



信阳师范学院  
数学与统计学院  
SCHOOL OF MATHEMATICS AND STATISTICS

# 第14章 神经网络与深度学习



讲授人：牛言涛



日期：2020年5月18日

# 目录

## CONTENTS

- A 机器学习简介
- B 单层神经网络
- C 多层神经网络
- D 神经网络及其分类
- E 深度学习
- F 卷积神经网络

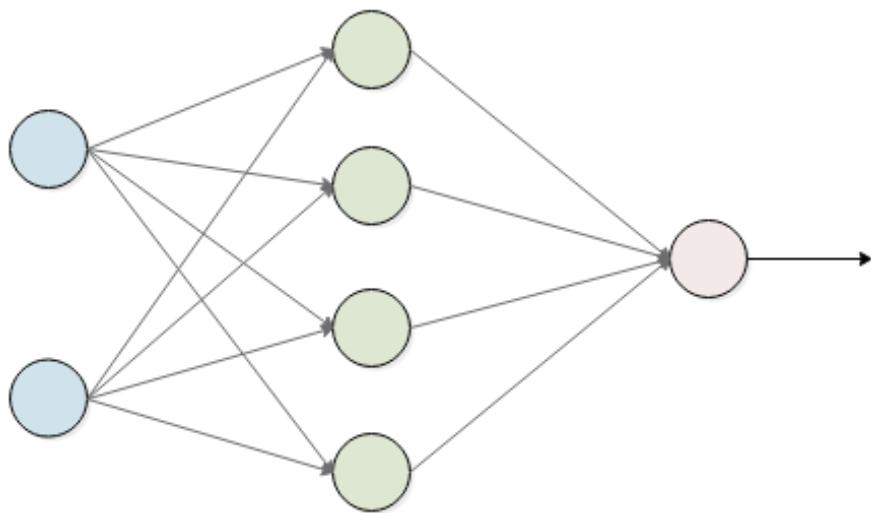


- 监督学习的主要应用是分类和回归。分类被用来确定数据所归属的类别，垃圾邮件过滤和文字识别是一些具有代表性的分类问题。回归被用来从给定的数据中预测未知值。
- 神经网络很少用于回归，非性能不好，而是多数回归问题使用简单的模型都能解决。
- 神经网络分类，输出层的结构通常取决于数据被分为多少类。所分的类数只影响输出层节点个数，不影响隐藏层节点个数。当分为更多的类时，与分成两类时所选择的节点个数与激活函数不同。

# 1. 二分类

- 前面两个数字分别表示和的坐标，后面的符号代表数据的类别。

坐标	类别
(5,7)	$\triangle$
(9,8)	●
$\vdots$	$\vdots$
(6,5)	●



神经网络输出是0~1之间的数值，需把符号 $\triangle$ 、 $\bullet$ 转换为数值，调用Sigmoid函数的最大值和最小值表示：

Class  $\triangle \rightarrow 1$

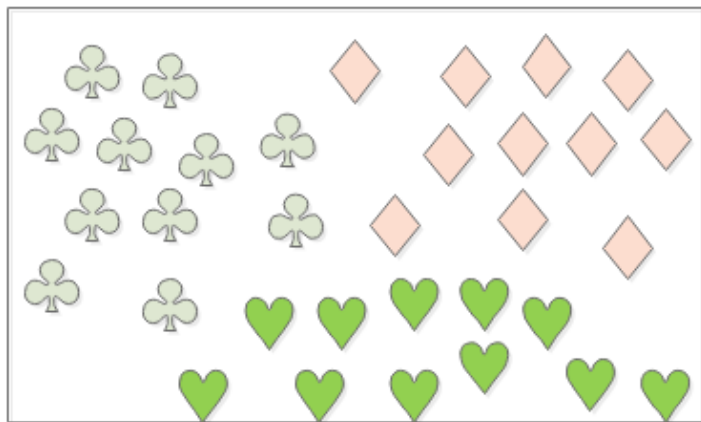
Class  $\bullet \rightarrow 0$

## 二分类网络训练步骤

- ① 二分类神经网络输出层只有一个节点，采用Sigmoid激活函数；
- ② 用Sigmoid函数的最大值和最小值将类别转化为数值；
- ③ 用适当的值初始化网络权重；
- ④ 计算节点的增量 $\delta$ ，即正常输出与模型输出之差 $E - D - Y, \delta = E$ ；
- ⑤ 反向传播，计算下一个隐藏层（左侧）节点的增量，即 $E^{(k)} = W^T \delta, \delta^{(k)} = \varphi'(V^{(k)})E^{(k)}$
- ⑥ 重复第⑤步，直至计算到输入层右侧的那一隐藏层为止；
- ⑦ 根据学习规则调整权重值，即 $\Delta w_{ij} = \alpha \delta_i x_j, w_{ij} \leftarrow w_{ij} + \Delta w_{ij}$
- ⑧ 对所有的训练数据节点重复第④~⑦步；
- ⑨ 重复第④~⑧步，直到神经网络得到了合适的训练。

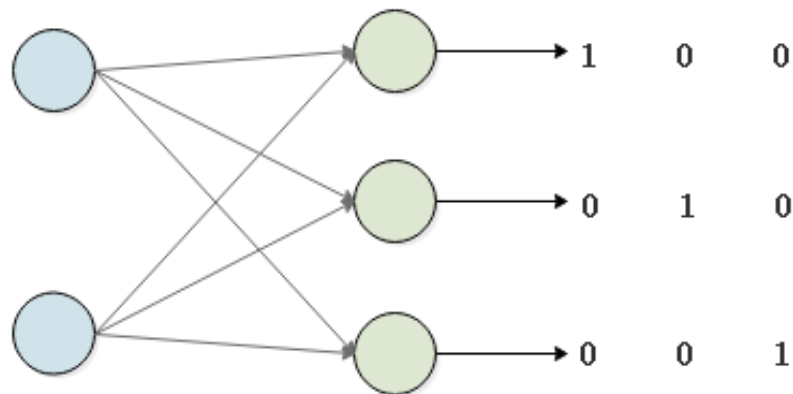
## 2. 多分类

- 多分类数据点



坐标	类别
(5,7)	♣
(9,8)	♦
⋮	⋮
(6,5)	♥

- 用于多分类的无隐藏层神经网络



每个输出节点被映射成一个类别向量，而且在相应的节点上生成“1”


Class ♣  $\rightarrow [1 \ 0 \ 0]$

Class ♦  $\rightarrow [0 \ 1 \ 0]$

Class ♥  $\rightarrow [0 \ 0 \ 1]$

## 2. 多分类

- one-hot编码（独热编码）或1-of N编码。

坐标	类别		坐标	类别
(5,7)	♣		(5,7)	1 0 0
(9,8)	♦		(9,8)	0 1 0
⋮	⋮		⋮	⋮
(6,5)	♥		(6,5)	0 0 1

- 多分类器大多采用Softmax函数作为输出节点的激活函数，正确地诠释神经网络多元分类的输出结果需要考虑所有节点输出的相对大小：

$$y_i = \varphi(v_i) = \frac{e^{v_i}}{\sum_{k=1}^M e^{v_k}}, \quad \sum_{k=1}^M \varphi(v_k) = 1$$

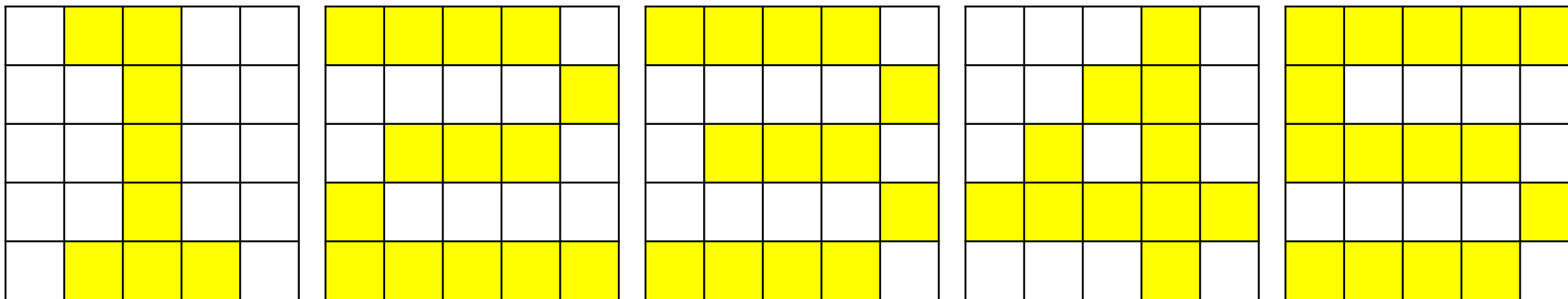
- ① 多分类网络输出层节点个数与类别数一致，采用Softmax激活函数；
- ② 通过one-hot编码方式将类别名称转换为数值向量；
- ③ 用适当的值初始化网络权重；
- ④ 计算节点的增量 $\delta$ ，即正常输出与模型输出之差 $E - D - Y, \delta = E$ ；
- ⑤ 反向传播，计算下一个隐藏层（左侧）节点的增量，即 $E^{(k)} = W^T \delta, \delta^{(k)} = \varphi'(V^{(k)})E^{(k)}$
- ⑥ 重复第⑤步，直至计算到输入层右侧的那一隐藏层为止；
- ⑦ 根据学习规则调整权重值，即 $\Delta w_{ij} = \alpha \delta_i x_j, w_{ij} \leftarrow w_{ij} + \Delta w_{ij}$
- ⑧ 对所有的训练数据节点重复第④~⑦步；
- ⑨ 重复第④~⑧步，直到神经网络得到了合适的训练。



### 3. 多分类神经网络示例



输入数据为5个 $5 \times 5$ 矩阵，分别表示1,2,3,4,5。网络结构为输入节点25个，输出节点5个，隐含节点25个。



## 4. 多分类神经网络MATLAB实现



```
NNMultiClass_image.m  x  +
1  function [Wh, Wo, Y, Error] = NNMulticlass_image(X, labels, nh, alpha, epochs)
2  % NNMulticlass_image函数实现多分类BP神经网络，单隐藏层，针对图片类型
3  % 输入X是一个三维矩阵，其中第三维表示样本数
4  % 输入labels是正确类别标签，nh是隐藏层神经元数
5
6  %% one-hot编码
7  [m, n, num] = size(X);
8  fn = length(unique(labels)); %类别数
9  D = zeros(num, fn);
10 for i = 1:num
11     D(i, labels(i)) = 1;
12 end
13
14 %% 权重的初始化及动量初始化
15 Wh = 2*rand(nh, m*n) - 1; %输入层到隐藏层权重初始化
16 Wo = 2*rand(fn, nh) - 1; %隐藏层到输出层权重初始化
17
18 %% 网络训练
19 Error = zeros(1, epochs);
20 for i = 1:epochs
21     y = zeros(fn, num);
22     for k = 1:num
23         %% 信号向前传播
24         x = reshape(X(:, :, k), m*n, 1); %k表示第k副图，25*1向量
25         d = D(k, :)'; %取正确输出，构成一列
26         v1 = Wh*x; %输入层到隐藏层
27         y1 = Sigmoid(v1); %激活函数
28         v = Wo*y1; %隐藏层到输出层
29         y(:, k) = Softmax(v); %激活函数
```

```
30
31 %% 误差向后传播
32 e = d - y(:, k); %误差
33 delta = e;
34 e1 = Wo'*delta;
35 delta1 = y1.*(1-y1).*e1;
36 dWh = alpha*delta1*x';
37 Wh = Wh + dWh;
38 dWo = alpha*delta*y1';
39 Wo = Wo + dWo;
40
41 end
42 Error(i) = mean(mean((D' - y).^2));
43 if mod(i, 10) == 0
44     fprintf('第【%d】次训练.....\n', i);
45 end
46
47 end
48
49 %% 模型最终输出
50 Y = zeros(fn, num);
51 for k = 1:num
52     x = reshape(X(:, :, k), m*n, 1);
53     v1 = Wh*x;
54     y1 = Sigmoid(v1);
55     v = Wo*y1;
56     Y(:, k) = Softmax(v);
57 end
58
59 %% 统计识别正确率
60 s2 = size(Y, 2);
61 hitNum = 0;
```

## 4. 多分类神经网络MATLAB实现



```
59 — for i = 1:s2
60 —     [~, Index] = max(Y(:, i));
61 —     if Index == labels(i)
62 —         hitNum = hitNum + 1;
63 —     end
64 — end
65 — fprintf(' 识别正确率为: %3.3f。 \n', 100 * hitNum / s2 )
66 —
67 — %% 误差损失曲线
68 — plot(Error(1:5:end), 'r.-', 'LineWidth', 1)
69 — grid on
70 — xlabel(' epochs/5'); ylabel(' Loss')
71 — title(strcat(' 多分类BP神经网络误差损失曲线 Loss = ', num2str(Error(end))))
72 —
73 — %% 激活函数
74 — function y = Sigmoid(x)
75 —     y = 1./(1+exp(-x));
76 — end
77 — %% 激活函数
78 — function y = Softmax(x)
79 —     ex = exp(x);
80 —     y = ex./sum(ex);
81 — end
82 — end
```

$X(:,1) = [0 \ 1 \ 1 \ 0 \ 0; 0 \ 0 \ 1 \ 0 \ 0; 0 \ 0 \ 1 \ 0 \ 0; 0 \ 0 \ 1 \ 0 \ 0; 0 \ 1 \ 1 \ 1 \ 0];$  %数字1

$X(:,2) = [1 \ 1 \ 1 \ 1 \ 0; 0 \ 0 \ 0 \ 0 \ 1; 0 \ 1 \ 1 \ 1 \ 0; 1 \ 0 \ 0 \ 0 \ 0; 1 \ 1 \ 1 \ 1 \ 1];$  %数字2

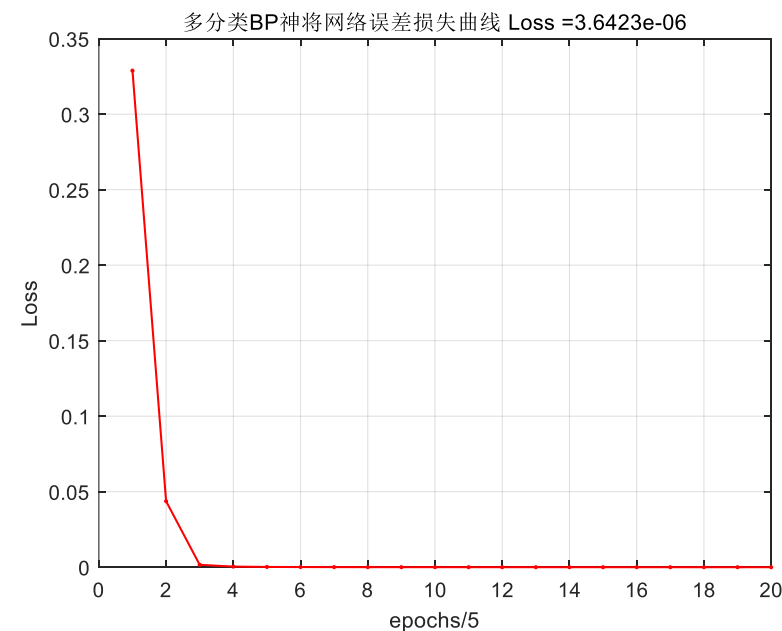
$X(:,3) = [1 \ 1 \ 1 \ 1 \ 0; 0 \ 0 \ 0 \ 0 \ 1; 0 \ 1 \ 1 \ 1 \ 0; 0 \ 0 \ 0 \ 0 \ 1; 1 \ 1 \ 1 \ 1 \ 0];$  %数字3

$X(:,4) = [0 \ 0 \ 0 \ 1 \ 0; 0 \ 0 \ 1 \ 1 \ 0; 0 \ 1 \ 0 \ 1 \ 0; 1 \ 1 \ 1 \ 1 \ 1; 0 \ 0 \ 0 \ 1 \ 0];$  %数字4

$X(:,5) = [1 \ 1 \ 1 \ 1 \ 1; 1 \ 0 \ 0 \ 0 \ 0; 1 \ 1 \ 1 \ 1 \ 0; 0 \ 0 \ 0 \ 0 \ 1; 1 \ 1 \ 1 \ 1 \ 0];$  %数字5

labels = [1 2 3 4 5]';

[Wh,Wo,Y] = NNMultiClass\_image(X,labels,25,0.9,100);



识别正确率为: 100.000。

# 多分类神经网络示例

真实数据未必与训练数据相符，用微微污染的数据简单检验一下上面所构建的神经网络，训练100次。

$X(:, :, 1) = [0\ 0\ 1\ 1\ 0; 0\ 0\ 1\ 1\ 0; 0\ 1\ 0\ 1\ 0; 0\ 0\ 0\ 1\ 0; 0\ 1\ 1\ 1\ 0];$

$X(:, :, 2) = [1\ 1\ 1\ 1\ 0; 0\ 0\ 0\ 0\ 1; 0\ 1\ 1\ 1\ 0; 1\ 0\ 0\ 0\ 1; 1\ 1\ 1\ 1\ 1];$

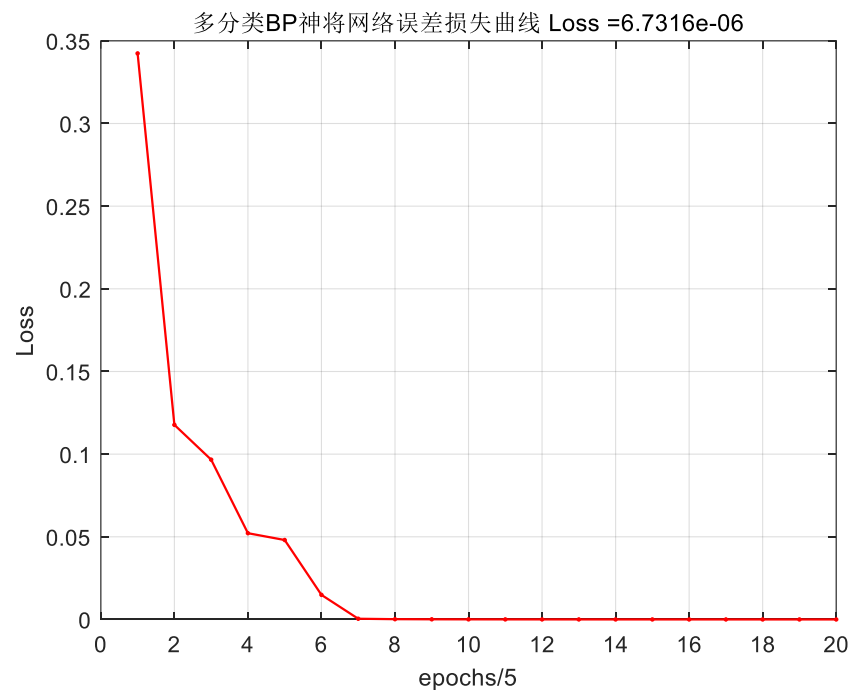
$X(:, :, 3) = [1\ 1\ 1\ 1\ 0; 0\ 0\ 0\ 0\ 1; 0\ 1\ 1\ 1\ 0; 1\ 0\ 0\ 0\ 1; 1\ 1\ 1\ 1\ 0];$

$X(:, :, 4) = [0\ 1\ 1\ 1\ 0; 0\ 1\ 0\ 0\ 0; 0\ 1\ 1\ 1\ 0; 0\ 0\ 0\ 1\ 0; 0\ 1\ 1\ 1\ 0];$

$X(:, :, 5) = [0\ 1\ 1\ 1\ 1; 0\ 1\ 0\ 0\ 0; 0\ 1\ 1\ 1\ 0; 0\ 0\ 0\ 1\ 0; 1\ 1\ 1\ 1\ 0];$

Y =

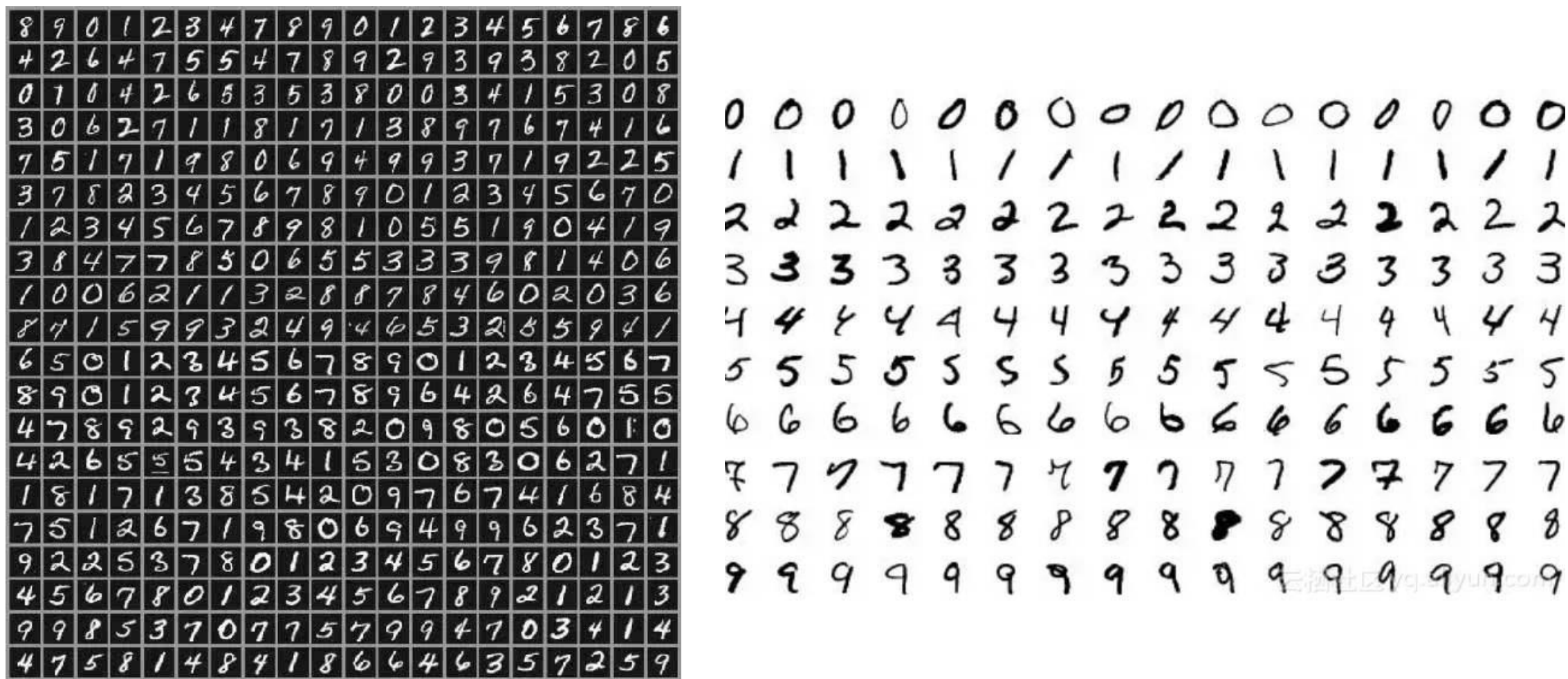
0.9977	0.0000	0.0002	0.0013	0.0004
0.0000	0.9953	0.0044	0.0002	0.0001
0.0005	0.0046	0.9946	0.0006	0.0005
0.0015	0.0001	0.0004	0.9954	0.0028
0.0003	0.0000	0.0004	0.0026	0.9962



- 99.77%的概率认为是1
- 99.53%的概率认为是2
- 99.46%的概率认为是3
- 99.54%的概率认为是4
- 99.62%的概率认为是5

# 识别MNIST手写数字

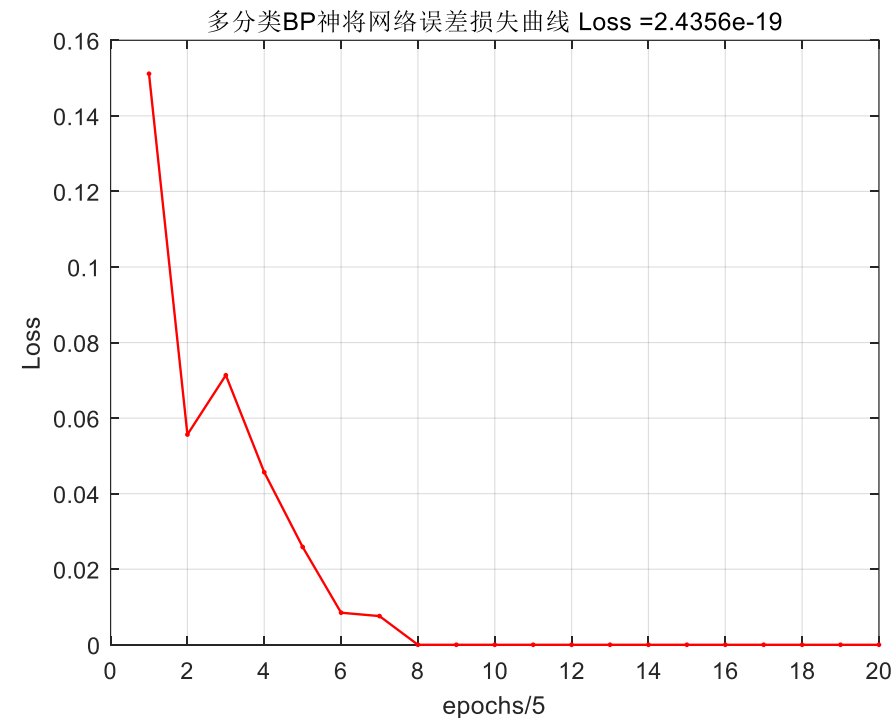
MNIST数据集有60000幅图像，每幅都是28\*28黑白图像。这里由于计算能力的要求，选取前100个数字图片用于训练，且训练100次。



# 识别MNIST手写数字

```
Images = loadMNISTImages('MNIST\t10k-images.idx3-ubyte');  
Images = reshape(Images, 28, 28, []);  
Labels = loadMNISTLabels('MNIST\t10k-labels.idx1-ubyte');  
Labels(Labels == 0) = 10; %类别为0的是10  
% 由于图片过多，且训练时间较长，这里选择前100幅图像  
X = Images(:, :, 1:100);  
labels = Labels(1:100);  
[Wh, Wo, Y, Error] = NNMultiClass_image(X, labels, 600, 0.9, 100);  
第【10】次训练.....  
第【20】次训练.....  
.....  
第【80】次训练.....  
第【90】次训练.....  
第【100】次训练.....  
识别正确率为：100.000。
```

```
for i = 1:100  
    x = X(:, :, i);  
    subplot(5, 20, i)  
    imshow(x)  
end
```



对手写数字回代预测正确率为100%，不代表模型具有很强的泛化能力。一是选择的样本数量比较少，二是训练次数不一定够，仅对当前样本训练100次。

# 识别MNIST手写数字

7 2 1 0 4 1 4 9 5 9 0 6 9 0 1 5 9 7 3 4

9 6 6 5 4 0 7 4 0 1 3 1 3 4 7 2 7 1 2 1

1 7 4 2 3 5 1 2 4 4 6 3 5 5 6 0 4 1 9 5

7 8 9 3 7 4 6 4 3 0 7 0 2 9 1 7 3 2 9 7

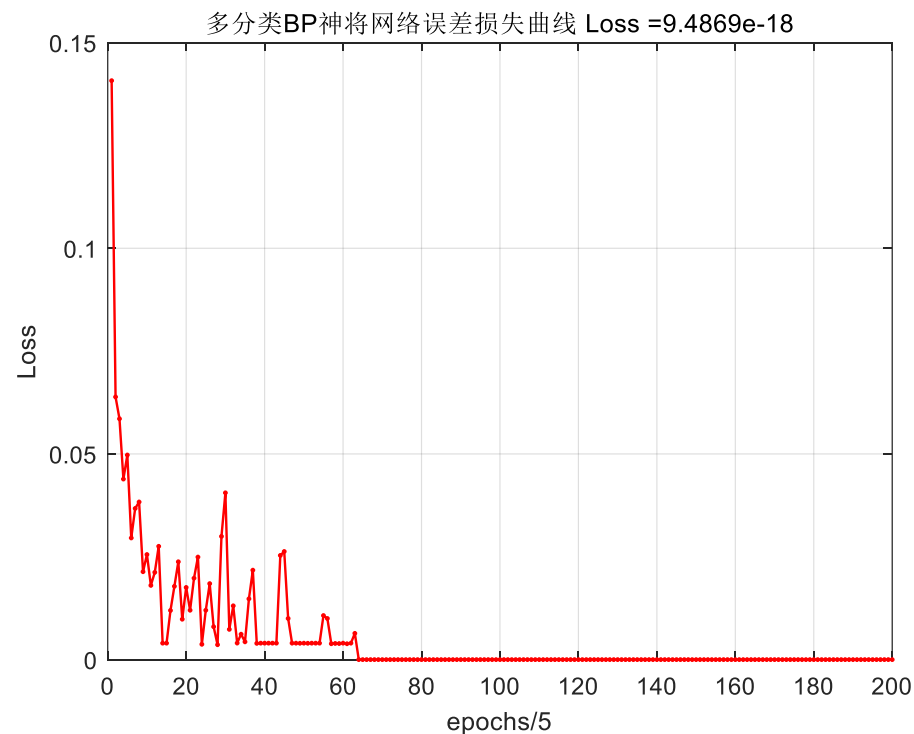
7 6 2 7 8 4 7 3 6 1 3 6 9 3 1 4 1 7 6 9



# 识别MNIST手写数字



```
MNIST_test.m
1 testX = Images(:, :, 101:150);
2 testlabels = Labels(101:150);
3 [m, n, num] = size(testX);
4 fn = length(unique(testlabels)); %类别数
5 %% 模型最终输出
6 Y = zeros(fn, num);
7 for k = 1:num
8     x = reshape(testX(:, :, k), m*n, 1);
9     v1 = Wh*x;
10    y1 = Sigmoid(v1);
11    v = Wo*y1;
12    Y(:, k) = Softmax(v);
13 end
14 %% 统计识别正确率
15 s2 = size(Y, 2);
16 hitNum = 0;
17 for i = 1:s2
18     [~, Index] = max(Y(:, i));
19     if Index == testlabels(i)
20         hitNum = hitNum + 1;
21     end
22 end
23 fprintf(' 识别正确率为: %3.3f。 \n', 100 * hitNum / s2 )
24 %% 激活函数
25 function y = Sigmoid(x)
26     y = 1. / (1 + exp(-x));
27 end
28 %% 激活函数
29 function y = Softmax(x)
30     ex = exp(x);
31     y = ex ./ sum(ex);
32 end
```



训练1000次后，测试50个样本，正确率仅为68%，不高。原因可能是：训练的样本量不够；训练次数不够；网络需要优化。



# The CIFAR-10 dataset 分类

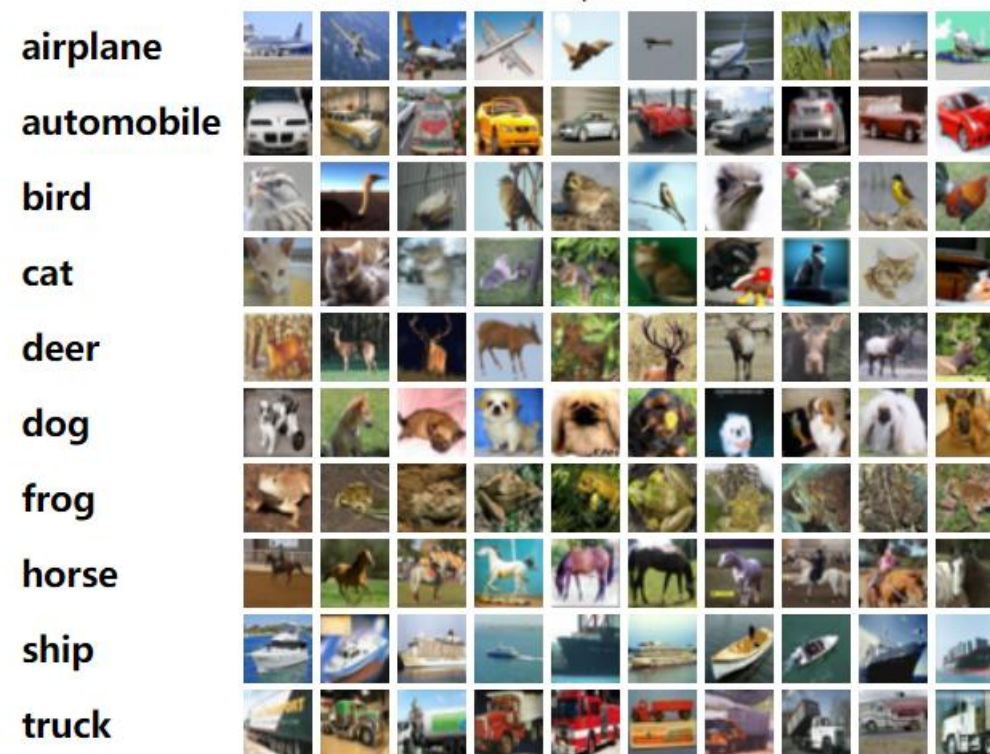


信阳师范学院  
数学与统计学院  
SCHOOL OF MATHEMATICS AND STATISTICS

CIFAR-10数据集由10个类别的60000 32x32彩色图像组成，每个类别有6000张图像。有50000个训练图像和10000个测试图像。数据集分为五个训练集和一个测试集，每个集有10000个图像。测试集包含来自每个类的正好1000个随机选择的图像。训练集的每个类别5000个图像。图像类别如下：

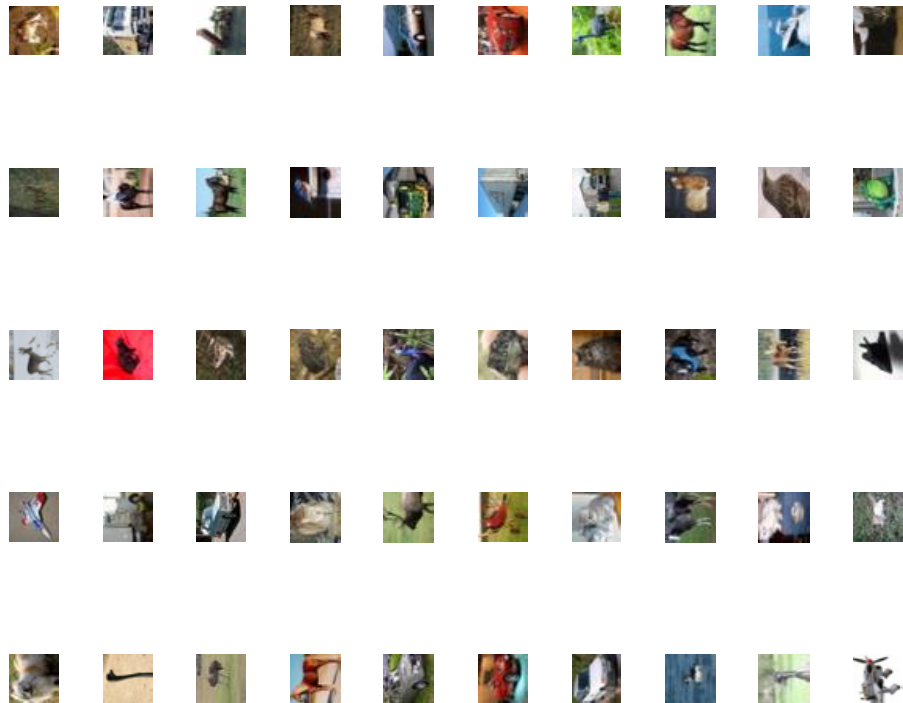
<https://www.cs.toronto.edu/~kriz/cifar.html>

- 由于计算能力的问题，这里选取50个样本用于训练和测试，训练500次，隐藏层节点500个。
- 首先对图像进行处理，数据集每一行代表一个图像，需要进行重塑为32\*32\*3的彩色图像，然后进行灰度处理和图像特征的提取和背景的删除，这会损失图像的一部分特征，导致网络训练的失败或识别率比较低。



# The CIFAR-10 dataset 分类

```
for i = 1:50  
    im = reshape(data(i,:),32,32,[]);  
    subplot(5,10,i); imshow(im)  
end
```



```
for i = 1:50  
    x = imgs(:,:,i);  
    subplot(5,10,i); imshow(x)  
end
```



# The CIFAR-10 dataset 分类

```
load data_batch_1.mat
```

```
for k = 1:50
```

```
    im = reshape(data(k,:),32,32,[]); %提取一行数据并进行重塑
```

```
    img = rgb2gray(im);
```

```
    c = cat(3,img,img,img); %三张二维灰度图a叠加在一起，形成彩图c
```

```
    c = c./255; %除以255，得到的彩图c。矩阵里元素的值都是0、1。
```

```
    e=c.*img; %此时乘以彩图b，（三维矩阵乘以三维矩阵）。
```

```
    %由于0乘以任何数都是0，也就是黑色，此时把背景给屏蔽了
```

```
    %蝴蝶的话因为是用1乘的，其值不变，所以还是彩色
```

```
    imgs(:,:,k) = rgb2gray(e); %再次转化为灰色
```

```
end
```

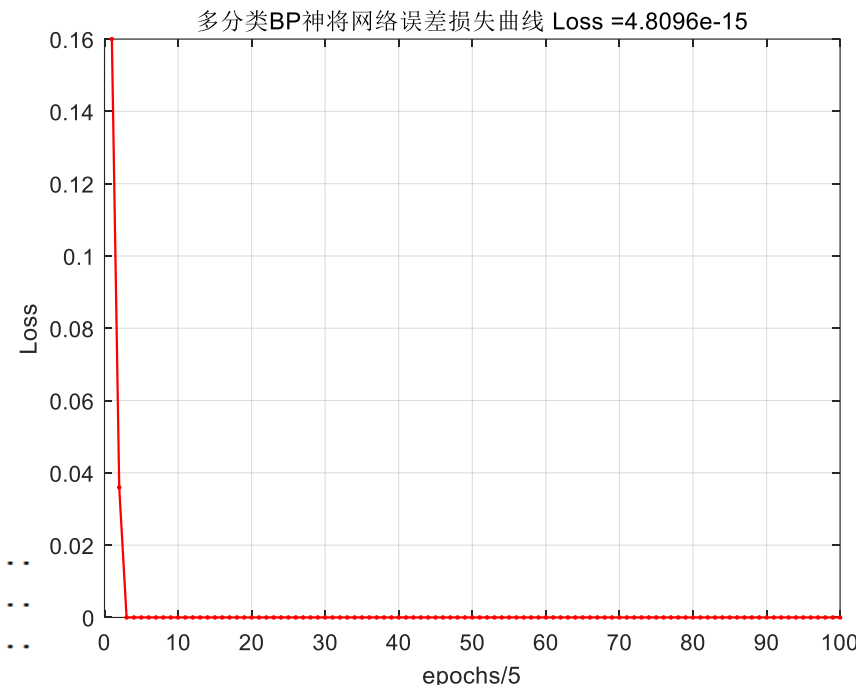
```
labels(labels == 0) = 10;
```

```
X = double(imgs);
```

```
lab = double(labels(1:50));
```

```
[Wh,Wo,Y>Error] = NNMultiClass_image(X,lab,500,0.9,500);
```

```
第【400】次训练.....  
第【410】次训练.....  
第【420】次训练.....  
第【430】次训练.....  
第【440】次训练.....  
第【450】次训练.....  
第【460】次训练.....  
第【470】次训练.....  
第【480】次训练.....  
第【490】次训练.....  
第【500】次训练.....  
识别正确率为：100.000。
```



训练500次，正确率为100%，但是模型容易训练失败，需要优化网络，增加稳定性和学习速度，或者采用特定的网络，如CNN。



---

# 感谢聆听

---