Chapter Four

BP: An Illustrating Example

内部资料，请勿传播！！
四川大学 计算机学院 章毅教授

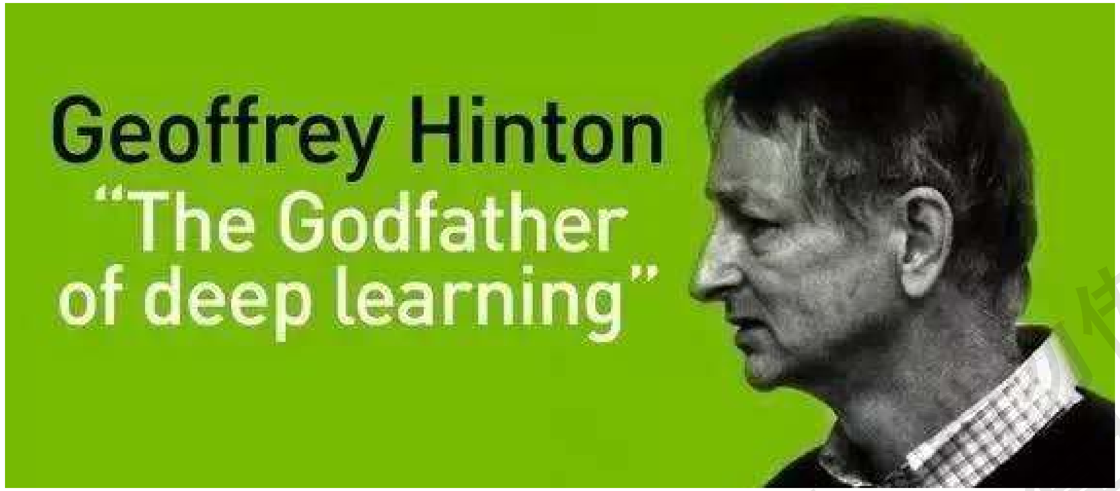
Zhang Yi, *IEEE Fellow*
Autumn 2018

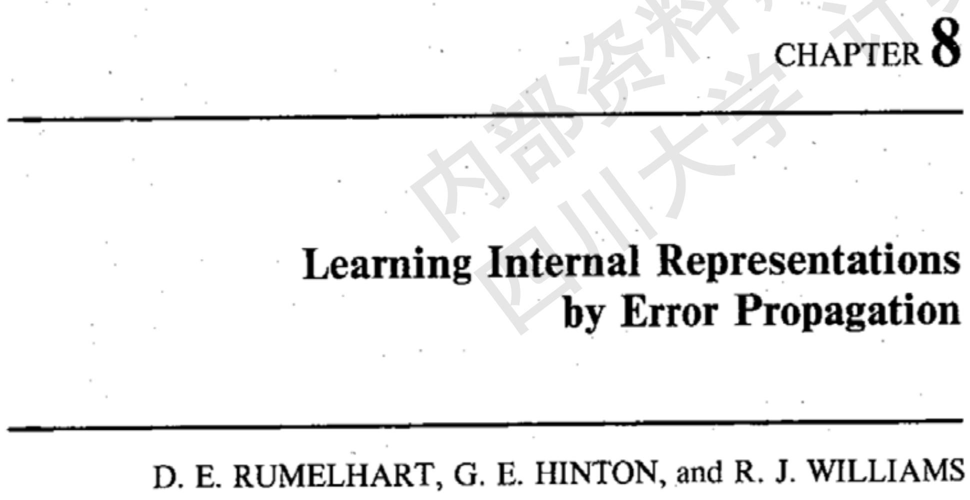
Outline

- Brief Review of Backpropagation Algorithm
- An Illustrating Example
- Experiments
- Assignment

内部资料，请勿传播！！
四川大学 计算机学院 章毅教授

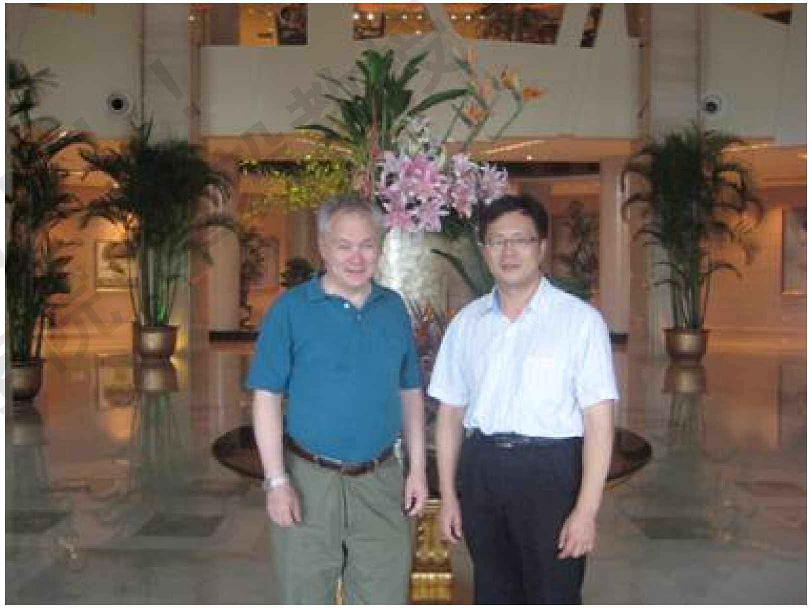
Brief History of BP




CHAPTER 8

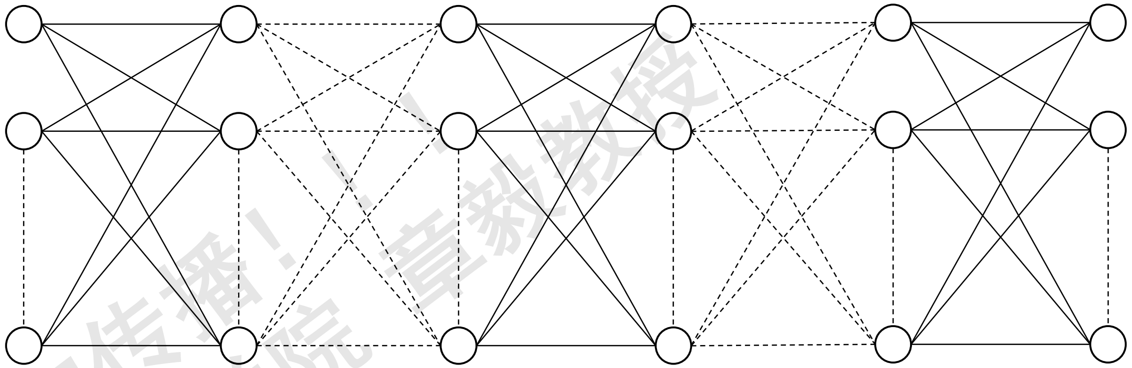
Learning Internal Representations by Error Propagation

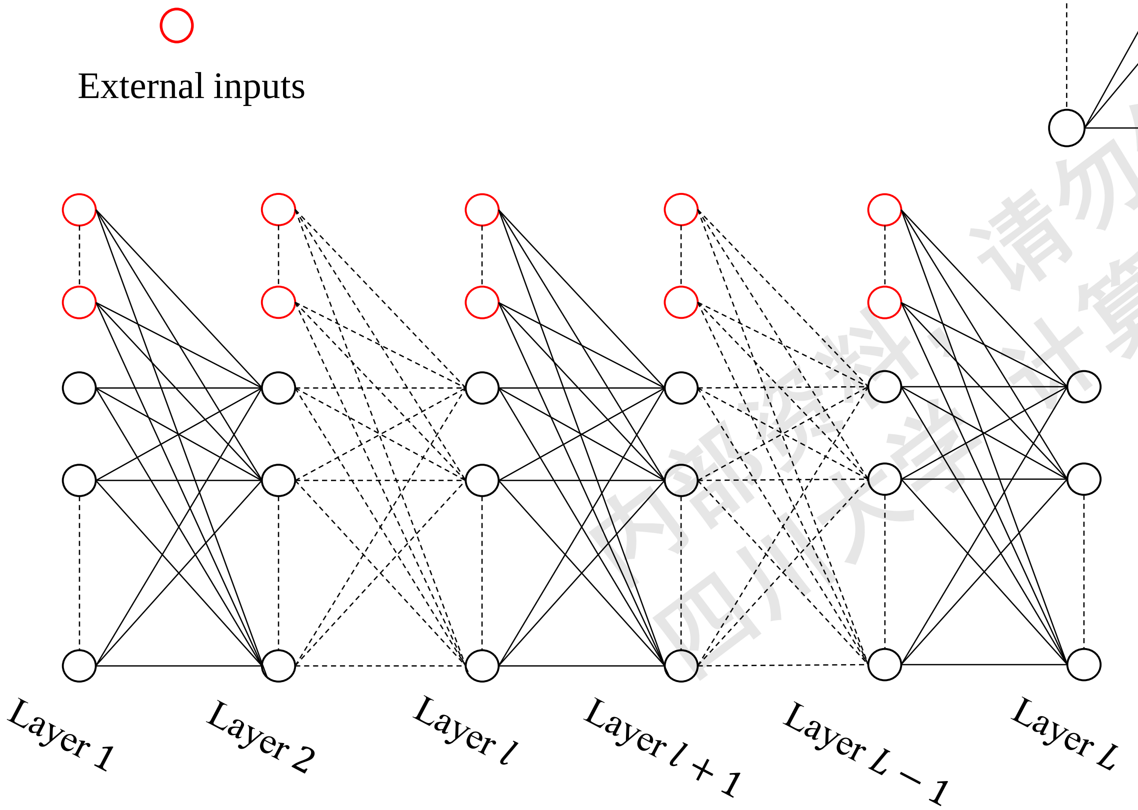
D. E. RUMELHART, G. E. HINTON, and R. J. WILLIAMS



Professor P. Werbos

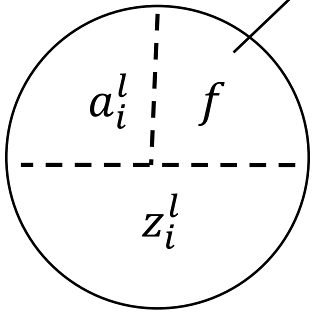
Computational Model of Neural Networks





内部资料，请勿传播！！
四川大学计算机学院 章毅教授

Local function defined on neuron



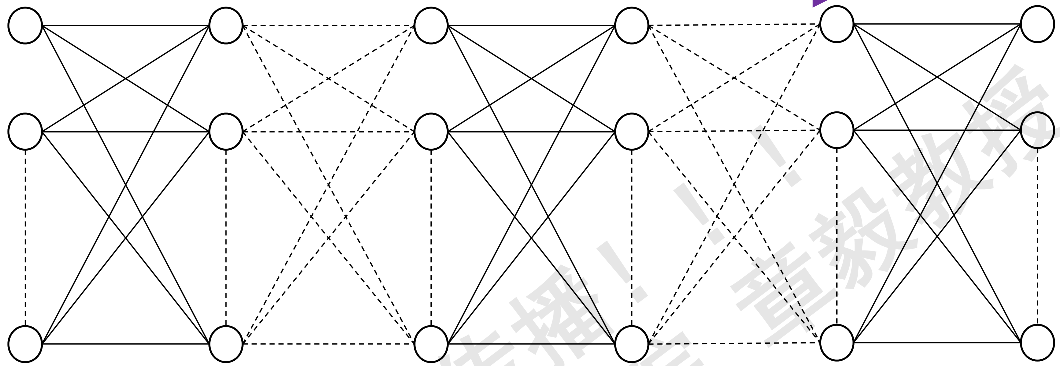
$$a_i^l = f(z_i^l)$$

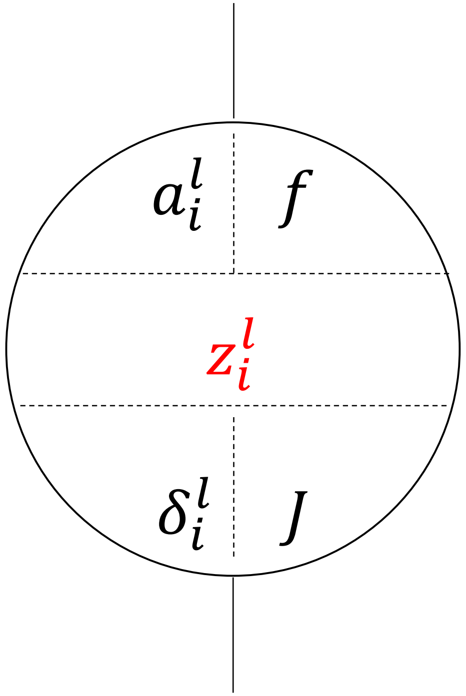
$$a^l = \begin{bmatrix} a_1^l \\ a_2^l \\ \vdots \\ a_{n_l}^l \end{bmatrix}$$

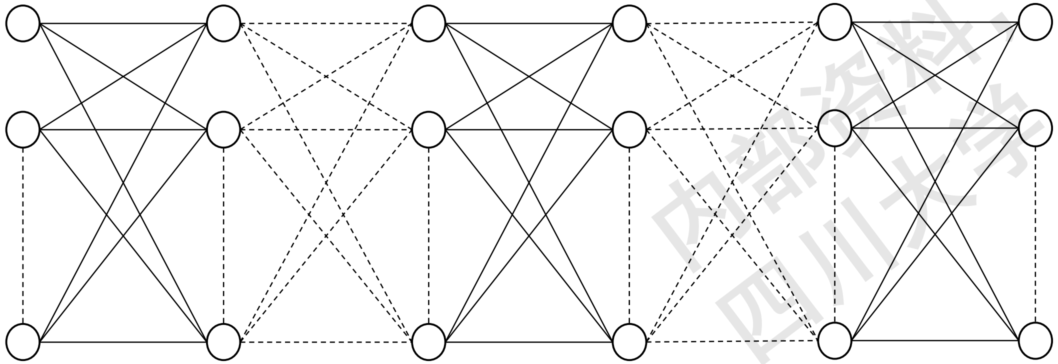
l layer i^{th} neuron

Local activation function f

Forward computing a^l





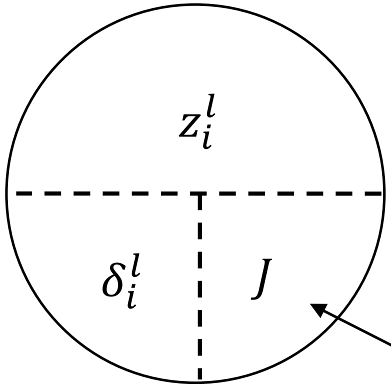


Backpropagation δ^l

Global cost function J

l layer i^{th} neuron

$$\delta_i^l = \frac{\partial J}{\partial z_i^l}$$



$$\delta^l = \begin{bmatrix} \delta_1^l \\ \delta_2^l \\ \vdots \\ \delta_{n_l}^l \end{bmatrix}$$

Global function defined on network

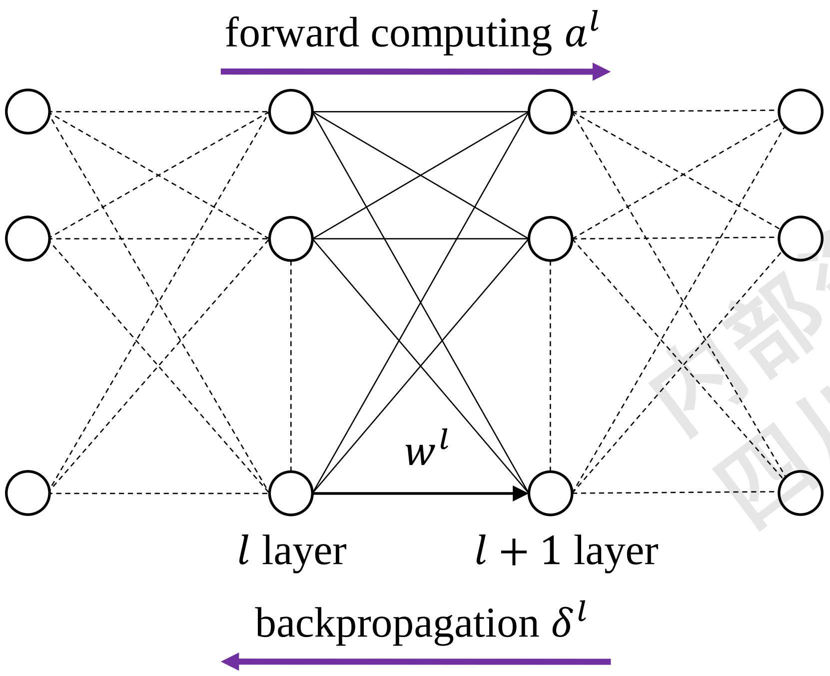
内部资料，请勿传播！！！
四川大学 计算机学院 章毅教授

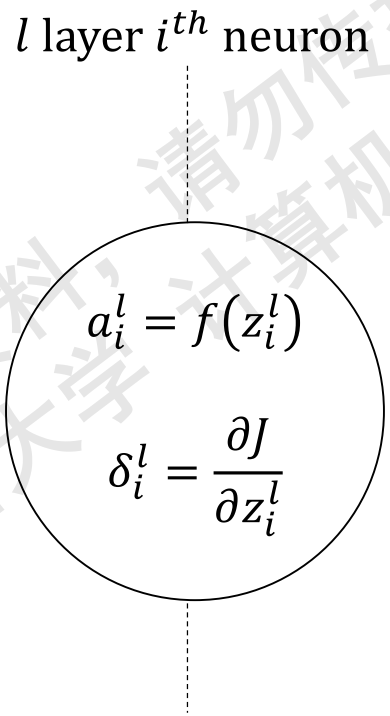
One Page to Understand BP

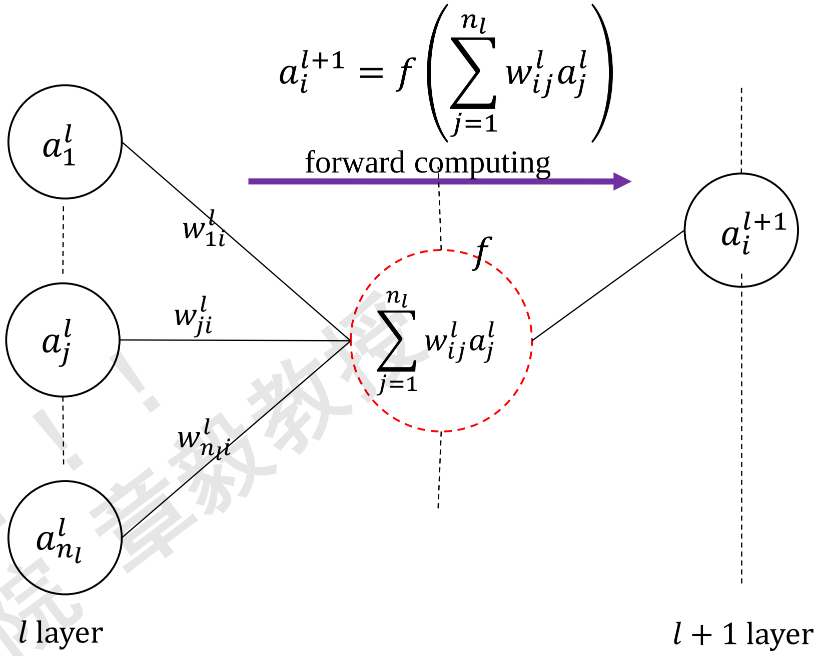
Cost function: $J(w^1, \cdots, w^L)$

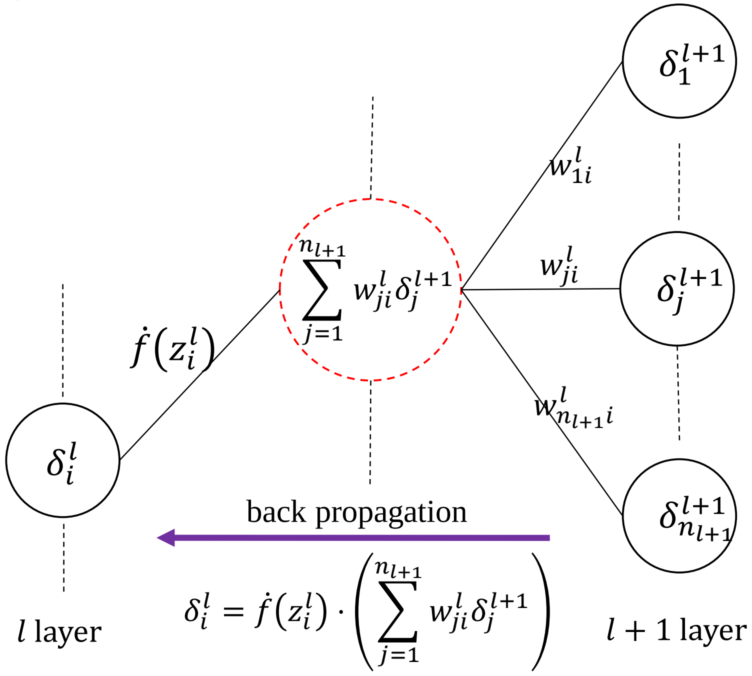
Updating rule: $w_{ji}^l \leftarrow w_{ji}^l - \alpha \cdot \frac{\partial J}{\partial w_{ji}^l}$

Relationship: $\frac{\partial J}{\partial w_{ji}^l} = \delta_j^{l+1} \cdot a_i^l$









内部资料，请勿传播！！
四川大学 计算机学院

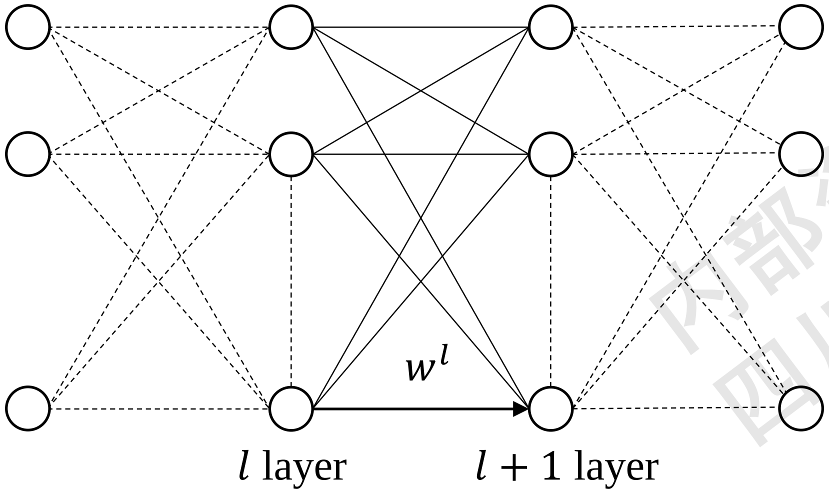
BP Functions

Cost function: $J(w^1, \cdots, w^L)$

Updating rule: $w_{ji}^l \leftarrow w_{ji}^l - \alpha \cdot \frac{\partial J}{\partial w_{ji}^l}$

Relationship: $\frac{\partial J}{\partial w_{ji}^l} = \delta_j^{l+1} \cdot a_i^l$

forward computing a^l



backpropagation δ^l

l layer i^{th} neuron

$$a_i^l = f(z_i^l)$$

$$\delta_i^l = \frac{\partial J}{\partial z_i^l}$$

%forward computing

function $fc(w^l, a^l)$

$for\ i = 1: n_{l+1}$

$$z_i^{l+1} = \sum_{j=1}^{n_l} w_{ij}^l a_j^l$$

$$a_i^{l+1} = f(z_i^{l+1})$$

end

%backpropagation

function $bc(w^l, \delta^{l+1})$

$for\ i = 1: n_l$

$$\delta_i^l = \dot{f}(z_i^l) \cdot \left(\sum_{j=1}^{n_{l+1}} w_{ji}^l \delta_j^{l+1} \right)$$

end

内部资料，请勿传播！！！
四川大学计算机学院 章毅教授

The BP Algorithm

BP Algorithm:

Step 1. Input the training data set $D = \{(x, y)\}$

Step 2. Initial each w_{ij}^l, and choose a learning rate α.

Step 3. For each mini-batch sample $D_m \subseteq D$

$$a^1 \leftarrow x \in D_m;$$

for $l = 2 : L$

$$a^l \leftarrow fc(w^l, a^l);$$

end

$$\delta^L = \frac{\partial J}{\partial z^L};$$

for $l = L - 1 : 2$

$$\delta^l \leftarrow bc(w^l, \delta^{l+1});$$

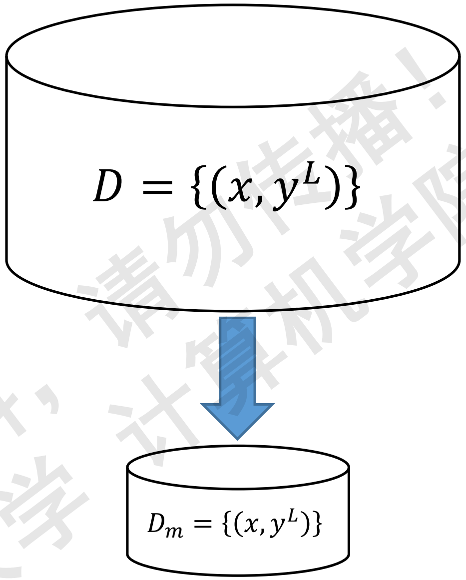
end

$$\frac{\partial J}{\partial w_{ji}^l} \leftarrow \frac{\partial J}{\partial w_{ji}^l} + \delta_j^{l+1} \cdot a_i^l;$$

Step 4. Updating

$$w_{ji}^l \leftarrow w_{ji}^l - \alpha \cdot \frac{\partial J}{\partial w_{ji}^l};$$

Step 5. Return to Step 3 until each w^lconverge.



function $fc(w^l, a^l)$

$for\ i = 1 : n_{l+1}$

$$z_i^{l+1} = \sum_{j=1}^{n_l} w_{ij}^l a_j^l$$

$$a_i^{l+1} = f(z_i^{l+1})$$

end

Relationship:

$$\frac{\partial J}{\partial w_{ji}^l} = \delta_j^{l+1} \cdot a_i^l$$

function $bc(w^l, \delta^{l+1})$

$for\ i = 1 : n_l$

$$\delta_i^l = \dot{f}(z_i^l) \cdot \left(\sum_{j=1}^{n_{l+1}} w_{ji}^l \delta_j^{l+1} \right)$$

end

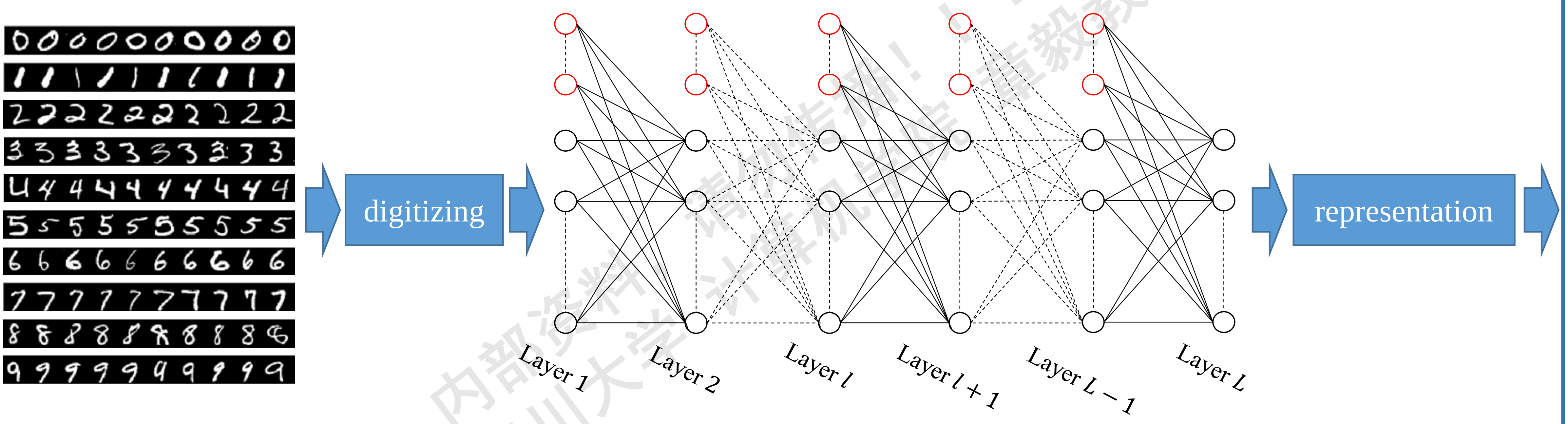
内部资料，请勿传播！！！
四川大学 计算机学院 章毅教授

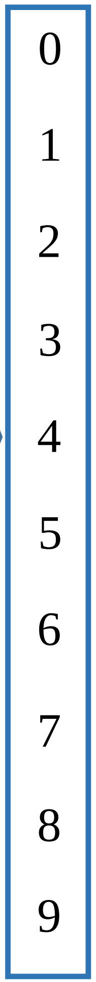
Outline

- Brief Review of Backpropagation Algorithm
- An Illustrating Example
- Experiments
- Assignment

内部资料，请勿传播！！
四川大学 计算机学院 章毅教授

Handwritten digits recognition problem





Task:
Use Backpropagation algorithm to train a neural network to recognize handwritten digits.

内部资料 请勿传播！！！
四川大学计算机学院 章毅教授

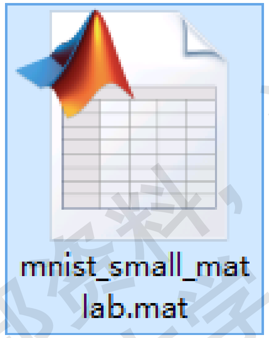
Step 1: Data Preparation

Dataset: **MNIST_small**

MNIST is a database of handwritten digits created by "re-mixing" the samples from MNIST's original datasets. It contains digits written by high school students and employees of the United States Census Bureau. The digits have been size-normalized and centered in 28×28 images.

MNIST_small dataset is a subset of MNIST containing 10000 training samples and 2000 testing samples.

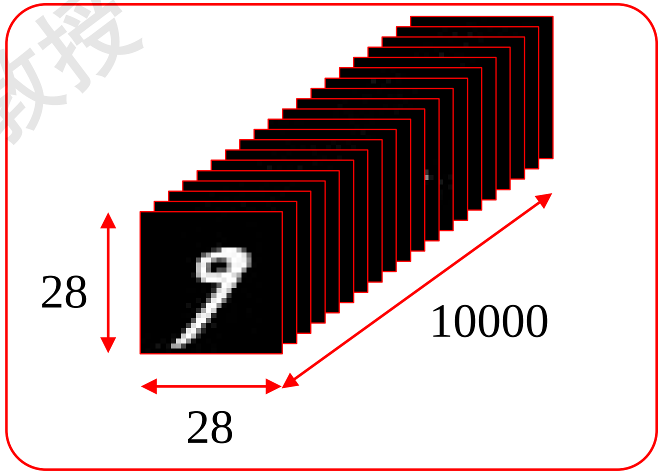
New York University

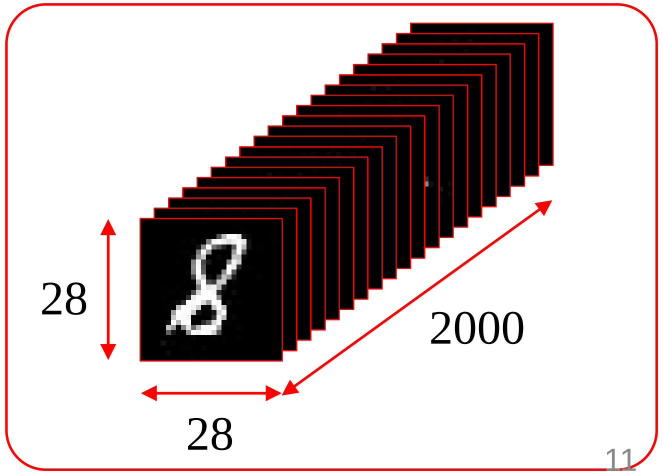


Training set
- Used for training network
- 10000 samples

Testing set
- Used for evaluating network performance
- 2000 samples

Data

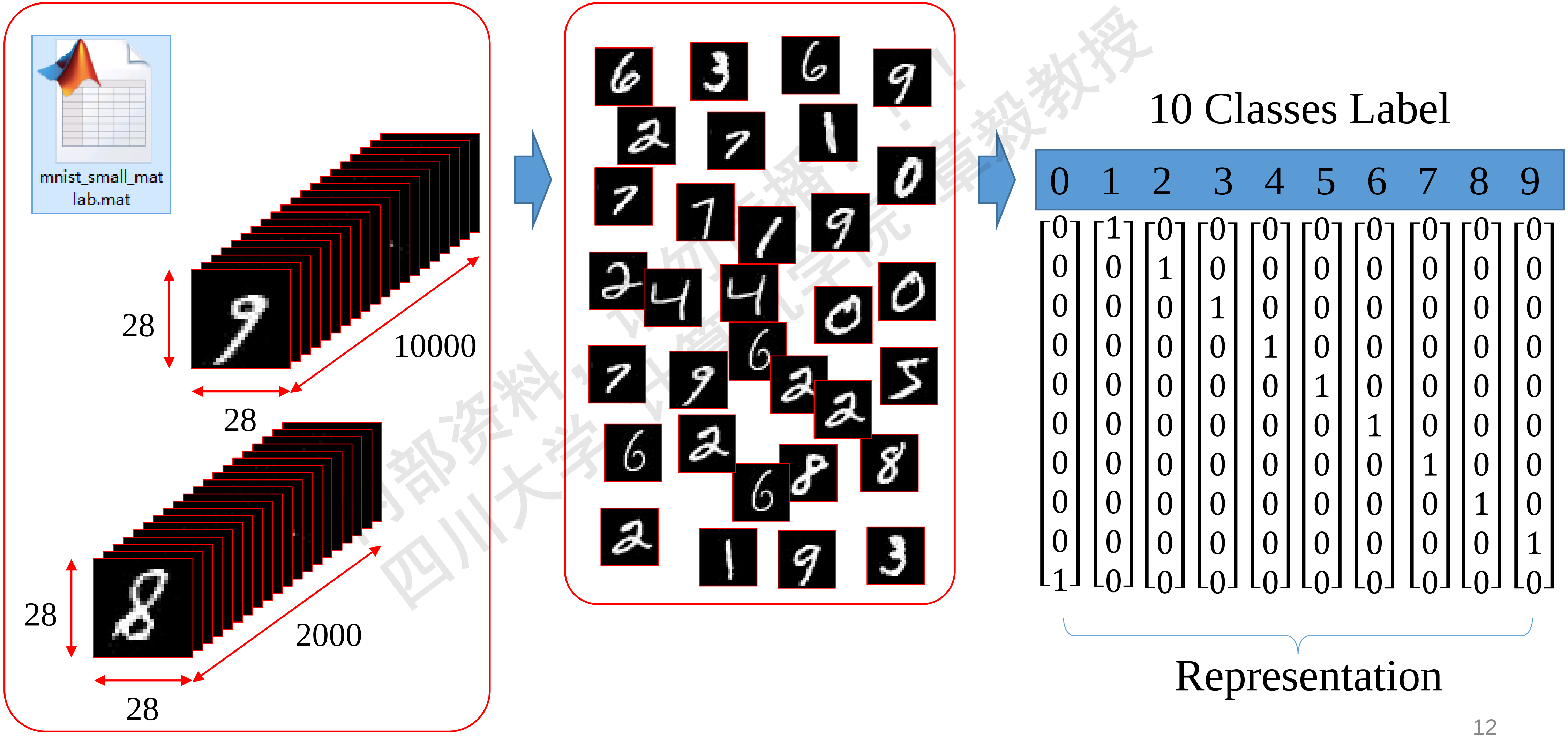




Download link:
MNIST http://yann.lecun.com/exdb/mnist/
MNIST_small: https://github.com/kswersky/nnet/blob/master/mnist_small.mat

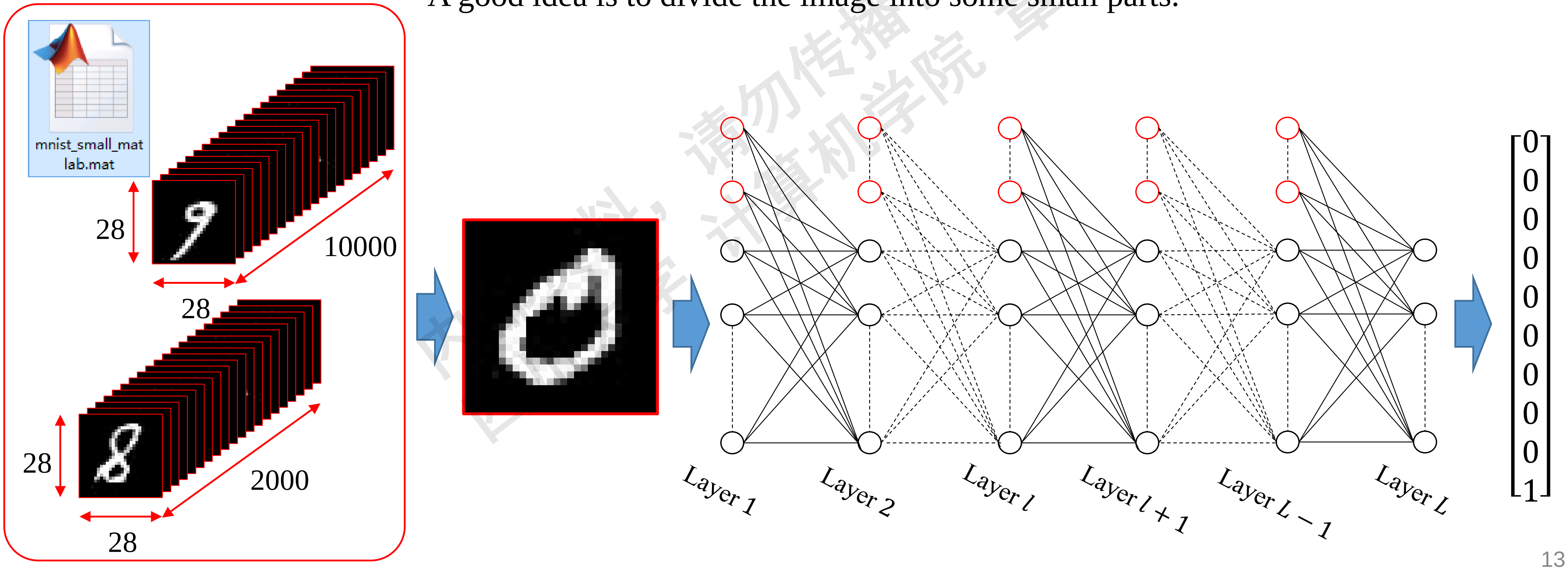
内部资料，请勿传播！！！
四川大学 计算机学院 章毅教授

Step 1: Data Preparation



Step 1: Data Preparation

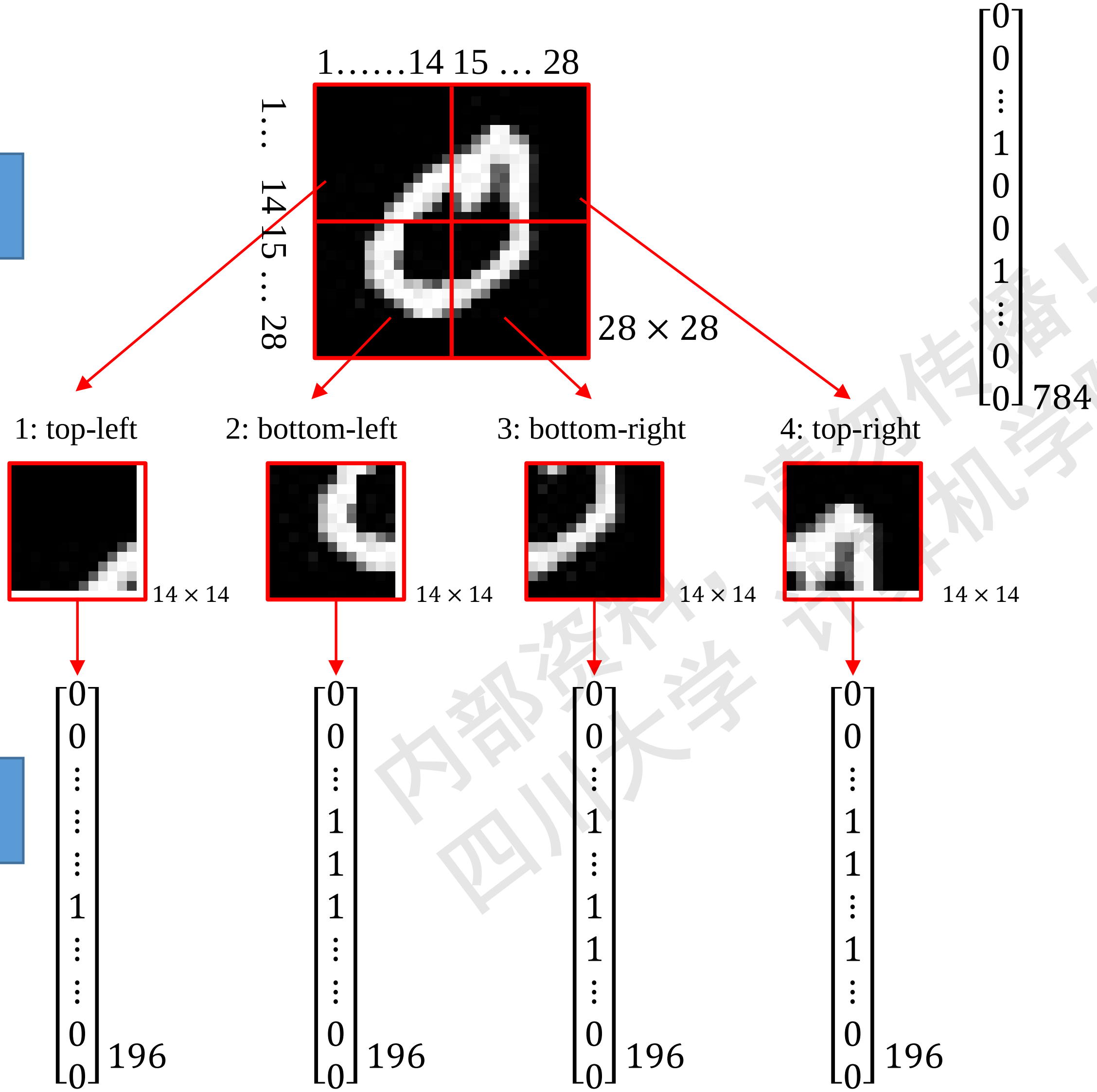
The input image is a $28 \times 28 = 784$ dimensional vector, relatively large in some situation.
A good idea is to divide the image into some small parts.



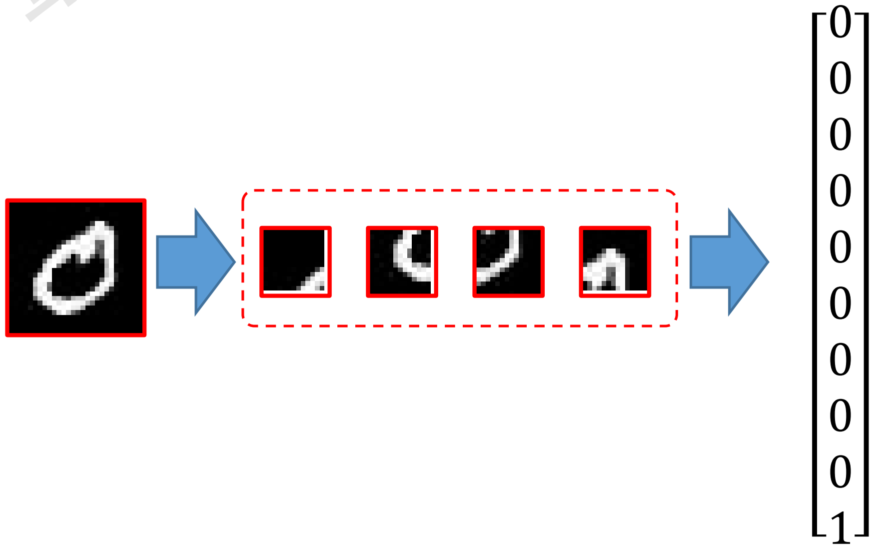
Step 1: Prepare Data Preparation

Dividing

digitizing

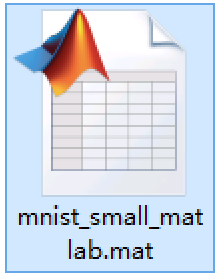


Representation



内部资料，请勿传播！！！ 章毅教授
四川大学 计算机学院

Step 1: Data Preparation



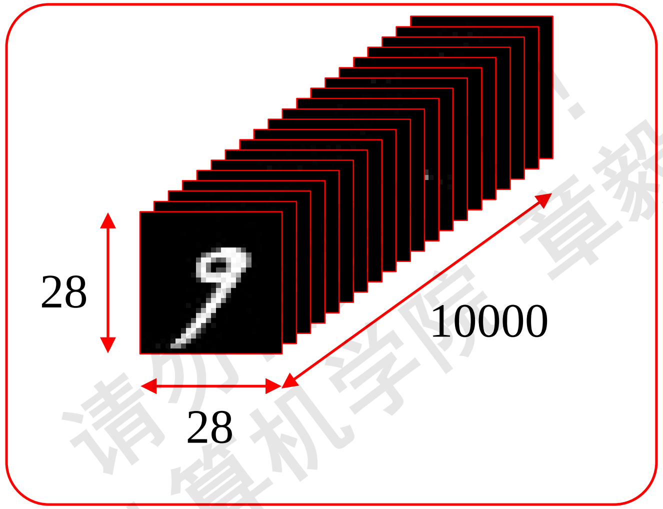
Training set

- ❑ Used for training network
- ❑ 10000 samples
- ❑ each sample contains four elements

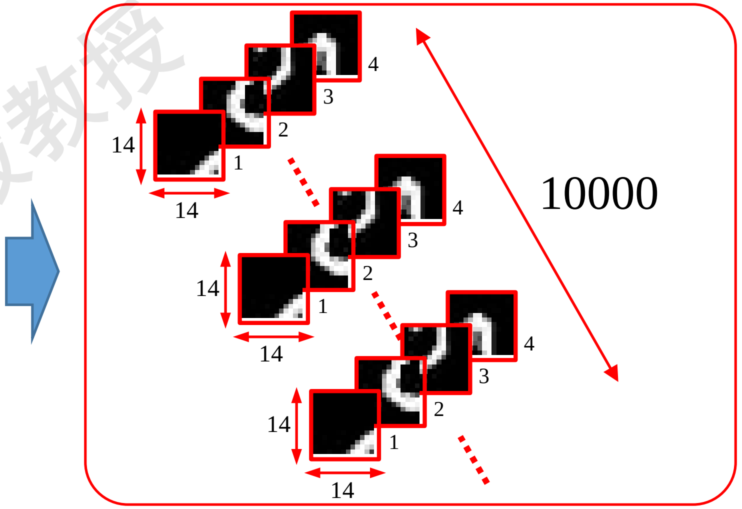
Testing set

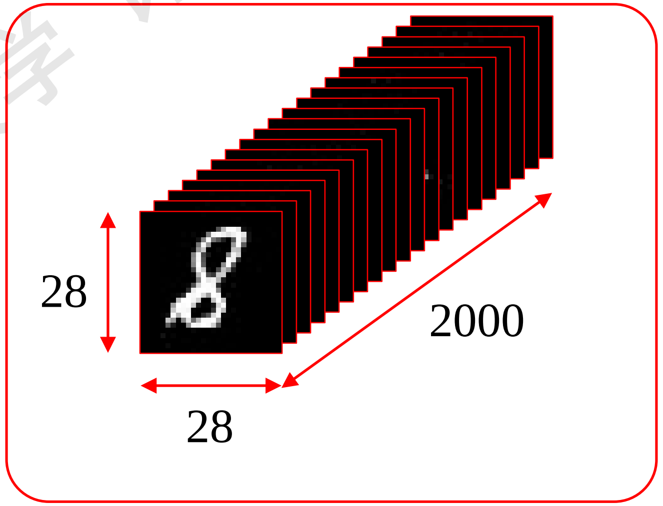
- ❑ Used for evaluating network performance
- ❑ 2000 samples
- ❑ each sample contains four elements

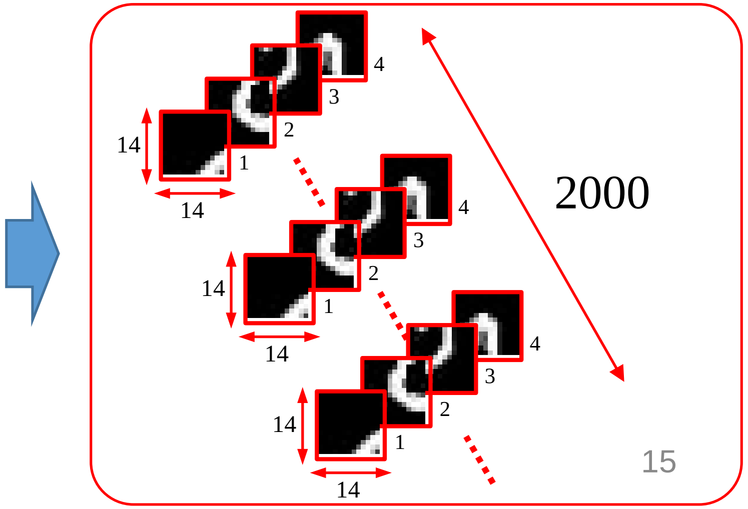
Training Data



Training Data

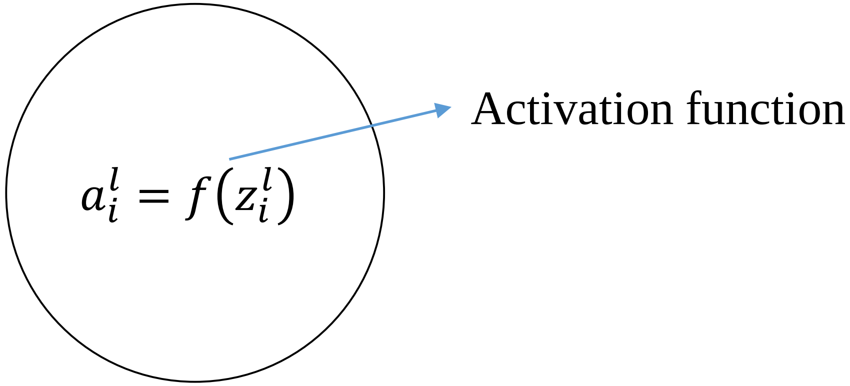






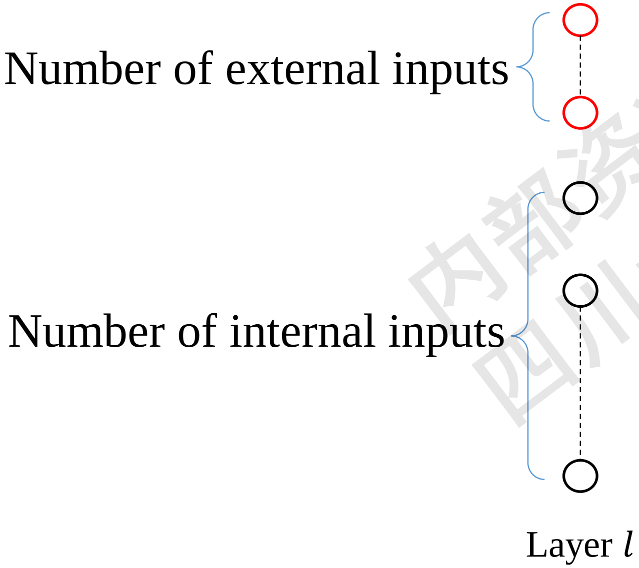
内部资料，请勿
四川大学 计算机学院 章毅教授

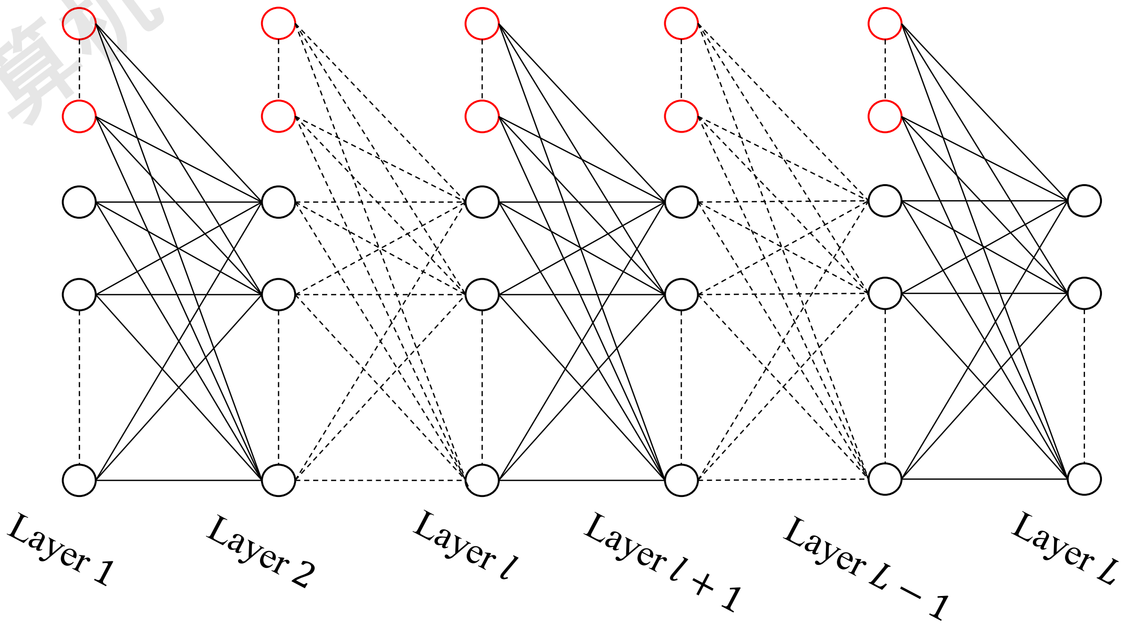
Step 2: Design Network Architecture



Network architecture design:

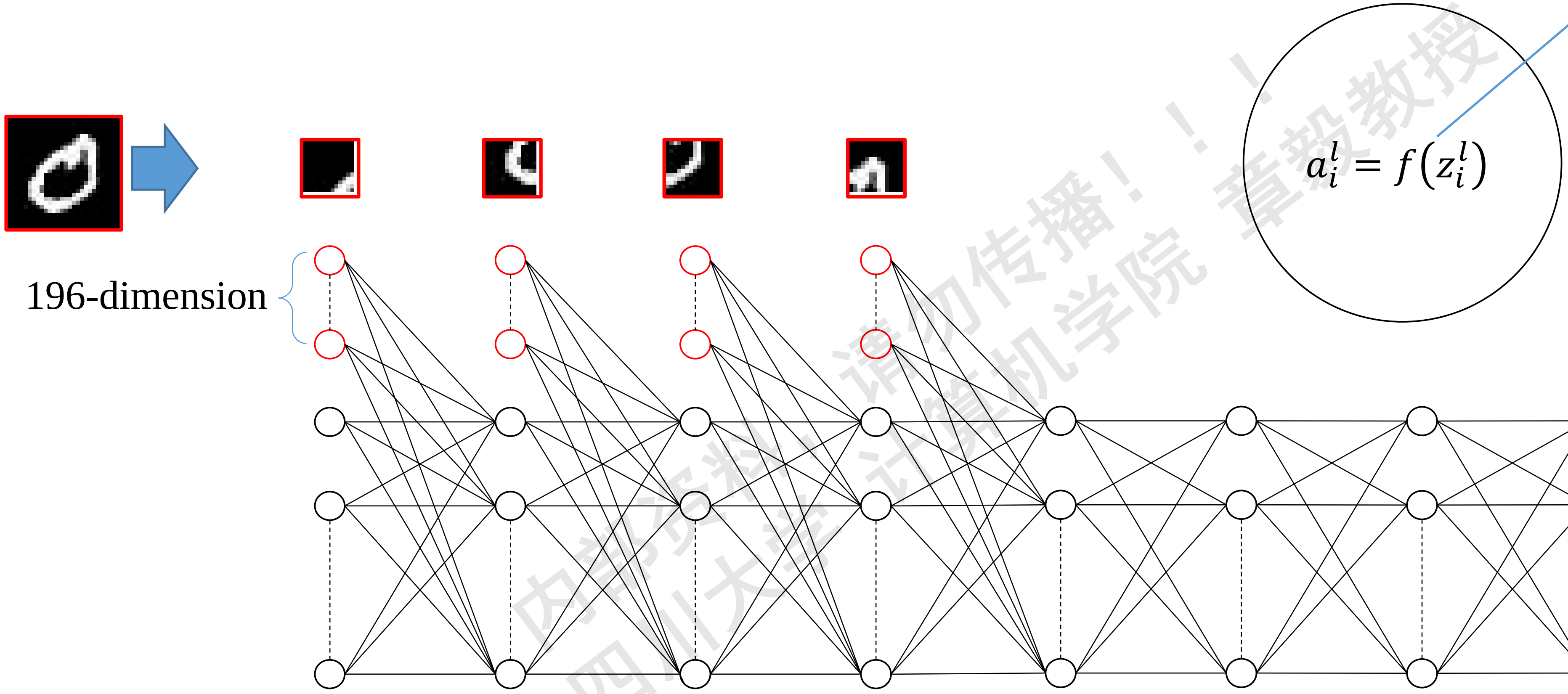
1. Number of layers
2. Number of neurons in each layer (external neurons and internal neurons)
3. Activation function





内部资料，请勿传播！
四川大学 计算机学院 章毅教授

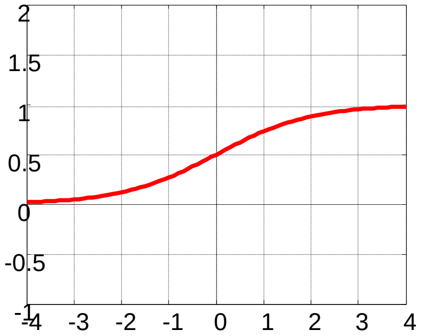
Step 2: Design Network Architecture



Layer 1 (196, 32) | Layer 2 (196, 32) | Layer 3 (196, 32) | Layer 4 (196, 32) | Layer 5 (0, 64) | Layer 6 (0, 64) | Layer 7 (0, 64) | Layer 8 (0, 10)

Sigmoid function

$$f(z) = \frac{1}{1+e^{-z}}$$



内部资料，请勿传播！！！
四川大学计算机学院 章毅教授

Step 3: Initial Weights and Learning Rate

Initialize Weight Connections

Random initialization:

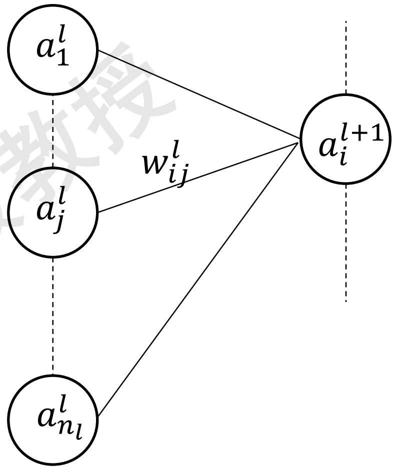
Method 1: Gaussian distribution: $w_{ij}^l \sim N(0,1)$

Method 2: Uniform distribution: $w_{ij}^l \sim U\left(-r^l, r^l\right)$

$$r^l = \sqrt{\frac{6}{p^l + q^{l+1}}}$$

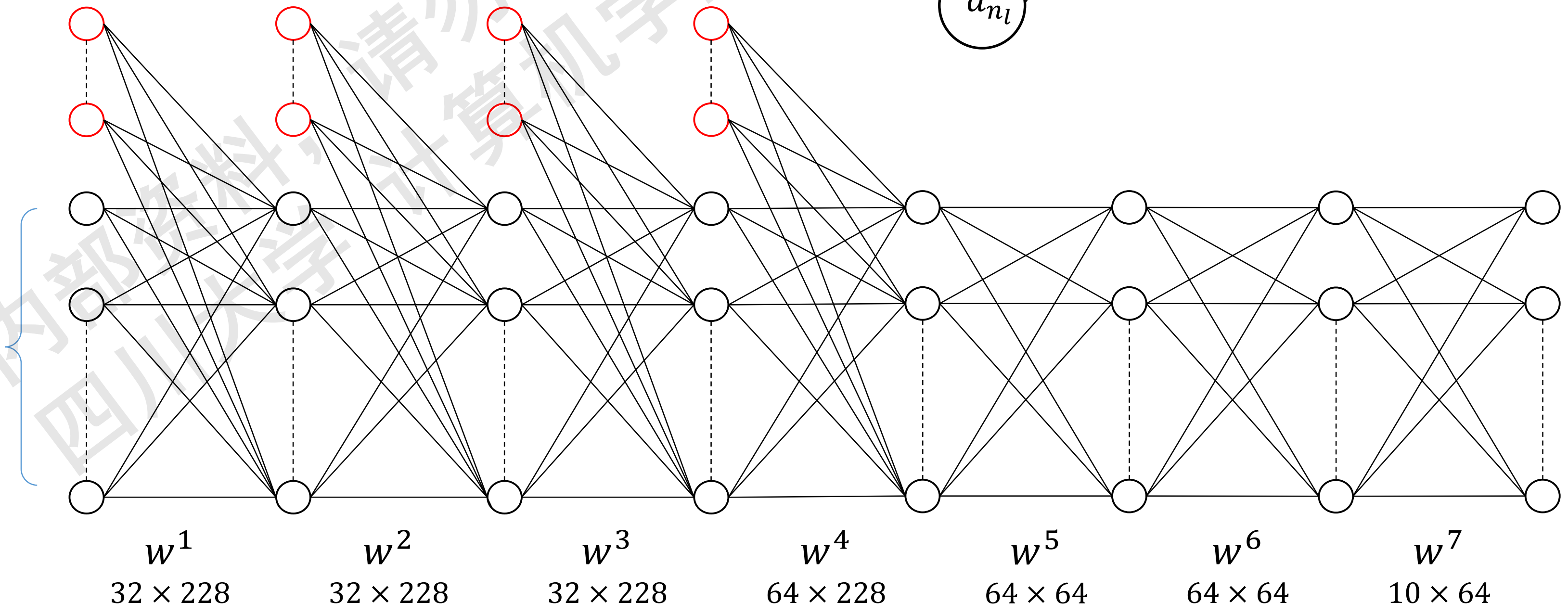
p^l: number of neurons in l layer

q^{l+1}: number of internal neurons in $l+1$ layer



Initialize Internal Representation of Layer 1:

$$a_i^1 = 0,$$
or
$$a_i^1 = 1,$$
…

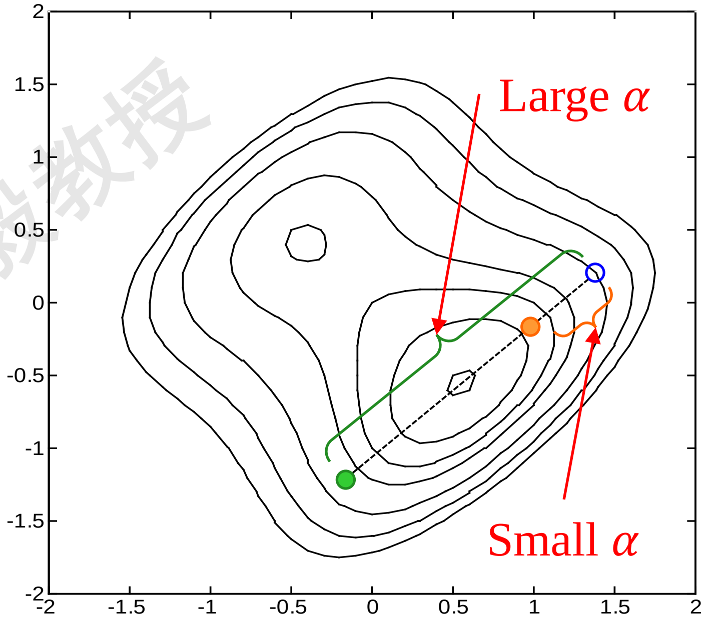


内部资料，请勿传播！！！
四川大学计算机学院 章毅教授

Step 3: Initialization and Learning Rate

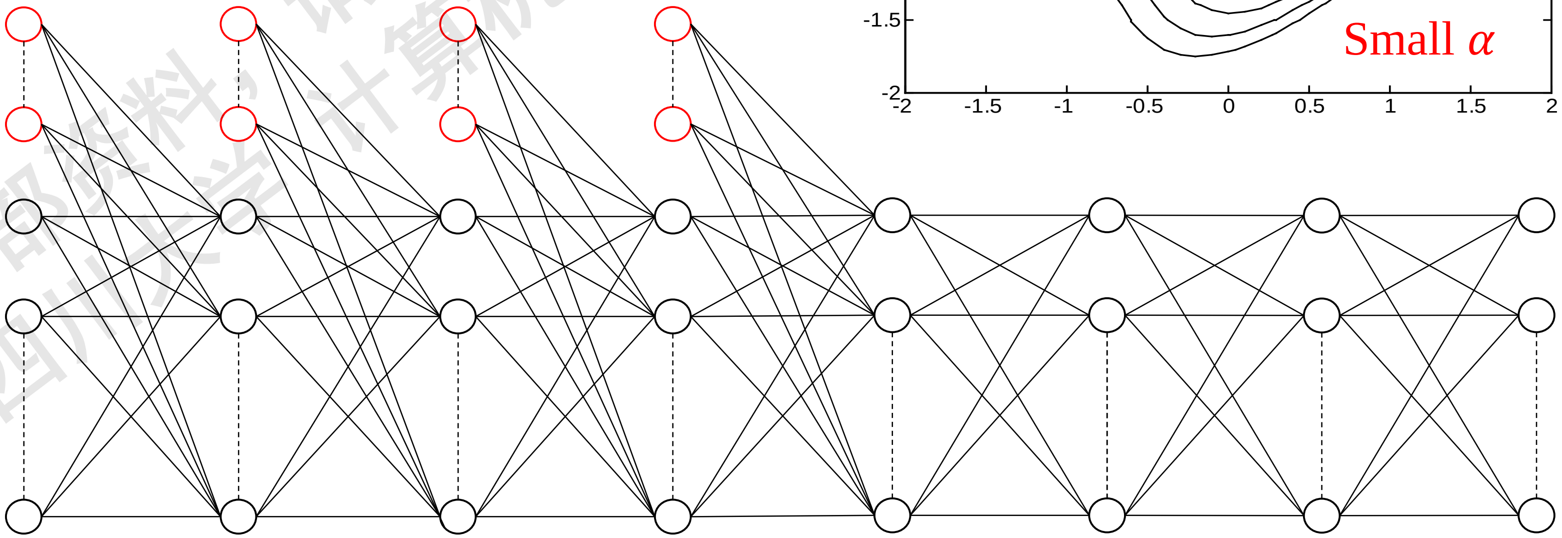
Learning rate:

- Small: slow learning, long learning time.
- Large: fast learning, possibly not converge to minima.



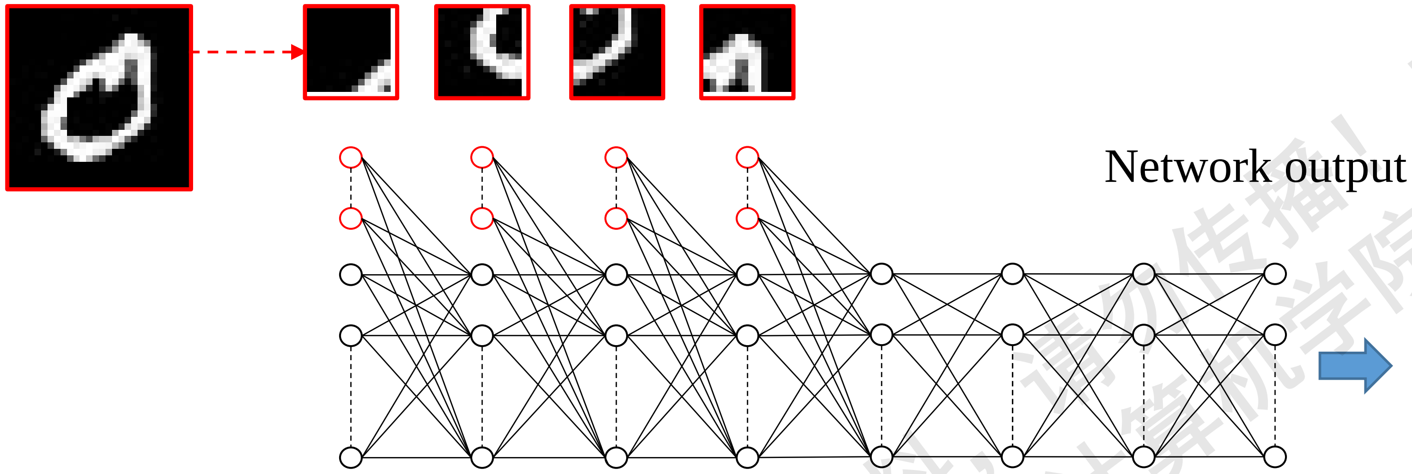
$$w_{ji}^{l} \leftarrow w_{ji}^{l} - \alpha \cdot \frac{\partial J}{\partial w_{ji}^{l}}$$

$$\alpha = \cdots, 0.5, 1, 2, 4, \cdots$$



内部资料，请勿传播！！！ 四川大学计算机学院 章毅教授

Step 4: Define Cost Function



Network output $\begin{bmatrix} 0 \\ 0 \\ 0.1 \\ 0 \\ 0 \\ 0 \\ 0.2 \\ 0 \\ 0 \\ 0.7 \end{bmatrix}$

Cost function

$$J = \frac{1}{2}\sum_{j=1}^{n_L} e_j^2$$

$$e_j = a_j^L - y_j^L \qquad e = \begin{bmatrix} 0 \\ 0 \\ 0.1 \\ 0 \\ 0 \\ 0 \\ 0.2 \\ 0 \\ 0 \\ -0.3 \end{bmatrix}$$

Target output $\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$

Label

0 → representation → Target output

内部资料，请勿传播！！！
四川大学 计算机学院 章毅教授

Step 5: Define Evaluation Index

$$\text{Accuracy} = \frac{\text{number of correct prediction}}{\text{number of samples}}$$

An example

Tested data 9

Prediction 7 9 0 4 8 8 8 3 1

Correct prediction
7

Incorrect prediction
2

$$\text{Accuracy} = \frac{7}{9} = 77.78\%$$

Test on training set:

- Reflect the progress of training.
- Evaluate the ability of the model to fit given data.

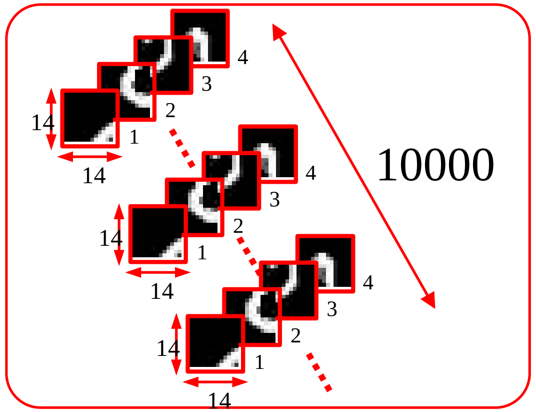
Test on testing set:

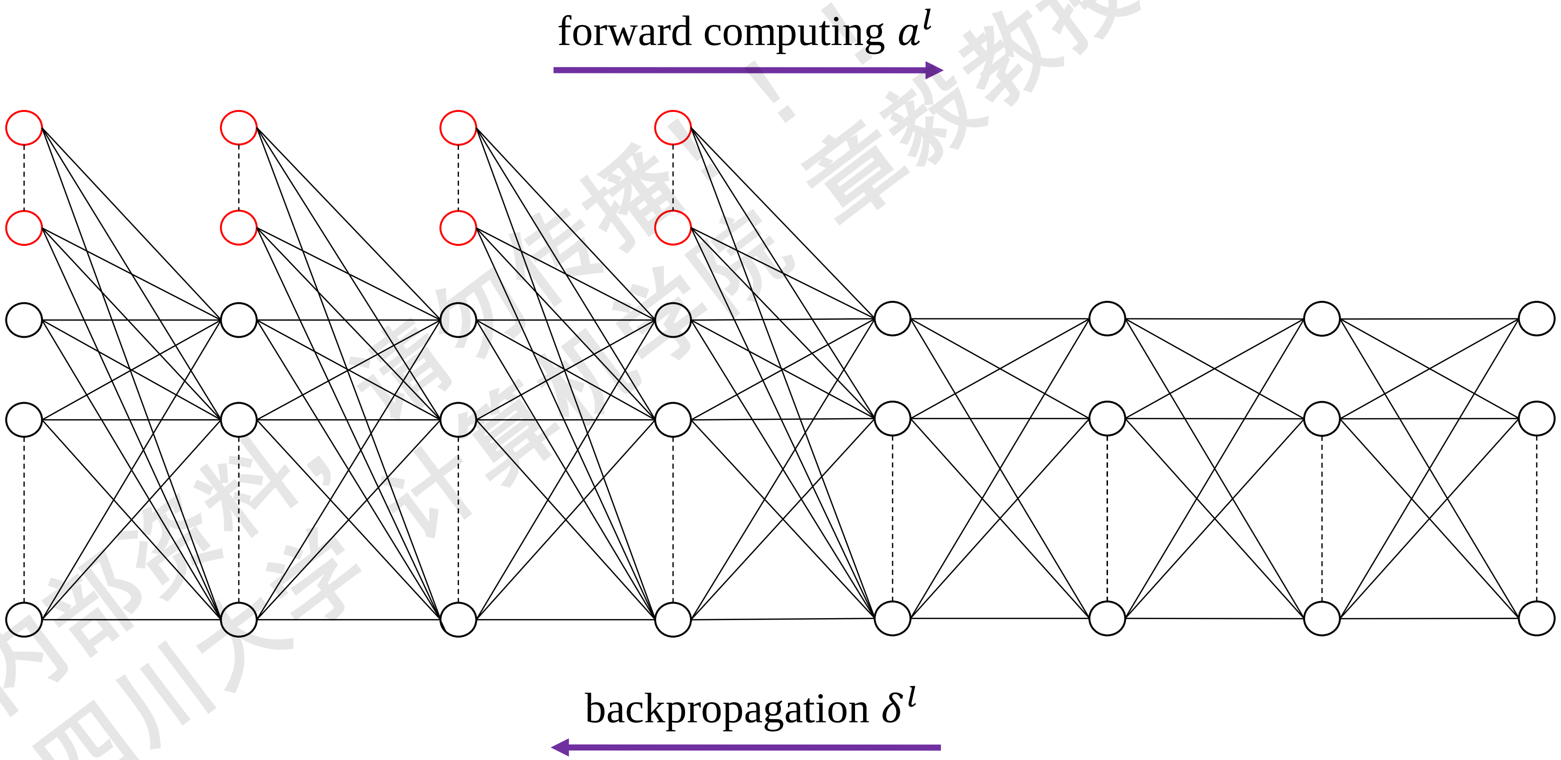
- Evaluate the ability of the model to generalize the knowledge.

内部资料 请勿传播！！！
四川大学 计算机学院 章毅教授

Step 6: Train the Network

Training Data





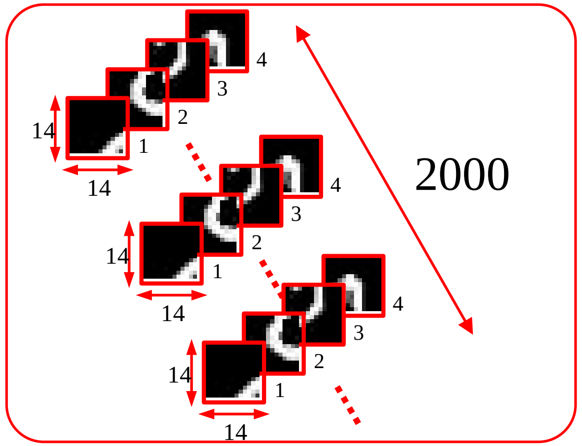
Updating weights

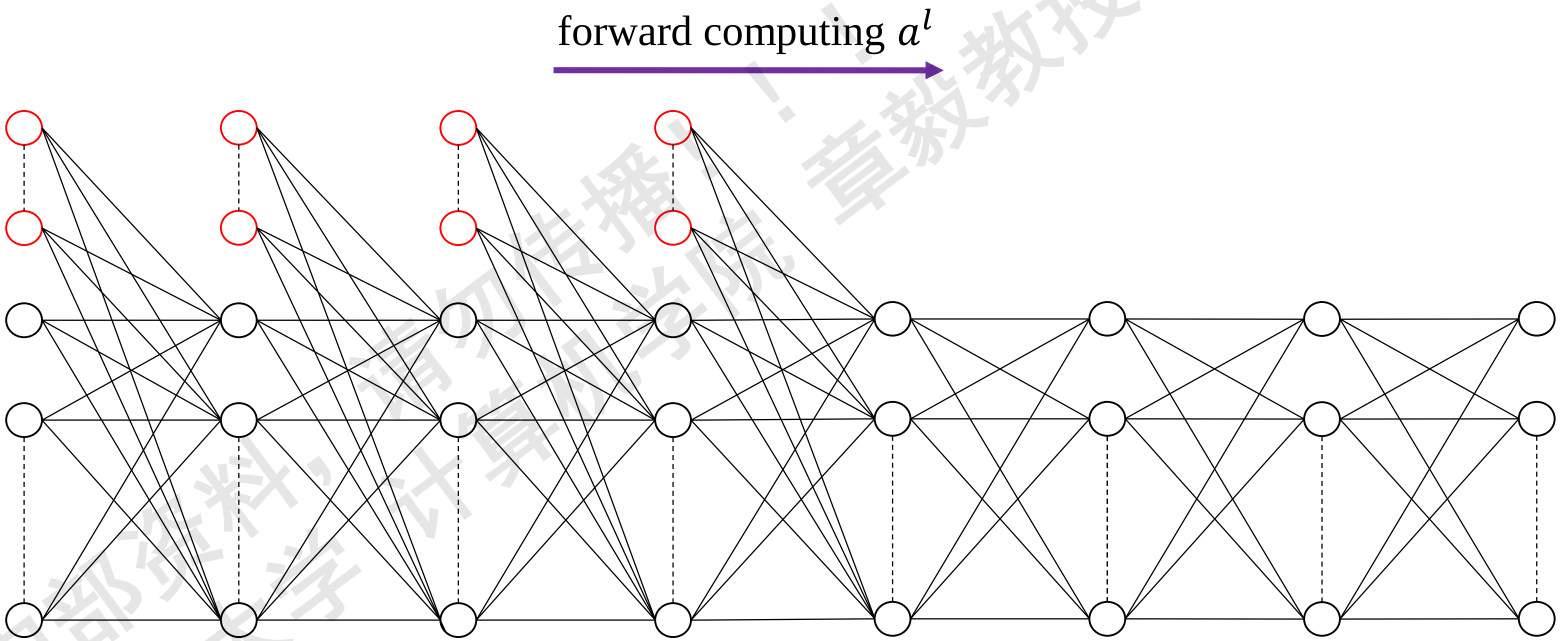
$$w_{ji}^l \leftarrow w_{ji}^l - \alpha \cdot \frac{\partial J}{\partial w_{ji}^l}$$

内部资料，请勿传播！
四川大学计算机学院 章毅教授

Step 7: Test the Network

Test Data





内部资料，请勿传播！！ 四川大学计算机学院 章毅教授

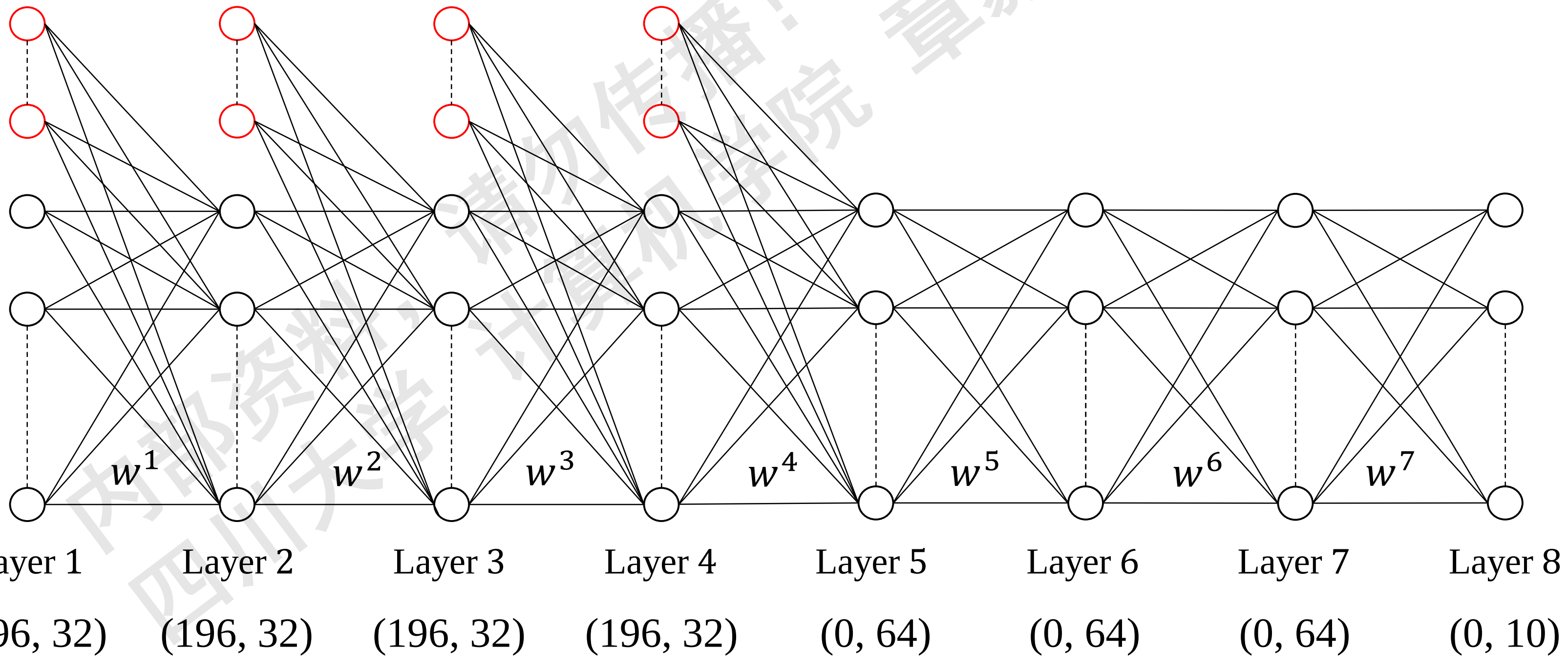
Calculating the Evaluation index

$$\text{Accuracy} = \frac{\text{number of correct prediction}}{2000}$$

Step 8: Store the Network Parameters

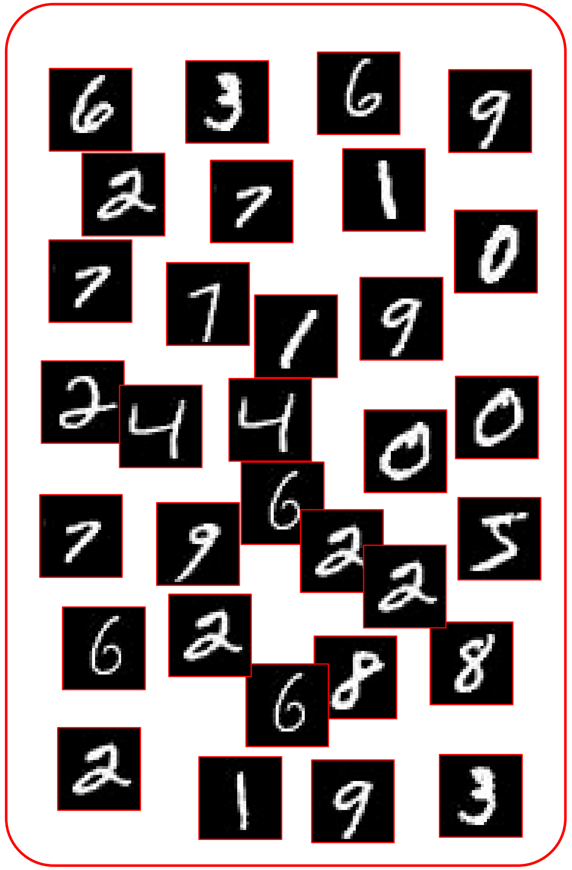
Learning rate α

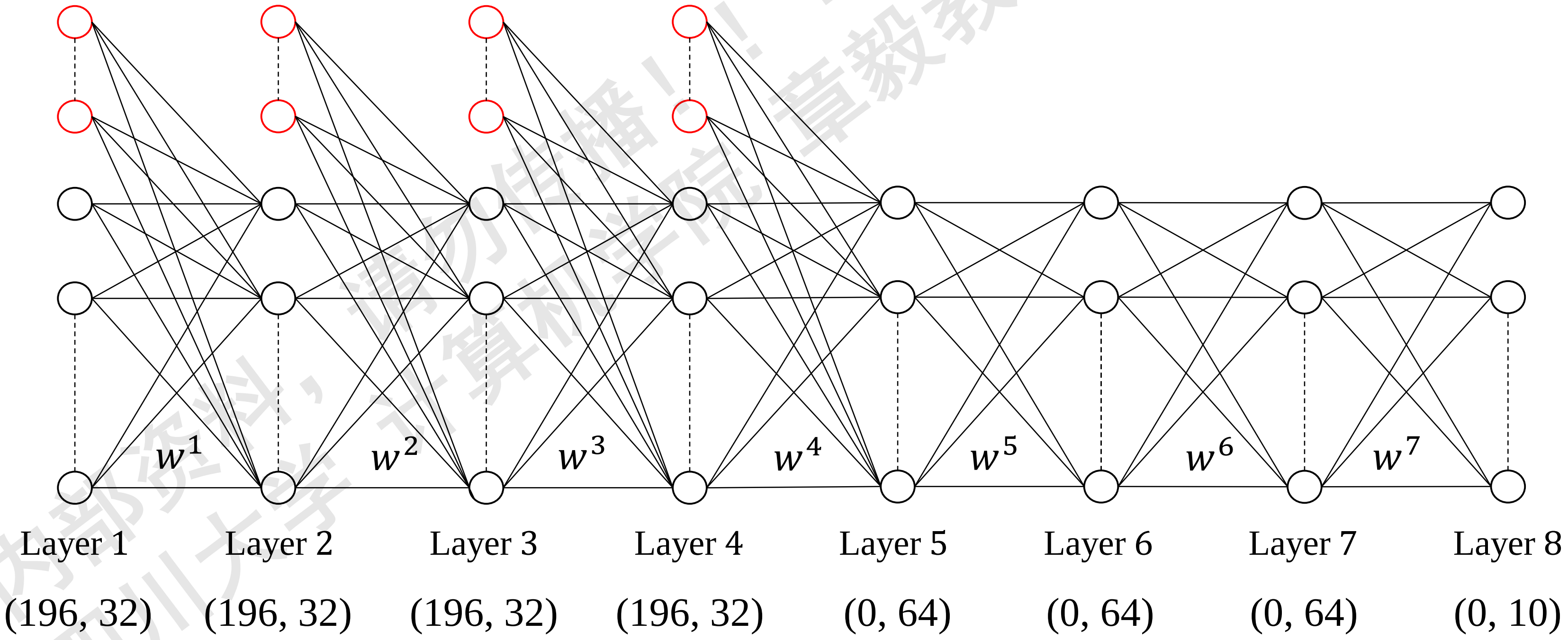
Initialize value of Layer 1



内部资料，请勿传播！！！ 四川大学 计算机学院 章毅教授

Step 9: Using Trained Network for Applications





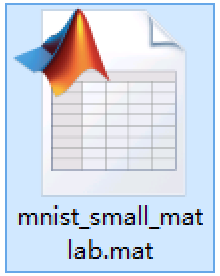
内部资料，请勿传播！！！
四川大学 计算机学院 章毅教授

Outline

- Brief Review of Backpropagation Algorithm
- An Illustrating Example
- Experiments
- Assignment

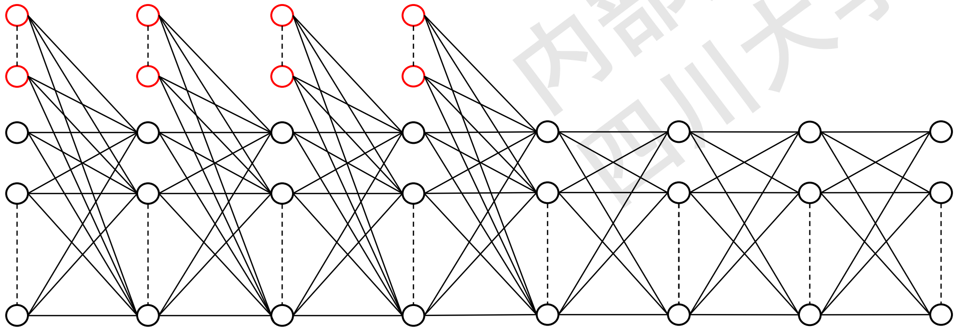
内部资料，请勿传播！！
四川大学 计算机学院 章毅教授

Experiments: Data Preparation

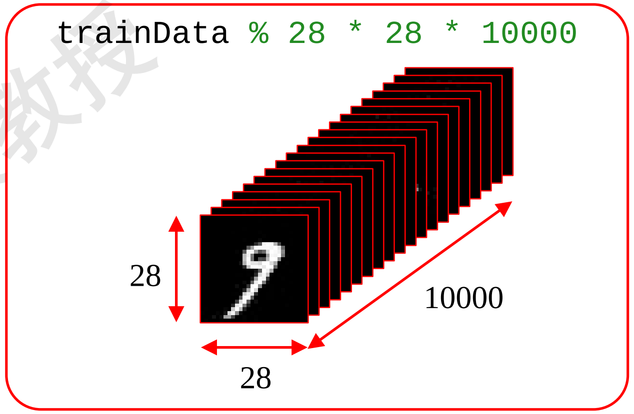


名称	值
trainData	28x28x10000 double
testData	28x28x2000 double
trainLabels	10x10000 double
testLabels	10x2000 double

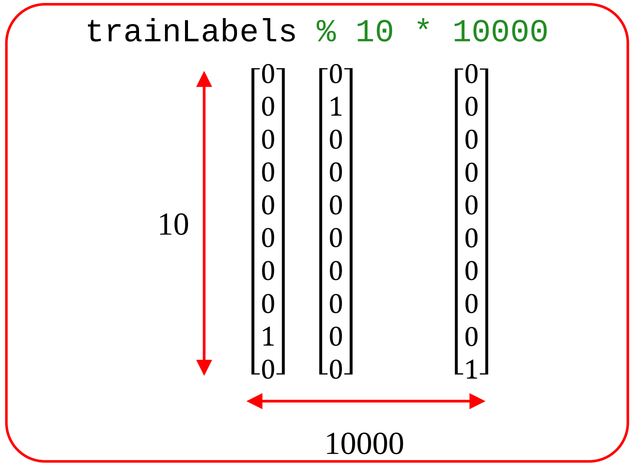
```
% prepare the data set
load mnist_small_matlab.mat
```



Training Data

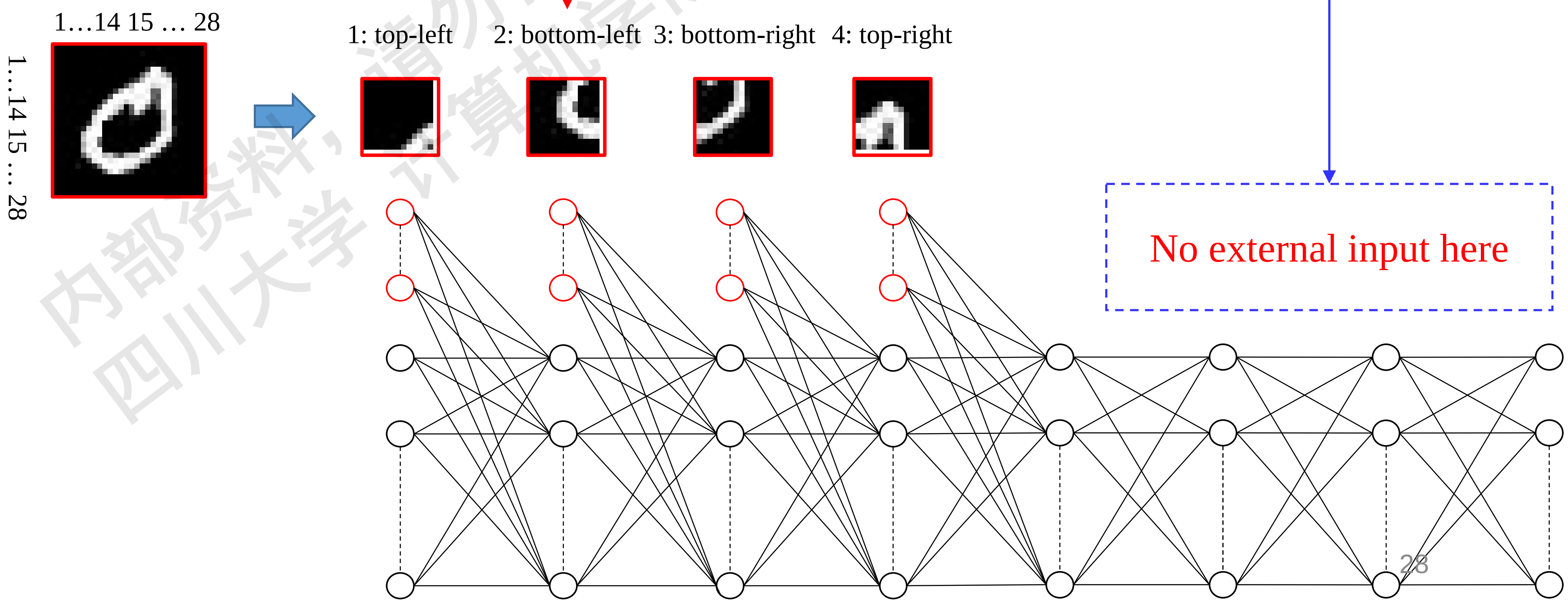


Label



内部资料，请勿传播！！！ 四川大学 计算机学院 章毅教授

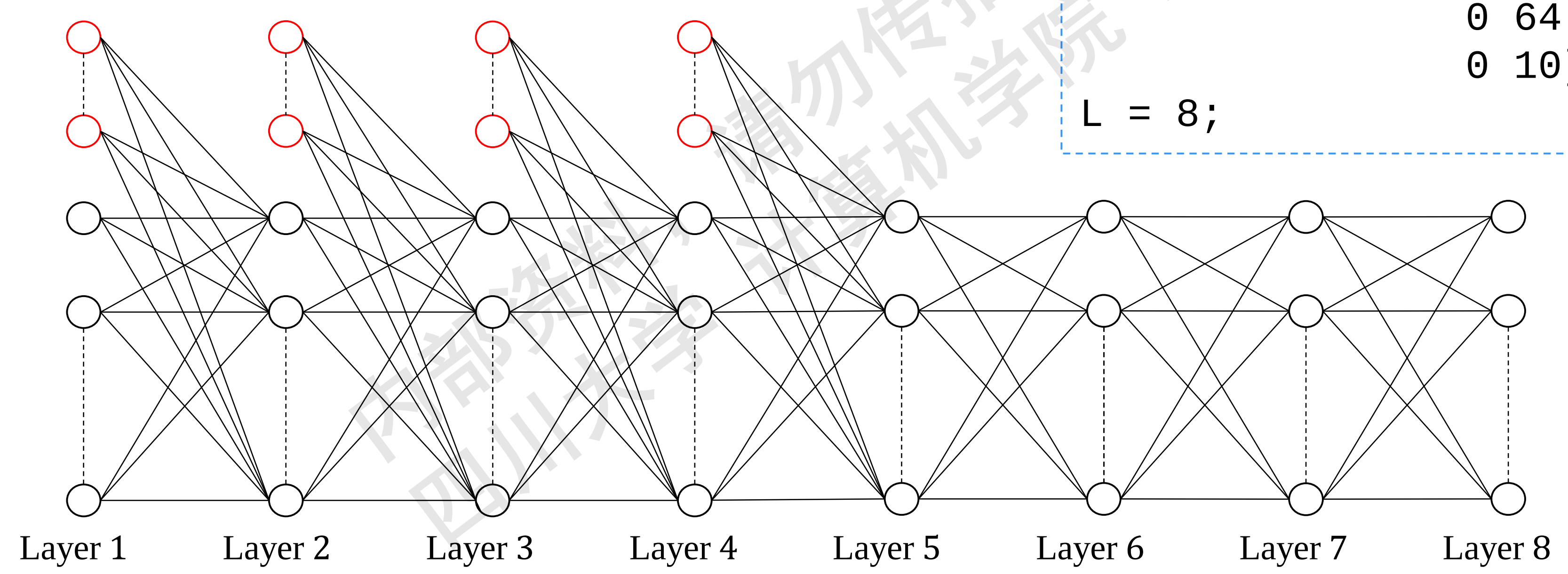
```
% prepare training data
train_size = 10000;
X_train{1} = reshape(trainData(1:14,1:14,:),[],train_size);% top-left
X_train{2} = reshape(trainData(15:28,1:14,:),[],train_size); % bottom-left
X_train{3} = reshape(trainData(15:28,15:28,:),[],train_size); % bottom-right
X_train{4} = reshape(trainData(1:14,15:28,:),[],train_size); % top-right
X_train{5} = zeros(0, train_size);
X_train{6} = zeros(0, train_size);
X_train{7} = zeros(0, train_size);
X_train{8} = zeros(0, train_size);
% prepare testing data
% ...
```



内部资料，请勿传播！！！ 章毅教授
四川大学计算机学院

Experiments: Architecture

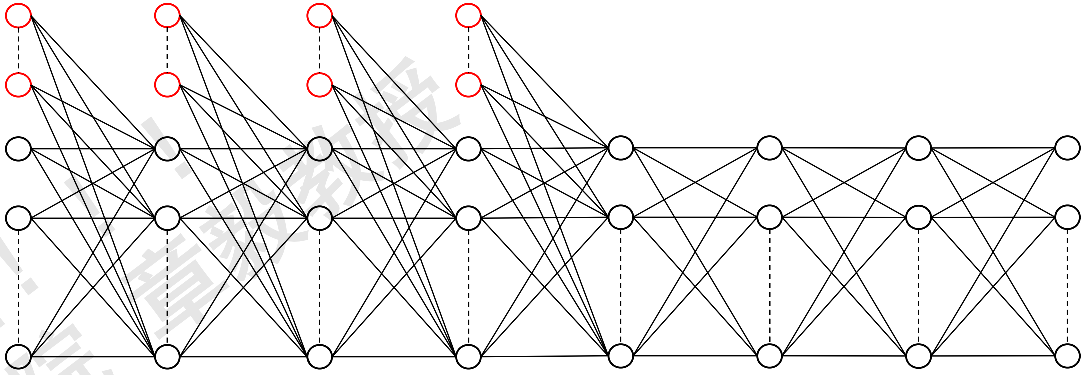
```
% define network architecture
% - 1st column: external neurons
% - 2nd column: internal neurons
layer_size = [196 32    % layer 1
              196 32    % layer 2
              196 32    % layer 3
              196 32    % layer 4
                0 64    % layer 5
                0 64    % layer 6
                0 64    % layer 7
                0 10];  % layer 8
L = 8;
```



Dimensions (196, 32) (196, 32) (196, 32) (196, 32) (0, 64) (0, 64) (0, 64) (0, 10)

内部资料，请勿传播！！！ 四川大学 计算机学院 章毅教授

Experiments: Initialize Weights



Gaussian distribution: $w_{ij}^{l} \sim N(0,1)$

```
% initialize weights
for l = 1:L-1
    w{l} = randn(layer_size(l+1,2), sum(layer_size(l,:)));
end
```

Uniform distribution: $w_{ij}^{l} \sim U\left(-r^{l}, r^{l}\right)$

```
% initialize weights
for l = 1:L-1
    % a tricky, but effective, initialization
    w{l} = (rand(layer_size(l+1,2), sum(layer_size(l,:))) * 2 -1)
               * sqrt(6/(layer_size(l+1,2)+sum(layer_size(l,:))));
end
```

$$r^{l} = \sqrt{\frac{6}{p^{l} + q^{l+1}}}$$

p^{l}: number of neurons in l layer
q^{l+1}: number of internal neurons in $l+1$ layer

内部资料，请勿传播！
四川大学 计算机学院

Experiments: Run the Network

Learning rate

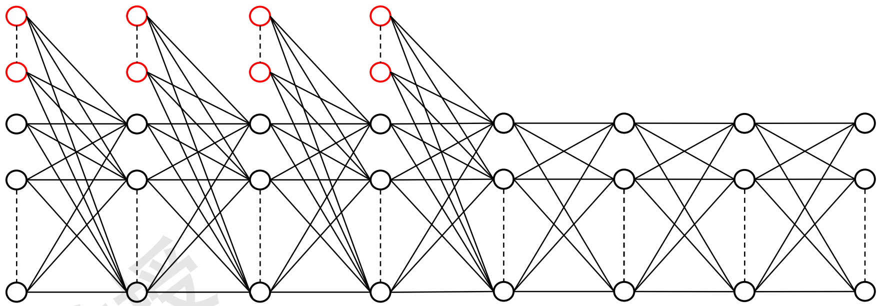
Number of iteration

Number of samples in a batch

```
% choose parameters
alpha = 1;
max_iter = 300;
mini_batch = 100;
```

```
% loop until converge
for iter = 1:max_iter
    % for each mini-batch
        % batch forward computation
        % batch backward computation
        % cumulate and update weight
end
```

Mini-batch BP implement



BP Algorithm:

Step 1. Input the training data set $D = \{(x, y)\}$

Step 2. Initial each w_{ij}^{l}, and choose a learning rate α.

Step 3. For each mini-batch sample $D_m \subset D$

$$a^1 \leftarrow samples\ in\ D_m$$

for $l = 2: L$

$$a^l \leftarrow fc(w^l, a^l);$$

end

$$\delta^L = \frac{\partial J}{\partial z^L};$$

for $l = L-1: 2$

$$\delta^l \leftarrow bc(w^l, \delta^{l+1});$$

end

$$\frac{\partial J}{\partial w_{ji}^l} \leftarrow \frac{\partial J}{\partial w_{ji}^l} + \delta_j^{l+1} \cdot a_i^l;$$

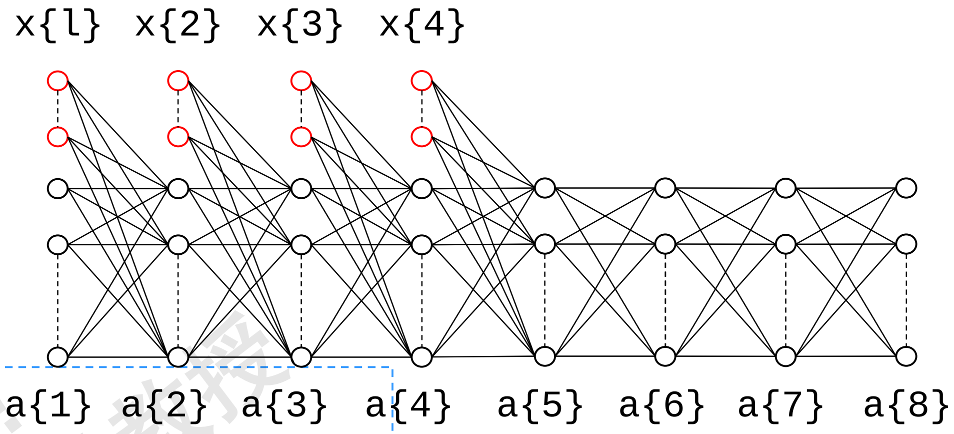
Step 4. Updating

$$w_{ji}^l \leftarrow w_{ji}^l - \alpha \cdot \frac{\partial J}{\partial w_{ji}^l};$$

Step 5. Return to Step 3 until each w^l converge.

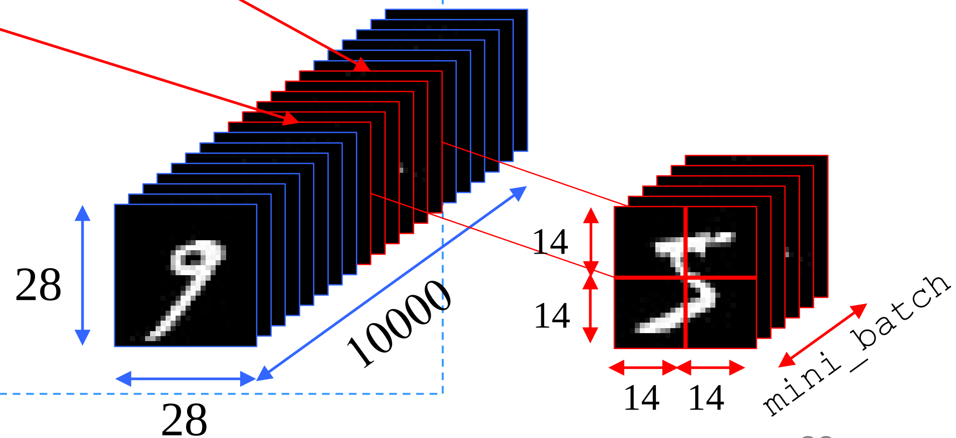
内部资料，请勿传播！！！章毅教授
四川大学计算机学院

Experiments: mini-batch BP



shuffle index

```
ind = randperm(train_size);
% for each mini-batch
for k = 1:ceil(train_size/mini_batch)
    % prepare internal inputs
    a{1} = zeros(layer_size(1,2),mini_batch);
    % prepare external inputs
    for l=1:L
        x{l} = X_train{l}(:,ind((k-1)*mini_batch+1:min(k*mini_batch, train_size)));
    end
    % prepare labels
    y = double(trainLabels(:,ind((k-1)*mini_batch+1:min(k*mini_batch, train_size))));
    % forward computation
    % ...
    % cost function and error
    % ...
    % backward computation
    % ...
    % update weight
    % ...
end
```

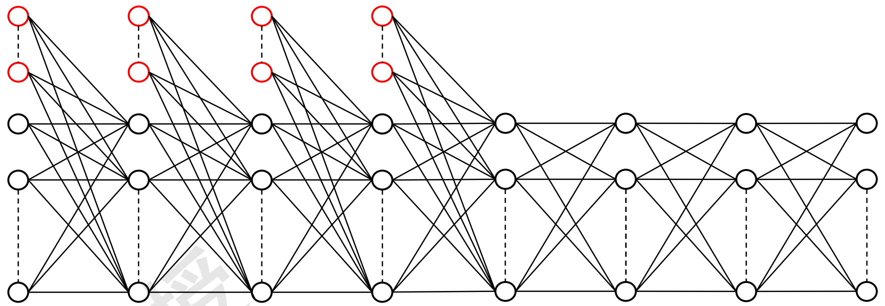


training data

mini-batch

内部资料，请勿传播！！ 章毅教授
四川大学 计算机学院

Experiments: BP



```
% forward computation
for l=1:L-1
    [a{l+1}, z{l+1}] = fc(w{l}, a{l}, x{l});
end

% Compute delta of last layer
delta{L} = (a{L} - y).* a{L} .*(1-a{L});

% backward computation
for l=L-1:-1:2
    delta{l} = bc(w{l}, z{l}, delta{l+1});
end

% update weight
for l=1:L-1
    gw = delta{l+1} * [x{l};a{l}]' / mini_batch;
    w{l} = w{l} - alpha * gw;
end
```

BP Algorithm:

Step 1. Input the training data set $D=\{(x,y)\}$

Step 2. Initial each w_{ij}^{l}, and choose a learning rate α.

Step 3. For each mini-batch sample $D_m \subset D$

$a^1 \leftarrow samples\ in\ D_m$

for $l = 2{:}L$

$a^l \leftarrow fc(w^l, a^l);$

end

$$\delta^L = \frac{\partial J}{\partial z^L};$$

for $l = L-1{:}2$

$\delta^l \leftarrow bc(w^l, \delta^{l+1});$

end

$$\frac{\partial J}{\partial w_{ji}^{l}} \leftarrow \frac{\partial J}{\partial w_{ji}^{l}} + \delta_j^{l+1} \cdot a_i^l;$$

Step 4. Updating

$$w_{ji}^{l} \leftarrow w_{ji}^{l} - \alpha \cdot \frac{\partial J}{\partial w_{ji}^{l}};$$

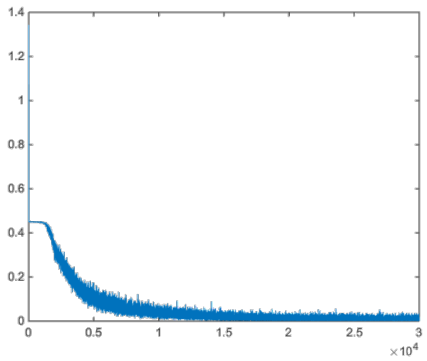
Step 5. Return to Step 3 until each w^l converge.

内部资料，请勿传播！！！ 章毅教授
四川大学 计算机学院

Experiments: Plotting

Cost function

$$J = \frac{1}{2}\sum_{j=1}^{n_L}\left(a_j^L - y_j^L\right)^2$$

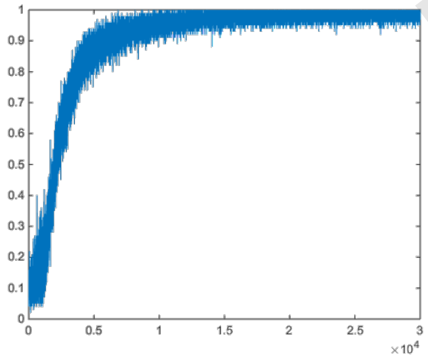



```
% cost function
J = [J 1/2/mini_batch*sum((a{L}(:)-y(:)).^2)];
figure
plot(J);
```

Accuracy

$$\text{Acc} = \frac{\textit{number of correct prediction}}{\textit{number of samples}}$$

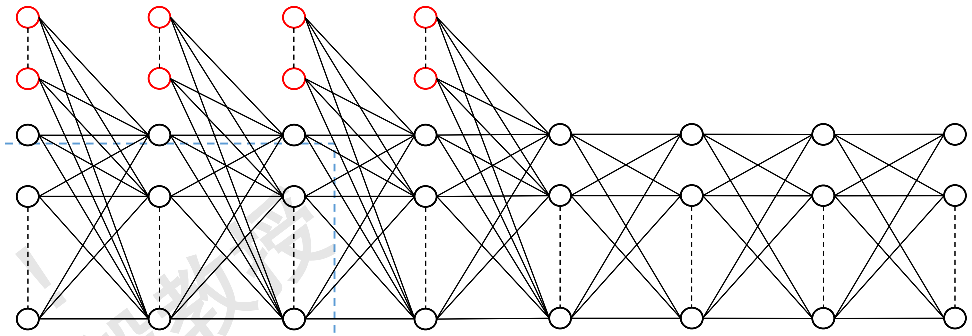
Use max output as prediction




```
% accuary on training batch
[~,ind_train] = max(y);
[~,ind_pred] = max(a{L});
Acc= [Acc sum(ind_train == ind_pred) / mini_batch];
figure
plot(Acc);
```

内部资料，请勿传播！！ 章毅教授
四川大学 计算机学院

Experiments: Testing



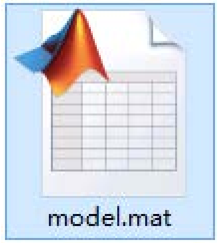
```
% test on training set
a{1} = zeros(layer_size(1,2),train_size);
for l=1:L-1
    a{l+1} = fc(w{l}, a{l}, X_train{l});
end
[~,ind_test] = max(trainLabels);
[~,ind_pred] = max(a{L});
train_acc = sum(ind_test == ind_pred)/train_size;
fprintf('Accuracy on training dataset is %f%%\n', train_acc*100);
```

```
% test on testing set
a{1} = zeros(layer_size(1,2),test_size);
for l=1:L-1
    a{l+1} = fc(w{l}, a{l}, X_test{l});
end
[~,ind_test] = max(testLabels);
[~,ind_pred] = max(a{L});
test_acc = sum(ind_test == ind_pred)/test_size;
fprintf('Accuracy on testing dataset is %f%%\n', test_acc*100);
```

内部资料，请勿传播！！
四川大学 计算机学院 章毅教授

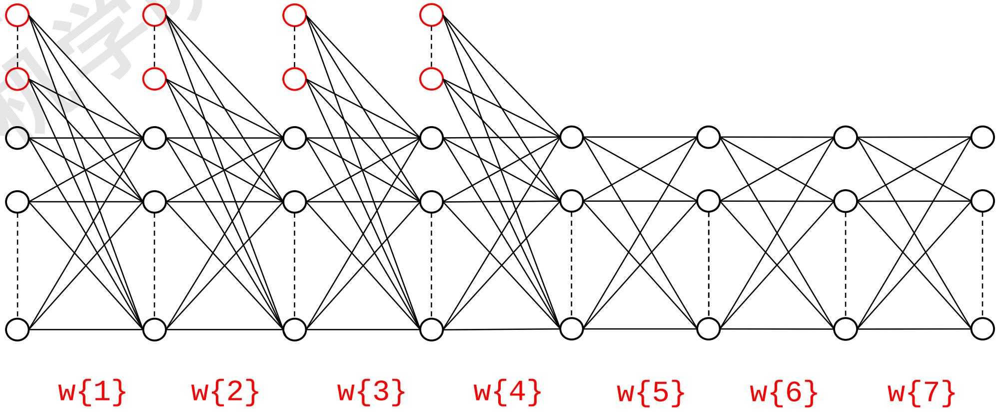
Experiments: Store the Network Parameters

```
% save model
save model.mat w layer_size
```



名称	值
layer_size	8x2 double
w	1x7 cell

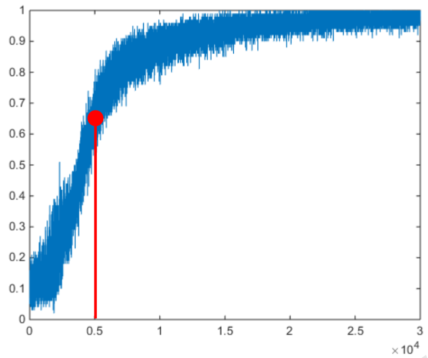
This is very important!



内部资料，请勿传播！！！
四川大学 计算机学院 章毅教授

Results: Learning Rate

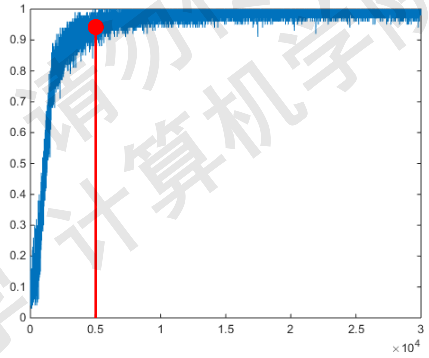
```
% learning rate
alpha = 0.5;
```



Accuracy

- Training=98.05%
- Testing=94.40%

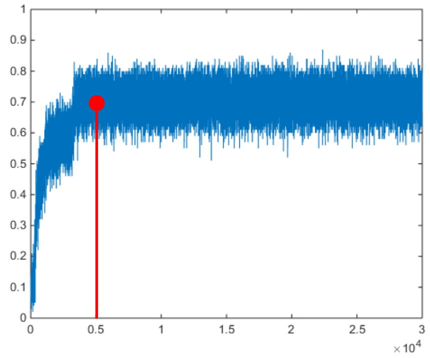
```
% learning rate
alpha = 2;
```



Accuracy

- Training=99.14%
- Testing=95.40%

```
% learning rate
alpha = 8;
```

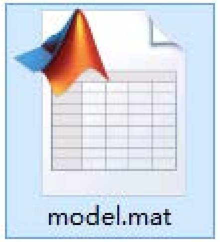


Accuracy

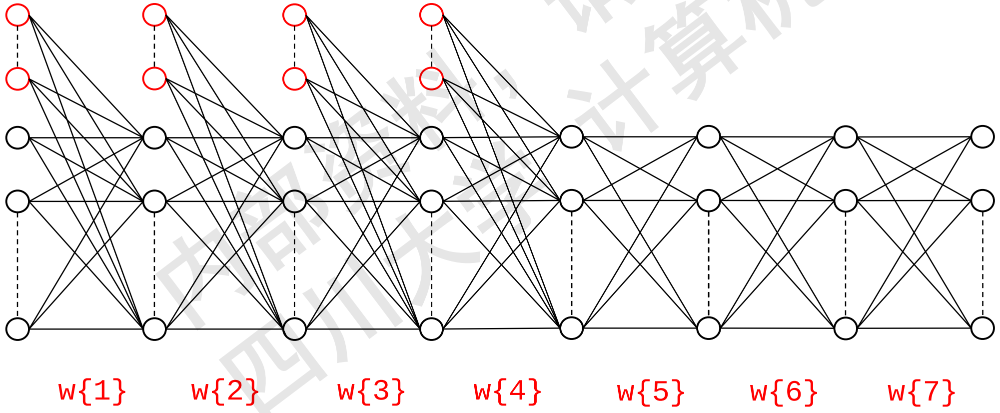
- Training=71.02%
- Testing=69.25%

内部资料，请勿传播！！
四川大学计算机学院 章毅教授

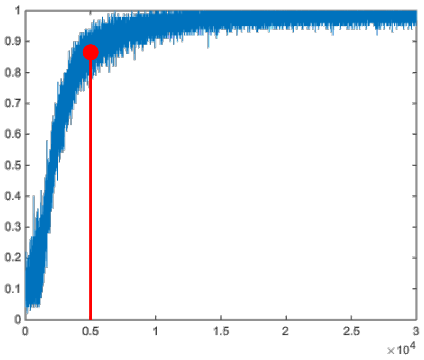
Results: Number of Layers



名称	值
layer_size	8x2 double
w	1x7 cell



8 layers



Accuracy
Training=98.65%
Testing=95.10%

请勿传播！！！ 章毅教授
内部资料 四川大学计算机学院

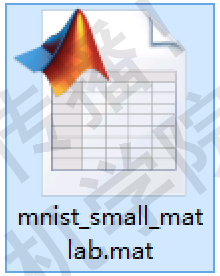
Outline

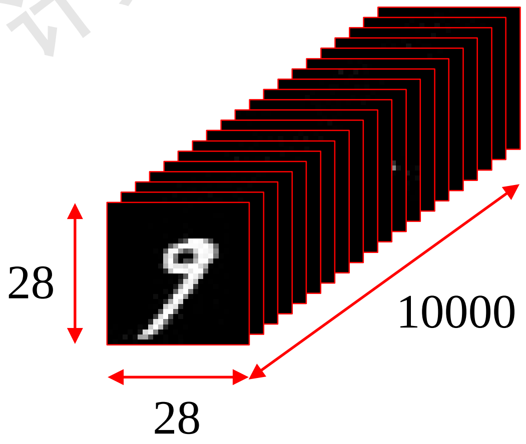
- Brief Review of Backpropagation Algorithm
- An Illustrating Example
- Experiments
- Assignment

内部资料，请勿传播！！
四川大学 计算机学院 章毅教授

Assignment

Implement the handwritten digits recognition by MATLAB using only one layer of external input.





内部资料，请勿传播！！
四川大学 计算机学院 章毅教授

Thanks

内部资料，请勿传播！！！
四川大学 计算机学院 章毅教授