zouxy09的专栏

悲喜枯荣如是本无分别，当来则来，当去则去，随心，随性，随缘！-zouxy09@qq.com

个人资料

zouxy09

访问： 4587026次
积分： 24754
等级： BLOG 7
排名： 第156名

原创： 116篇 转载： 11篇
译文： 1篇 评论： 3150条

个人简介

关注：机器学习、计算机视觉、人机交互和人工智能等领域。
邮箱：zouxy09@qq.com
微博：Erik-zou
交流请发邮件，不怎么看博客私信^-^

文章搜索

文章分类

OpenCV (29)
机器学习 (46)
计算机视觉 (73)
Deep Learning (18)
语音识别与TTS (13)
图像处理 (55)
Linux (15)
Linux驱动 (4)
嵌入式 (18)
OpenAL (3)
Android (1)
C/C++编程 (18)
摄像头相关 (5)
数学 (5)
Kinect (9)
神经网络 (8)

Deep Learning论文笔记之（四）CNN卷积神经网络推导和实现

标签： Deep Learning

2013-08-16 00:40 149844人阅读 评论(57) 收藏 举报

分类： 计算机视觉（72） Deep Learning（17） 机器学习（45）

Deep Learning论文笔记之（四）**CNN**卷积神经网络推导和实现

zouxy09@qq.com

http://blog.csdn.net/zouxy09

自己平时看了一些论文，但老感觉看完过后就会慢慢的淡忘，某一天重新拾起来的时候又好像没有看过一样。所以想习惯地把一些感觉有用的论文中的知识点总结整理一下，一方面在整理过程中，自己的理解也会更深，另一方面也方便未来自己的勘察。更好的还可以放到博客上面与大家交流。因为基础有限，所以对论文的一些理解可能不太正确，还望大家不吝指正交流，谢谢。

本文的论文来自：

Notes on Convolutional Neural Networks, Jake Bouvrie。

这个主要是CNN的推导和实现的一些笔记，再看懂这个笔记之前，最好具有CNN的一些基础。这里也先列出一个资料供参考：

[1] Deep Learning（深度学习）学习笔记整理系列之（七）

[2] LeNet-5, convolutional neural networks

[3]卷积神经网络

[4] Neural Network for Recog... f Handwritten Digits

[5] Deep learning：三十八(Stacked CNN简单介绍)

[6] Gradient-based learning applied to document recognition.

[7]Imagenet classification with deep convolutional neural networks.

[8] UFLDL中的“卷积特征提取”和“池化”。

另外，这里有个matlab的Deep Learning的toolbox，里面包含了CNN的代码，在下一个博文中，我将会详细注释这个代码。这个笔记对这个代码的理解非常重要。

点的理解：

《Notes on Convolutional Neural Networks》

一、介绍

这个文档讨论的是CNNs的推导和实现。CNN架构的连接比权值要多很多，这实际上就隐含着实现了某种形式的规则化。这种特别的网络假定了我们希望通过数据驱动的方式学习到一些滤波器，作为提取输入的特征的一种方法。

本文中，我们先对训练全连接网络的经典BP算法做一个描述，然后推导2D CNN网络的卷积层和子采样层的BP权值更新方法。在推导过程中，我们更强调实现的效率，所以会给出一些Matlab代码。最后，我们转向讨论如何自动地学习组合前一层的特征maps，特别地，我们还学习特征maps的稀疏组合。

二、全连接的反向传播算法

典型的CNN中，开始几层都是卷积和下采样的交替，然后在最后一些层（靠近输出层的），都是全连接的一维网络。这时候我们已经将所有两维2D的特征maps转化为全连接的一维网络的输入。这样，当你准备好将最终的2D特征maps输入到1D网络中时，一个非常方便的方法就是把所有输出的特征maps连接成一个长的输入向量。然后我们回到BP算法的讨论。（更详细的基础推导可以参考UFLDL中“反向传导算法”）。

2.1、Feedforward Pass前向传播

在下面的推导中，我们采用平方误差代价函数。我们讨论的是多类问题，共c类，共N个训练样本。

$$E^N = \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{c}(t_k^n - y_k^n)^2.$$

这里t_k^n表示第n个样本对应的标签的第k维。y_k^n表示第n个样本对应的网络输出的第k个输出。对于多类问题，输出一般组织为“one-of-c”的形式，也就是只有该输入对应的类的输出节点输出为正，其他类的位或者节点为0或者负数，这个取决于你输出层的激活函数。sigmoid就是0，tanh就是-1.

因为在全部训练集上的误差只是每个训练样本的误差的总和，所以这里我们先考虑对于一个样本的BP。对于第n个样本的误差，表示为：

$$E^n = \frac{1}{2}\sum_{k=1}^{c}(t_k^n - y_k^n)^2 = \frac{1}{2}\|\mathbf{t}^n - \mathbf{y}^n\|_2^2.$$

传统的全连接神经网络中，我们需要根据BP规则计算代价函数E关于网络每一个权值的偏导数。我们用l来表示当前层，那么当前层的输出可以表示为：

$$\mathbf{x}^\ell = f(\mathbf{u}^\ell), \text{ with } \mathbf{u}^\ell = \mathbf{W}^\ell\mathbf{x}^{\ell-1} + \mathbf{b}^\ell$$

输出激活函数f(.)可以有很多种，一般是sigmoid函数或者双曲线正切函数。sigmoid将输出压缩到[0, 1]，所以最后的输出平均值一般趋于0 。所以如果将我们的训练数据归一化为零均值和方差为1，可以在梯度下降的过程中增加收敛性。对于归一化的数据集来说，双曲线正切函数也是不错的选择。

2.2、Backpropagation Pass反向传播

反向传播回来的误差可以看做是每个神经元的基的灵敏度sensitivities（灵敏度的意思

就是我们的基b变化多少，误差会变化多少，也就是误差对基的变化率，也就是导数了），定义如下：（第二个等号是根据求导的链式法则得到的）

$$\frac{\partial E}{\partial b} = \frac{\partial E}{\partial u}\frac{\partial u}{\partial b} = \delta$$

因为∂u/∂b=1，所以∂E/∂b=∂E/∂u=δ，也就是说bias基的灵敏度∂E/∂b=δ和误差E对一个节点全部输入u的导数∂E/∂u是相等的。这个导数就是让高层误差反向传播到底层的神来之笔。反向传播就是用下面这条关系式：（下面这条式子表达的就是第l层的灵敏度，就是）

$$\delta^{\ell} = (W^{\ell+1})^T \delta^{\ell+1} \circ f'(\mathbf{u}^{\ell})$$ 公式（1）

这里的“∘”表示每个元素相乘。输出层的神经元的灵敏度是不一样的：

$$\delta^{L} = f'(\mathbf{u}^{L}) \circ (\mathbf{y}^{n} - \mathbf{t}^{n}).$$

最后，对每个神经元运用delta（即δ）规则进行权值更新。具体来说就是，对一个给定的神经元，得到它的输入，然后用这个神经元的delta（即δ）来进行缩放。用向量的形式表述就是，对于第l层，误差对于该层每一个权值（组合为矩阵）的导数是该层的输入（等于上一层的输出）与该层的灵敏度（该层每个神经元的δ组合成一个向量的形式）的叉乘。然后得到的偏导数乘以一个负学习率就是该层的神经元的权值的更新了：

$$\frac{\partial E}{\partial \mathbf{W}^{\ell}} = \mathbf{x}^{\ell-1}(\delta^{\ell})^T$$
$$\Delta \mathbf{W}^{\ell} = -\eta \frac{\partial E}{\partial \mathbf{W}^{\ell}}$$ 公式（2）

对于bias基的更新表达式差不多。实际上，对于每一个权值$(W)_{ij}$都有一个特定的学习率η_{lj}。

三、Convolutional Neural Networks 卷积神经网络

3.1、Convolution Layers 卷积层

我们现在关注网络中卷积层的BP更新。在一个卷积层，上一层的特征maps被一个可学习的卷积核进行卷积，然后通过一个激活函数，就可以得到输出特征map。每一个输出map可能是组合卷积多个输入maps的值：

$$\mathbf{x}_j^{\ell} = f\left(\sum_{i \in M_j} \mathbf{x}_i^{\ell-1} * \mathbf{k}_{ij}^{\ell} + b_j^{\ell}\right)$$

这里M_j表示选择的输入maps的集合，那么到底选择哪些输入maps呢？有选择一对的或者三个的。但下面我们会讨论如何去自动选择需要组合的特征maps。每一个输出map会给一个额外的偏置b，但是对于一个特定的输出map，卷积每个输入maps的卷积核是不一样的。也就是说，如果输出特征map j和输出特征map k都是从输入map i中卷积求和得到，那么对应的卷积核是不一样的。

3.1.1、Computing the Gradients梯度计算

我们假定每个卷积层l都会接一个下采样层l+1 。对于BP来说，根据上文我们知道，要想求得层l的每个神经元对应的权值的权值更新，就需要先求层l的每一个神经节点的灵敏度δ（也就是权值更新的公式（2））。为了求这个灵敏度我们就需要先对下一层的节点（连接到当前层l的感兴趣节点的第l+1层的节点）的灵敏度求和（得到δ^{l+1}），然后乘以这些连接对应的权值（连接第l层感兴趣节点和第l+1层节点的权值）W。再乘以当前层l的该

神经元节点的输入u的激活函数f的导数值（也就是那个灵敏度反向传播的公式（1）的δ^l的求解），这样就可以得到当前层l每个神经节点对应的灵敏度δ^l了。

然而，因为下采样的存在，采样层的一个像素（神经元节点）对应的灵敏度δ对应于卷积层（上一层）的输出map的一块像素（采样窗口大小）。因此，层l中的一个map的每个节点只与l+1层中相应map的一个节点连接。

为了有效计算层l的灵敏度，我们需要上采样upsample 这个下采样downsample层对应的灵敏度map（特征map中每个像素对应一个灵敏度，所以也组成一个map），这样才使得这个灵敏度map大小与卷积层的map大小一致，然后再将层l的map的激活值的偏导数与从第l+1层的上采样得到的灵敏度map逐元素相乘（也就是公式（1））。

在下采样层map的权值都取一个相同值β，而且是一个常数。所以我们只需要将上一个步骤得到的结果乘以一个β就可以完成第l层灵敏度δ的计算。

我们可以对卷积层中每一个特征map j重复相同的计算过程。但很明显需要匹配相应的子采样层的map（参考公式（1））：

$$\boldsymbol{\delta}_j^\ell = \beta_j^{\ell+1}\left(f'(\mathbf{u}_j^\ell) \circ \mathrm{up}(\boldsymbol{\delta}_j^{\ell+1})\right)$$

up(.)表示一个上采样操作。如果下采样的采样因子是n的话，它简单的将每个像素水平和垂直方向上拷贝n次。这样就可以恢复原来的大小了。实际上，这个函数可以用Kronecker乘积来实现：

$$\mathrm{up}(\mathbf{x}) \equiv \mathbf{x} \otimes \mathbf{1}_{n\times n}.$$

好，到这里，对于一个给定的map，我们就可以计算得到其灵敏度map了。然后我们就可以通过简单的对层l中的灵敏度map中所有节点进行求和快速的计算bias基的梯度了：

$$\frac{\partial E}{\partial b_j} = \sum_{u,v} (\boldsymbol{\delta}_j^\ell)_{uv}.$$ 公式（3）

最后，对卷积核的权值的梯度就可以用BP算法来计算了（公式（2））。另外，很多连接的权值是共享的，因此，对于一个给定的权值，我们需要对所有与该权值有联系（权值共享的连接）的连接对该点求梯度，然后对这些梯度进行求和，就像上面对bias基的梯度计算一样：

$$\frac{\partial E}{\partial \mathbf{k}_{ij}^\ell} = \sum_{u,v} (\boldsymbol{\delta}_j^\ell)_{uv} (\mathbf{p}_i^{\ell-1})_{uv}$$

这里，$(\mathbf{p}_i^{\ell-1})_{uv}$是$\mathbf{x}_i^{\ell-1}$中的在卷积的时候与$\mathbf{k}_{ij}^\ell$逐元素相乘的patch，输出卷积map的(u, v)位置的值是由上一层的(u, v)位置的patch与卷积核k_ij逐元素相乘的结果。

咋一看，好像我们需要煞费苦心地记住输出map（和对应的灵敏度map）每个像素对应于输入map的哪个patch。但实际上，在Matlab中，可以通过一个代码就实现。对于上面的公式，可以用Matlab的卷积函数来实现：

$$\frac{\partial E}{\partial \mathbf{k}_{ij}^\ell} = \mathrm{rot180}(\mathrm{conv2}(\mathbf{x}_i^{\ell-1}, \mathrm{rot180}(\boldsymbol{\delta}_j^\ell), 'valid'))$$

我们先对delta灵敏度map进行旋转，这样就可以进行互相关计算，而不是卷积（在卷积的数学定义中，特征矩阵（卷积核）在传递给conv2时需要先翻转（flipped）一下。也就是颠倒下特征矩阵的行和列）。然后把输出反旋转回来，这样我们在前向传播进行卷积的时候，卷积核才是我们想要的方向。

3.2、Sub-sampling Layers 子采样层

对于子采样层来说，有N个输入maps，就有N个输出maps，只是每个输出map都变小了。

$$\mathbf{x}_j^\ell = f\left(\beta_j^\ell \operatorname{down}(\mathbf{x}_j^{\ell-1}) + b_j^\ell\right)$$

down(.)表示一个下采样函数。典型的操作一般是对输入图像的不同nxn的块的所有像素进行求和。这样输出图像在两个维度上都缩小了n倍。每个输出map都对应一个属于自己的乘性偏置β和一个加性偏置b。

3.2.1、Computing the Gradients 梯度计算

这里最困难的是计算灵敏度map。一旦我们得到这个了，那我们唯一需要更新的偏置参数β和b就可以轻而易举了（公式（3））。如果下一个卷积层与这个子采样层是全连接的，那么就可以通过BP来计算子采样层的灵敏度maps。

我们需要计算卷积核的梯度，所以我们必须找到输入map中哪个patch对应输出map的哪个像素。这里，就是必须找到当前层的灵敏度map中哪个patch对应与下一层的灵敏度map的给定像素，这样才可以利用公式（1）那样的δ递推，也就是灵敏度反向传播回来。另外，需要乘以输入patch与输出像素之间连接的权值，这个权值实际上就是卷积核的权值（已旋转的）。

$$\boldsymbol{\delta}_j^\ell = f'(\mathbf{u}_j^\ell) \circ \operatorname{conv2}(\boldsymbol{\delta}_j^{\ell+1}, \operatorname{rot180}(\mathbf{k}_j^{\ell+1}), \text{'full'})$$

在这之前，我们需要先将核旋转一下，让卷积函数可以实施互相关计算。另外，我们需要对卷积边界进行处理，但在Matlab里面，就比较容易处理。Matlab中全卷积会对缺少的输入像素补0 。

到这里，我们就可以对b和β计算梯度了。首先，加性基b的计算和上面卷积层的一样，对灵敏度map中所有元素加起来就可以了：

$$\frac{\partial E}{\partial b_j} = \sum_{u,v} (\boldsymbol{\delta}_j^\ell)_{uv}.$$

而对于乘性偏置β，因为涉及到了在前向传播过程中下采样map的计算，所以我们最好在前向的过程中保存好这些maps，这样在反向的计算中就不用重新计算了。我们定义：

$$\mathbf{d}_j^\ell = \operatorname{down}(\mathbf{x}_j^{\ell-1}).$$

这样，对β的梯度就可以用下面的方式计算：

$$\frac{\partial E}{\partial \beta_j} = \sum_{u,v} (\boldsymbol{\delta}_j^\ell \circ \mathbf{d}_j^\ell)_{uv}.$$

3.3、Learning Combinations of Feature Maps 学习特征map的组合

大部分时候，通过卷积多个输入maps，然后再对这些卷积值求和得到一个输出map，这样的效果往往是比较好的。在一些文献中，一般是人工选择哪些输入maps去组合得到一个输出map。但我们这里尝试去让CNN在训练的过程中学习这些组合，也就是让网络自己学习挑选哪些输入maps来计算得到输出map才是最好的。我们用αij表示在得到第j个输出map的其中第i个输入map的权值或者贡献。这样，第j个输出map可以表示为：

$$\mathbf{x}_j^\ell = f\left(\sum_{i=1}^{N_{in}} \alpha_{ij}(\mathbf{x}_i^{\ell-1} * \mathbf{k}_i^\ell) + b_j^\ell\right)$$

需要满足约束：

$$\sum_i \alpha_{ij} = 1, \quad \text{and} \quad 0 \le \alpha_{ij} \le 1.$$

这些对变量αij的约束可以通过将变量αij表示为一个组无约束的隐含权值c_{ij}的softmax函数来加强。（因为softmax的因变量是自变量的指数函数，他们的变化率会不同）。

$$\alpha_{ij} = \frac{\exp(c_{ij})}{\sum_k \exp(c_{kj})}.$$

因为对于一个固定的j来说，每组权值c_{ij}都是和其他组的权值独立的，所以为了方面描述，我们把下标j去掉，只考虑一个map的更新，其他map的更新是一样的过程，只是map的索引j不同而已。

Softmax函数的导数表示为：

$$\frac{\partial \alpha_k}{\partial c_i} = \delta_{ki}\alpha_i - \alpha_i\alpha_k$$

这里的δ是Kronecker delta。对于误差对于第l层变量αi的导数为：

$$\frac{\partial E}{\partial \alpha_i} = \frac{\partial E}{\partial u^\ell}\frac{\partial u^\ell}{\partial \alpha_i} = \sum_{u,v}\left(\boldsymbol{\delta}^\ell \circ (\mathbf{x}_i^{\ell-1} * \mathbf{k}_i^\ell)\right)_{uv}.$$

最后就可以通过链式规则去求得代价函数关于权值c_i的偏导数了：

$$\begin{aligned}\frac{\partial E}{\partial c_i} &= \sum_k \frac{\partial E}{\partial \alpha_k}\frac{\partial \alpha_k}{\partial c_i} \\ &= \alpha_i\left(\frac{\partial E}{\partial \alpha_i} - \sum_k \frac{\partial E}{\partial \alpha_k}\alpha_k\right).\end{aligned}$$

3.3.1、Enforcing Sparse Combinations 加强稀疏性组合

为了限制αi是稀疏的，也就是限制一个输出map只与某些而不是全部的输入maps相连。我们在整体代价函数里增加稀疏约束项Ω(α)。对于单个样本，重写代价函数为：

$$\tilde{E}^n = E^n + \lambda \sum_{i,j} |(\boldsymbol{\alpha})_{ij}|$$

然后寻找这个规则化约束项对权值ci求导的贡献。规则化项Ω(α)对αi求导是：

$$\frac{\partial \Omega}{\partial \alpha_i} = \lambda \operatorname{sign}(\alpha_i)$$

然后，通过链式法则，对ci的求导是：

$$\frac{\partial \Omega}{\partial c_i} = \sum_k \frac{\partial \Omega}{\partial \alpha_k} \frac{\partial \alpha_k}{\partial c_i}$$

$$= \lambda \left(|\alpha_i| - \alpha_i \sum_k |\alpha_k| \right).$$

所以，权值ci最后的梯度是：

$$\frac{\partial \tilde{E}^n}{\partial c_i} = \frac{\partial E^n}{\partial c_i} + \frac{\partial \Omega}{\partial c_i}.$$

3.4、Making it Fast with MATLAB

CNN的训练主要是在卷积层和子采样层的交互上，其主要的计算瓶颈是：

1）前向传播过程：下采样每个卷积层的maps；

2）反向传播过程：上采样高层子采样层的灵敏度map，以匹配底层的卷积层输出maps的大小；

3）sigmoid的运用和求导。

对于第一和第二个问题，我们考虑的是如何用Matlab内置的图像处理函数去实现上采样和下采样的操作。对于上采样，imresize函数可以搞定，但需要很大的开销。一个比较快速的版本是使用Kronecker乘积函数kron。通过一个全一矩阵ones来和我们需要上采样的矩阵进行Kronecker乘积，就可以实现上采样的效果。对于前向传播过程中的下采样，imresize并没有提供在缩小图像的过程中还计算nxn块内像素的和的功能，所以没法用。一个比较好和快速的方法是用一个全一的卷积核来卷积图像，然后简单的通过标准的索引方法来采样最后卷积结果。例如，如果下采样的域是2x2的，那么我们可以用2x2的元素全是1的卷积核来卷积图像。然后再卷积后的图像中，我们每个2个点采集一次数据，y=x(1:2:end,1:2:end)，这样就可以得到了两倍下采样，同时执行求和的效果。

对于第三个问题，实际上有些人以为Matlab中对sigmoid函数进行inline的定义会更快，其实不然，Matlab与C/C++等等语言不一样，Matlab的inline反而比普通的函数定义更非时间。所以，我们可以直接在代码中使用计算sigmoid函数及其导数的真实代码。

上一篇 Deep Learning论文笔记之（三）单层非监督学习网络分析
下一篇 Deep Learning论文笔记之（五）CNN卷积神经网络代码理解

我的同类文章

计算机视觉（72） Deep Learning（17） 机器学习（45）

- 图像卷积与滤波的一些知识点 2015-10-12 阅读 14276
- 人脸识别之特征脸方法（Eig… 2015-04-25 阅读 15078
- 关于计算机视觉（随谈） 2014-08-17 阅读 15172
- 基于meanshift的手势跟踪… 2014-01-06 阅读 14106
- 基于感知哈希算法的视觉目… 2013-12-21 阅读 23006
- 时空上下文视觉跟踪（STC… 2013-11-22 阅读 33418
- 最简单的目标跟踪（模版匹… 2013-10-28 阅读 27674
- 图像分割之（六）交叉视觉… 2013-10-28 阅读 7464
- zigzag模式提取矩阵元素 2013-10-28 阅读 5466
- Deep Learning源代码收集-… 2013-09-22 阅读 55568

更多文章

参考知识库

算法与数据结构知识库

1038 关注 | 2080 收录

大型网站架构知识库

1700 关注 | 532 收录

猜你在找

查看评论

32楼 anan1205 2016-03-15 09:45发表

请教一下，博主，公式（1）也就是英文论文的公式（4）是怎么推导出来的呀，期待您的回复，谢谢！

Re: qiusuoxiaozi 2016-04-26 19:19发表

回复anan1205：认真看看这个
http://ufldl.stanford.edu/wiki/index.php/%E5%8F%8D%E5%90%91%E4%BC%A0%E5%AF%BC%E7%AE%97%E6%

31楼 i_1123 2016-03-02 22:56发表

前辈，对于CNN反向传播中kij梯度表达式的解释是翻译的原文，很难理解，我在这里有个疑问，为什么关于kij梯度的表达式中，灵敏度需要转180度？如果前辈你理解了请不吝赐教

Re: i_1123 2016-03-02 23:08发表

回复i_1123：不好意思，我又看了一遍，理解了！

30楼 wykqhyjs 2016-02-07 13:50发表

我们先对delta灵敏度map进行旋转，这样就可以进行互相关计算，而不是卷积

这句话，我不是很理解，我们的目的 到底 是完成 互相关，还是卷积呢？ 你表达的意思是？ 真心求教，感激无限

29楼 wykqhyjs 2016-02-03 16:09发表

楼主，我学习机器学习2年了，对你说的“神来之笔”这几个字有疑惑，能解释下吗？感激不尽！

28楼 qq_31958941 2015-12-27 16:54发表

本来就难懂，看完更不懂了

27楼 Jay_Sherry 2015-12-23 21:22发表

老师您好，有个问题咨询您一下。随着网络层数的增加，使用BP算法反向调参时会出现梯度弥散的问题。那卷积神经网络解决了梯度弥散问题了吗？？？

Re: 朝朝与暮暮 2016-01-03 21:29发表

回复Jay_Sherry：能哦

26楼 土洋葱 2015-11-05 15:36发表

你好，博主.

公式2怎么来的？

也就是下面这句话“对于第l层，误差对于该层每一个权值（组合为矩阵）的导数是该层的输入（等于上一层的输出）与该层的灵敏度（该层每个神经元的δ组合成一个向量的形式）的叉乘”，怎么理解呢？

麻烦讲解一下..

Re: 迷雾forest 2015-11-25 16:08发表

回复土洋葱：这里的确应该详细说说，将误差函数等价成为“向量转置乘以向量”的形式，然后根据矩阵求导法则和链式法则，就能够得到这样的形式。其实，如果分别对每个标量求导，得到一批等式之后再矩阵化，会更清晰，但是这样的推导量比较大。

25楼 qq_23225871 2015-09-27 10:32发表

楼主你好，我最近在训练一个cnn网络图片去模糊的，但是就是不知道什么原因，网络是收敛的，但是处理的结果还不如不用网络处理的好。

24楼 hdc 2015-07-10 10:29发表

公式2有个地方翻译错了，原文是outer product，是外积，但是作者翻译成了叉乘

Re: 土洋葱 2015-11-05 15:40发表

回复hdc：你好，公式2为什么是这样呢，还请讲解一下...

23楼 Journey-Dream 2015-06-18 22:35发表

博主你好，请教一个问题：
在学习多个feature maps组合的部分【3.3节】，公式里K的下标应该是j才说的过去啊？我看了原文，也是Ki。无法理解，请博主指点！

Re: lipeng19930407 2016-04-08 10:36发表

回复Journey-Dream：同问3.3节中，对softmax的导数是怎么得到的

22楼 jidongsong 2015-01-30 10:20发表

楼主，你好，我有一个关于CNN的问题，一直想不明白，想请教你一下，CNN的网络结构是：input(20x20x1)-conv(16x16x16 ReLU)-Max-pooling(8x8x16)-conv(4x4x64 ReLU)-Max-pooling(2x2x64)-Full-connect(2x2x64)-Softmax(3754x1) Softmax层由于ReLU输出值过大导致数据溢出，这个问题该如何解决？

Re: mobery9 2015-03-09 12:51发表

回复jidongsong：这个问题UFLDL教程关于Softmax练习部分有技巧提醒。很简单，为softmax计算时分子分母指数同时减掉一个常数就行，这个常数一般是指数中最大的一个。

21楼 mobery9 2015-01-19 15:50发表

很不错。

但是有几点不是特别明白，希望能得到您的回答：
1）文章多次提到像“为了求这个灵敏度我们就需要先对下一层的节点（连接到当前层l的感兴趣节点的第l+1层的节点）的灵敏度求和（得到δl+1）”这样的，求和得出δl+1。此处是说对于当前l层的某个节点，需要投入反向传播的δl+1是l+1层所有与该节点连接的l+1层节点灵敏度之和么？下面多次提到的“求和”均是这个意思吧？
2）关于3.1.1节相关性计算反向传播的理论依据不是特别明白。另外，此处描述的是参数共享的卷积层，在目前某些网络中会采用patch共享（如2*2）或者完全不共享的方式，那又应该如何计算灵敏度呢？
3）3.1.1节所提及的up上采样运算，在使用max-pooling时有没有什么特别的处理呢？

Re: mobery9 2015-01-28 10:55发表

回复mobery9：最近又看了原文，关于上述问题有一些了解了。

20楼 虎子王 2014-11-26 16:08发表

sigmoid将输出压缩到[0, 1]，所以最后的输出平均值一般趋于0 。与原文表述不同：therefore while the output of the hyperbolic tangent function will typically be near zero, the outputs of a sigmoid will be non-zero on average.应该是sigmoid函数非趋于0.

19楼 波小妞 2014-09-11 08:51发表

请问这里的” 神经元的基“，这个基指的是什么

Re: wykqhyjs 2014-11-09 14:47发表

回复波小妞：这里说的神来之笔没看懂，楼主能告诉我吗

18楼 IT小飞飞 2014-08-19 15:24发表

第一次卷积的时候，我怎么得到多个feature map？

Re: 在你左边001 2015-07-17 09:17发表

回复IT小飞飞：使用卷积模型的个数就是feature map的个数

17楼 宋康宁 2014-08-13 15:42发表

感谢分享，不过作者都是字面翻译过来的，如果作者能够结合自身丰富的经验将其讲解的更通俗易懂些就更好了。。。

16楼 pengji256 2014-07-02 16:31发表

您好，3.1.1这句“层l中的一个map的每个节点只与l+1层中相应map的一个节点连接。”
可是我看matlab的toolbox里面，cnnff()计算时，用了两个循环遍历卷积层的输出map和采样层的输出map，也就是说l层的每个结点会与l+1层的outmapsize个结点连接。这是实现上的差异，还是我理解的问题呢？
代码：
for j = 1 : net.layers{l}.outputmaps
for i = 1 : inputmaps
计算卷积
end
...
end

15楼 空穴来风 2014-06-11 14:31发表

您好，我看到下面这段话就看不懂了。请问“采样层”，“卷积层”指的是什么？“层l中的一个map的每个节点只与l+1层中相应map的一个节点连接”是什么意思？这个层l指的是pooling层吧？

“然而，因为下采样的存在，采样层的一个像素（神经元节点）对应的灵敏度δ对应于卷积层（上一层）的输出map的一块像素（采样窗口大小）。因此，层l中的一个map的每个节点只与l+1层中相应map的一个节点连接。”

14楼 szbwk1 2014-06-10 11:53发表

博主，或者各位同学，公式一到底是怎么推出来的啊

Re: 打湿井盖 2015-01-15 10:44发表

回复szbwk1：真实值减去预测值

13楼 Novice_wen 2014-06-09 23:28发表

博主，你好，看了你的这篇文章很受启发，但是里面有些问题，实在没想明白，请教下：
对于公式3，其中u,v到底是什么意思啊?为什么误差关于偏执的偏导数是当前层灵敏度的求和？？？特别是这个u,v是要把这个灵敏度MAP的值全部加起来??

Re: junedwx_123 2016-03-26 16:45发表

回复Novice_wen：你好，这个问题我也不了解，想请教一下

Re: 土洋葱 2015-11-12 21:59发表

回复Novice_wen：你好，我这边也无法理解，你能讲解下么.

12楼 始于长虹 2014-06-04 09:58发表

博主，这篇论文有相应的代码吗？

11楼 良仁 2014-05-21 14:35发表

Q1：字符识别里的输入特征层选择Lecun已经给出来了。他是不是也是通过学习combinations of feature maps中的alpha来得到这些输入特征层的。
Q2：整个网络训练的时候输出层是不是和上一层所有的输入层相链接，Nin等于上一层数，前向传播之后，反向传播时，更新alpha是不是和k,b同时进行的？最后选择系数比较大的alpha对应的输入层。是么？

10楼 liwei_yezi 2014-03-05 20:21发表

感觉BP最关键部分——反向传播可以反向的原因，在这里也没讲清楚，只是把文章翻译了一下，如果作者了解，希望可以补上，代表广大读者谢过楼主的奉献精神！！！

Re: 良仁 2014-05-21 14:40发表

回复liwei_yezi：反向传播其实就是所谓的敏感度反向传播，由于无法对隐藏的权值进行微分，因此使用链式微分，

得到的结果恰好包含上一层的敏感度，因此敏感度要反向传播。

9楼 hualiandie1234 2014-01-10 10:10发表

楼主你好，请问mnist数据库可以换成其他的吗？比如语音文件，需要特定的什么格式吗？谢谢您！

Re: zouxy09 2014-01-10 19:48发表

回复hualiandie1234：CNNs接受的是二维的图像输入。视频的话有拓展的3D CNN。因为语音是一维信号，所以不能直接用CNN。个人感觉应该用的是延时神经网络TDNN。CNNs也是是受早期的延时神经网络（TDNN）的启发。延时神经网络通过在时间维度上共享权值降低学习复杂度，适用于语音和时间序列信号的处理。

Re: xbgd1990 2014-04-23 17:09发表

回复zouxy09：楼主，麻烦问一下3D CNN有没有公开的代码，有的话能给个链接吗？

Re: hualiandie1234 2014-01-15 15:36发表

回复zouxy09：可以用DBN吗？最近实验得到了一些csv格式的语音特征文件，就是一个语音特征矩阵，可以直接用DBN吗？谢谢。

Re: zouxy09 2014-01-15 23:10发表

回复hualiandie1234：我没了解过DL在语音方面的应用，不过目前是存在不少的，你可以去搜索写论文看看。然后我感觉DBN是ok的。

Re: hualiandie1234 2014-01-18 15:32发表

回复zouxy09：嗯，好的，谢谢。

8楼 chocolate9624 2013-12-31 11:22发表

请问博主一个问题，在Enforcing Sparse Combinations 加强稀疏性组合 这块，aij的和本身就是一，那个正则化的项的和一直都是个常数，这样的情况下如何做到sparse的呢？

7楼 tornadomeet 2013-12-08 09:02发表

不错不错

6楼 耕农 2013-12-02 20:03发表

楼主，你好！一直不明白卷积神经网络的样本标签是怎么找的，就是求误差时的那个Tn-Xn中的Tn

Re: zouxy09 2013-12-02 21:30发表

回复耕农：监督训练，样本便签是给定的。换句话说，你给的是一堆（x_i, y_i）的样本（y_i就是Tn了）去训练卷积神经网络。例如你给30张车的照片，和三十张人的图片，让CNN训练去区分车和人，那假设车的标签是1，人的标签是0，也就是他们的y_i。当然了，这也只是标签表达的一种方法，还是输出10或者01的方法，也就是一个神经元输出1，其他输出0. 总之，监督训练，样本标签是我们给的。

5楼 wangzhebupt 2013-11-11 11:08发表

严重同意楼主看论文整理的说法。。。表示看完的论文木有办法记住啊。。。幸亏做了些笔记，不过楼主有没有试过翻译论文的方法，也许会帮助记忆哦

Re: zouxy09 2013-11-11 23:21发表

回复wangzhebupt：嗯嗯，我其实也算翻译的，然后加上自己的某些理解而已。

4楼 wallyell 2013-11-07 14:45发表

楼主最好能用自己的话去理解这篇文字，也会让大家受益匪浅。看到的都是大段大段的原文翻译.....

3楼 wantong1017 2013-10-22 14:29发表

一直不明白权值和bias的更新用的是高层map所有神经元的delta的求和而不是求平均的原因，不知道怎么推导的

Re: wangpengwei2 2014-03-17 20:38发表

回复wantong1017：我这里也不是很明白，note中更本没讲为什么

2楼 井底之蛙-hzq 2013-10-05 11:20发表

亲，我正在用您的日志学习卷积神经网络。
我写过层与层间全连接的人工神经网络（C++），是按照某个论文的思路写的。
在ann中，每个神经元有一个bias，层间的每两个神经元都有一个连接权重w。
看了您的文章，不太明白在cnn中，卷积层和子采样层的bias和连接权重w是怎样设置的？

Re: 良仁 2014-05-21 14:43发表

回复井底之蛙-hzq：初始时随机化，然后用梯度下降法或BFGS更新这些参数。

Re: wantong1017 2013-10-22 14:28发表

回复井底之蛙-hzq：按这篇论文的说法，子采样层一个map应该只有一个w和bias吧但是我也看过采样层只采样不用f()再输出一下的用法，这样就没有w和bias

Re: 井底之蛙-hzq 2013-11-02 19:57发表

回复wantong1017：权值其实就是卷积核，卷积核是3*3的就有9个w，5*5的就有25个w

1楼 DeepSea 2013-08-19 15:21发表

你好，我最近也看了这篇文章。在3.2.1节中介绍的：如果下一个卷积层与这个子采样层是全连接的，那么就可以通过BP来计算子采样层的灵敏度maps。我想问一下，此处的全连接是什么意思？是这个子采样层的所有map都是下一个卷积层每个map的输入的意思吗，如果是这样是不是计算delta的时候，就需要利用下一个卷积层的所有map的delta来计算该子采样层的delta，最后求和？