

天然桑蚕丝

马上抢购>>



秒杀

货到付款

木已成林的博客

http://blog.sina.com.cn/sisilin0211 [订阅] [手机订阅]

首页 博文目录 图片 关于我



新浪博客

新版客户端



扫一扫
下载



木已成林

微博

加好友

发纸条

写留言

加关注

博客地图



博客等级: 13
博客积分: 337
博客访问: 25,503
关注人气: 29
获赠金笔: 3
赠出金笔: 0
荣誉徽章:

JD.COM 京东



¥ 209.00

5/8

相关博文

大盘流星示警 变盘指日可待
金池

题材股炒作是否对社会有益
唐史主任

正文

字体大小: 大 中 小

数据挖掘系列（10）——卷积神经网络算法的一个实现(转) (2015-06-26 12:51:05)

转 载 ▼

分类: 数据挖掘

前言

从理解卷积神经到实现它，前后花了一个月时间，现在也还有一些地方没有理解透彻，CNN还是有一定难度的，不是看哪个的博客和一两篇论文就明白了，主要还是靠自己专研，阅读推荐列表在末尾的参考文献。目前实现的CNN在MINIT数据集上效果还不错，但是还有一些bug，因为最近比较忙，先把之前做的总结一下，以后再继续优化。

卷积神经网络CNN是Deep Learning的一个重要算法，在很多应用上表现出卓越的效果，[1]中对比多重算法在文档字符识别的效果，结论是CNN优于其他所有的算法。CNN在手写体识别取得最好的效果，[2]将CNN应用在基于人脸的性别识别，效果也非常不错。前段时间我用BP神经网络对手机拍照图片的数字进行识别，效果还算不错，接近98%，但在汉字识别上表现不佳，于是想试试卷积神经网络。

1、CNN的整体网络结构

卷积神经网络是在BP神经网络的改进，与BP类似，都采用了前向传播计算输出值，反向传播调整权重和偏置；CNN与标准的BP最大的不同是：CNN中相邻层之间的神经单元并不是全连接，而是部分连接，也就是某个神经单元的感知区域来自于上层的部分神经单元，而不是像BP那样与所有的神经单元相连接。CNN的有三个重要的思想架构：

- 局部区域感知
- 权重共享
- 空间或时间上的采样

局部区域感知能够发现数据的一些局部特征，比如图片上的一个角，一段弧，这些基本特征是构成动物视觉的基础[3]；而BP中，所有的像素点是一堆混乱的点，相互之间的关系没有被挖掘。

CNN中每一层的由多个map组成，每个map由多个神经单元组成，同一个map的所有神经单元共用一个卷积核（即权重），卷积核往往代表一个特征，比如某个卷积和代表一段弧，那么把这个卷积核在整个图片上滚一下，卷积值较大的区域就很有可能是一段弧。注意卷积核其实就是权重，我们并不需要单独去计算一个卷积，而是一个固定大小的权重矩阵去图像上匹配时，这个操作与卷积类似，因此我们称为卷积神经网络，实际上，BP也可以看做一种特殊的卷积神经网络，只是这个卷积核就是某层的所有权重，即感知区域是整个图像。权重共享策略减少了需要训练的参数，使得训练出来的模型的泛华能力更强。

采样的目的主要是混淆特征的具体位置，因为某个特征找出来后，它的具体位置已经不重要了，我们只需要这个特征与其他的位置，比如一个“8”，当我们得到了上面一个“o”时，我们不需要知道它在图像的具体位置，只需要知道它下面又是一个“o”我们就可以知道是一个‘8’了，因为图片中“8”在图片中偏左或者偏右都不影响我们认识它，这种混淆具体位置的策略能对变形和扭曲的图片进行识别。

CNN的这三个特点是其对输入数据在空间（主要针对图像数据）上和时间（主要针对时间序列数据，参考TDNN）上的扭曲有很强的鲁棒性。CNN一般采用卷积层与采样层交替设置，即一层卷积层接一层采样层，采样层后接一层卷积...这样卷积层提取出特征，再进行组合形成更抽象的特征，最后形成对图片对象的描述特征，CNN后面还可以跟全连接层，全连接层跟BP一样。下面是一个卷积神经网络的示例：

如何解读消息？如何解读财报？
长余量化

指数不会有大的调整空间
田渭东

忠言虽逆耳 征战需记清
安阳

大盘短期维持这种震荡格局
王九洲

八大主力机构一季度持股揭秘
黄珏老鹅

外围市场接连受挫 继续深挖超跌个
邢星

喝酒吃药会是这次的主线么？
李世红

市场怎么走关键看成交量变化
道无兄

更多>>

弹力·亲肤·舒适

点击试穿 包邮到家

抢!

天然蚕丝

货到付款

只做好面料

聚划算

全场5折起

母亲节专场疯抢中

推荐博文

“紫五月”将在沉默中悄然爆发

5月5日操作策略

周四指数空间与操作策略

推荐一只中报预增10倍低价股

5.5早间要闻评论

周四重点关注的板块及个股（图）

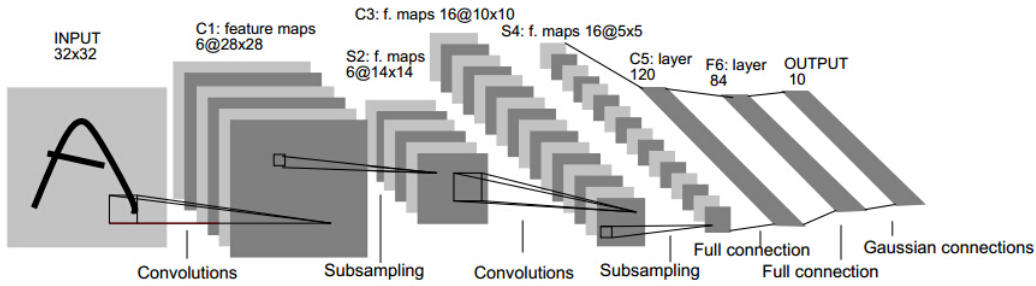


图1（图片来源）

卷积神经网络的基本思想是这样，但具体实现有多重版本，我参考了matlab的Deep Learning的工具箱DeepLearnToolbox，这里实现的CNN与其他最大的差别是采样层没有权重和偏置，仅仅只对卷积层进行一个采样过程，这个工具箱的测试数据集是MINIST，每张图像是28*28大小，它实现的是下面这样一个CNN：

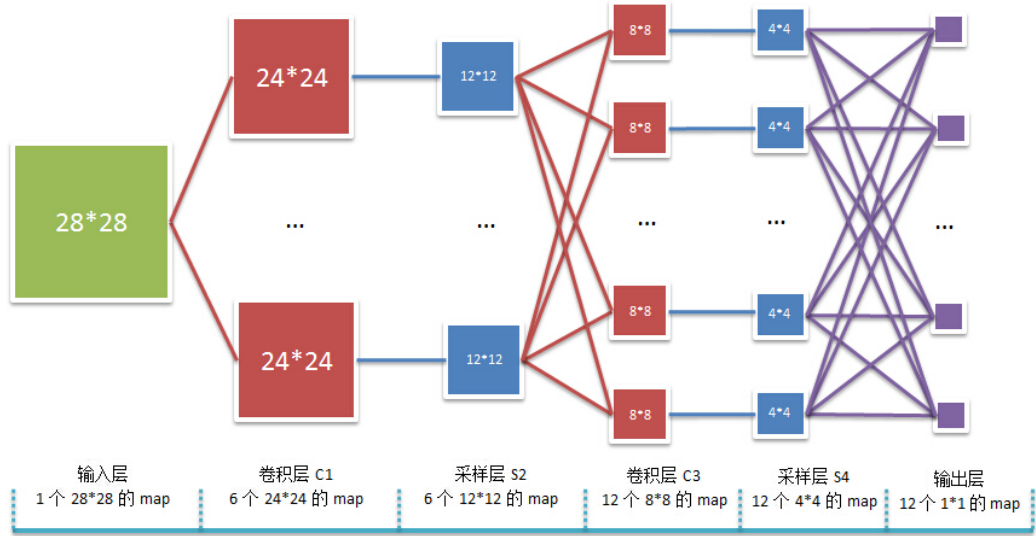


图2

2、网络初始化

CNN的初始化主要是初始化卷积层和输出层的卷积核（权重）和偏置，DeepLearnToolbox里面面对卷积核和权重进行随机初始化，而对偏置进行全0初始化。

3、前向传输计算

前向计算时，输入层、卷积层、采样层、输出层的计算方式不相同。

3.1 输入层：输入层没有输入值，只有一个输出向量，这个向量的大小就是图片的大小，即一个28*28矩阵；

3.2 卷积层：卷积层的输入要么来源于输入层，要么来源于采样层，如上图红色部分。卷积层的每一个map都有一个大小相同的卷积核，Toolbox里面是5*5的卷积核。下面是一个示例，为了简单起见，卷积核大小为2*2，上一层的特征map大小为4*4，用这个卷积在图片上滚一遍，得到一个(4-2+1)*(4-2+1)=3*3的特征map，卷积核每次移动一步，因此。在Toolbox的实现中，卷积层的一个map与上层的所有map都关联，如上图的S2和C3，即C3共有6*12个卷积核，卷积层的每一个特征map是不同的卷积核在前一层所有map上作卷积并将对应元素累加后加一个偏置，再求sigmoid得到的。还有需要注意的是，卷积层的map个数是在网络初始化指定的，而卷积层的map的大小是由卷积核和上一层输入map的大小决定的，假设上一层的map大小是n*n、卷积核的大小是k*k，则该层的map大小是(n-k+1)*(n-k+1)，比如上图的24*24的map大小24=(28-5+1)。斯坦福的深度学习教程更加详细的介绍了卷积特征提取的计算过程。

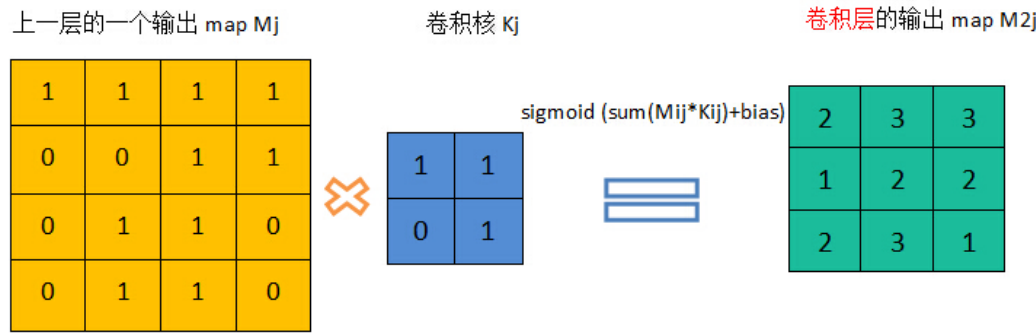


图3

3.3 采样层 (subsampling, Pooling)：采样层是对上一层map的一个采样处理，这里的采样方式是对上一层map的相邻小区域进行聚合统计，区域大小为scale*scale，有些实现是取小区域的最大值，而ToolBox里面的实现是采用2*2小区域的均值。注意，卷积的计算窗口是有重叠的，而采用的计算窗口没有重叠，ToolBox里

楼市有了新变化（早盘必读）

两种概念股后市或走牛

3000点关口资金缘何不敢进场

周四热点概念与题材前瞻（附股）



[查看更多>>](#)

面计算采样也是用卷积(conv2(A,K,'valid'))来实现的，卷积核是2*2，每个元素都是1/4，去掉计算得到的卷积结果中有重叠的部分，即：

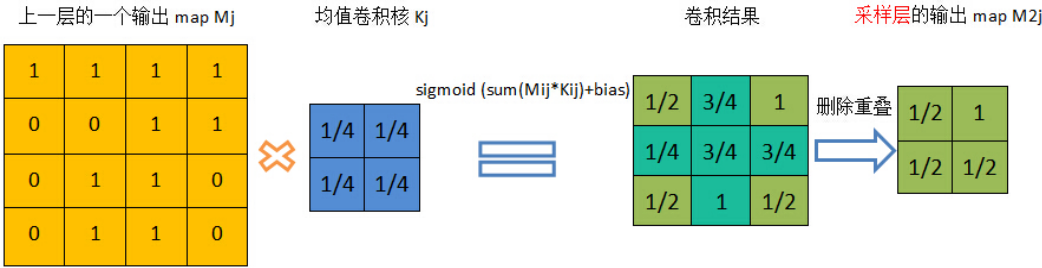


图4

4、反向传输调整权重

反向传输过程是CNN最复杂的地方，虽然从宏观上来看基本思想跟BP一样，都是通过最小化残差来调整权重和偏置，但CNN的网络结构并不像BP那样单一，对不同的结构处理方式不一样，而且因为权重共享，使得计算残差变得很困难，很多论文[1][5]和文章[4]都进行了详细的讲述，但我发现还是有一些细节没有讲明白，特别是采样层的残差计算，我会在这里详细讲述。

4.1输出层的残差

和BP一样，CNN的输出层的残差与中间层的残差计算方式不同，输出层的残差是输出值与类标值得误差值，而中间各层的残差来源于下一层的残差的加权和。输出层的残差计算如下：

$$\delta_i^{(n_l)} = \frac{\partial}{\partial z_i^{(n_l)}} \frac{1}{2} \|y - h_{W,b}(x)\|^2 = -(y_i - a_i^{(n_l)}) \cdot f'(z_i^{(n_l)})$$

公式来源

这个公式不做解释，可以查看公式来源，看斯坦福的深度学习教程的解释。

4.2 下一层为采样层（ subsampling ）的卷积层的残差

当一个卷积层L的下一层(L+1)为采样层，并假设我们已经计算得到了采样层的残差，现在计算该卷积层的残差。从最上面的网络结构图我们知道，采样层（L+1）的map大小是卷积层L的1/（scale*scale），ToolBox里面，scale取2，但这两层的map个数是一样的，卷积层L的某个map中的4个单元与L+1层对应map的一个单元关联，可以对采样层的残差与一个scale*scale的全1矩阵进行克罗内克积进行扩充，使得采样层的残差的维度与上一层的输出map的维度一致，Toolbox的代码如下，其中d表示残差，a表示输出值：

```
net.layers{1}.d{j} = net.layers{1}.a{j} .* (1 - net.layers{1}.a{j}) .* expand(net.layers{1 + 1}.d{j}, [net.layers{1 + 1}.scale net.layers{1 + 1}.scale 1])
```

扩展过程：

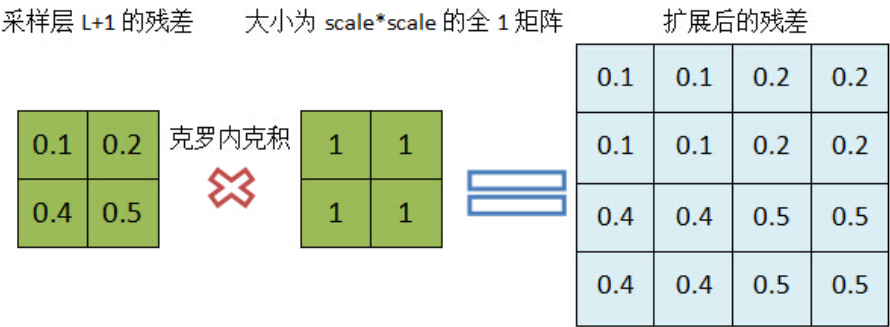


图5

利用卷积计算卷积层的残差：



图6

4.3 下一层为卷积层（ subsampling ）的采样层的残差

当某个采样层L的下一层是卷积层(L+1)，并假设我们已经算出L+1层的残差，现在计算L层的残差。采样层到卷积层直接的连接是有权重和偏置参数的，因此不像卷积层到采样层那样简单。现再假设L层第j个map Mj与

L+1层的M_{2j}关联，按照BP的原理，L层的残差D_j是L+1层残差D_{2j}的加权和，但是这里的困难在于，我们很难理清M_{2j}的那些单元通过哪些权重与M_j的哪些单元关联，Toolbox里面还是采用卷积（稍作变形）巧妙的解决了这个问题，其代码为：

```
convn(net.layers{l + 1}.d{j}, rot180(net.layers{l + 1}.k{i}{j}),'full');
```

rot180表示对矩阵进行180度旋转（可通过行对称交换和列对称交换完成），为什么这里要对卷积核进行旋转，答案是：通过这个旋转，'full'模式下得卷积的正好抓住了前向传输计算上层map单元与卷积和及当期层map的关联关系，需要注意的是matlab的内置函数convn在计算卷积前，会对卷积核进行一次旋转，因此我们之前的所有卷积的计算都对卷积核进行了旋转：



```
a = 1 1 1 1 1 1 1 1 1 k = 1 2 3 4 5 6 7 8 9 >> convn(a,k,'full') ans = 1 3 6 5 3 5 12 21 16 9 12 27 45 33 18 11 24 39 28 15 7 15 24 17 9
```



convn在计算前还会对待卷积矩阵进行0扩展，如果卷积核为k*k，待卷积矩阵为n*n，需要以n*n原矩阵为中心扩展到(n+2(k-1))*(n+2(k-1))，所有上面convn(a,k,'full')的计算过程如下：

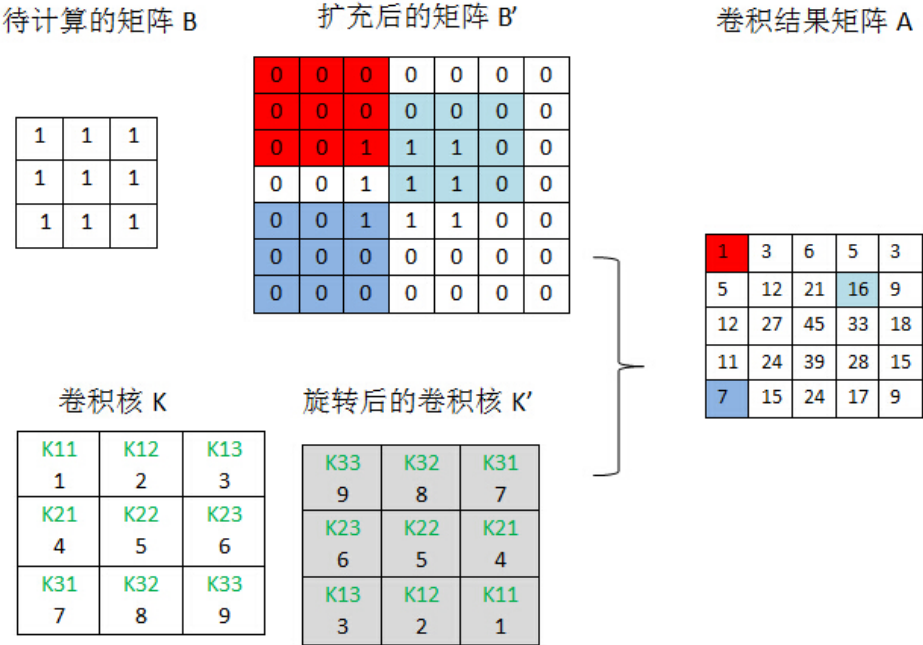


图7

实际上convn内部是否旋转对网络训练没有影响，只要内部保持一致（即都要么旋转，要么都不旋转），所有我的卷积实现里面没有对卷积核旋转。如果在convn计算前，先对卷积核旋转180度，然后convn内部又对其旋转180度，相当于卷积核没有变。

为了描述清楚对卷积核旋转180与卷积层的残差的卷积所关联的权重与单元，正是前向计算所关联的权重与单元，我们选一个稍微大一点的卷积核，即假设卷积层采用用3*3的卷积核，其上一层采样层的输出map的大小是5*5，那么前向传输由采样层得到卷积层的过程如下：

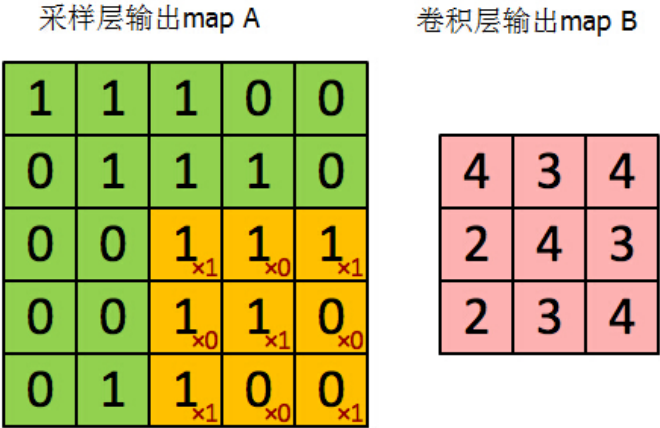


图8

这里我们采用自己实现的convn（即内部不会对卷积核旋转），并假定上面的矩阵A、B下标都从1开始，那么有：



$$\begin{aligned}
 B11 &= A11 * K11 + A12 * K12 + A13 * K13 + A21 * K21 + A22 * K22 + A23 * K23 + A31 * K31 + A32 * K32 + A33 * K33 \\
 B12 &= A12 * K11 + A13 * K12 + A14 * K13 + A22 * K21 + A23 * K22 + A24 * K23 + A32 * K31 + A33 * K32 + A34 * K33 \\
 B13 &= A13 * K11 + A14 * K12 + A15 * K13 + A23 * K21 + A24 * K22 + A25 * K23 + A33 * K31 + A34 * K32 + A35 * K33 \\
 B21 &= A21 * K11 + A22 * K12 + A23 * K13 + A31 * K21 + A32 * K22 + A33 * K23 + A41 * K31 + A42 * K32 + A43 * K33 \\
 B22 &= A22 * K11 + A23 * K12 + A24 * K13 + A32 * K21 + A33 * K22 + A34 * K23 + A42 * K31 + A43 * K32 + A44 * K33 \\
 B23 &= A23 * K11 + A24 * K12 + A25 * K13 + A33 * K21 + A34 * K22 + A35 * K23 + A43 * K31 + A44 * K32 + A45 * K33 \\
 B31 &= A31 * K11 + A32 * K12 + A33 * K13 + A41 * K21 + A42 * K22 + A43 * K23 + A51 * K31 + A52 * K32 + A53 * K33 \\
 B32 &= A32 * K11 + A33 * K12 + A34 * K13 + A42 * K21 + A43 * K22 + A44 * K23 + A52 * K31 + A53 * K32 + A54 * K33 \\
 B33 &= A33 * K11 + A34 * K12 + A35 * K13 + A43 * K21 + A44 * K22 + A45 * K23 + A53 * K31 + A54 * K32 + A55 * K33
 \end{aligned}$$


我们可以得到B矩阵每个单元与哪些卷积核单元和哪些A矩阵的单元之间有关联：



A11 [K11, B11] A12 [K12, K11] [B12, B11] A13 [K13, K12, K11] [B12, B13, B11] A14 [K13, K12] [B12, B13] A15 [K13] [B13] A21 [K21, K11] [B21, B11] A22 [K22, K21, K12, K11] [B12, B22, B21, B11] A23 [K23, K22, K21, K13, K12, K11] [B23, B22, B21, B12, B13, B11] **A24 [K23, K22, K13, K12] [B23, B12, B13, B22]** A25 [K23, K13] [B23, B13] A31 [K31, K21, K11] [B31, B21, B11] A32 [K32, K31, K22, K21, K12, K11] [B31, B32, B22, B12, B21, B11] A33 [K33, K32, K31, K23, K22, K21, K13, K12, K11] [B23, B22, B21, B31, B12, B13, B11, B33, B32] A34 [K33, K32, K23, K22, K13, K12] [B23, B22, B32, B33, B12, B13] A35 [K33, K23, K13] [B23, B13, B33] A41 [K31, K21] [B31, B21] A42 [K32, K31, K22, K21] [B32, B22, B21, B31] A43 [K33, K32, K31, K23, K22, K21] [B31, B23, B22, B32, B33, B21] A44 [K33, K32, K23, K22] [B23, B22, B32, B33] A45 [K33, K23] [B23, B33] A51 [K31] [B31] A52 [K32, K31] [B31, B32] A53 [K33, K32, K31] [B31, B32, B33] A54 [K33, K32] [B32, B33] A55 [K33] [B33]



然后再用matlab的convn(内部会对卷积核进行180度旋转)进行一次convn(B,K,'full')，结合图7，看红色部分，除去0，A11=B'33*K'33=B11*K11，发现A11正好与K11、B11关联对不对；我们再看一个A24=B'34*K'21+B'35*K'22+B'44*K'31+B'45*K'32=B12*K23+B13*K22+B22*K13+B23*K12，发现参与A24计算的卷积核单元与B矩阵单元，正好是前向计算时关联的单元，所以我们可以通过旋转卷积核后进行卷积而得到采样层的残差。

残差计算出来后，剩下的就是用更新权重和偏置，这和BP是一样的，因此不再细究，有问题欢迎交流。

5、代码实现

详细的代码不再这里贴了，我依旧放在了[github](#)，欢迎参考和指正。我又是在重造车轮了，没有使用任何第三方的库类，这里贴一下调用代码：



```

public static void runCnn() { //创建一个卷积神经网络
    LayerBuilder builder = new LayerBuilder();
    builder.addLayer(Layer.buildInputLayer(new Size(28, 28)));
    builder.addLayer(Layer.buildConvLayer(6, new Size(5, 5)));
    builder.addLayer(Layer.buildSampLayer(new Size(2, 2)));
    builder.addLayer(Layer.buildConvLayer(12, new Size(5, 5)));
    builder.addLayer(Layer.buildSampLayer(new Size(2, 2)));
    builder.addLayer(Layer.buildOutputLayer(10));
    CNN cnn = new CNN(builder, 50); //导入数据集
    String fileName = "dataset/train.format";
    Dataset dataset = Dataset.load(fileName, ",", 784);
    cnn.train(dataset, 3); // String modelName = "model/model.cnn";
    cnn.saveModel(modelName);
    dataset.clear();
    dataset = null; //预测
    CNN cnn = CNN.loadModel(modelName);
    Dataset testset = Dataset.load("dataset/test.format", ",", -1);
    cnn.predict(testset, "dataset/test.predict");
}

```



6、参考文献

- [1].YANN LECUN. Gradient-Based Learning Applied to Document Recognition.
- [2].Shan Sung LIEW. Gender classification: A convolutional neural network approach.
- [3] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction teraction, and functional architecture in the cat' s visual cortex,"
- [4] tornadomeet. <http://www.cnblogs.com/tornadomeet/p/3468450.html>.
- [5] Jake Bouvrie. Notes on Convolutional Neural Networks.

[6] C++实现的详细介绍. <http://www.codeproject.com/Articles/16650/Neural-Network-for-Recognition-of-Handwritten-Digi>
[7] matlab DeepLearnToolbox <https://github.com/rasmusbergpalm/DeepLearnToolbox>

转载请注明出处：<http://www.cnblogs.com/fengfenggirl>

17 3
喜欢 赠金笔

分享：
阅读(14887) | 评论 (3) | 收藏(3) | 转载(11) | 喜欢▼ | 打印 | 举报 已投稿到： 排行榜

前一篇：数据挖掘系列（9）——BP神经网络算法与实践(转)
后一篇：K-means算法及文本聚类实践（转）

评论 重要提示：警惕虚假中奖信息 [发评论]









用户5762808822
博主你好，这篇文章是挺好，（因此。在Toolbox的实现中，卷积层的一个map与上层的所有map都关联，如上图的S2和C3，即C3共有6*12个卷积核，）中6*12个卷积核是不是有误，望指教
2015-11-16 19:47 回复(0)

Micropoint_Li

膜拜
2015-12-21 10:02 回复(0)

周扒皮没错
“卷积层的每一个特征map是不同的卷积核在前一层所有map上作卷积并将对应元素累加后加一个偏置，再求sigmoid得到的”
楼主，这句话是不是有点问题，我怎么感觉是相同的卷积核，你是不是打错了
3月19日 14:54 回复(0)

发评论

登录名： 密码： 找回密码 注册 ☒ 记住登录状态
☒ 分享到微博 ☐ 评论并转载此博文

按住左边滑块，拖动完成上方拼图

发评论

以上网友发言只代表其个人观点，不代表新浪网的观点或立场。

[< 前一篇](#)

[后一篇 >](#)

[数据挖掘系列（9）——BP神经网络算法与实践\(转\)](#)

[K-means算法及文本聚类实践（转）](#)

新浪BLOG意见反馈留言板 不良信息反馈 电话：4006900000 提示音后按1键（按当地市话标准计费） 欢迎批评指正
[新浪简介](#) | [About Sina](#) | [广告服务](#) | [联系我们](#) | [招聘信息](#) | [网站律师](#) | [SINA English](#) | [会员注册](#) | [产品答疑](#)

Copyright © 1996 - 2016 SINA Corporation, All Rights Reserved
新浪公司 版权所有