

null的专栏


我是一个菜鸟，只希望每天自己能够进步一点点

目录视图

摘要视图

RSS 订阅

个人资料



zhiyong\_will

访问: 229014次

积分: 3361

等级: BLOG > 5

排名: 第6163名

原创: 105篇

转载: 1篇

译文: 0篇

评论: 189条

个人声明

博客的主要内容主要是自己的学习笔记，并结合个人的理解，供各位在学习过程中参考，若有疑问，欢迎提出；若有侵权，请告知博主删除，原创文章转载还请注明出处。

联系我

Email: zhaozhiyong1989@126.com

2016攒课第二期之你听课我买单，快来攒你想听的课！[架构免费公开课报名](#) [【UDN沙龙】Intel大拿分享HTML5时代的跨平台开发解](#)

简单易学的机器学习算法——神经网络之BP神经网络

标签: 神经网络 机器学习 前向反馈 BP神经网络

2014-06-21 11:49 9186人阅读 评论(15) 收藏 举报

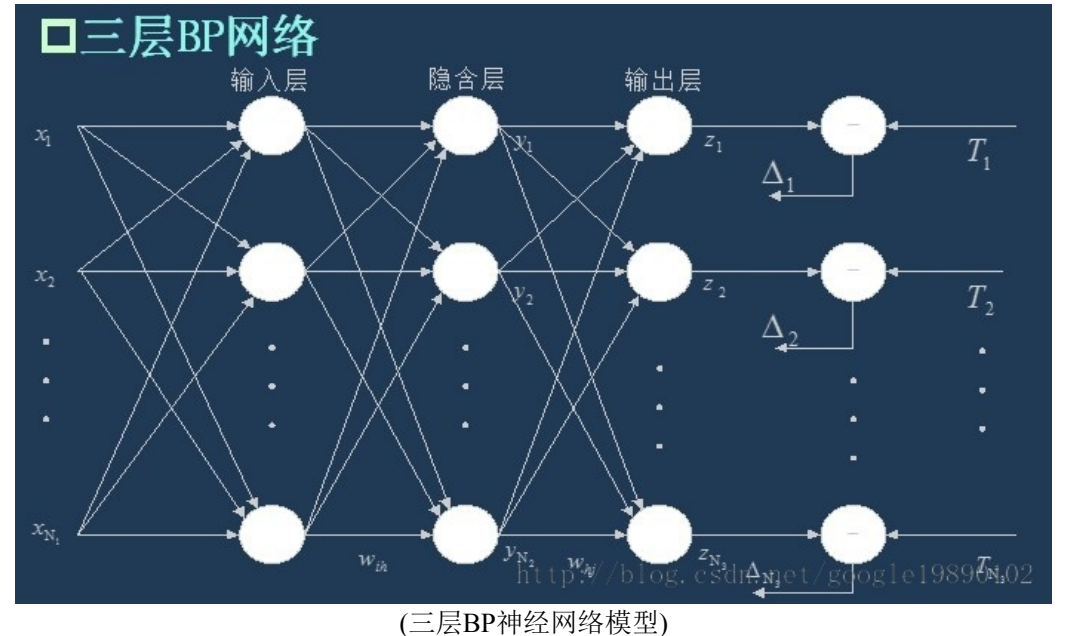
分类: machine learning (41)

版权声明：本文为博主原创文章，未经博主允许不得转载。

目录(?)

一、BP神经网络的概念

BP神经网络是一种多层的前馈神经网络，其主要的特点是：信号是前向传播的，而误差是反向传播的。具体来说，对于如下的只含一个隐层的神经网络模型：



(三层BP神经网络模型)

BP神经网络的过程主要分为两个阶段，第一阶段是信号的前向传播，从输入层经过隐含层，最后到达输出层；第二阶段是误差的反向传播，从输出层到隐含层，最后到输入层，依次调节隐含层到输出层的权重和偏置，输入层到隐含层的权重和偏置。

二、BP神经网络的流程

在知道了BP神经网络的特点后，我们需要依据信号的前向传播和误差的反向传播来构建整个网络。

1、网络的初始化

假设输入层的节点个数为 $n$ ，隐含层的节点个数为 $l$ ，输出层的节点个数为 $m$ 。输入层

X Access Denied

**YOUR  
ATTEMPT  
TO  
ACCESS  
FOLLOWING  
WEBSITE  
HAS  
BEEN  
DENIED:**  
**http://widget.v**  
**language=&wi**

**This  
restriction  
is to  
prevent  
you from  
inadvertently  
bringing  
malicious/offer  
material  
into the  
workplace.**

URL in  
question is News/  
categorized Netwo  
as:  
Exception  
that  
triggered  
the event:  
Your IP  
address has  
been  
recorded  
10.102

博客专栏



机器学习，数据  
挖掘算法  
文章：32篇  
阅读：128166

文章分类

machine learning (42)  
Deep Learning (9)  
optimization algorithm (14)  
论文与材料的学习笔记 (28)  
Computational Advertising (1)  
自然语言处理 (2)  
Recommender System (1)  
每周算法练习 (9)  
设计模式 (1)  
Hadoop (3)  
python (11)  
Java (0)  
Matlab (0)

到隐含层的权重 $\omega_{ij}$ ，隐含层到输出层的权重为 $\omega_{jk}$ ，输入层到隐含层的偏置为 $a_j$ ，隐含层到输出层的偏置为 $b_k$ 。学习速率为 $\eta$ ，激励函数为 $g(x)$ 。其中激励函数为 $g(x)$ 取Sigmoid函数。形式为：

$$g(x) = \frac{1}{1 + e^{-x}}$$

## 2、隐含层的输出

如上面的三层BP网络所示，隐含层的输出 $H_j$ 为

$$H_j = g\left(\sum_{i=1}^n \omega_{ij}x_i + a_j\right)$$

## 3、输出层的输出

$$O_k = \sum_{j=1}^l H_j \omega_{jk} + b_k$$

## 4、误差的计算

我们取误差公式为：

$$E = \frac{1}{2} \sum_{k=1}^m (Y_k - O_k)^2$$

其中 $Y_k$ 为期望输出。我们记 $Y_k - O_k = e_k$ ，则 $E$ 可以表示为

$$E = \frac{1}{2} \sum_{k=1}^m e_k^2$$

以上公式中， $i = 1 \cdots n$ ， $j = 1 \cdots l$ ， $k = 1 \cdots m$ 。

## 5、权值的更新

权值的更新公式为：

$$\begin{cases} \omega_{ij} = \omega_{ij} + \eta H_j (1 - H_j) x_i \sum_{k=1}^m \omega_{jk} e_k \\ \omega_{jk} = \omega_{jk} + \eta H_j e_k \end{cases}$$

这里需要解释一下公式的由来：

这是误差反向传播的过程，我们的目标是使得误差函数达到最小值，即 $\min E$ ，我们使用梯度下降法：

- 隐含层到输出层的权重更新

$$\frac{\partial E}{\partial \omega_{jk}} = \sum_{k=1}^m (Y_k - O_k) \left( -\frac{\partial O_k}{\partial \omega_{jk}} \right) = (Y_k - O_k) (-H_j) = -e_k H_j$$

则权重的更新公式为：

$$\omega_{jk} = \omega_{jk} + \eta H_j e_k$$

- 输入层到隐含层的权重更新

$$\frac{\partial E}{\partial \omega_{ij}} = \frac{\partial E}{\partial H_j} \cdot \frac{\partial H_j}{\partial \omega_{ij}}$$

其中

$$\begin{aligned} \frac{\partial E}{\partial H_j} &= (Y_1 - O_1) \left( -\frac{\partial O_1}{\partial H_j} \right) + \cdots + (Y_m - O_m) \left( -\frac{\partial O_m}{\partial H_j} \right) \\ &= -(Y_1 - O_1) \omega_{jk} - \cdots - (Y_m - O_m) \omega_{jk} \\ &= -\sum_{k=1}^m (Y_k - O_k) \omega_{jk} = -\sum_{k=1}^m \omega_{jk} e_k \end{aligned}$$

$$\begin{aligned} \frac{\partial H_j}{\partial \omega_{ij}} &= \frac{\partial g(\sum_{i=1}^n \omega_{ij} x_i + a_j)}{\partial \omega_{ij}} \\ &= g(\sum_{i=1}^n \omega_{ij} x_i + a_j) \cdot [1 - g(\sum_{i=1}^n \omega_{ij} x_i + a_j)] \cdot \frac{\partial (\sum_{i=1}^n \omega_{ij} x_i + a_j)}{\partial \omega_{ij}} \\ &= H_j (1 - H_j) x_i \end{aligned}$$

则权重的更新公式为：

$$\omega_{ij} = \omega_{ij} + \eta H_j (1 - H_j) x_i \sum_{k=1}^m \omega_{jk} e_k$$

## 6、偏置的更新

偏置的更新公式为：

$$\begin{cases} a_j = a_j + \eta H_j (1 - H_j) \sum_{k=1}^m \omega_{jk} e_k \\ b_k = b_k + \eta e_k \end{cases}$$

Spark (1)  
Linux (1)  
PHP (0)  
Lua (0)  
R (0)

文章存档

2016年04月 (7)  
2016年03月 (3)  
2016年01月 (2)  
2015年12月 (6)  
2015年11月 (7)

展开

阅读排行

简单易学的机器学习算法 (14789)  
简单易学的机器学习算法 (9658)  
简单易学的机器学习算法 (9156)  
简单易学的机器学习算法 (7065)  
简单易学的机器学习算法 (5803)  
简单易学的机器学习算法 (5481)  
简单易学的机器学习算法 (5393)  
优化算法——梯度下降法 (5180)  
优化算法——人工蜂群算 (4865)  
简单易学的机器学习算法 (4802)

评论排行

简单易学的机器学习算法 (32)  
优化算法——人工蜂群算 (18)  
简单易学的机器学习算法 (17)  
简单易学的机器学习算法 (15)  
简单易学的机器学习算法 (13)  
《数学之美》拾遗——潜 (8)  
机器学习中的特征——特 (7)  
简单易学的机器学习算法 (5)  
简单易学的机器学习算法 (5)  
简单易学的机器学习算法 (5)

最新评论

优化算法——人工蜂群算法(ABC qq\_34315659: 你好请问可以提供那个txt文件吗? 谢谢了!  
优化算法——人工蜂群算法(ABC qq\_34315659: 请问可以提供txt文件吗? 谢谢了!  
优化算法——人工蜂群算法(ABC qq\_34315659: 你好请问可以提供那个txt文件吗? 谢谢了!  
简单易学的机器学习算法——协方差 zhanglingxia7: 这个可以用asp.net做吗  
简单易学的机器学习算法——极坐标 sinat\_33960729: 请问一下有没有更多的数据集  
优化算法——截断梯度法(TG) zhiyong\_will: @fanzitao:符合条

• 隐含层到输出层的偏置更新

$$\frac{\partial E}{\partial b_k} = (Y_k - O_k) \left( -\frac{\partial O_k}{\partial b_k} \right) = -e_k$$

则偏置的更新公式为:

$$b_k = b_k + \eta e_k$$

• 输入层到隐含层的偏置更新

$$\frac{\partial E}{\partial a_j} = \frac{\partial E}{\partial H_j} \cdot \frac{\partial H_j}{\partial a_j}$$

其中

$$\begin{aligned} \frac{\partial H_j}{\partial a_j} &= \frac{\partial g(\sum_{i=1}^n \omega_{ij} x_i + a_j)}{\partial a_j} \\ &= g(\sum_{i=1}^n \omega_{ij} x_i + a_j) \cdot [1 - g(\sum_{i=1}^n \omega_{ij} x_i + a_j)] \cdot \frac{\partial (\sum_{i=1}^n \omega_{ij} x_i + a_j)}{\partial a_j} \\ &= H_j (1 - H_j) \end{aligned}$$

$$\begin{aligned} \frac{\partial E}{\partial H_j} &= (Y_1 - O_1) \left( -\frac{\partial O_1}{\partial H_j} \right) + \dots + (Y_m - O_m) \left( -\frac{\partial O_m}{\partial H_j} \right) \\ &= -(Y_1 - O_1) \omega_{jk} - \dots - (Y_m - O_m) \omega_{jk} \\ &= -\sum_{k=1}^m (Y_k - O_k) \omega_{jk} = -\sum_{k=1}^m \omega_{jk} e_k \end{aligned}$$

则偏置的更新公式为:

$$a_k = a_k + \eta H_j (1 - H_j) \sum_{k=1}^m \omega_{jk} e_k$$

7、判断算法迭代是否结束

有很多的方法可以判断算法是否已经收敛，常见的有指定迭代的代数，判断相邻的两次误差之间的差别是否小于指定的值等等。

三、实验的仿真

在本试验中，我们利用BP神经网络处理一个四分类问题，最终的分类结果为:

righttridio =  
  
0.6735    1.0000    0.9550    0.9524

MATLAB代码

主程序

```
[plain]
01. %% BP的主函数
02.
03. % 清空
04. clear all;
05. clc;
06.
07. % 导入数据
08. load data;
09.
10. %从1到2000间随机排序
11. k=rand(1,2000);
12. [m,n]=sort(k);
13.
14. %输入输出数据
15. input=data(:,2:25);
16. output1 =data(:,1);
17.
18. %把输出从1维变成4维
19. for i=1:2000
20.     switch output1(i)
21.         case 1
22.             output(i,:)= [1 0 0 0];
23.         case 2
24.             output(i,:)= [0 1 0 0];
25.         case 3
26.             output(i,:)= [0 0 1 0];
27.         case 4
```

件为1, 不符合条件为0

优化算法——截断梯度法(TG)

fanzitao: 感觉指示函数I的定义那一块是不是写反了

优化算法——拟牛顿法之L-BFGS

zhiyong\_will: @richardzrc:平时自己学习的笔记

优化算法——拟牛顿法之L-BFGS

richardzrc: 这是一门课的作业还是自己研究的

简单易学的机器学习算法——支持

ozone.zz: 啧啧, 谢谢

```

28.         output(i,:)= [0 0 0 1];
29.     end
30. end
31.
32. %随机提取1500个样本为训练样本, 500个样本为预测样本
33. trainCharacter=input(n(1:1600),:);
34. trainOutput=output(n(1:1600),:);
35. testCharacter=input(n(1601:2000),:);
36. testOutput=output(n(1601:2000),:);
37.
38. % 对训练的特征进行归一化
39. [trainInput,inputs]=mapminmax(trainCharacter');
40.
41. %% 参数的初始化
42.
43. % 参数的初始化
44. inputNum = 24;%输入层的节点数
45. hiddenNum = 50;%隐含层的节点数
46. outputNum = 4;%输出层的节点数
47.
48. % 权重和偏置的初始化
49. w1 = rands(inputNum,hiddenNum);
50. b1 = rands(hiddenNum,1);
51. w2 = rands(hiddenNum,outputNum);
52. b2 = rands(outputNum,1);
53.
54. % 学习率
55. yita = 0.1;
56.
57. %% 网络的训练
58. for r = 1:30
59.     E(r) = 0;% 统计误差
60.     for m = 1:1600
61.         % 信息的正向流动
62.         x = trainInput(:,m);
63.         % 隐含层的输出
64.         for j = 1:hiddenNum
65.             hidden(j,:) = w1(:,j)'*x+b1(j,:);
66.             hiddenOutput(j,:) = g(hidden(j,:));
67.         end
68.         % 输出层的输出
69.         outputOutput = w2'*hiddenOutput+b2;
70.
71.         % 计算误差
72.         e = trainOutput(m,:)'-outputOutput;
73.         E(r) = E(r) + sum(abs(e));
74.
75.         % 修改权重和偏置
76.         % 隐含层到输出层的权重和偏置调整
77.         dw2 = hiddenOutput*e';
78.         db2 = e;
79.
80.         % 输入层到隐含层的权重和偏置调整
81.         for j = 1:hiddenNum
82.             partOne(j) = hiddenOutput(j)*(1-hiddenOutput(j));
83.             partTwo(j) = w2(j,:)*e;
84.         end
85.
86.         for i = 1:inputNum
87.             for j = 1:hiddenNum
88.                 dw1(i,j) = partOne(j)*x(i,:)*partTwo(j);
89.                 db1(j,:) = partOne(j)*partTwo(j);
90.             end
91.         end
92.
93.         w1 = w1 + yita*dw1;
94.         w2 = w2 + yita*dw2;
95.         b1 = b1 + yita*db1;
96.         b2 = b2 + yita*db2;
97.     end
98. end
99.

```



```

100. %% 语音特征信号分类
101. testInput=mapminmax('apply',testCharacter',inputsps);
102.
103. for m = 1:400
104.     for j = 1:hiddenNum
105.         hiddenTest(j,:) = w1(:,j)*testInput(:,m)+b1(j,:);
106.         hiddenTestOutput(j,:) = g(hiddenTest(j,:));
107.     end
108.     outputOfTest(:,m) = w2'*hiddenTestOutput+b2;
109. end
110.
111. %% 结果分析
112. %根据网络输出找出数据属于哪类
113. for m=1:400
114.     output_fore(m)=find(outputOfTest(:,m)==max(outputOfTest(:,m)));
115. end
116.
117. %BP网络预测误差
118. error=output_fore-output1(n(1601:2000))';
119.
120. k=zeros(1,4);
121. %找出判断错误的分类属于哪一类
122. for i=1:400
123.     if error(i)~=0
124.         [b,c]=max(testOutput(i,:));
125.         switch c
126.             case 1
127.                 k(1)=k(1)+1;
128.             case 2
129.                 k(2)=k(2)+1;
130.             case 3
131.                 k(3)=k(3)+1;
132.             case 4
133.                 k(4)=k(4)+1;
134.         end
135.     end
136. end
137.
138. %找出每类的个体和
139. kk=zeros(1,4);
140. for i=1:400
141.     [b,c]=max(testOutput(i,:));
142.     switch c
143.         case 1
144.             kk(1)=kk(1)+1;
145.         case 2
146.             kk(2)=kk(2)+1;
147.         case 3
148.             kk(3)=kk(3)+1;
149.         case 4
150.             kk(4)=kk(4)+1;
151.     end
152. end
153.
154. %正确率
155. rightridio=(kk-k)./kk

```

#### 激活函数

```

[plain]  

01. %% 激活函数
02. function [ y ] = g( x )
03.     y = 1./(1+exp(-x));
04. end

```

顶

1

踩

0

上一篇

简单易学的机器学习算法——CART之回归树

下一篇

简单易学的机器学习算法——支持向量机(开篇：基本概念)

我的同类文章

machine learning (41)

• 图解机器学习总结——1、...

2016-04-09

阅读 1949

• Latent Dirichlet Allocation...

2016-01-25

阅读 3694

• 可扩展机器学习——分类—...

2015-12-14

阅读 489

• 可扩展机器学习——线性回...

2015-12-12

阅读 556

• 可扩展机器学习——概述

2015-12-01

阅读 346

• 机器学习中的常见问题——...

2016-03-31

阅读 271

• 广告计算——平滑CTR

2016-01-11

阅读 1039

• 可扩展机器学习——梯度下...

2015-12-12

阅读 484

• 可扩展机器学习——Spark...

2015-12-01

阅读 445

• 机器学习中的常见问题——...

2015-09-27

阅读 1209

更多文章

猜你在找

- 《C语言/C++学习指南》加密解密篇（安全相关算法）
- Spark零基础入门（1）：Scala基本数据类型及程序控制
- C语言系列之 字符串压缩算法与结构体初探
- C语言系列之 快速排序与全排列算法
- C语言系列之 递归算法示例与 Windows 趣味小项目

查看评论

9楼 依然静谧 2016-02-27 03:02发表



只有一个隐层？

8楼 Ailon\_\_ 2015-12-11 11:21发表



Unable to read file data: No such file or directory.

Re: zhiyong\_will 2015-12-11 11:32发表



回复Ailon\_\_：这提示你要导入数据，我提供的只是实验的代码，没有实验数据

7楼 chaosimpler 2015-12-03 15:37发表



有个问题不知道是不是我理解的有误呢，在第3步输出层的输出公式中，最后加上的偏置应该是  $b_k$  而不是  $b_j$  吧？

Re: zhiyong\_will 2015-12-03 18:29发表



回复u012973416：建议与<http://blog.csdn.net/google19890102/article/details/49736619>一起读

Re: zhiyong\_will 2015-12-03 18:29发表



回复u012973416：已修正，下面是正确的，谢谢

Re: chaosimpler 2015-12-04 10:55发表



回复google19890102：非常感谢回复

6楼 [KDFZ\\_SUSU](#) 2015-10-06 04:03发表



没注意输出层的激励函数和隐层不一样 为什么输出层选择线性激励函数呢？

5楼 [KDFZ\\_SUSU](#) 2015-10-05 03:43发表



E对Wjk的偏导计算少算了H对Wjk的偏导数吧 Wjk的权重更新公式后面那一项应该再乘（1-Hj)xj

Re: [zhiyong\\_will](#) 2015-10-05 21:57发表



回复KDFZ\_SUSU：已检查过，你再看下，应该没问题

4楼 [MineralterMan](#) 2015-04-10 23:05发表



博主的输出层少了一个激励函数，让反馈偏移量的计算简化了不少。。  
请参见[blog.csdn.net/mineralterman/article/details/44986125](http://blog.csdn.net/mineralterman/article/details/44986125)，  
按照神经元的工作方式，有一次求和，就应该存在一次激励。

Re: [zhiyong\\_will](#) 2015-05-02 11:18发表



回复MineralterMan：这里使用的是一种线性激活函数，你可以参见这篇博客：  
<http://www.cnblogs.com/heaad/archive/2011/03/07/1976443.html>里面有个讲激活函数的

3楼 [MineralterMan](#) 2015-04-10 22:19发表



推导的公式特别简单，我看懂了。而且用latex语言写公式，太棒了！

2楼 [zhiyong\\_will](#) 2014-07-14 08:27发表



是不是你给我的邮件，我回复了一个邮件

1楼 [tim110629](#) 2014-07-11 16:02发表



请问您的data可以提供吗？

您还没有登录,请[\[登录\]](#)或[\[注册\]](#)

\* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

核心技术类目

全部主题   Hadoop   AWS   移动游戏   Java   Android   iOS   Swift   智能硬件   Docker   OpenStack  
VPN   Spark   ERP   IE10   Eclipse   CRM   JavaScript   数据库   Ubuntu   NFC   WAP   jQuery  
BI   HTML5   Spring   Apache   .NET   API   HTML   SDK   IIS   Fedora   XML   LBS   Unity  
Splashtop   UML   components   Windows Mobile   Rails   QEMU   KDE   Cassandra   CloudStack  
FTC   coremail   OPhone   CouchBase   云计算   iOS6   Rackspace   Web App   SpringSide  
Maemo   Compuware   大数据   aptech   Perl   Tornado   Ruby   Hibernate   ThinkPHP   HBase  
Pure   Solr   Angular   Cloud Foundry   Redis   Scala   Django   Bootstrap