

Augster的博客

目录视图

摘要视图

RSS 订阅

个人资料



秋水长天q

关注

发私信

访问：32288次

积分：637

等级：BLOG > 3

排名：千里之外

原创文章：34篇

转载：0篇

译文：0篇

评论：25条

文章搜索

文章分类

机器学习 (15)

Java基础 (6)

Hadoop (6)

Python (1)

线性代数 (1)

Leetcode (4)

数据结构 (1)

JavaWeb (1)

文章存档

2017年09月 (1)

2017年07月 (1)

2017年04月 (2)

2017年03月 (4)

2017年02月 (1)

展开

阅读排行

神经网络之BP神经网络（Pyt... (12732)

迁移学习算法之TrAdaBoost (6125)

图灵赠书——程序员11月书单

【思考】Python这么厉害的原因竟然是！

感恩节赠书：《深度学习》等异步社区优秀图书和译者评选启动！

每周荐书：京东架构、Linux内核、Python全栈

神经网络之BP神经网络（Python实现）

2016-10-17 17:46

12794人阅读

评论(6)

收藏

举报

分类：

机器学习 (14)

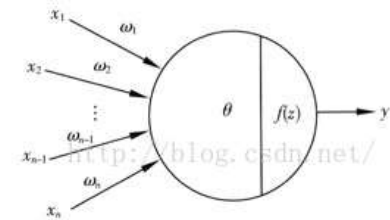
版权声明：本文为博主原创文章，未经博主允许不得转载。

文中部分截图出自周志华《机器学习》一书中，感谢周老师！！

1. 神经元模型

神经网络：神经网络是由具有适应性的简单单元组成的广泛并行互联的网络，它的组织可以模拟生物神经系统对真实世界所做出的交互反应。

简单说来，神经网络是由多个基本的神经元组成的网络系统，每一个神经元有对应的输入和相应的输出。利用下面的这幅图来说明单一的神经元的工作过程。



$x_1 \sim x_n$ 神经元的输入 y 神经元的输出 $f(z)$ 激活函数
 θ 阈值 $w_1 \sim w_n$ 神经元之间的连接权重

如图所示，此神经元有n个输入(x_1, x_2, \dots, x_n), 每一个输入都有一个对应的权重 w ，神经元会有一个阈值，将输入和权重进行乘积之后再与阈值相比较，作用于 f 函数之后得到输出。

$$y = f\left(\sum_{i=1}^n w_i x_i - \theta\right)$$

在实际应用中，这里的 f 函数往往采用Sigmoid函数，这样可以将输出映射到0,1范围

2. BP神经网络

误差逆向传播(err BackPropagation)简称BP算法是神经网络中常见的训练算法。BP神经网络模型拓扑结构包括输入层（input），隐层(hidden layer)和输出层(output layer)。它的学习规则是使用梯度下降法，通过反向传播来不断调整网络的权值和阈值，使网络的误差平方和最小。

http://blog.csdn.net/Augster/article/details/52837474

1/6

基于用户的协同过滤推荐一实...	(1062)
范数汇总	(789)
SVD矩阵奇异值分解	(785)
决策树学习	(747)
迁移学习概述	(722)
集体智慧编程第三章之发现群组	(629)
集体智慧编程第二章之提供推荐	(581)
MapReduce实现KNN	(540)

评论排行	
迁移学习算法之TrAdaBoost	(14)
神经网络之BP神经网络（Pyt...	(6)
范数汇总	(3)
MapReduce实现KNN	(1)
k-近邻算法	(0)
决策树学习	(0)
贝叶斯分类算法	(0)
SSM框架整合	(0)
equals和hashCode方法	(0)
Java线程	(0)

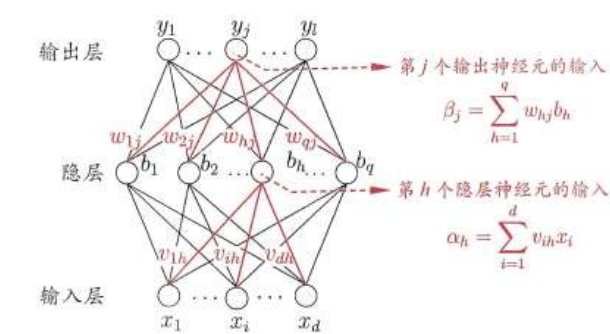
代码学习

最新评论	
神经网络之BP神经网络（Python实现） 香香的七仔：使用√（输入层节点数×输出层节点数）作为隐藏层节点数（88个）可以将正确率提升到大概95%	
范数汇总 yujianglan：你好，请问F范数和21范数之间怎么转化呢？谢谢	
迁移学习算法之TrAdaBoost confina：想问下什么样本的权重怎么和基本的学习器结合啊 哪一种学习器会考虑样本的权重呢	
范数汇总 秋水长天q：@aoqiulei7117机器学习中的很多地方都会有范数的应用，比如说公式中的一些约束之类的：	
范数汇总 aoqiulei7117：范数有啥应用呢	
神经网络之BP神经网络（Python实现） weixin_39518878：博主你好，请问这段代码的迭代次数在哪里啊	
神经网络之BP神经网络（Python实现）	



达内可靠吗

现在给定数据集 $D=\{(x_1, y_1), \dots, (x_m, y_m)\}$,其中，输入样本是有d维的向量组成，输出样本是有L维的向量。所以现在可以规划这样的一个神经网络，其中有d个输入，L个输出，中间的隐层可以设置成q个，如下图（图片来自《机器学习》）所示：



如图所示，第h个隐层神经元的输入 α_h 为d个输入神经元与各自对应的权值v的乘积之和，第j个输出神经元的输入为全部的q个隐层神经元的输出与各自对应的权值乘积之和。隐层第h个神经元的阈值用 γ_h 表示，输出层第j个神经元的阈值用 θ_j 表示。

现在假设神经元的输出表示为 $\hat{y}_k = (\hat{y}_1^k, \hat{y}_2^k, \dots, \hat{y}_l^k)$ ，即：

$$\hat{y}_j^k = f(\beta_j - \theta_j),$$

采用均方误差最小化来最优化系数：

$$E_k = \frac{1}{2} \sum_{j=1}^l (\hat{y}_j^k - y_j^k)^2.$$

BP神经网络采用梯度下降算法进行最优化对于优化目标的Ek，给定一个学习率，可以这样来进行优化：

$$\Delta w_{hj} = -\eta \frac{\partial E_k}{\partial w_{hj}}.$$

由于 $w_{h,j}$ 先影响到输出层第j个神经元的输入，再影响到其输出，最后影响到Ek，所以采用链式求导法则可得：

$$\frac{\partial E_k}{\partial w_{hj}} = \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial w_{hj}}.$$

其中 $\frac{\partial \beta_j}{\partial w_{hj}} = b_h$ ，注意到Sigmoid函数求导之后有 $f'(x) = f(x) * (1 - f(x))$ 所以



综合可得：

$$\begin{aligned} g_j &= -\frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j} \\ &= -(\hat{y}_j^k - y_j^k) f'(\beta_j - \theta_j) \\ &= \hat{y}_j^k (1 - \hat{y}_j^k) (y_j^k - \hat{y}_j^k) . \end{aligned}$$

$\Delta_{h,j} = \eta g_j b_h$
所以
类似可得：

$$\begin{aligned} \Delta \theta_j &= -\eta g_j , \\ \Delta v_{ih} &= \eta e_h x_i , \\ \Delta \gamma_h &= -\eta e_h , \end{aligned}$$

其中的 e_h 表示为：

$$e_h = b_h (1 - b_h) \sum_{j=1}^l w_{hj} g_j$$

对于每个训练样本，BP算法先将输入样例提供给输入神经元，然后逐层的将信号向前传播，直到产生输出层的结果，然后计算输出层的误差，再将误差逆向传播到隐层神经元，然后根据神经元的误差来对连接权值和与之进行调整优化，知道训练达到很小的误差值或者迭代到一定的次数。具体过程如下图所示（图片来自于《机器学习》）

输入：训练集 $D = \{(x_k, y_k)\}_{k=1}^m$;
学习率 η .

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 $(x_k, y_k) \in D$ do
4: 根据当前参数和式(5.3) 计算当前样本的输出 \hat{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 θ_j , γ_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

图 5.8 误差逆传播算法

接下来用一个具体的实例来说明BP神经网络的实现过程，用Python在模mnist数据集上进行实验，mnist数据集是一个手写字的数据集。在本文中，为了便于实验，采用的是.mat格式的文件进行操作。

3. 实验过程

首先需要将.mat格式的文件读入到程序当中，我们直接利用scipy模块的IO函数来进行读取：

```
1 #定义数据文件读取函数
2 def LoadFile(filename):
3     data=sio.loadmat(filename)
4
```



达内可靠吗



```
5 data=data[filename[0:-4]]
   return data
```

定义Sigmoid函数：

```
1 #定义Sigmoid函数
2 def get(x):
3     act_vec=[]
4     for i in x:
5         act_vec.append(1/(1+math.exp(-i)))
6     act_vec=np.array(act_vec)
7     return act_vec
```

将训练数据读入之后，接下来就可以利用训练数据来训练了，输入神经元的个数毫无疑问是数据的维数，在这里训练样本是一个60000*784的矩阵，所以输入神经元的个数就是784，设置隐层的神经元个数为8个，输出层因为对应的是0到9个数字类别，所以设置输出层的神经元个数为10，在学习的过程中，设置学习每一次的学习率为0.2，开始的时候随机初始化w1，w2作为输入层到隐层，隐层到输出层的权值矩阵，设置隐层和输出层神经元的初始阈值为0。具体代码如下：

```
1 #训练BP神经网络
2 def TrainNetwork(sample, label):
3     sample_num = len(sample)
4     sample_len = len(sample[0])
5     out_num = 10
6     hid_num = 8
7     w1 = 0.2 * np.random.random((sample_len, hid_num)) - 0.1
8     w2 = 0.2 * np.random.random((hid_num, out_num)) - 0.1
9     hid_offset = np.zeros(hid_num)
10    out_offset = np.zeros(out_num)
11    input_learnrate = 0.2
12    hid_learnrate = 0.2
13    for i in range(0, len(sample)):
14        t_label=np.zeros(out_num)
15        t_label[label[i]]=1
16        #前向的过程
17        hid_value=np.dot(sample[i],w1)+hid_offset #隐层的输入
18        hid_act=get(hid_value) #隐层对应的输出
19        out_value=np.dot(hid_act,w2)+out_offset
20        out_act=get(out_value) #输出层最后的输出
21
22        #后向过程
23        err=t_label-out_act
24        out_delta=err*out_act*(1-out_act) #输出层的方向梯度方向
25        hid_delta = hid_act*(1 - hid_act) * np.dot(w2, out_delta)
26        for j in range(0, out_num):
27            w2[:,j]+=hid_learnrate*out_delta[j]*hid_act
28        for k in range(0, hid_num):
29            w1[:,k]+=input_learnrate*hid_delta[k]*sample[i]
30
31        out_offset += hid_learnrate * out_delta #阈值的更新
32        hid_offset += input_learnrate * hid_delta
33
34    return w1,w2,hid_offset,out_offset
```



达内可靠吗



最后利用测试集来测试训练的网络的分类正确率即可：

```
1 #测试过程
2 def Test():
3     train_sample=LoadFile('mnist_train.mat')
4     train_sample=train_sample/256.0
5     train_label=LoadFile('mnist_train_labels.mat')
6     test_sample=LoadFile('mnist_test.mat')
7     test_sample=test_sample/256.0
8     test_label=LoadFile('mnist_test_labels.mat')
9     w1,w2,hid_offset,out_offset=TrainNetwork(train_sample, train_label)
10    right = np.zeros(10)
11    numbers = np.zeros(10)
12    for i in test_label:
13        numbers[i]+=1
14    print(numbers)
15    for i in range(0, len(test_label)):
16        hid_value=np.dot(test_sample[i],w1)+hid_offset
17        hid_act=get(hid_value)
18        out_value=np.dot(hid_act,w2)+out_offset
19        out_act=get(out_value)
20        if np.argmax(out_act) == test_label[i]:
21            right[test_label[i]] += 1
22    print(right.sum()/ len(test_label))
```

最中通过实验可知，在mnist数据集上利用BP神经网络识别的正确率大致在87.78%左右。

4. 参考文章

- 周志华《机器学习》
- BP神经网络Python实现
- BPNetwork

顶 7 踩 0

- 上一篇 决策树学习
- 下一篇 贝叶斯分类算法

相关文章推荐

- BP神经网络python简单实现
- python实现bp神经网络
- MySQL在微信支付下的高可用运营--莫晓东
- 腾讯云容器服务架构实现介绍--董晓杰
- MATLAB神经网络编程（七）——BP神经网络的...
- Python使用numpy实现BP神经网络
- 容器技术在58同城的实践--姚远
- 微博热点事件背后的数据库运维心得--张冬洪
- Python实现一个最简单的2层BP神经网络
- 神经网络中 BP 算法的原理与 PYTHON 实现源码...
- SDCC 2017之容器技术实战线上峰会
- BP神经网络——Python 实现
- Python实现一个简单的3层BP神经网络
- 深度学习笔记一：BP神经网络的介绍和Python代...
- SDCC 2017之数据库技术实战线上峰会
- Python实现三层BP神经网络



达内可靠吗



查看评论



香香的七仔

4楼 2017-12-15 11:33发表

使用√（输入层节点数×输出层节点数）作为隐藏层节点数（88个）可以将正确率提升到大概95%



weixin_39518878

3楼 2017-08-07 15:03发表

博主你好，请问这段代码的迭代次数在哪里啊



Fengming1220

2楼 2017-07-30 14:30发表

你好，请问原始数据怎么转化为.mat格式文件？



Coding_ForFun

Re: 2017-07-30 20:19发表

回复Fengming1220：用matlab就可以转化为mat格式了~



sinat_38326000

1楼 2017-04-15 17:28发表

为什么在输入后向过程语句时显示语法无效



秋水长天q

Re: 2017-04-17 21:29发表

回复语法正确的：

发表评论

用户名： wujuxKkoolerter

评论内容：



提交

* 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

