



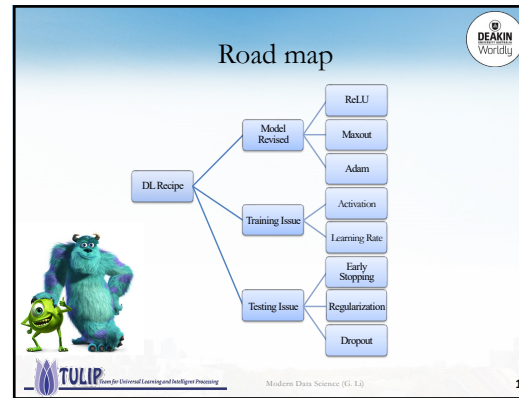
Lecture Notes on  
Pattern Recognition

Module 09(B): DL Training and Tuning

Gang Li  
School of Information Technology  
Deakin University, VIC 3125, Australia

0



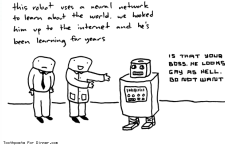


1

Training and Tuning

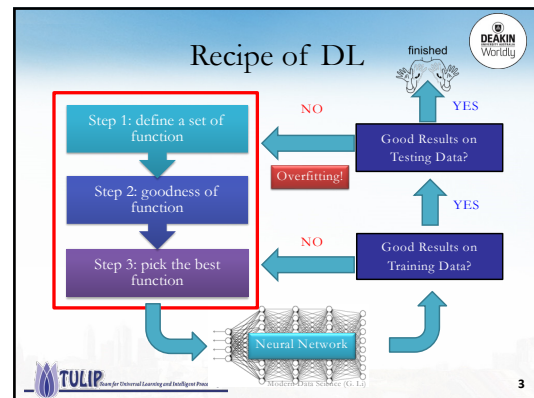
- Recipe of DL
- ReLU
- Maxout
- RMSProp
- Momentum
- Adam

*This robot uses a neural network to learn about the world, not looking him up to the internet and he's been learning the options.*

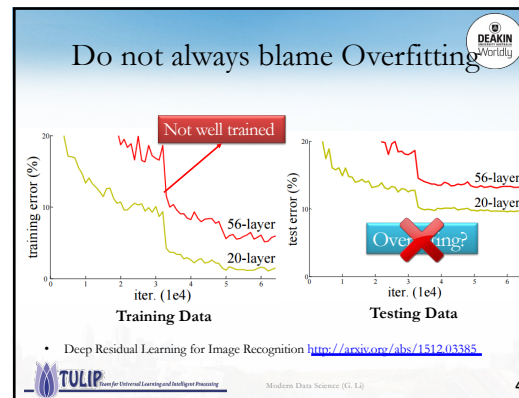
*IS THAT YOUR BEST HE LEARNS OF HIS WORLD, DO NOT WAIT*

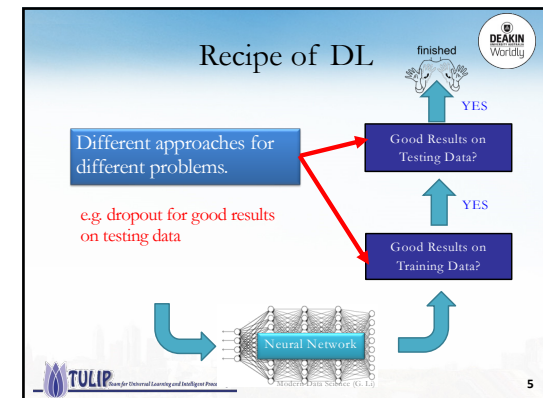
2



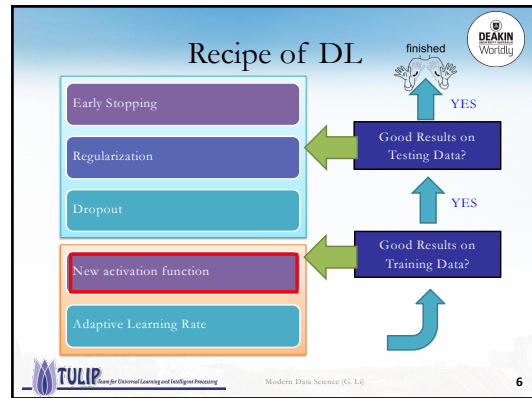
3



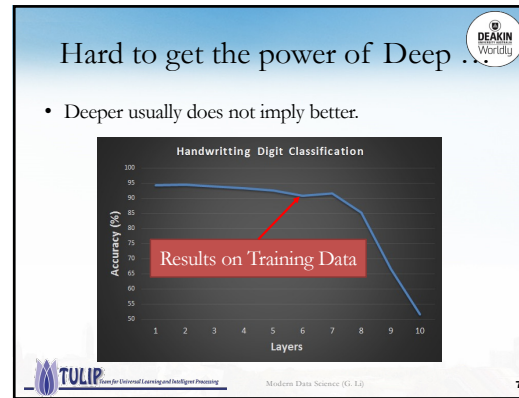
4



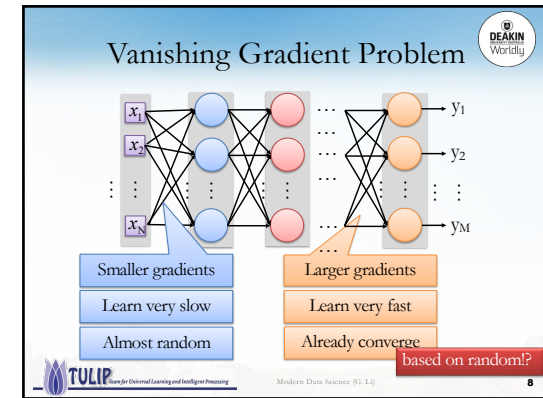
5



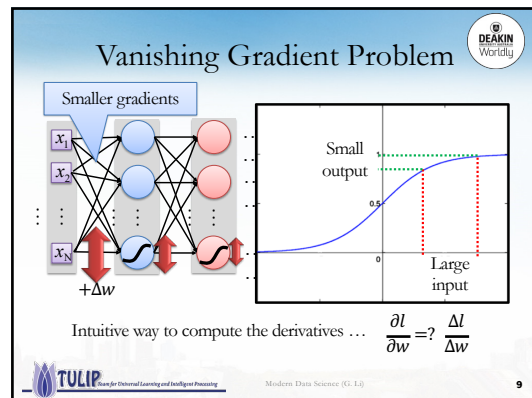
6



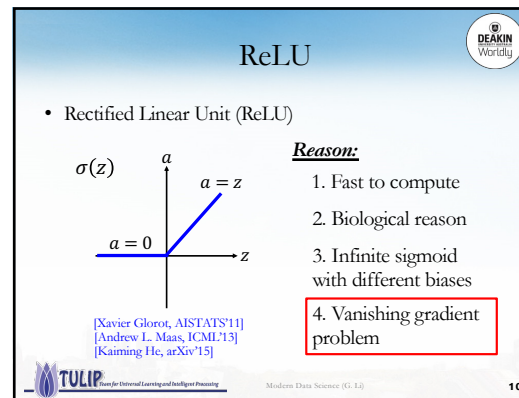
7



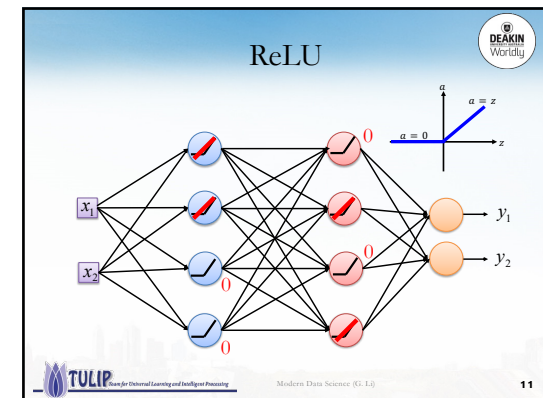
8



9



10



11

### ReLU

- A Thinner linear network

Do not have smaller gradients

TULIP: Towards Universal Learning and Intelligent Processing

Modern Data Science (G. Li)

12

### ReLU - variant

*Leaky ReLU*

*Parametric ReLU*

•  $\alpha$  also learned by gradient descent

TULIP: Towards Universal Learning and Intelligent Processing

Modern Data Science (G. Li)

13

### Maxout

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

You can have more than 2 elements in a group.

TULIP: Towards Universal Learning and Intelligent Processing

Modern Data Science (G. Li)

14

### Maxout

$z = wx + b$

$z_1 = wx + b$

$z_2 = 0$

TULIP: Towards Universal Learning and Intelligent Processing

Modern Data Science (G. Li)

15

### Maxout

$z = wx + b$

$z_1 = wx + b$

$z_2 = w'x + b'$

Learnable Activation Function

TULIP: Towards Universal Learning and Intelligent Processing

Modern Data Science (G. Li)

16

### 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

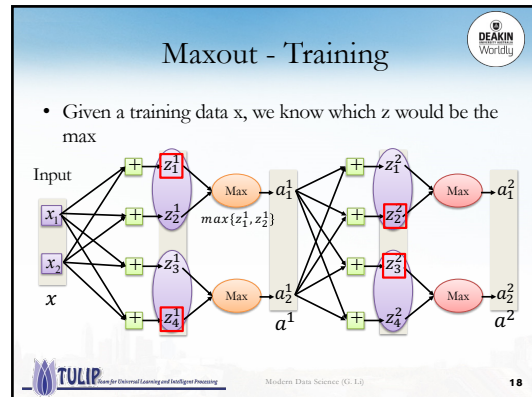
2 elements in a group

3 elements in a group

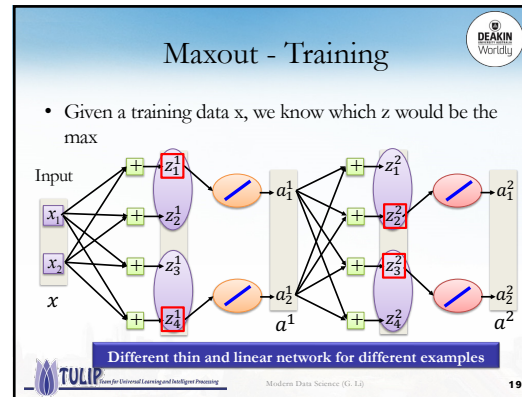
TULIP: Towards Universal Learning and Intelligent Processing

Modern Data Science (G. Li)

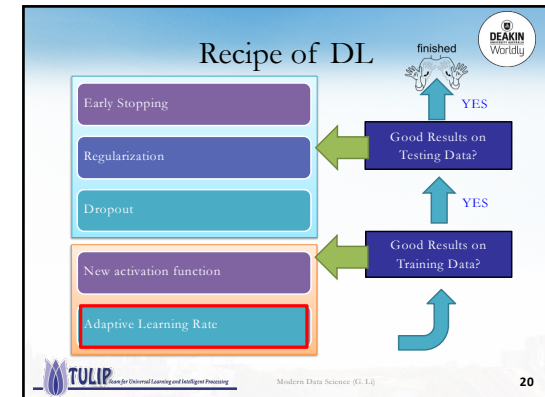
17



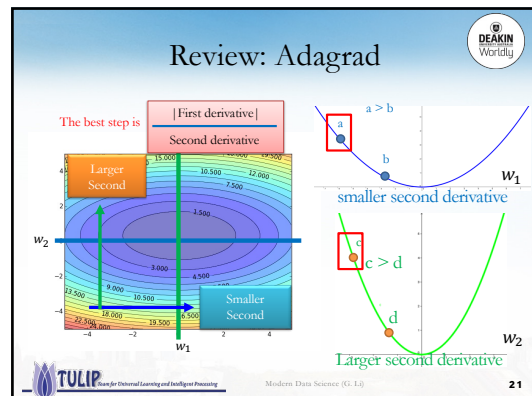
18



19



20



21

### Review: Adagrad

$$\theta^{t+1} \leftarrow \theta^t - \frac{\eta}{\sqrt{\sum_{i=0}^t (g^i)^2}} g^t$$

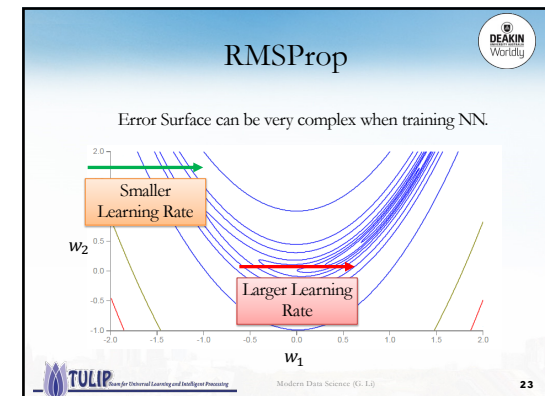
The best step is

|First derivative|

Second derivative

TULIP: Towards Universal Learning and Intelligent Processing  
Modern Data Science (G. Li)

22



23

### RMSProp

$$w^1 \leftarrow w^0 - \frac{\eta}{\sigma^0} g^0 \quad \sigma^0 = g^0$$

$$w^2 \leftarrow w^1 - \frac{\eta}{\sigma^1} g^1 \quad \sigma^1 = \sqrt{\alpha(\sigma^0)^2 + (1-\alpha)(g^1)^2}$$

$$w^3 \leftarrow w^2 - \frac{\eta}{\sigma^2} g^2 \quad \sigma^2 = \sqrt{\alpha(\sigma^1)^2 + (1-\alpha)(g^2)^2}$$

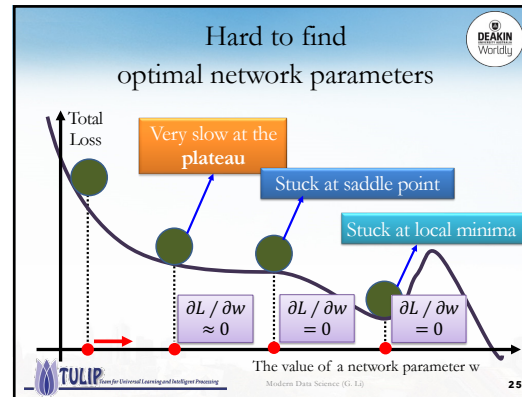
$$\vdots$$

$$w^{t+1} \leftarrow w^t - \frac{\eta}{\sigma^t} g^t \quad \sigma^t = \sqrt{\alpha(\sigma^{t-1})^2 + (1-\alpha)(g^t)^2}$$

• Root Mean Square of the gradients with previous gradients being decayed

TULIP Centre for Data Science and Intelligent Processing Modern Data Science (G.1.1) **24**

24



25

### In physical world .....

- Momentum

How about put this phenomenon in gradient descent?

TULIP Centre for Data Science and Intelligent Processing Modern Data Science (G.1.1) **26**

26

### Review: Vanilla Gradient Descent

Start at position  $\theta^0$

Compute gradient at  $\theta^0$

Move to  $\theta^1 = \theta^0 - \eta \nabla L(\theta^0)$

Compute gradient at  $\theta^1$

Move to  $\theta^2 = \theta^1 - \eta \nabla L(\theta^1)$

$\vdots$

Stop until  $\nabla L(\theta^t) \approx 0$

→ Gradient

→ Movement

TULIP Centre for Data Science and Intelligent Processing Modern Data Science (G.1.1) **27**

27

### Momentum

Movement: movement of last step minus gradient at present

Start at point  $\theta^0$

Movement  $v^0 = 0$

Compute gradient at  $\theta^0$

Movement  $v^1 = \lambda v^0 - \eta \nabla L(\theta^0)$

Move to  $\theta^1 = \theta^0 + v^1$

Compute gradient at  $\theta^1$

Movement  $v^2 = \lambda v^1 - \eta \nabla L(\theta^1)$

Move to  $\theta^2 = \theta^1 + v^2$

Movement not just based on gradient, but previous movement

→ Gradient

→ Movement

→ Movement of last step

TULIP Centre for Data Science and Intelligent Processing Modern Data Science (G.1.1) **28**

28

### Momentum

Movement: movement of last step minus gradient at present

Start at point  $\theta^0$

Movement  $v^0 = 0$

Compute gradient at  $\theta^0$

Movement  $v^1 = \lambda v^0 - \eta \nabla L(\theta^0)$

Move to  $\theta^1 = \theta^0 + v^1$

Compute gradient at  $\theta^1$

Movement  $v^2 = \lambda v^1 - \eta \nabla L(\theta^1)$

Move to  $\theta^2 = \theta^1 + v^2$

Movement not just based on gradient, but previous movement

$v^i$  is actually the weighted sum of all the previous gradient:  
 $\nabla L(\theta^0), \nabla L(\theta^1), \dots, \nabla L(\theta^{i-1})$

$v^0 = 0$

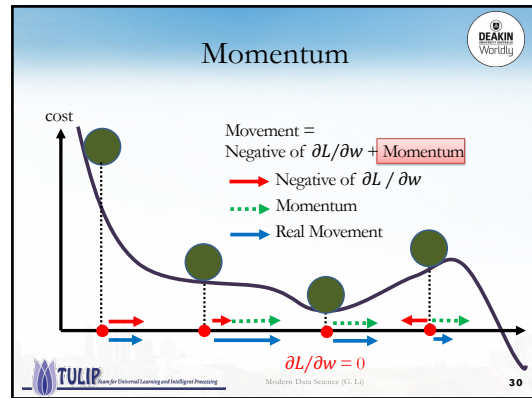
$v^1 = -\eta \nabla L(\theta^0)$

$v^2 = -\lambda \eta \nabla L(\theta^0) - \eta \nabla L(\theta^1)$

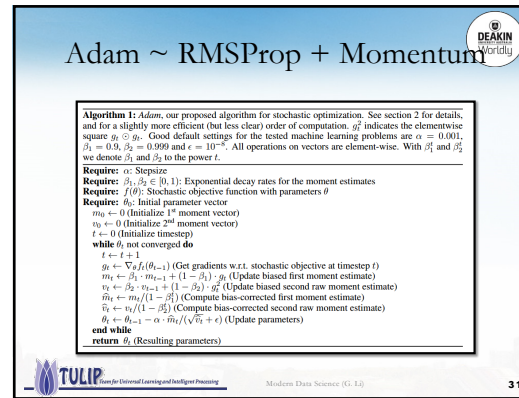
$\vdots$

TULIP Centre for Data Science and Intelligent Processing Modern Data Science (G.1.1) **29**

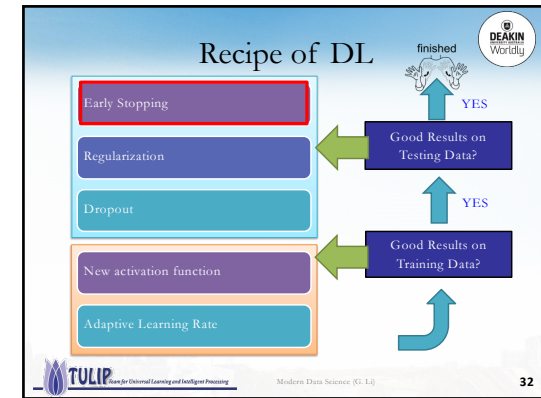
29



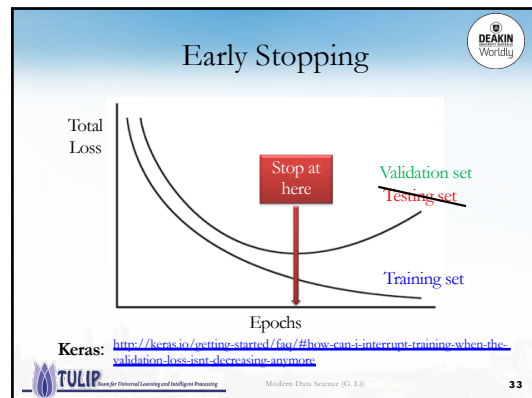
30



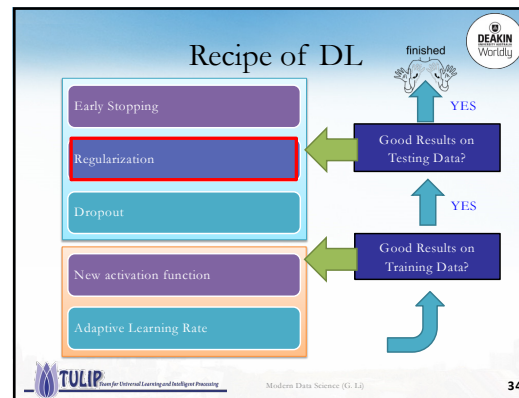
31



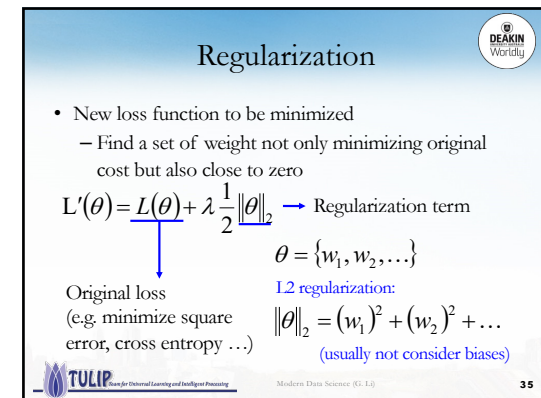
32



33



34



35



## Regularization

- New loss function to be minimized L2 regularization:  
 $\|\theta\|_2 = (w_1)^2 + (w_2)^2 + \dots$

$$L'(\theta) = L(\theta) + \lambda \frac{1}{2} \|\theta\|_2^2 \quad \text{Gradient: } \frac{\partial L'}{\partial w} = \frac{\partial L}{\partial w} + \lambda w$$

Update:  $w^{t+1} \rightarrow w^t - \eta \frac{\partial L'}{\partial w} = w^t - \eta \left( \frac{\partial L}{\partial w} + \lambda w^t \right)$

$$= (1 - \eta \lambda) w^t - \eta \frac{\partial L}{\partial w} \quad \text{Weight Decay}$$

↓  
Closer to zero

36

36

## Regularization

- New loss function to be minimized L1 regularization:  
 $\|\theta\|_1 = |w_1| + |w_2| + \dots$

$$L'(\theta) = L(\theta) + \lambda \frac{1}{2} \|\theta\|_1 \quad \frac{\partial L'}{\partial w} = \frac{\partial L}{\partial w} + \lambda \text{sgn}(w)$$

Update:  $w^{t+1} \rightarrow w^t - \eta \frac{\partial L'}{\partial w} = w^t - \eta \left( \frac{\partial L}{\partial w} + \lambda \text{sgn}(w^t) \right)$

$$= w^t - \eta \frac{\partial L}{\partial w} - \eta \lambda \text{sgn}(w^t) \quad \text{Always delete}$$

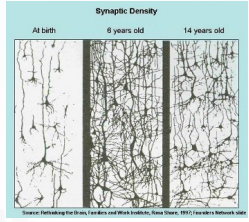
$$= (1 - \eta \lambda) w^t - \eta \frac{\partial L}{\partial w} \quad \dots \dots \text{L2}$$

37

37

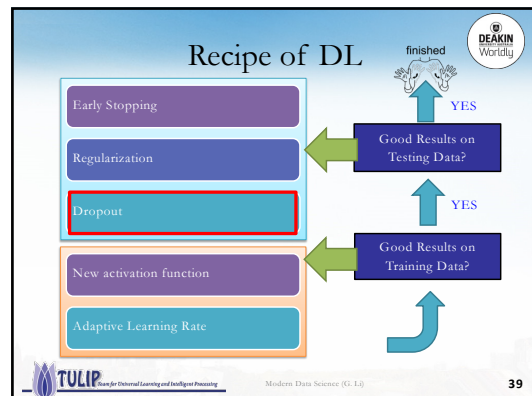
## Regularization - Weight Decay

- Our brain prunes out the useless link between neurons.

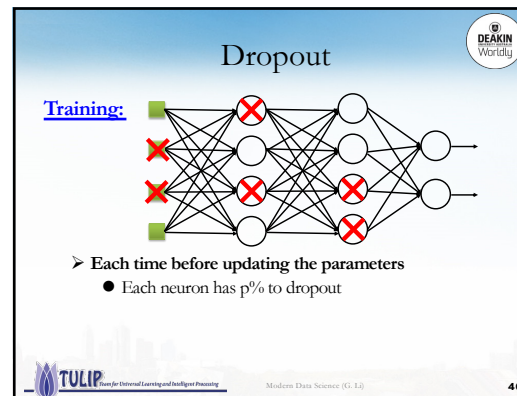


38

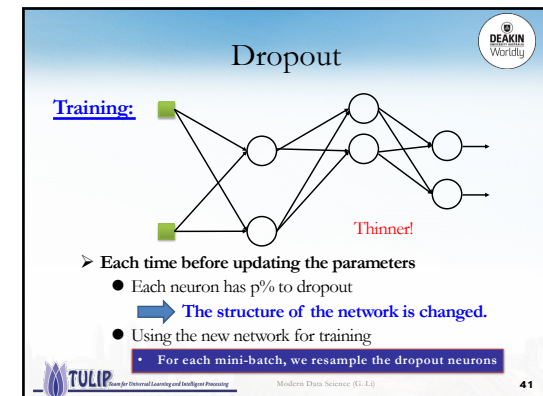
38



39



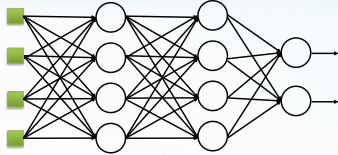
40



41

### 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.

TULIP: Towards Universal Learning and Intelligent Processing  
Modern Data Science (G.L.)


42

### Dropout - Intuitive Reason

**Training**  
Dropout (make life difficult)



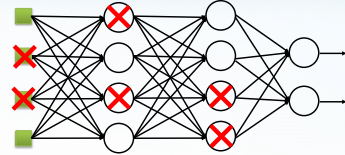
**Testing**  
No dropout (feel easier)



TULIP: Towards Universal Learning and Intelligent Processing  
Modern Data Science (G.L.)

43

### 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.

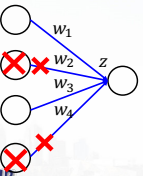
TULIP: Towards Universal Learning and Intelligent Processing  
Modern Data Science (G.L.)

44

### Dropout - Intuitive Reason

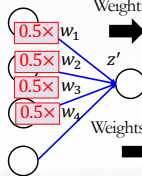
- Why the weights should multiply  $(1-p)\%$  (dropout rate) when testing?

**Training of Dropout**  
Assume dropout rate is  $50\%$



**Testing of Dropout**  
No dropout

Weights from training  $\rightarrow z' \approx 2z$



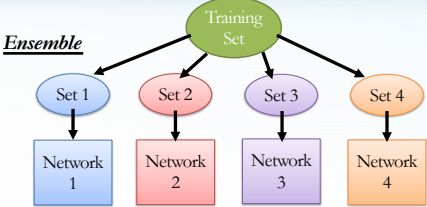
Weights multiply  $1-p\%$   $\rightarrow z' \approx z$

TULIP: Towards Universal Learning and Intelligent Processing  
Modern Data Science (G.L.)

45

### Dropout is a kind of ensemble.

**Ensemble**



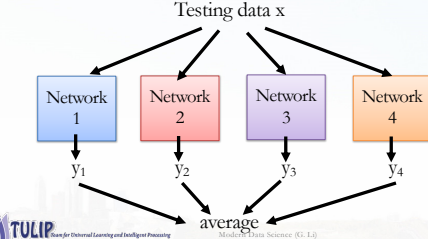
Train a bunch of networks with different structures

TULIP: Towards Universal Learning and Intelligent Processing  
Modern Data Science (G.L.)

46

### Dropout is a kind of ensemble.

**Ensemble**



average

TULIP: Towards Universal Learning and Intelligent Processing  
Modern Data Science (G.L.)

47



