

# 中国地质大学



题    目：    基于 C++ 与 MNIST 数据集训练 BP 神经网络

姓    名：                    常文瀚

院    系：                    计算机学院

班    级：                    191181

学    号：                    20181001095

2020 年 12 月 1 日

目录

一、题目背景.....1

二、题目需求.....1

三、题目原理.....1

    1.神经网络.....1

    2 神经元模型.....1

    3 误差逆传播算法.....2

四、处理过程.....3

五、使用数据.....3

六、算法实现.....3

七、运行结果.....8

八、总结.....9

九、参考文献.....10

# 一、 题目背景

我们在现实生活中，会在多个场合，都能运用到字体的识别技术。如道路监控摄像头，拍下过往车辆的图片，需要识别车辆的车牌号；对身份证号码的手写或者影印版，需要提取录入到数据库等等。如果采用传统的人工识别方法，不仅效率低下，而且在遇到大量数据的情况下，人工识别的错误率也会直线上升。根据这个问题背景，本实验的目标，就是通过 BP 神经网络，识别出手写的数字 0-9 。

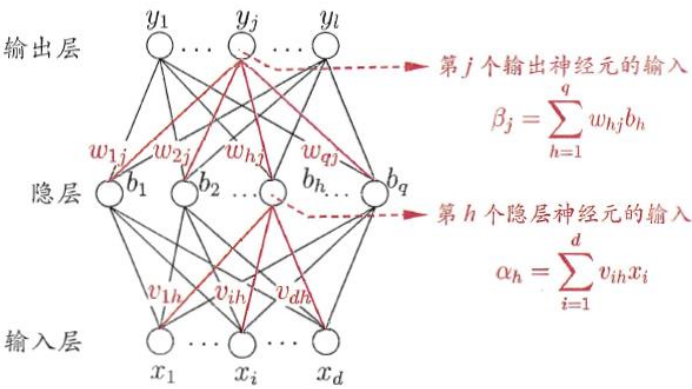
# 二、 题目需求

基于 C++编程语言，实现 BP 神经网络，并使用 MNIST 数据集训练搭建的神经网络，最后利用测试集对最后训练的模型进行精确度测试，同时可以将结果显示在控制台中。

# 三、 题目原理

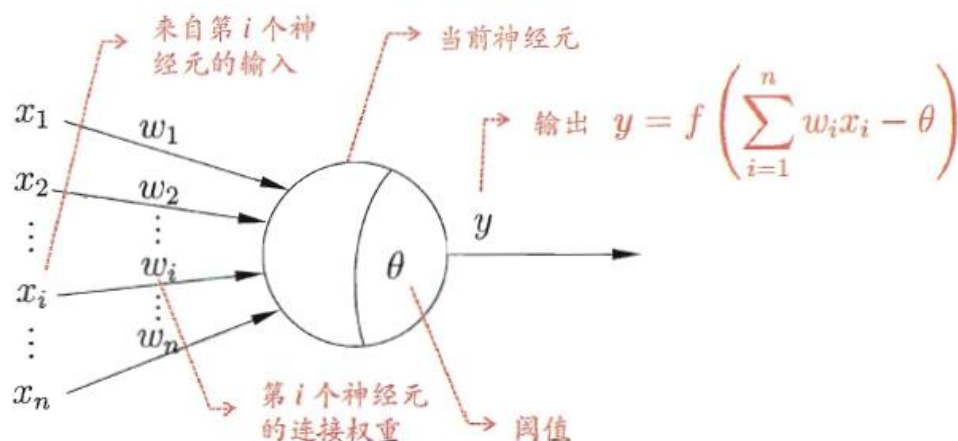
## 1. 神经网络

神经网络(neural networks)方面的研究很早就已出现,今天“神经网络”已是一个相当大的、多学科交叉的学科领域.各相关学科对神经网络的定义多种多样,本书采用目前使用得最广泛的一种，即“神经网络是由具有适应性的简单单元组成的广泛并行互连的网络,它的组织能够模拟生物神经系统对真实世界物体所做出的交互反应” [Kohonen, 1988].我们在机器学习中谈论神经网络时指的是“神经网络学习”,或者说，是机器学习与神经网络这两个学科领域的交叉部分，下图为本实验所参考的 BP 神经网络模型：



## 2. 神经元模型

神经网络中最基本的成分是神经元(neuron)模型,即上述定义中的“简单单元”在生物神经网络中,每个神经元与其他神经元相连,当它“兴奋”时,就会向相连的神经元发送化学物质,从而改变这些神经元内的电位;如果某神经元的电位超过了一个“阈值”,那么它就会被激活,即“兴奋”起来,向其他神经元发送化学物质。



### 3. 误差逆传播算法

BP 神经网络是一种具有三层或者三层以上的多层神经网络，每一层都由若干个神经元组成，它的左、右各层之间各个神经元实现全连接，即靠近输入层的每一个神经元与右层的每个神经元都由连接，而上下各神经元之间无连接。

BP 神经网络按有监督学习方式训练，当一对学习模式提供给神经网络后，其神经元的激活值将从输入层经各隐含层向输出层传播，在输出层的各神经元输出对应输入样例的识别结果，这一结果往往是一组概率，程序会将概率最大对应结果作为最终结果。

在这之后，根据希望的结果与实际的结果求出误差，再通过误差求取连接权重、神经元激活阈值等可以被更新的参数，把他们从输出层传送到隐含层，最后回到输入层（从右到左）逐层修正各连接权和神经元激活阈值。由于这种修正过程是从输出到输入逐层进行的，所以称它为“误差逆传播算法”。随着这种误差逆传播训练的不断修正，网络对输入样例识别的正确率也将不断提高。

---

**输入：**训练集  $D = \{(\mathbf{x}_k, \mathbf{y}_k)\}_{k=1}^m$ ;  
学习率  $\eta$ .

**过程：**

- 1: 在(0, 1)范围内随机初始化网络中所有连接权和阈值
- 2: **repeat**
- 3:   **for all**  $(\mathbf{x}_k, \mathbf{y}_k) \in D$  **do**
- 4:     根据当前参数和式(5.3) 计算当前样本的输出  $\hat{\mathbf{y}}_k$ ;
- 5:     根据式(5.10) 计算输出层神经元的梯度项  $g_j$ ;
- 6:     根据式(5.15) 计算隐层神经元的梯度项  $e_h$ ;
- 7:     根据式(5.11)-(5.14) 更新连接权  $w_{hj}$ ,  $v_{ih}$  与阈值  $\theta_j$ ,  $\gamma_h$
- 8:   **end for**
- 9: **until** 达到停止条件

**输出：**连接权与阈值确定的多层前馈神经网络

---

## 四、 处理过程

BP 神经网络是一种按误差逆传播算法训练的多层前馈网络，BP 网络能学习和存贮大量的输入-输出模式映射关系，而无需事前揭示描述这种映射关系的数学方程。

它的学习规则是使用最速下降法，通过反向传播来不断调整网络的权值和阈值，使网络的误差最小。BP 神经网络模型拓扑结构包括输入层、隐藏层和输出层。

针对此实验，我们对每一张手写图片，先把它处理成一个  $28 * 28$  的 01 矩阵，其中 1 代表数字的笔画着色部分，0 则代表空白。然后我们把该矩阵，扁平成一个 784 维的输入向量，输入到输入层。

经过隐藏层（此次实验隐藏层结点数取 100），到达输出层时，是一个 10 维的输出向量，每一位分别对应数字的 0-9。

通过比较输出层的实际输出与期望输出，进行反向反馈调节，并循环重复上述步骤。

## 五、 使用数据

1. `train-images.idx3-ubyte`: 跳过开头 16 个字节，每 784 个字节代表一张图片，总共有 60000 张。其中每一个字节代表  $28 * 28$  中的一个像素点，取值范围为  $0 \sim 256$ 。（本次实验中，像素点值大于 128，输入层的 01 矩阵对应维取 1；小于 128，01 矩阵对应维取 0）。
2. `train-labels.idx1-ubyte`: 跳过开头 8 个字节，每一个字节，为 1）中对应图片的期望输出，取值范围 0-9，数量也是 60000。
3. `t10k-images.idx3-ubyte`: 与 `train-images.idx3-ubyte` 数据格式等规格相同，但图片数量为 10000 张
4. `t10k-labels.idx1-ubyte`: 与 `train-labels.idx1-ubyte` 数据格式等规格相同，但数量为 10000

## 六、 算法实现

1. 激活函数

$$f(x) = 1 / (1 + e^{-x}) ;$$

2. 隐藏层的输出

$$\text{output1}[j] = f(\text{sigma}(\text{input}[i] * \text{weight1}[i][j]) + b1[j])$$

3. 输出层的输出

$$\text{output2}[k] = f(\text{sigma}(\text{output1}[j] * \text{weight2}[j][k]) + b2[k]);$$

4. 输出层的 delta 误差

```
delta2[k] = (output2[k]) * (1.0 - output2[k]) * (output2[k] - target[k]);
```

5. 隐藏层的 delta 误差

```
delta1[j] = (output1[j]) * (1.0 - output1[j]) * sigma(weight2[j][k] *  
delta2[k]);
```

6. 隐藏层反向调整

权值:  $\text{weight1}[i][j] = \text{weight1}[i][j] - \alpha * \text{input}[i] * \text{delta1}[j];$

阈值:  $b1[j] = b1[j] - \alpha * \text{delta1}[j];$

7. 输出层反向调整

权值:  $\text{weight2}[j][k] = \text{weight2}[j][k] - \alpha * \text{output1}[j] * \text{delta2}[k];$

阈值:  $b2[k] = b2[k] - \alpha * \text{delta2}[k];$

8. 训练函数

```
void training(){  
  
    FILE *image_train;  
  
    FILE *image_label;  
  
    image_train = fopen("../tc/train-images.idx3-ubyte", "rb");  
    image_label = fopen("../tc/train-labels.idx1-ubyte", "rb");  
    if (image_train == NULL || image_label == NULL){  
        cout << "can't open the file!" << endl;  
        exit(0);  
    }  
  
    unsigned char image_buf[784];  
    unsigned char label_buf[10];  
  
    int useless[1000];  
    fread(useless, 1, 16, image_train);  
    fread(useless, 1, 8, image_label);  
  
    int cnt = 0;  
  
    cout << "Start training..." << endl;
```

```

//60000 times

while (!feof(image_train) && !feof(image_label)){

    memset(image_buf, 0, 784);

    memset(label_buf, 0, 10);

    fread(image_buf, 1, 784, image_train);
    fread(label_buf, 1, 1, image_label);


    //initialize the input by 28 x 28 (0,1)matrix of the images
    for (int i = 0; i < 784; i++){

        if ((unsigned int)image_buf[i] < 128){

            input[i] = 0;

        }

        else{

            input[i] = 1;

        }

    }


    //initialize the target output

    int target_value = (unsigned int)label_buf[0];

    for (int k = 0; k < third; k++){

        target[k] = 0;

    }

    target[target_value] = 1;


    //get the output and start training

    op1_();

    op2_();

    dt2_();

    dt1_();

```

```

        feedback_second();

        feedback_third();

        cnt++;

        if (cnt % 1000 == 0){

            cout << "training image: " << cnt << endl;

        }

    }

    cout << endl;
}

```

## 9. 测试函数

```

void testing(){

    FILE *image_test;

    FILE *image_test_label;

    image_test = fopen("../tc/t10k-images.idx3-ubyte", "rb");

    image_test_label = fopen("../tc/t10k-labels.idx1-ubyte", "rb");

    if (image_test == NULL || image_test_label == NULL){

        cout << "can't open the file!" << endl;

        exit(0);

    }

    unsigned char image_buf[784];

    unsigned char label_buf[10];

    int useless[1000];

    fread(useless, 1, 16, image_test);

    fread(useless, 1, 8, image_test_label);

    while (!feof(image_test) && !feof(image_test_label)){

        memset(image_buf, 0, 784);
    }
}

```



```

memset(label_buf, 0, 10);

fread(image_buf, 1, 784, image_test);

fread(label_buf, 1, 1, image_test_label);


//initialize the input by 28 x 28 (0,1)matrix of the images
for (int i = 0; i < 784; i++){
    if ((unsigned int)image_buf[i] < 128){
        input[i] = 0;
    }
    else{
        input[i] = 1;
    }
}

//initialize the target output
for (int k = 0; k < third; k++){
    target[k] = 0;
}

int target_value = (unsigned int)label_buf[0];
target[target_value] = 1;


//get the ouput and compare with the targe

op1_();

op2_();


double max_value = -99999;
int max_index = 0;
for (int k = 0; k < third; k++){
    if (output2[k] > max_value){
        max_value = output2[k];
    }
}

```

```

        max_index = k;
    }
}

if (target[max_index] == 1){
    test_success_count++;
}

test_num++;

if ((int)test_num % 1000 == 0){
    cout << "test num: " << test_num << "    success: " << test_success_count << endl;
}
}

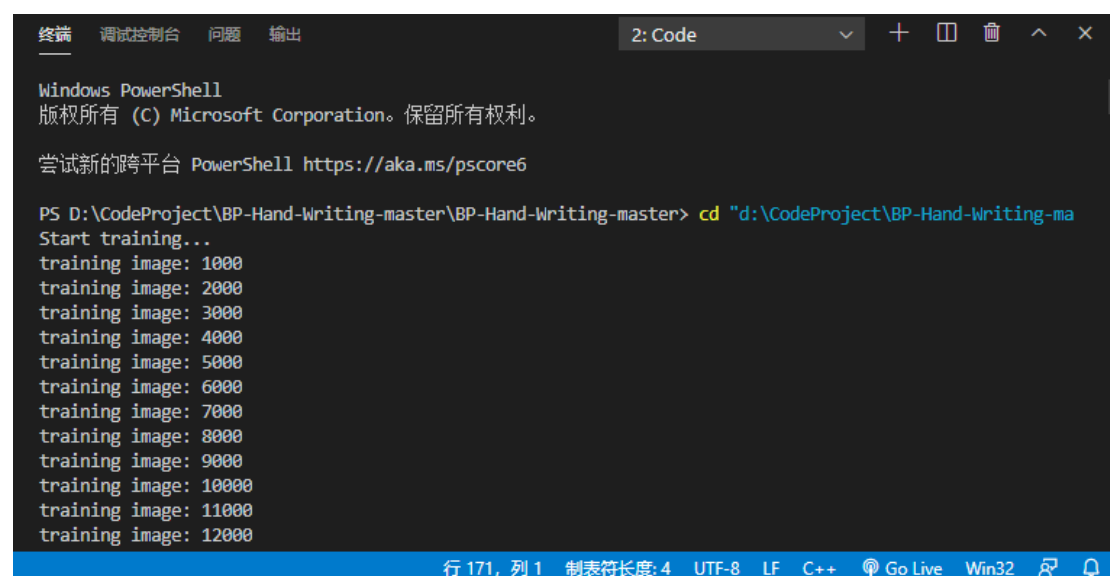
cout << endl;

cout << "The success rate: " << test_success_count / test_num << endl;
}

```

## 七、 运行结果

### 1. 开始训练



```

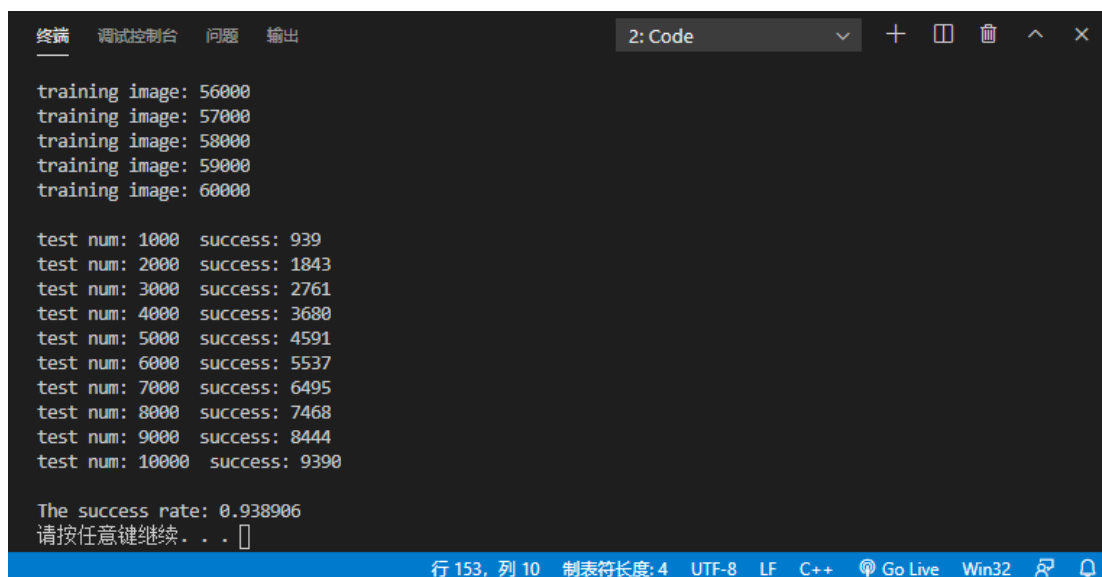
Windows PowerShell
版权所有 (C) Microsoft Corporation。保留所有权利。

尝试新的跨平台 PowerShell https://aka.ms/pscore6

PS D:\CodeProject\BP-Hand-Writing-master\BP-Hand-Writing-master> cd "d:\CodeProject\BP-Hand-Writing-master"
Start training...
training image: 1000
training image: 2000
training image: 3000
training image: 4000
training image: 5000
training image: 6000
training image: 7000
training image: 8000
training image: 9000
training image: 10000
training image: 11000
training image: 12000

```

## 2. 训练结束显示精确度



```
训练  调试控制台  问题  输出  2: Code
training image: 56000
training image: 57000
training image: 58000
training image: 59000
training image: 60000

test num: 1000  success: 939
test num: 2000  success: 1843
test num: 3000  success: 2761
test num: 4000  success: 3680
test num: 5000  success: 4591
test num: 6000  success: 5537
test num: 7000  success: 6495
test num: 8000  success: 7468
test num: 9000  success: 8444
test num: 10000 success: 9390

The success rate: 0.938906
请按任意键继续. . .
```

行 153, 列 10 制表符长度: 4 UTF-8 LF C++ Go Live Win32

## 八、 总结

数字图像是我们生活中接触最多的图像各类,图像作为信息的重要载体在信息传输方面有着声音、文字等信息载体不可替代的作用,作为计算机类专业的大学生更加有必要对数字图像处理技术有一定的掌握,而大多数人对于数字图像的知识却不全面,甚至一些基础知识也很模糊,比如各类繁多的各种图像格式之间的特点,不同的情况该用何种图像格式,还有关于图像的一些基本术语也不甚了解。所以对于数字图像处理这门课大家有着极大兴趣,更加想利用这门课的学习加深自己数字图像处理的理解并提高在数字图像处理方面的能力。

通过一学期的课程学习我们虽说还没有完全掌握数字图像处理技术,但也收获了不少,对于数字图像方面的知识有了深入的了解,更加理解了数字图像的本质,即是一些数字矩阵,但灰度图像和彩色图像的矩阵形式是不同的。对于一些耳熟能详的数字图像相关术语有了明确的认识,比如常见的:像素(衡量图像的大小)、分辨率(衡量图像的清晰程度)、位图(放大后会失真)、矢量图(经过放大不会失真)等大家都能叫上口却知识模糊的名词。也了解图像处理技术中一些常用处理技术的实质,比如锐化处理是使模糊的图像变清晰,增强图像的边缘等细节。而平滑处理是的目的是消除噪声,模糊图像,在提取大目标之前去除小的细节或弥合目标间的缝隙。

当然通过这些课程学习还是远远不够的,课程的内容更偏重于如何通过编程实现实现如何对图像进行一些类似于锐化、边缘提取、模糊、去除噪声等基础功能的实现,这其中涉及很多算法、函数,需要扎实的数学基础和编程基础,并且需要利用大量时间在课下编写代码,在今后的学习或者工作中,我也希望可以将所学到的知识充分的利用在任务中。

## 九、 参考文献

- [1] 《机器学习》 周志华
- [2] 戚德虎, 康继昌. BP 神经网络的设计[D]. , 1998.
- [3] 刘天舒. BP 神经网络的改进研究及应用[D]. 哈尔滨: 东北农业大学, 2011.
- [4] 王钰, 郭其一, 李维刚. 基于改进 BP 神经网络的预测模型及其应用[J]. 计算机测量与控制, 2005, 13(1): 39-42.
- [5] 沈花玉, 王兆霞, 高成耀, 等. BP 神经网络隐含层单元数的确定[J]. 天津理工大学学报, 2008, 24(5): 13-15.
- [6] <http://blog.csdn.net/zouxy09/article/details/45288129>
- [7] <http://blog.csdn.net/wanglp094/article/details/7702907>
- [8] <http://www.cnblogs.com/wentingtu/archive/2012/06/05/2536425.html>
- [9] <http://blog.csdn.net/appleml/article/details/48623303>