手写数字识别

卷积神经网络的一些应用

WyEhNf

上海交通大学 2025级ACM班

2025 年 10 月 17 日

CNN的工作模式

1

基本结构

结构

CNN网络的主要结构如下：

- 输入层
- 卷积层
- 池化层
- 输出层

运行流程

在单次训练中，CNN网络主要完成以下流程：

- 接受输入向量
- 前向传播
- 计算梯度
- 反向传播

2

各层实现

卷积神经元

考虑一般的神经元，对于输入向量X,其转换函数为：

$$f(x) = \sum x_i w_i + b$$

而卷积神经元相当于将向量间的内积变为权值矩阵与输入向量/矩阵的卷积，即对于输入向量X,其转换函数为：

$$f(x) = W^T * X + b$$

这对二维图像明显有这个更好的特征反映能力。

激活函数

常见的激活函数有两种：

Relu函数

$$f(x) = x[x > 0]$$

Sigmoid函数

$$f(x) = \frac{1}{1 + \mathbf{exp}(-x)}$$

激活函数的目的是将卷积这一线性操作非线性化。

卷积核的移动与边界填充

注意到卷积核在移动的时候对边界信息的获取是弱的，一般由以下填充模式：

Valid模式

不进行额外填充，$r*c$的矩阵经卷积后输出为$(r-n+1)*(c-m+1)$的矩阵。

Full模式

充分填充使得卷积核中心经过输入矩阵每个位置，$r*c$的矩阵经卷积后输出为$(r+n-1)*(c+m-1)$的矩阵。

Same模式

适当填充使得输出矩阵大小不变。

池化层

池化层的目的是降低进入后续层的数据规模，仅保留每个局部的整体特征。池化窗口的移动和卷积核类似，但不存在填充的问题。一般可以取窗口内平均或者max/min。

全连接层

全连接层相当于对所得到的数据的总结，可以看作是一个收束的神经元,将数据数量约束到最终期望的规模，
特别的，全连接层的激活函数需要将数据转化为可供与标签比较的量。这个激活后的输出即为最终输出。

二

CNN在数字识别中的应用

3

网络搭建

5层CNN网络

MINST数据集中摹刻数字的灰度矩阵是28*28的，最终对每张图片的判断应该是一个其为0-9的概率。

- 卷积层（5*5,6核）输出6张24*24的图片（选用Relu函数激活）
- 池化层（2*2,max）输出6张12*12的图片
- 卷积层（5*5,12核）输出12张8*8的图片
- 池化层（2*2,max）输出12张4*4的图片
- 全连接层接受一个192维向量并转换为一个10维的描述概率的向量,并评估误差

全连接层的激活函数

这里采用softmax函数，这个函数可以将一组数据归一化到$(0,1)$，且恰好能表示概率分布。

Softmax函数

$$\mathbf{Softmax(y_i)} = \frac{\mathbf{exp(y_i)}}{\sum \mathbf{exp(y_j)}}$$

误差评估

每次评估误差后根据误差修正卷积核的参量。

交叉熵损失函数

$$E = -\sum p_i ln Y_i$$

其中p_i是真实概率，Y_i是Softmax函数所得。

4

误反向传播

输出层-Softmax函数的梯度

$$\frac{\partial Y_j}{\partial y_i} = \frac{\mathbf{exp}(y_i)(\sum \mathbf{exp}(y_j) - \mathbf{exp}(y_i))}{(\sum \mathbf{exp}(y_j))^2} = Y_i(1 - Y_i) \quad (i = j)$$

$$\frac{\partial Y_j}{\partial y_i} = \frac{\mathbf{exp}(y_j)(0 - \mathbf{exp}(y_i))}{(\sum \mathbf{exp}(y_j))^2} = -Y_i Y_j \quad (i \neq j)$$

$$grad_i = \frac{\partial E}{\partial y_i} = \frac{\partial(\sum E_j)}{\partial y_i} = \sum \frac{\partial E_j}{\partial Y_j} \cdot \frac{\partial Y_j}{\partial y_i} = -\frac{p_i}{Y_i} Y_i(1 - Y_i) + \sum_{j \neq i} -\frac{p_j}{Y_j} Y_j Y_i$$

结合$\sum p_i = 1$,可以得到:

$$grad_i = Y_i - p_i$$

输出层-全连接神经元的梯度

方便起见，此后我们把上一层（也就是正向传播的下一层）所得梯度成为pre_i

$$grad_i = \frac{\partial E}{\partial y_i} = \sum_{j=0}^{9} \frac{\partial E_j}{\partial y_j} \cdot \frac{\partial y_j}{\partial x_i} = \sum_{j=0}^{9} pre_j \cdot w_{ij}$$

其中w_{ij}为全输出层神经元将x_i贡献到y_j的权重。

池化层

在全连接层中我们得到了12个4*4矩阵的梯度，那么经过池化层的反向传播后，矩阵扩展为原本的8*8的大小. 由于这里池化采取的是取max，所以反向传播的时候只要维持好这个性质就可以了。
一种可行的构造是维持池化窗口中最大值位置上的数不变，其余位置填上0。

卷积层

显然Relu函数的导数为:$Relu'(x) = [x > 0]$。
结合矩阵卷积的求导方法，有：

$$grad_i = \sum (pre_i \cdot Relu'(y_i)) * w'_{ij}$$

其中w'_{ij}为将输入的第j张图片贡献到输出第i张图片的权值矩阵的旋转矩阵。前两层的梯度推导是类似的。

参数更新

我们已经求出每一层的输出y_i关于E的梯度。鉴于每一层的转换都形如：

$$y_i = w_{ij}x_j + b_i$$

显然有：

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_i} \cdot x_j$$

$$\frac{\partial E}{\partial b_i} = \frac{\partial E}{\partial y_i}$$

据此我们可以更新任意参量X:

$$X' = X - \alpha \cdot grad_X$$

其中α为学习率。

5

实际训练

训练参数与结果

- 学习率：初始设置为0.03，随着训练的进行逐渐递减到0.001，避免在训练后期步长过大跳过最优点
- 训练集：MINST训练集提供了60000份样本，出于时间考量，循环训练5次。
- 测试集：10000份与训练集格式相同的样本

训练结果：成功率约为98.2%

6

不同模型的比较

卷积核的数量

- 双卷积层（6核，12核）：成功率98.2%
- 双卷积层（12核，16核）：成功率98.5%
- 双卷积层（16核，32核）：成功率98.7%

卷积层数

- 双卷积层（6核，12核）：成功率98.2%
- 三卷积层（6核，12核，16核（1*1））：成功率96.5%
- 三卷积层（16核，32核，32核（3*3）：成功率97.8%

对数据的重复训练

```
success: 0.9798
success: 0.9788
success: 0.9778
success: 0.9779
success: 0.9779
success: 0.9786
success: 0.9764
success: 0.9768
success: 0.9787
success: 0.979
success: 0.9782
success: 0.9788
success: 0.9776
```

(a) Without Dropout

```
success: 0.9772
success: 0.9738
success: 0.9736
success: 0.9731
success: 0.9748
success: 0.9724
success: 0.9697
success: 0.9715
success: 0.9688
success: 0.9711
success: 0.9725
success: 0.969
success: 0.9708
```

(b) With Dropout

可以看到由于数据的复杂性较低，dropout层对减缓过拟合的作用甚至不如其丢失数据对准确性造成的影响大

参考资料

资料链接：

- 大话卷积神经网络
- weijiakai's blog（C++/OpenCV实现CNN系列）

谢 谢