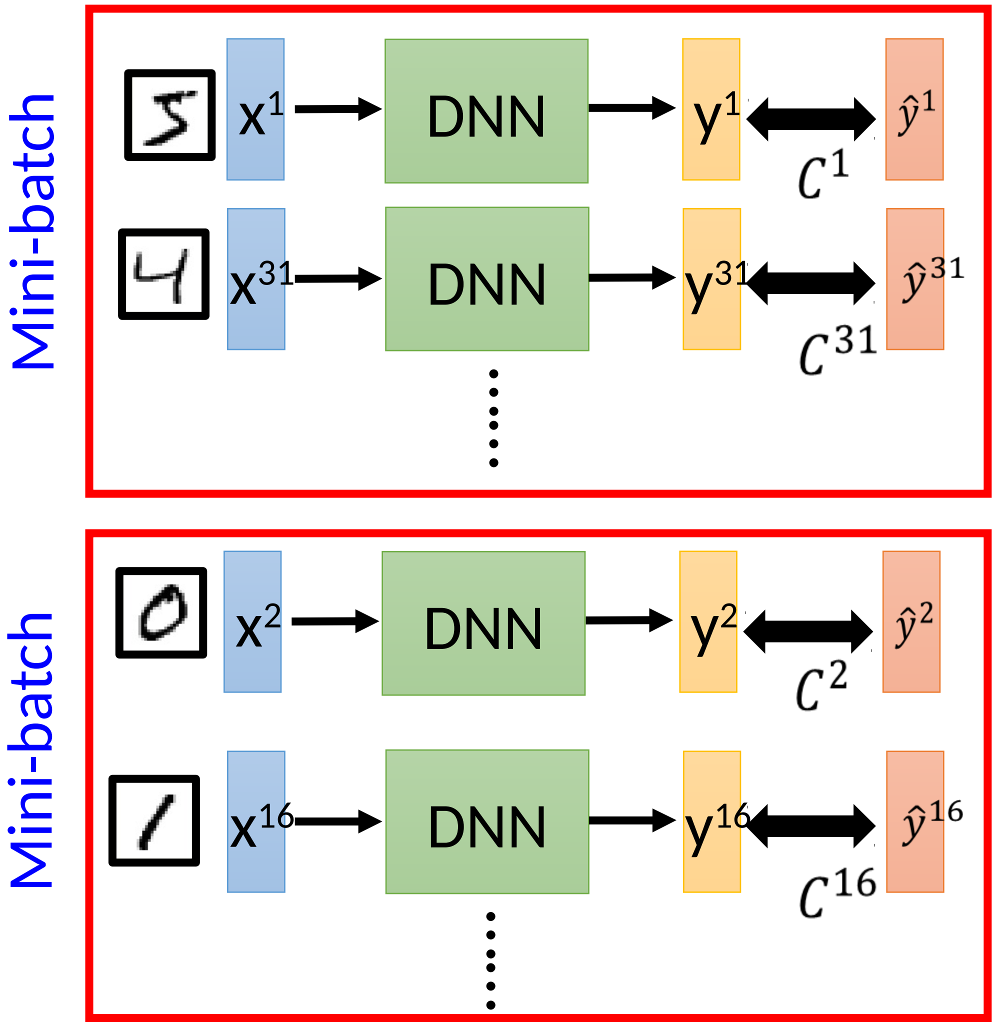
Training with Mini-batch

Faster | Better!

- Randomly initialize θ^0
- Pick the 1st batch
  $$C = C^1 + C^{31} + \cdots$$
  $$\theta^1 \leftarrow \theta^0 - \eta \nabla C(\theta^0)$$
- Pick the 2nd batch
  $$C = C^2 + C^{16} + \cdots$$
  $$\theta^2 \leftarrow \theta^1 - \eta \nabla C(\theta^1)$$
  $\vdots$
- Until all mini-batches have been picked

one epoch

Repeat the above process



Backpropagation

- A network can have millions of parameters.
  - Backpropagation is the way to compute the gradients efficiently (not today)
  - Ref: http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/Lecture/DNN%20backprop.ecm.mp4/index.html
- Many toolkits can compute the gradients automatically

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.

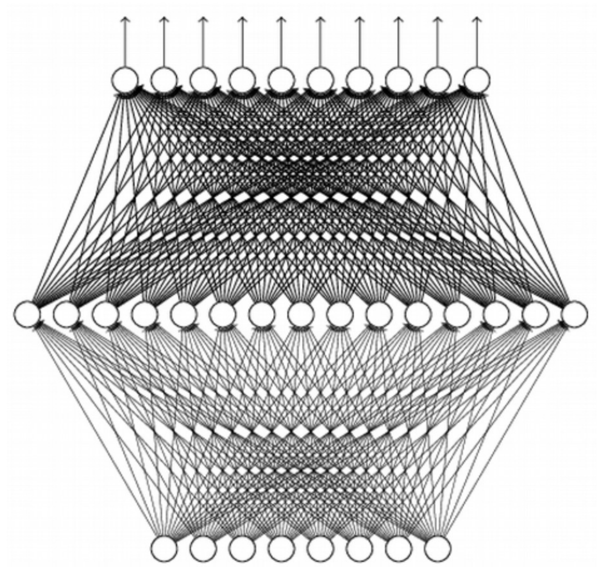
Universality Theorem

Any continuous function f

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

Can be realized by a network with one hidden layer

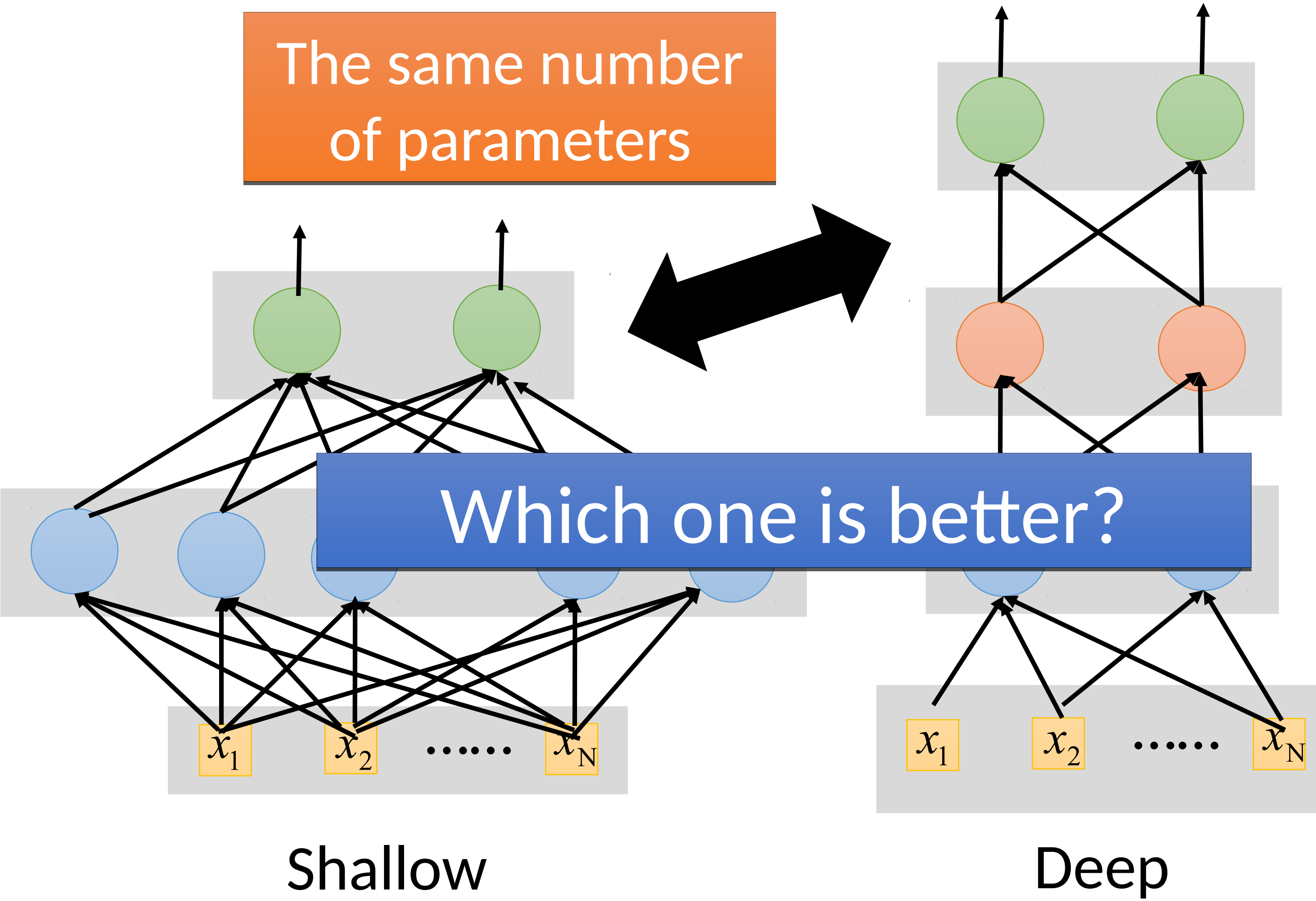
(given **enough** hidden neurons)



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

Why "Deep" neural network not "Fat" neural network?

Fat + Short v.s. Thin + Tall



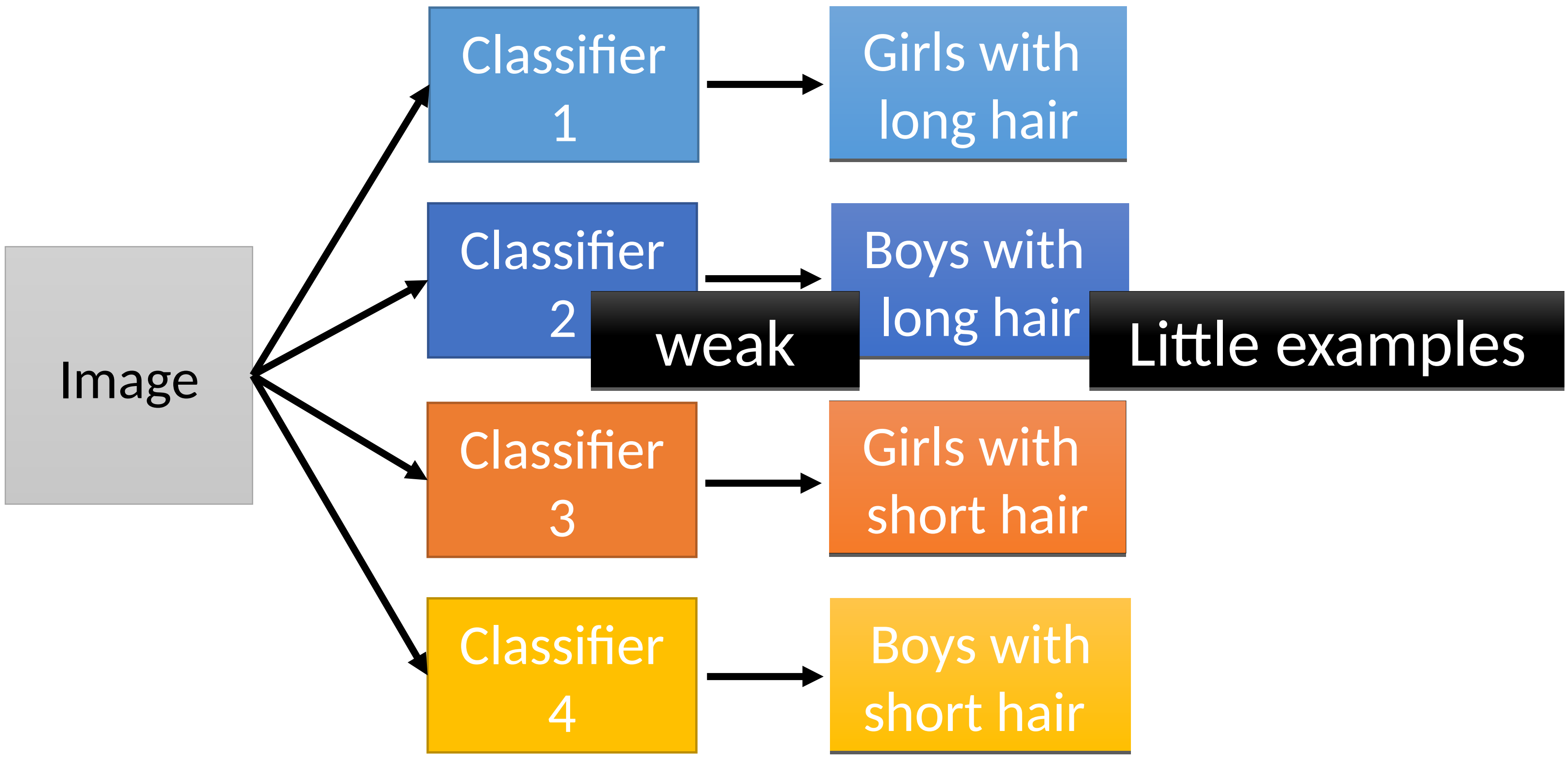
Fat + Short v.s. Thin + Tall

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.

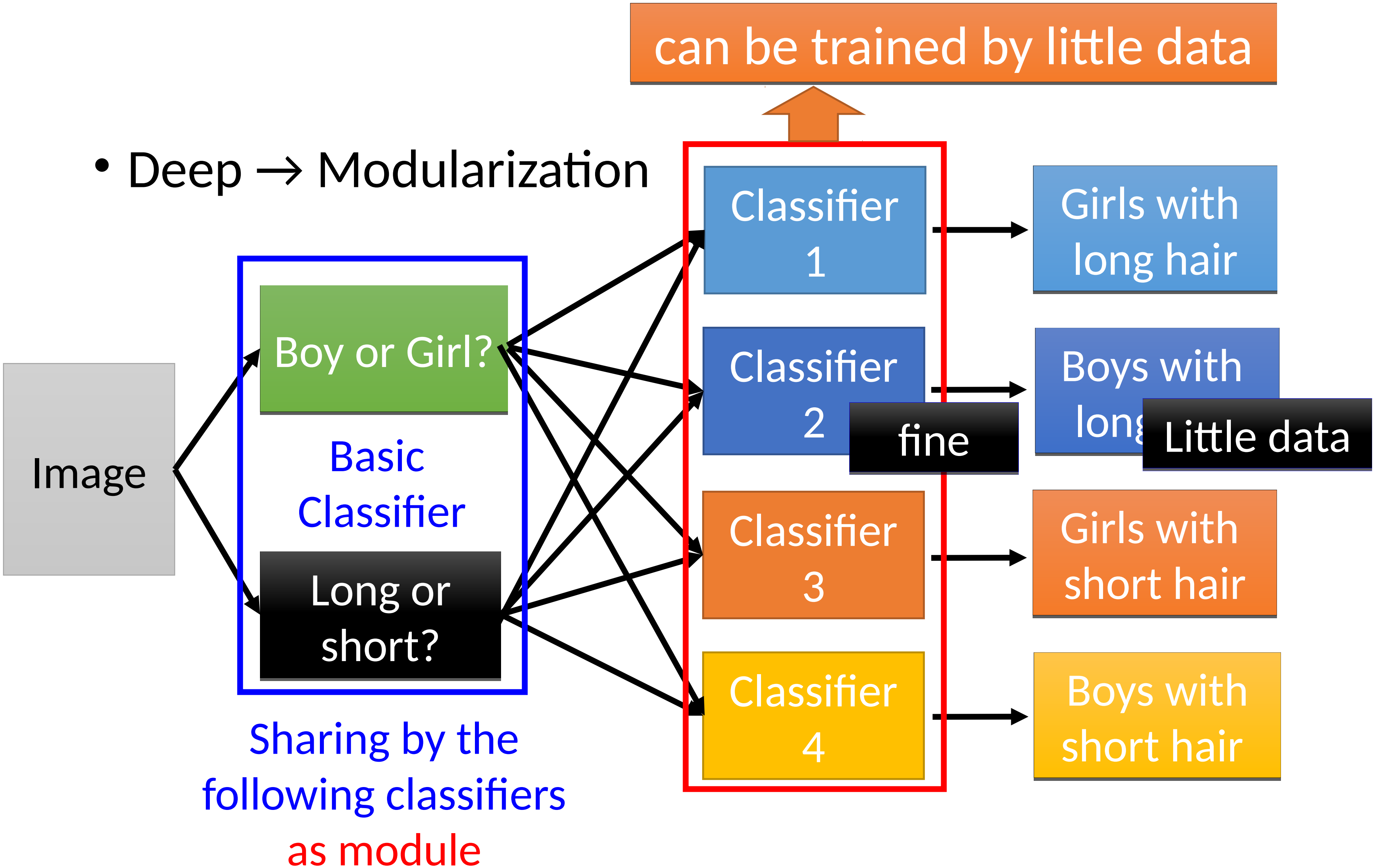
Why Deep?

- Deep → Modularization



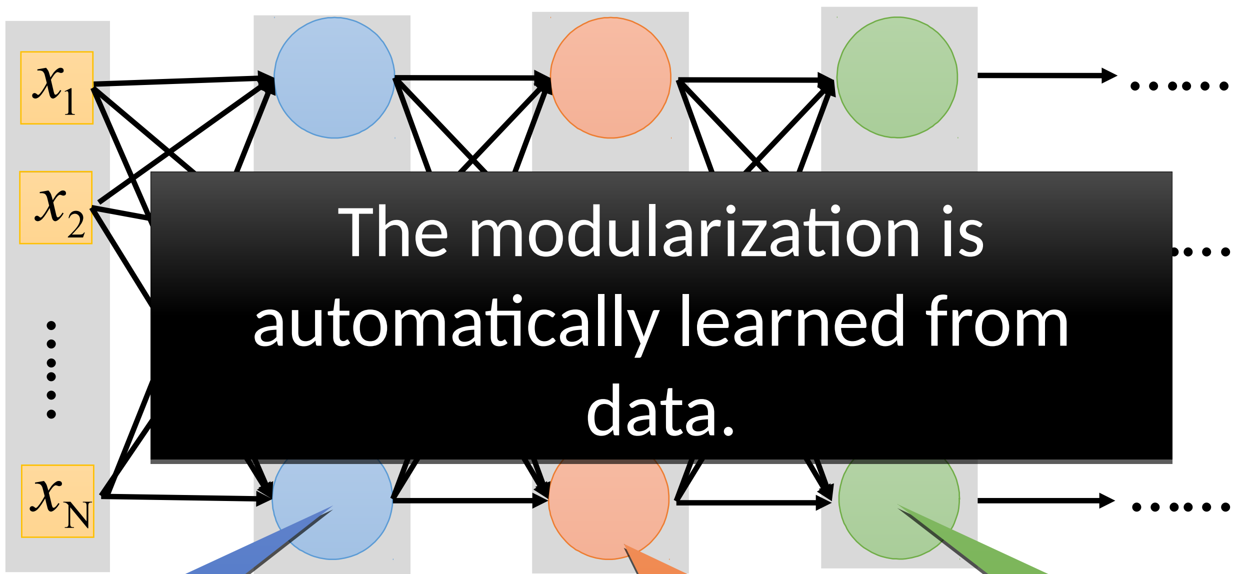
Why Deep?

- Deep → Modularization



Why Deep?

- Deep → Modularization → Less training data?



The modularization is automatically learned from data.

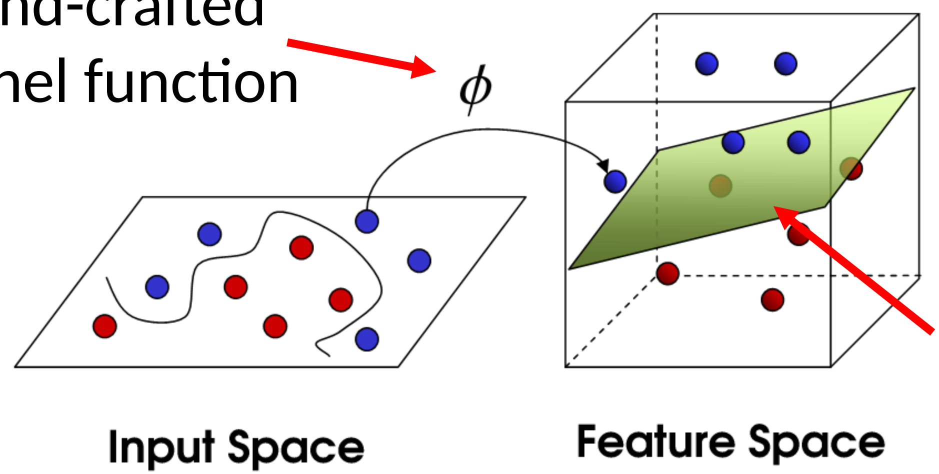
The most basic classifiers

Use 1st layer as module to build classifiers

Use 2nd layer as module ......

Hand-crafted kernel function

SVM



Apply simple classifier

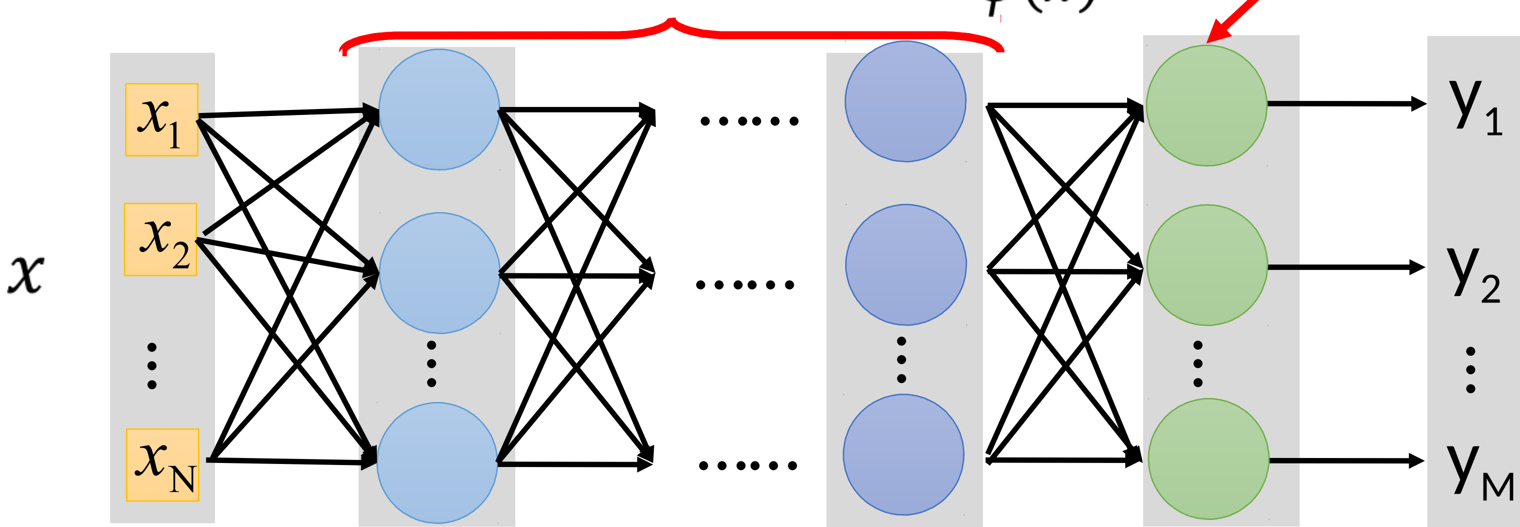
Source of image: http://www.gipsa-lab.grenoble-inp.fr/transfert/seminaire/455_Kadri2013Gipsa-lab.pdf

Deep Learning

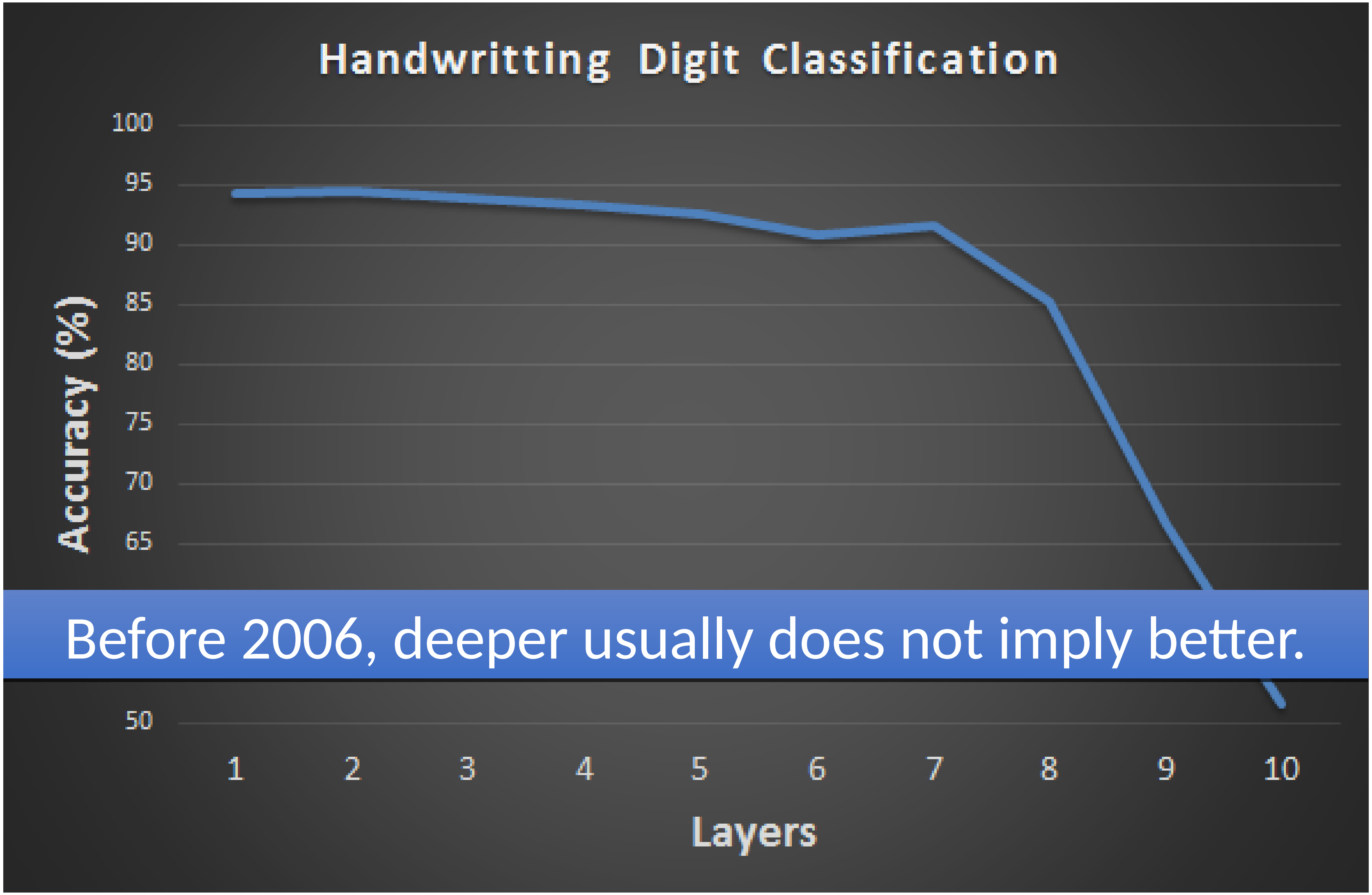
Learnable kernel

$\phi(x)$

simple classifier

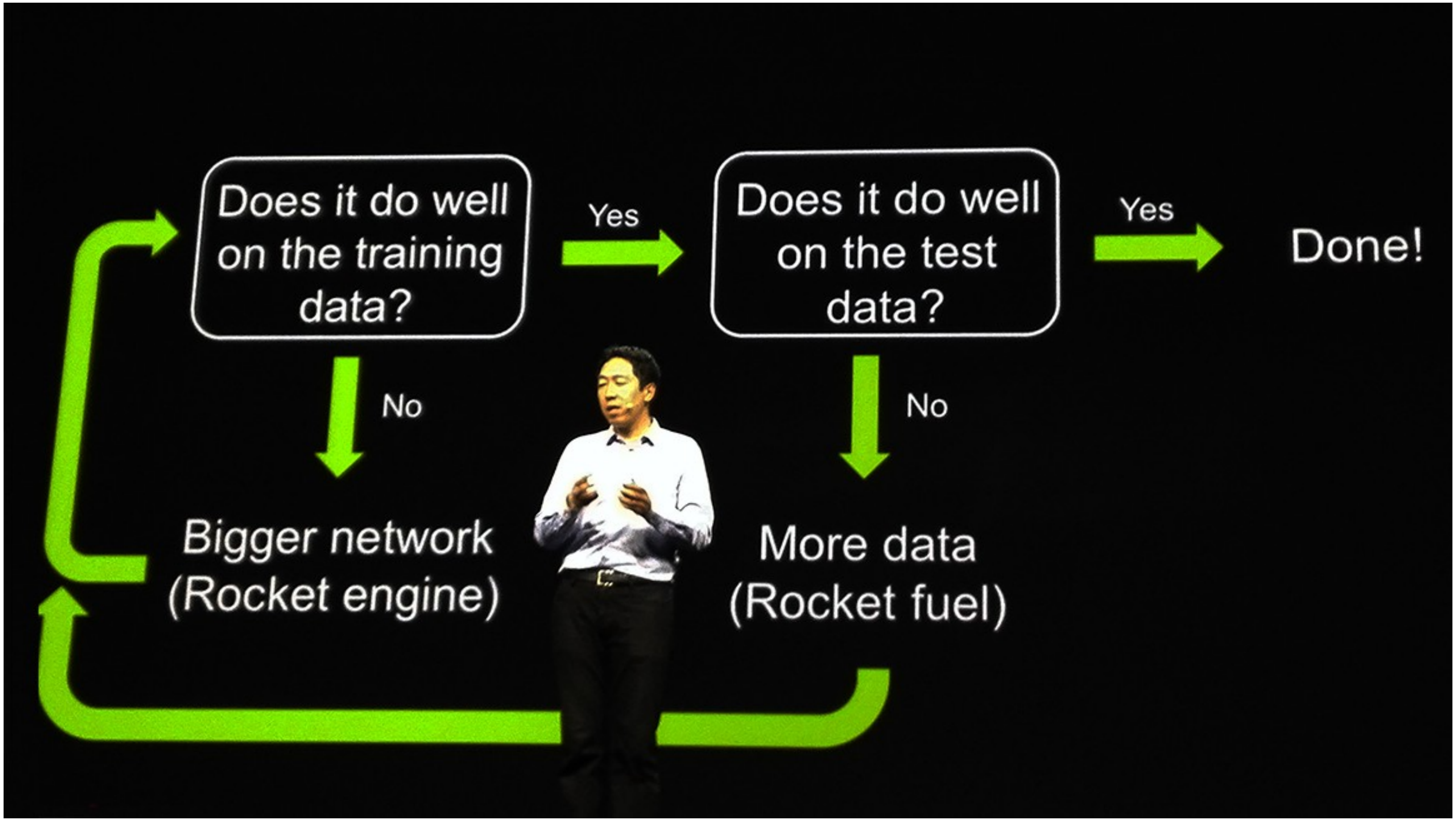


Hard to get the power of Deep



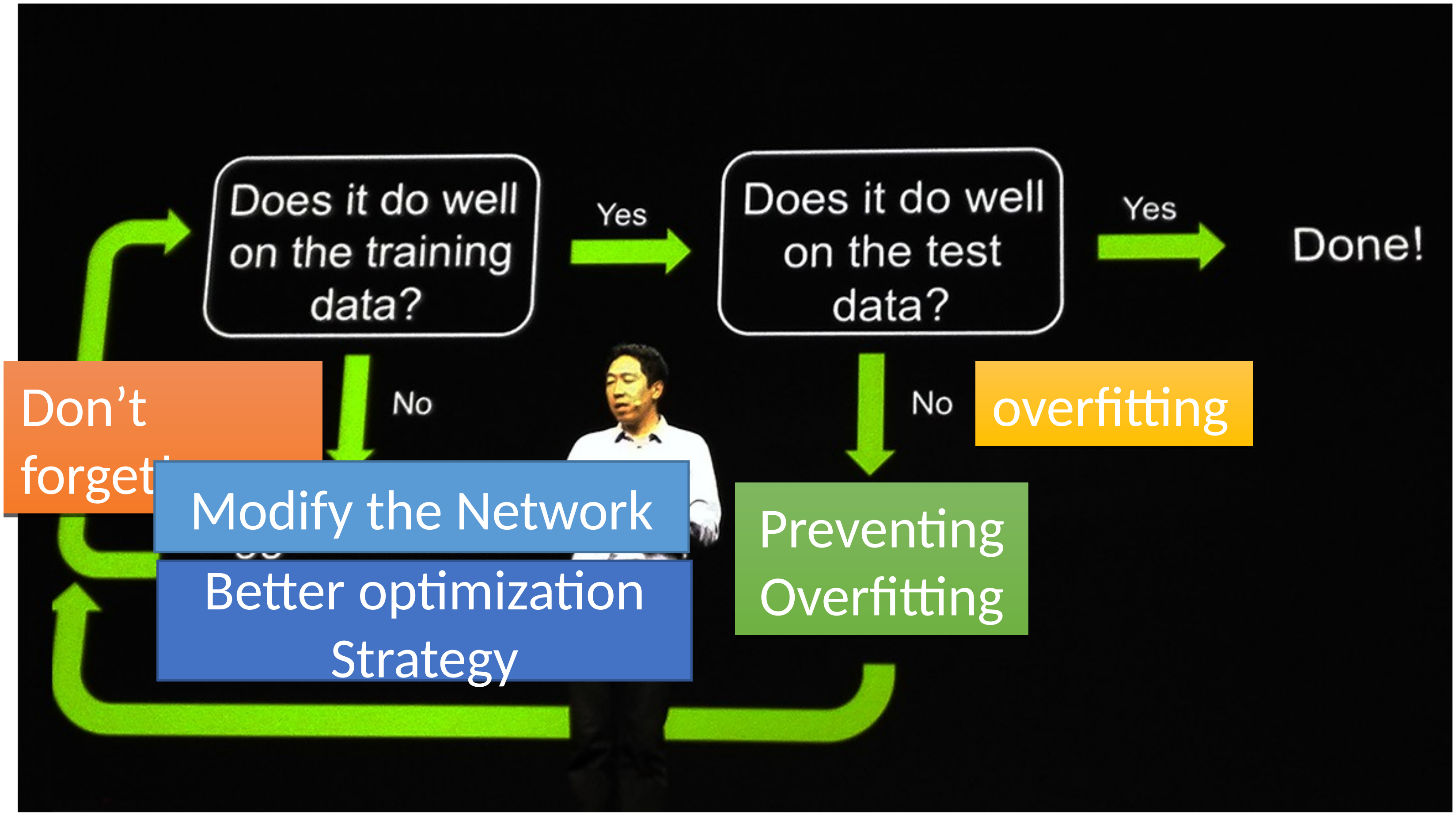
Before 2006, deeper usually does not imply better.

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



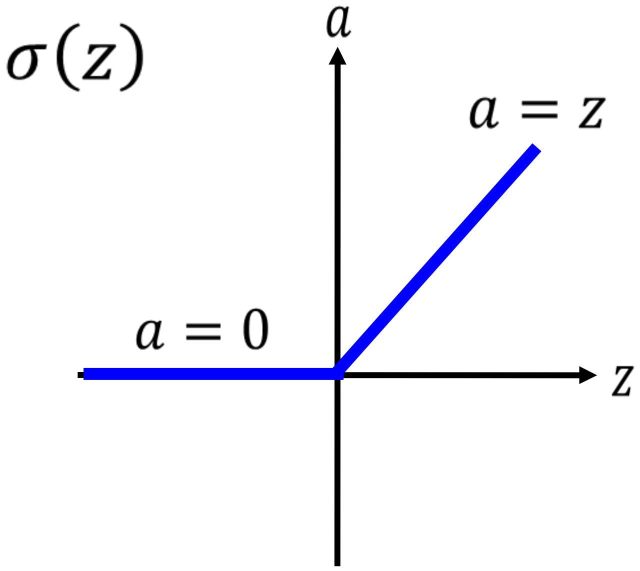
http://www.gizmodo.com.au/2015/04/the-basic-recipe-for-machine-learning-explained-in-a-single-powerpoint-slide/

Training DNN

- New Activation Function
- Adaptive Learning Rate
- Network Regularization

ReLU

- Rectified Linear Unit (ReLU)

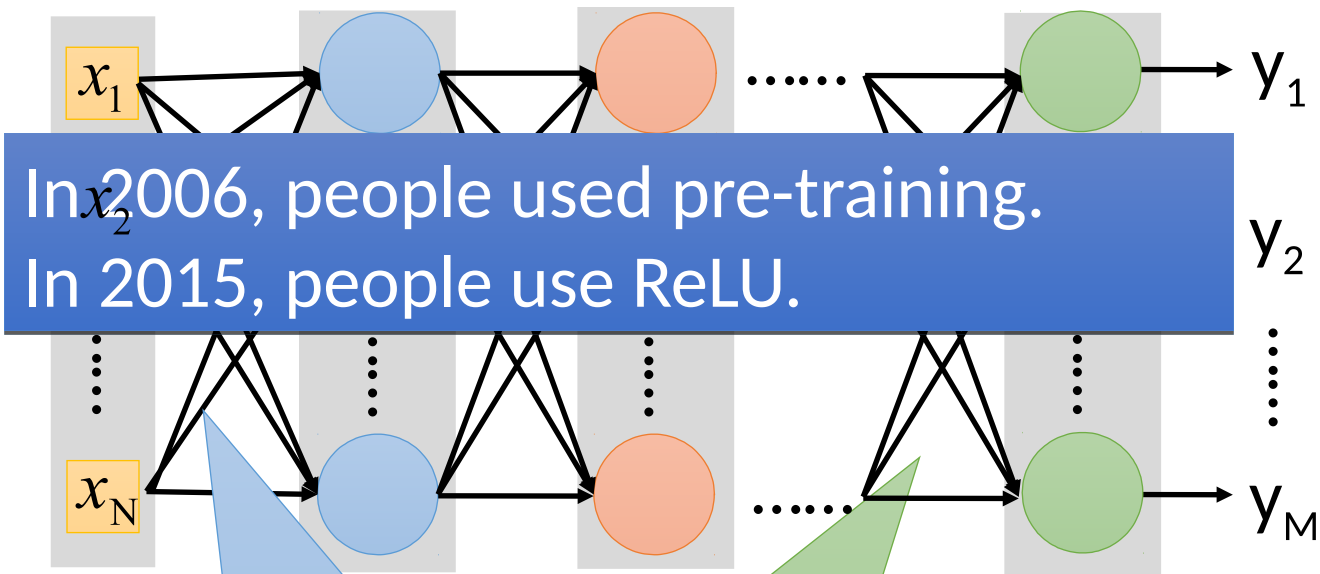


[Xavier Glorot, AISTATS'11]
[Andrew L. Maas, ICML'13]
[Kaiming He, arXiv'15]

Reason:

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

Vanishing Gradient Problem



In 2006, people used pre-training.
In 2015, people use ReLU.

Smaller gradients

Learn very slow

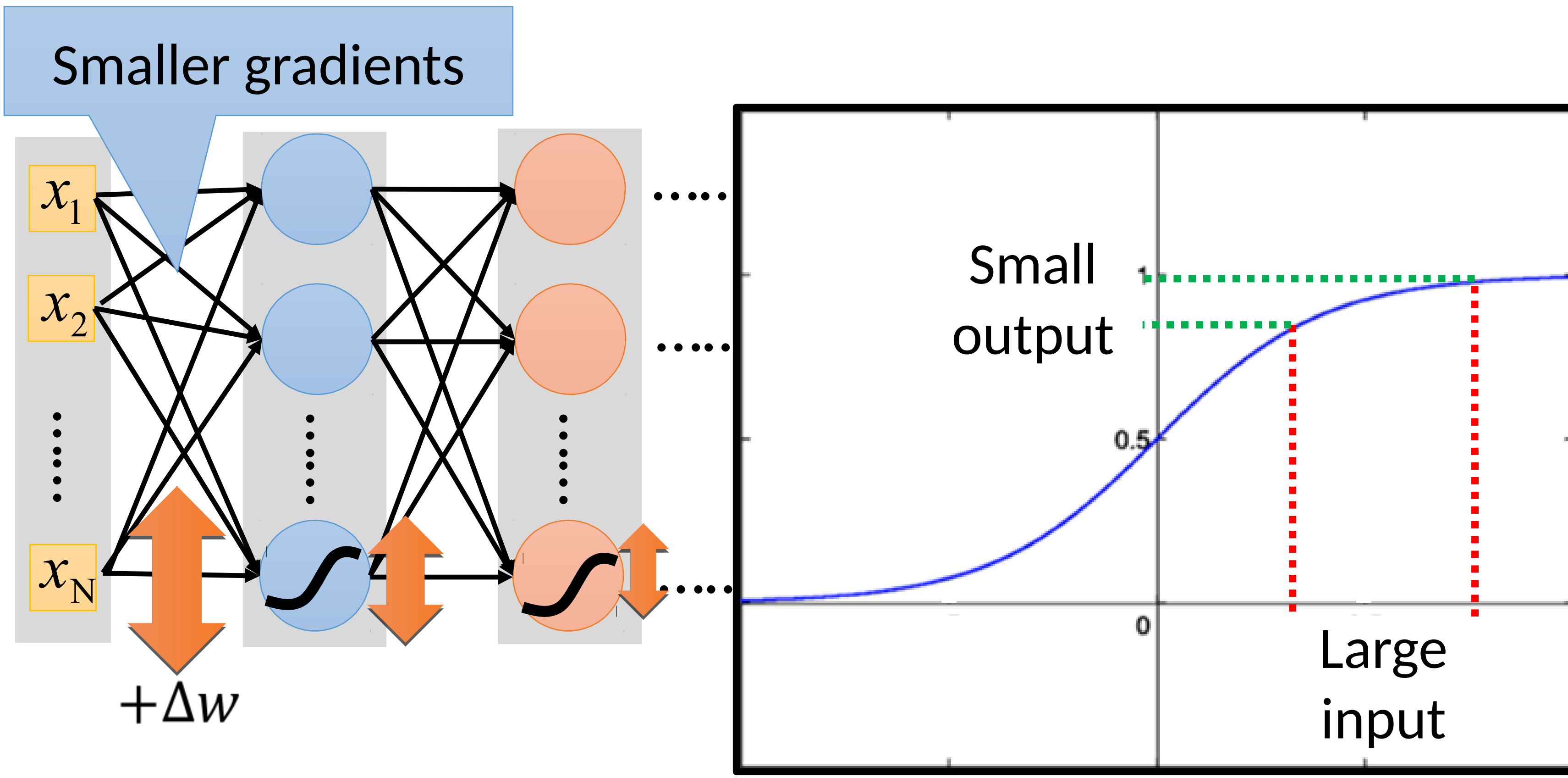
Almost random

Larger gradients

Learn very fast

Already converge

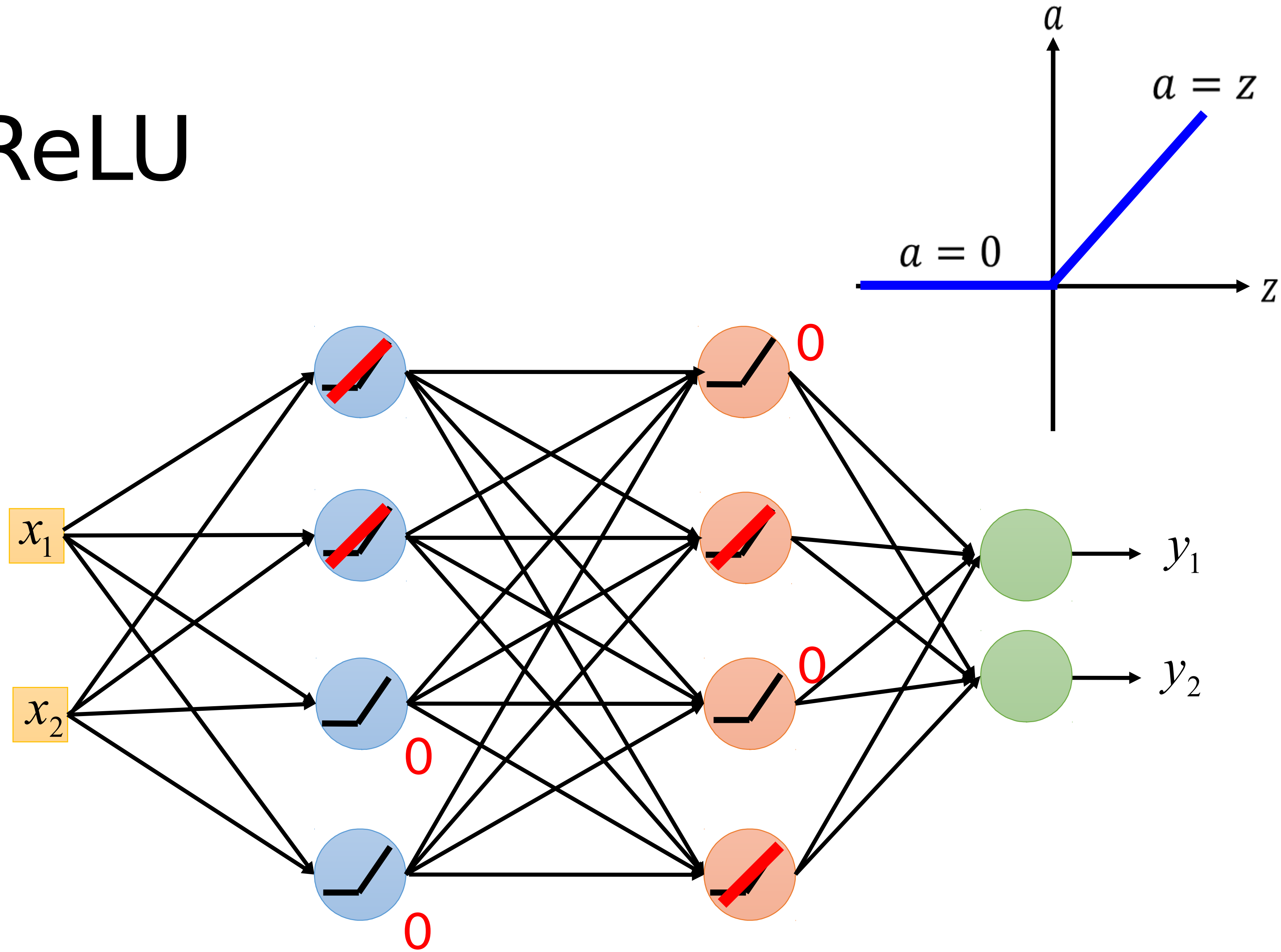
Vanishing Gradient Problem



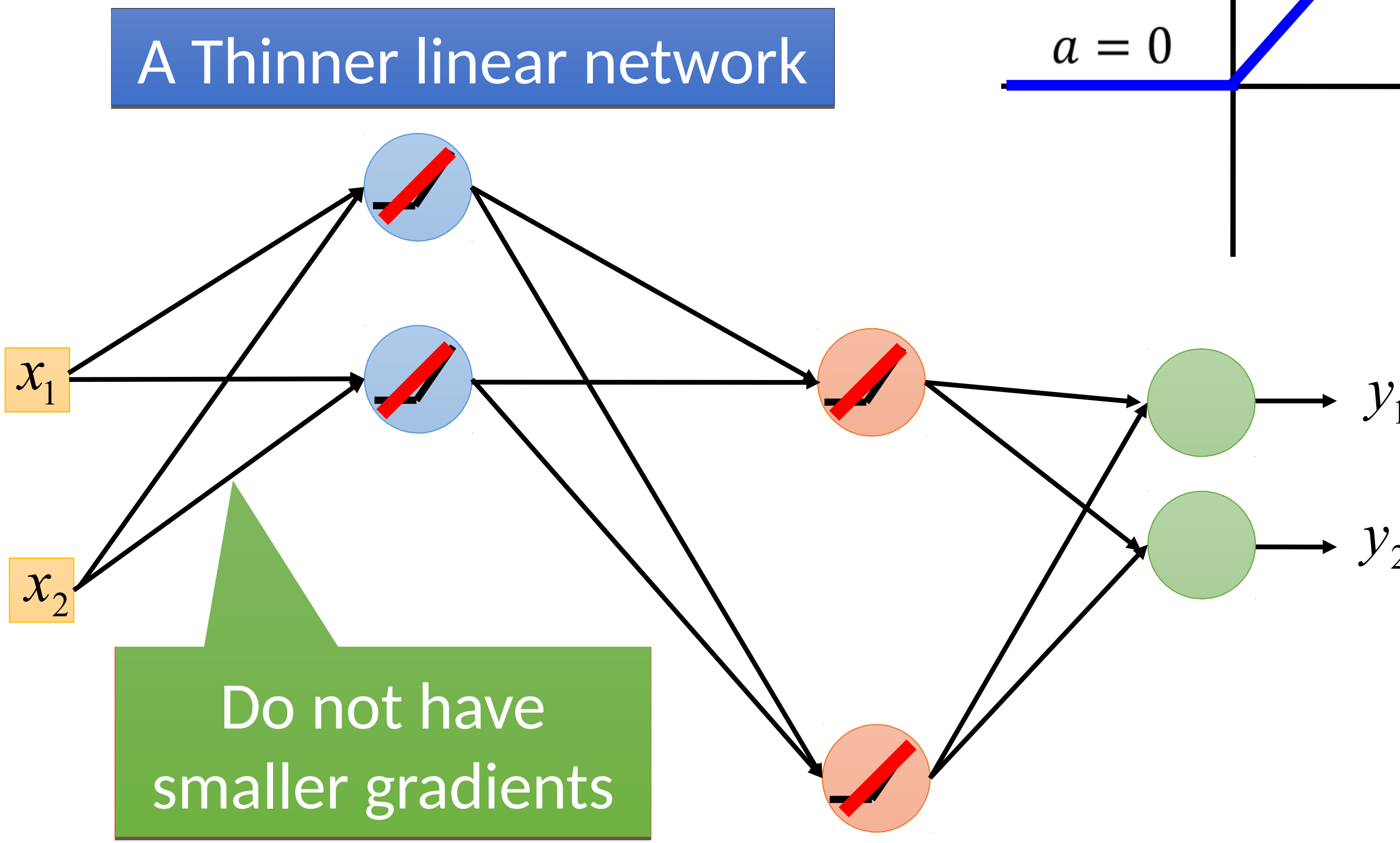
Intuitive way to compute the gradient …

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

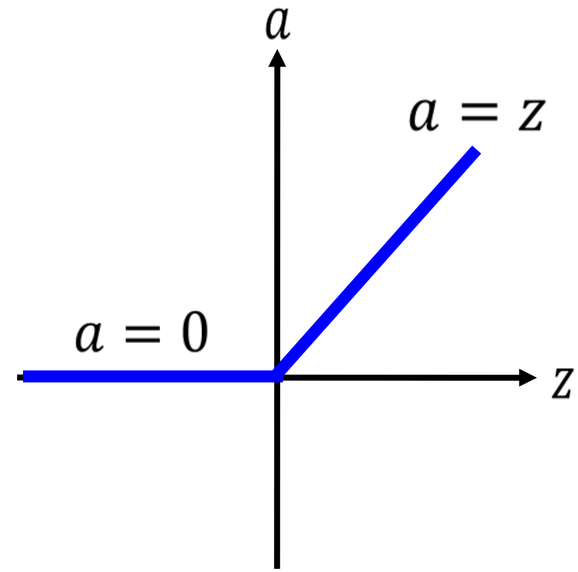
ReLU



ReLU



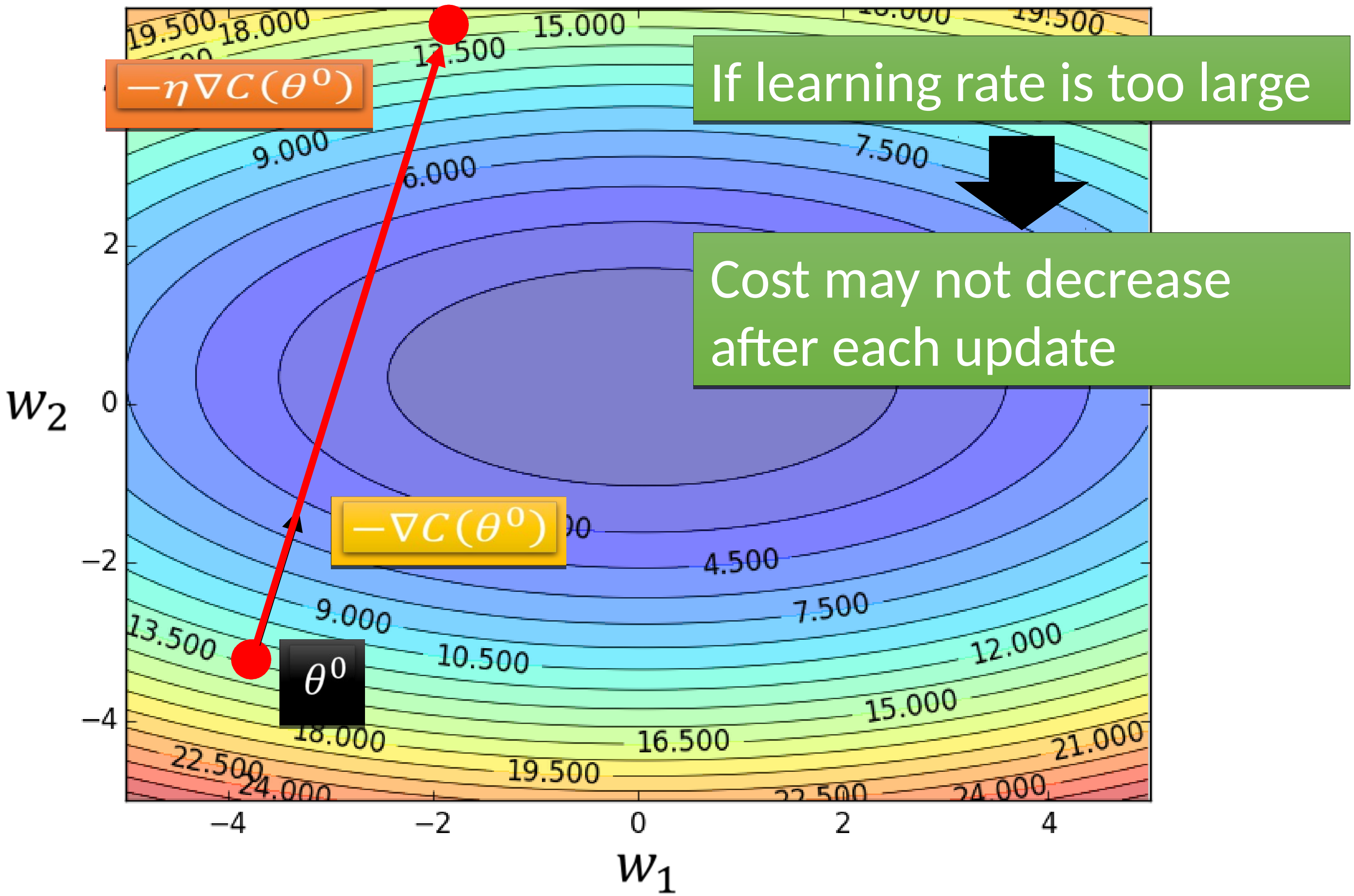
A Thinner linear network
x_1
x_2
y_1
y_2
Do not have smaller gradients



a
$a = z$
$a = 0$
z

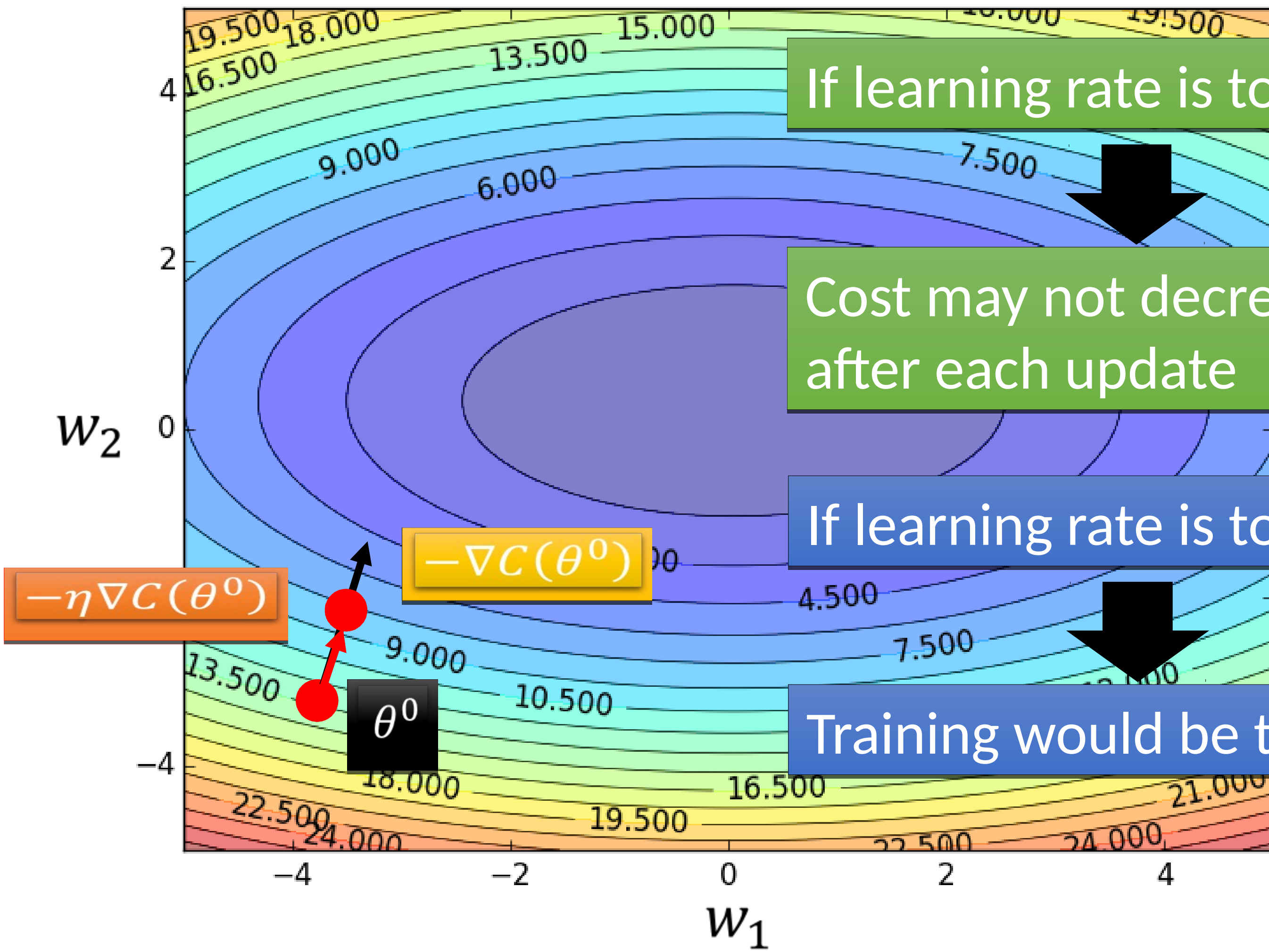
Learning Rate

Set the learning rate η carefully



Learning Rate

Can we give different parameters different learning rates?



If learning rate is too large

Cost may not decrease after each update

If learning rate is too small

Training would be too slow

Not the whole story ……

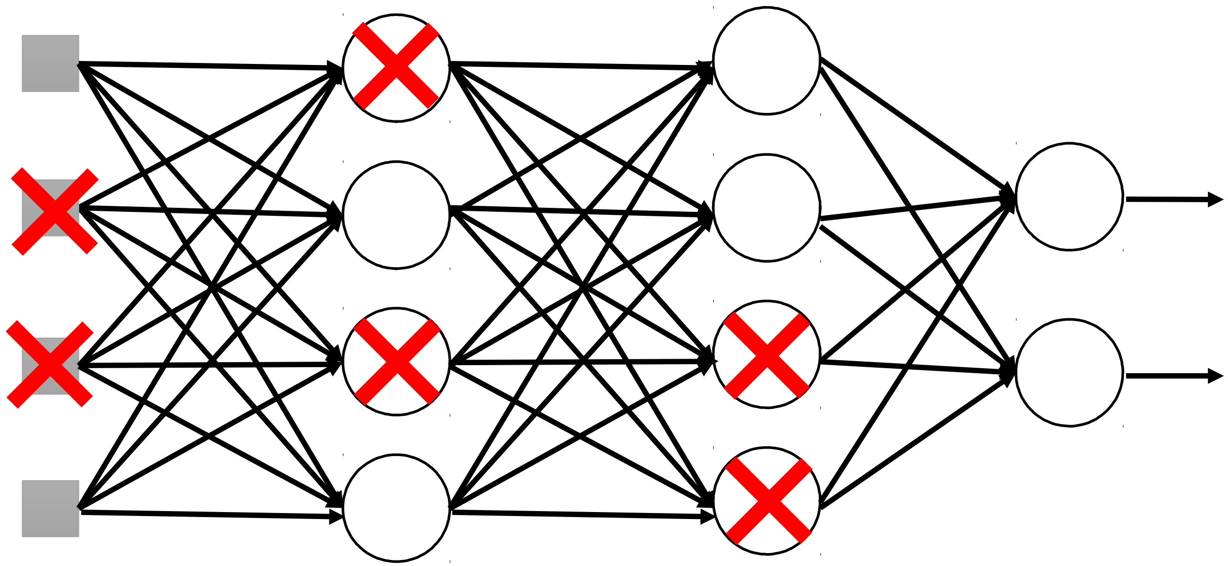
- Adagrad [John Duchi, JMLR’11]
- RMSprop
  - https://www.youtube.com/watch?v=O3sxAc4hxZU
- Adadelta [Matthew D. Zeiler, arXiv’12]
- Adam [Diederik P. Kingma, ICLR’15]
- AdaSecant [Caglar Gulcehre, arXiv’14]
- “No more pesky learning rates” [Tom Schaul, arXiv’12]

Regularization: via Dropout

Pick a mini-batch

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

Training:



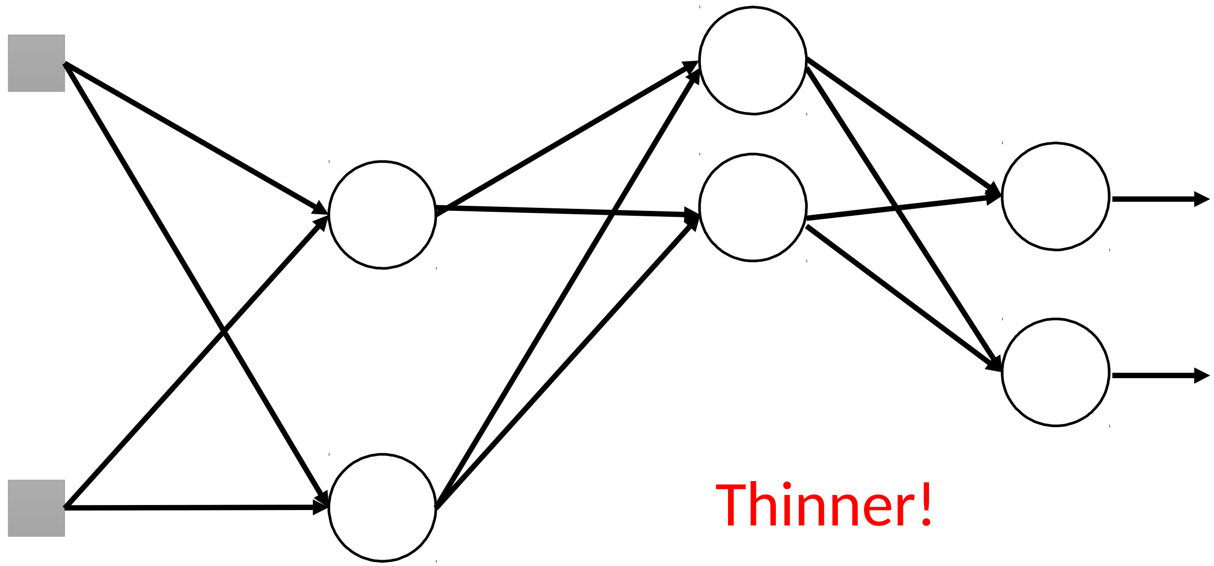
- **Each time before computing the gradients**
  - Each neuron has p% to dropout

Pick a mini-batch

Dropout

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

Training:



Thinner!

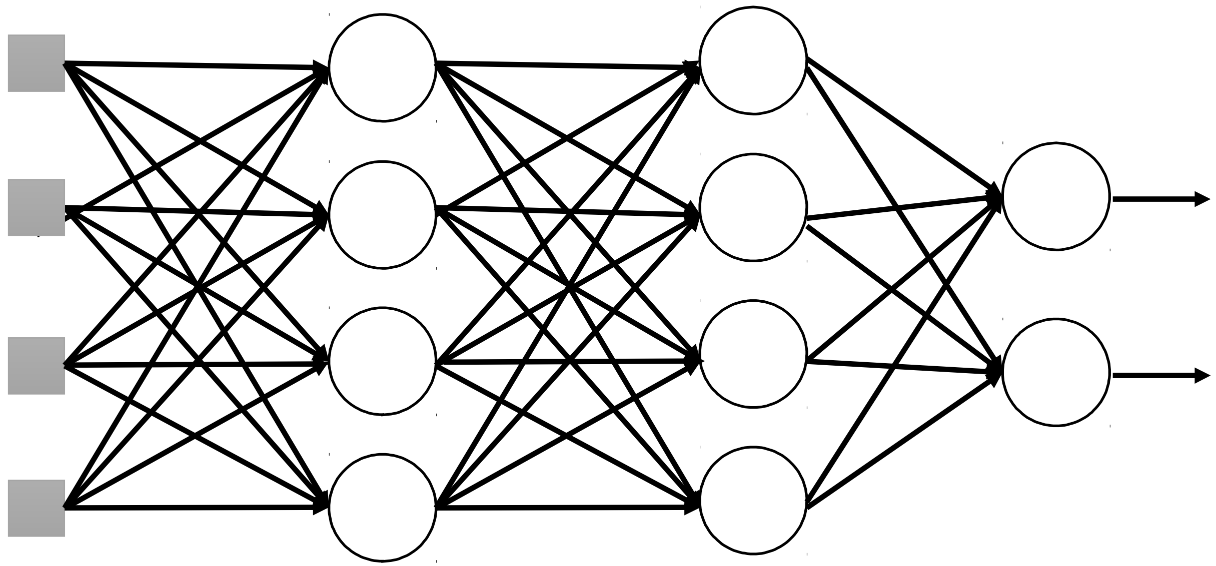
- **Each time before computing the gradients**
  - Each neuron has p% to dropout

    ➡ **The structure of the network is changed.**
  - Using the new network for training

For each mini-batch, we resample the dropout neurons

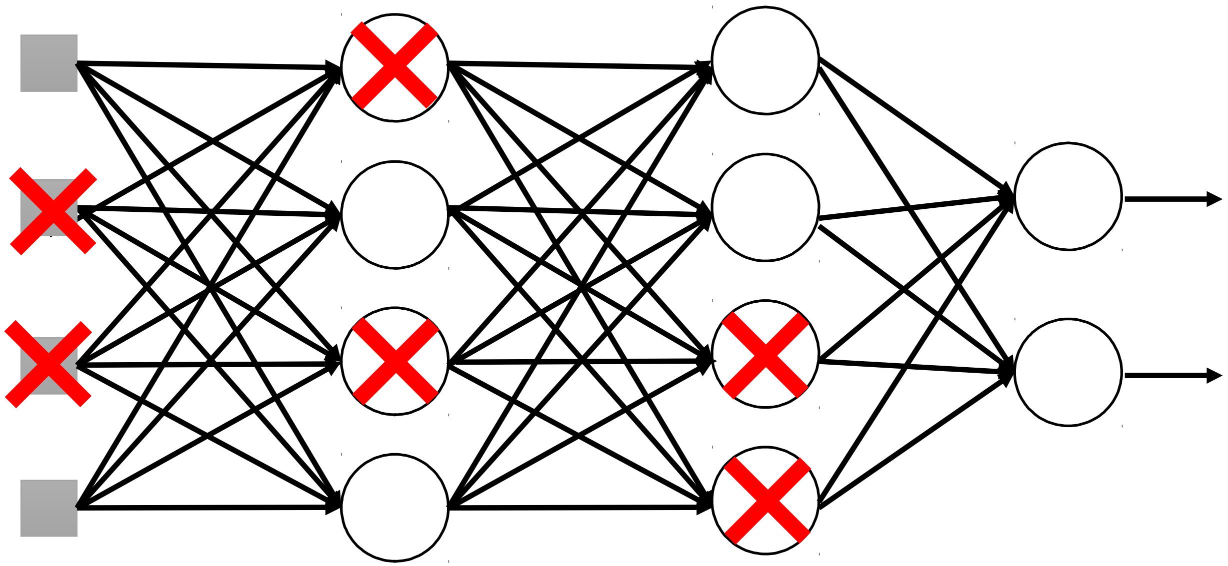
Dropout

Testing:



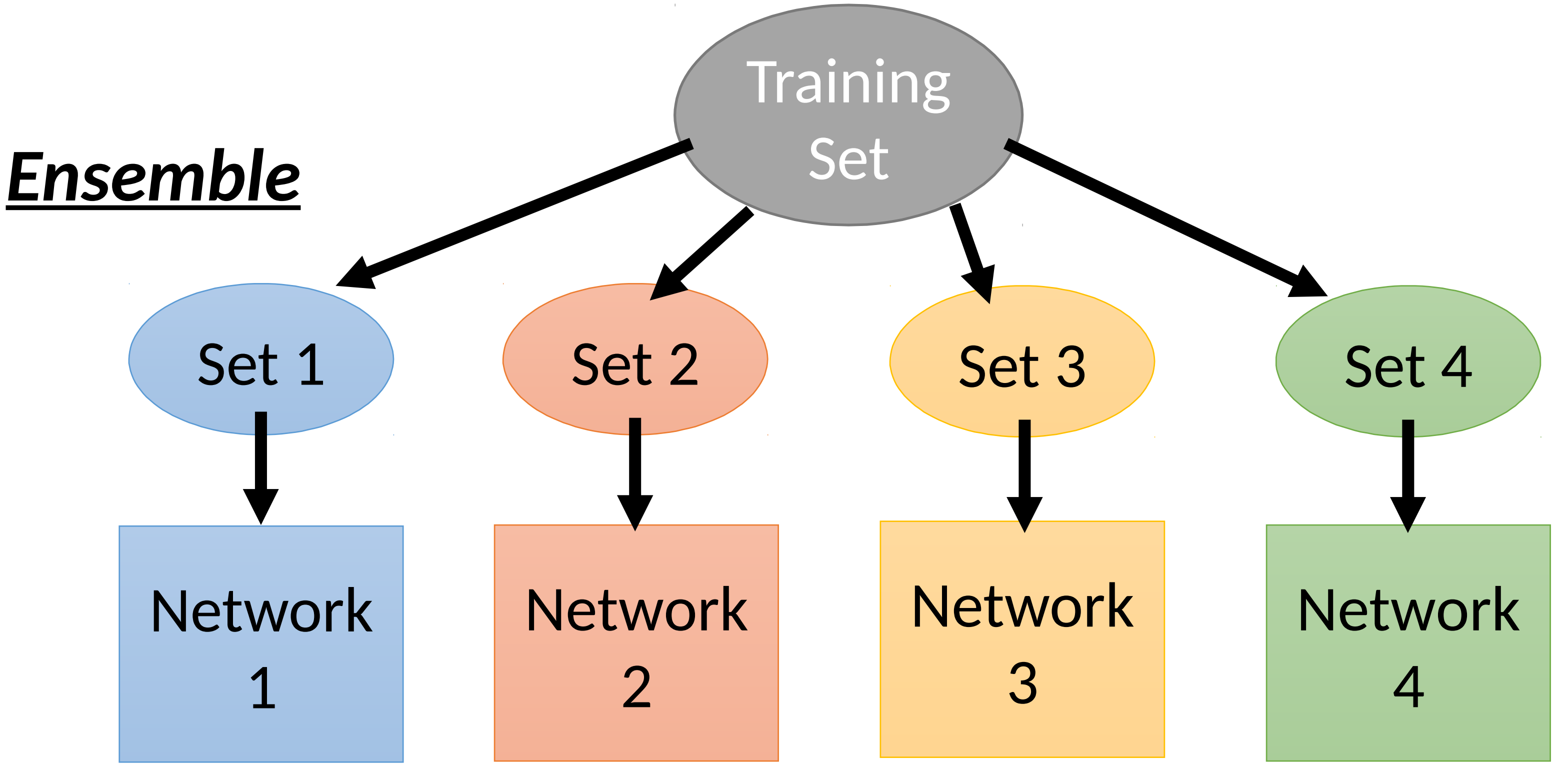
- **No dropout**
  - If the dropout rate at training is p%, all the weights times (1-p)%
  - Assume that the dropout rate is 50%. If a weight $w = 1$ by training, set $w = 0.5$ for testing.

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.

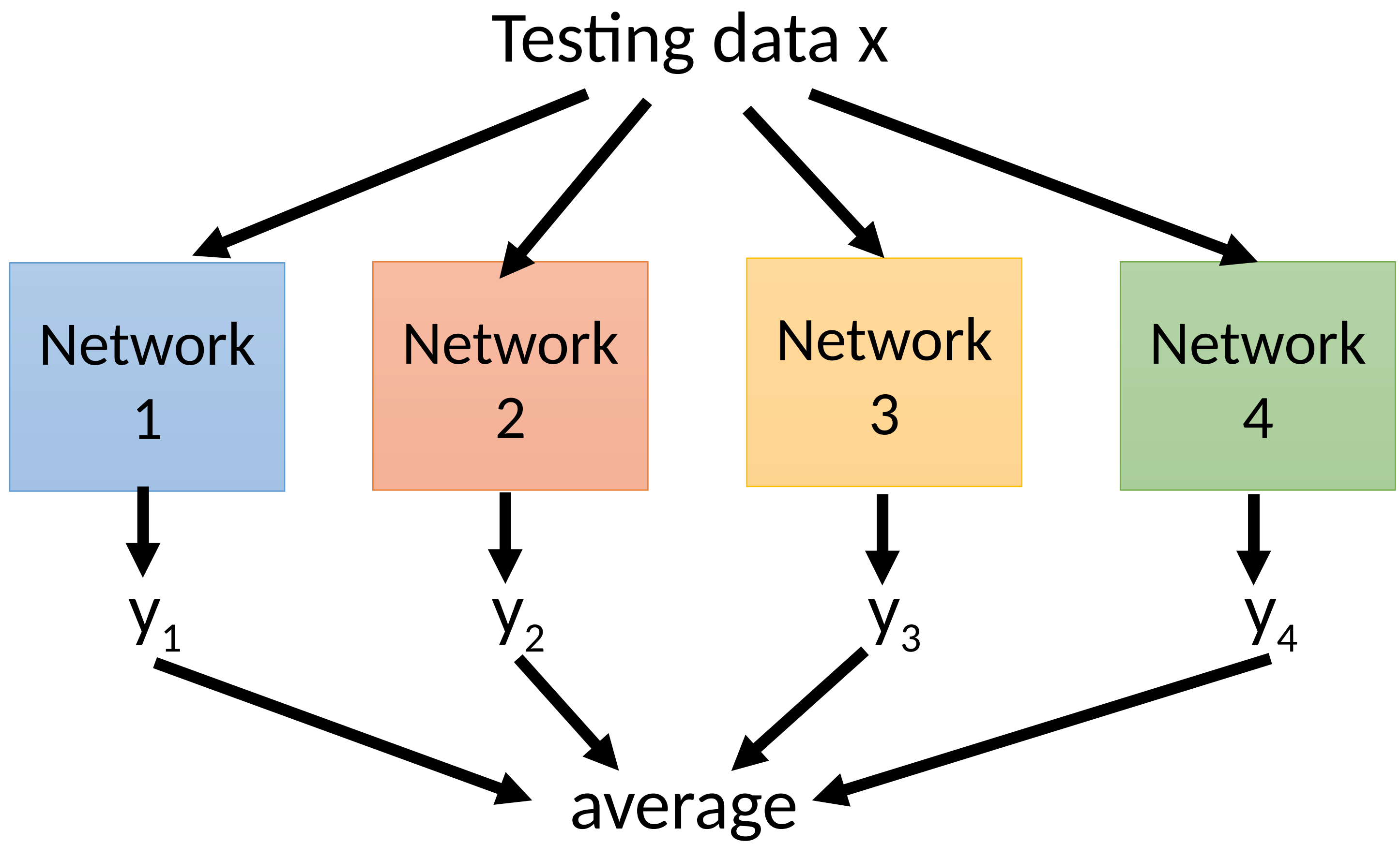
Dropout is a kind of ensemble.



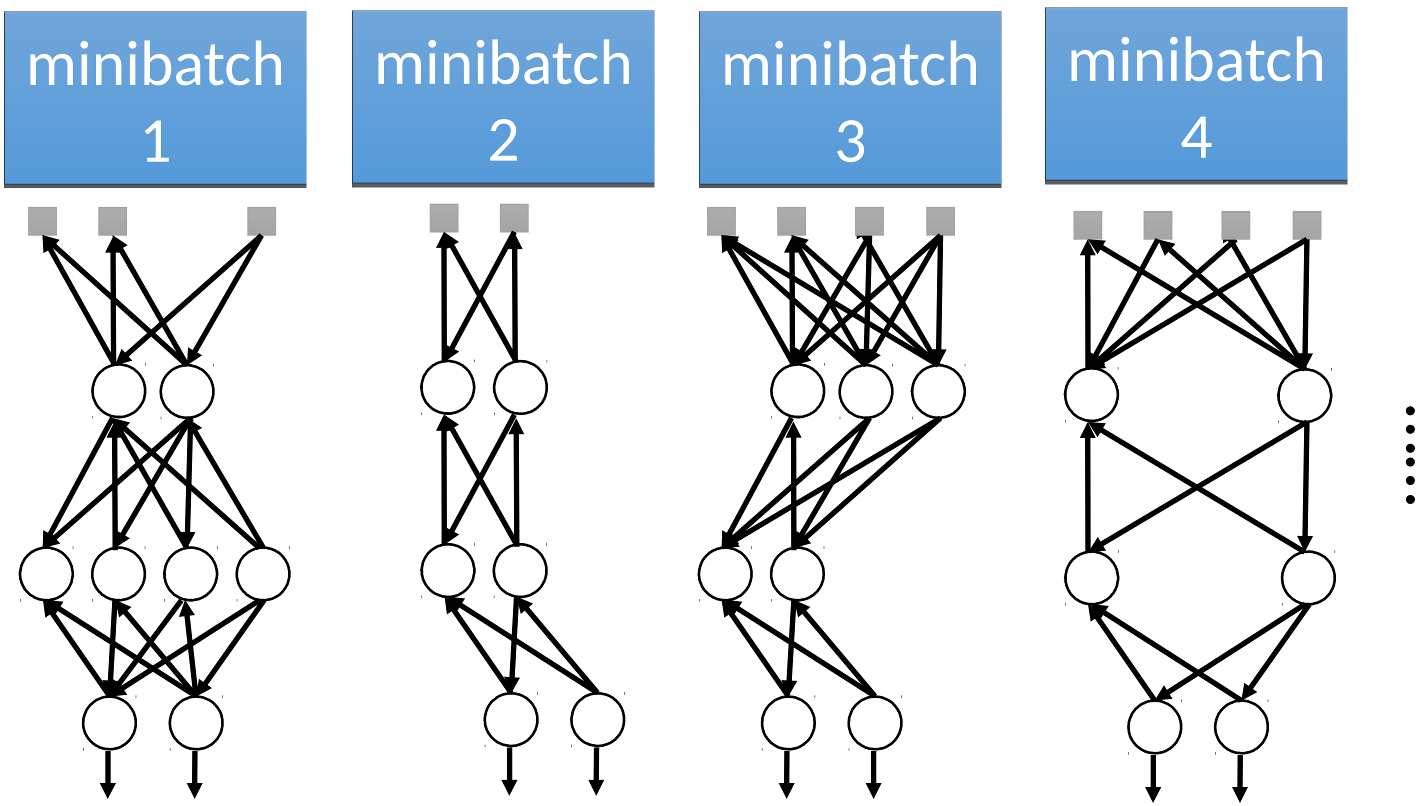
Train a bunch of networks with different structures

Dropout is a kind of ensemble

Ensemble



Dropout is a kind of ensemble



<u>Training of Dropout</u>

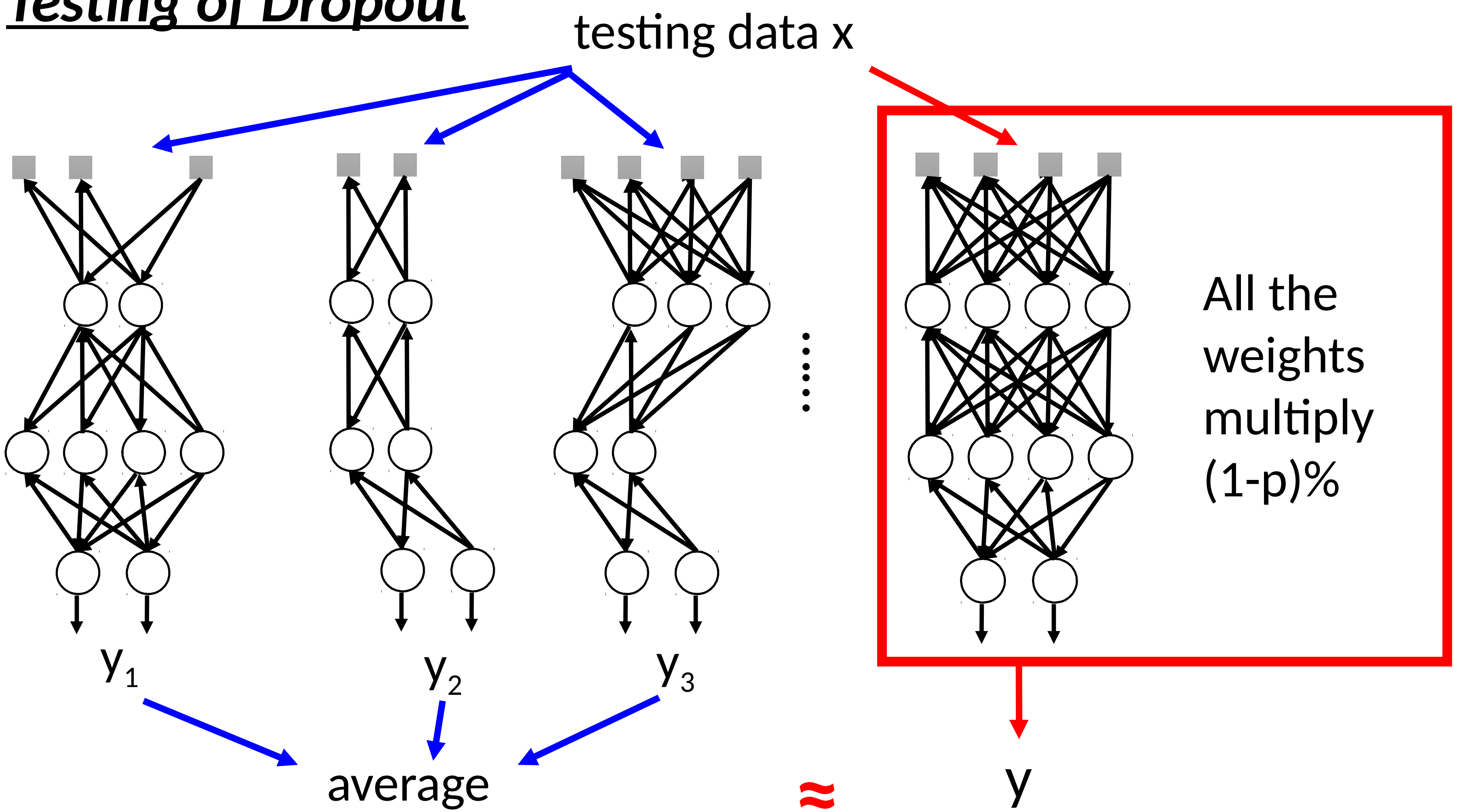
M neurons

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



Summary

- Introduction to deep learning
  - Fully connected neural networks
- Some training issues and solutions
  - Adaptive learning rate
  - New activation functions
  - Dropout
- Next lectures:
  - Convolution neural networks
  - Recurrent neural networks