
hjimce
博客专家

深度学习（二十二）Dropout浅层理解与实现

发表于2015/12/27 17:30:41 8816人阅读
分类： 深度学习

Dropout浅层理解与实现

原文地址：http://blog.csdn.net/hjimce/article/details/50413257

作者：hjimce

一、相关工作

本来今天是要搞《Maxout Networks》和《Network In Network》的，结果发现maxout和dropout有点类似，所以就对dropout做一下相关的总结，了解一下其代码层面的实现。

Dropout是2012年深度学习视觉领域的开山之作paper：《ImageNet Classification with Deep Convolutional》所提到的算法，用于防止过拟合。在我刚入门深度学习，搞视觉的时候，就有所耳闻，当时只知道它是为了防止过拟合。记得[illegible]前啥也不懂，看到《ImageNet Classification with

感觉自己模仿设计出来的网络，感觉精度都好烂，然后也不会分析网络设计哪些合理，哪些不合理。当时要么就是模仿别人，要么就是直接用别人的网络，被领导鄙视了一番……还是不啰嗦了，说多了都是泪。

网上都说dropout是让某些神经元以一定的概率不工作，但是具体代码怎么实现？原理又是什么，还是有点迷糊，所以就大体扫描了文献：《Improving neural networks by preventing co-adaptation of feature detectors》、《Improving Neural Networks with Dropout》、《Dropout: A Simple Way to Prevent Neural Networks from Overtting》，不过感觉看完以后，还是收获不是很大。下面是我的学习笔记，因为看的不是很细，也没有深入理解，有些地方可能有错，如有错误还请指出。

二、算法概述

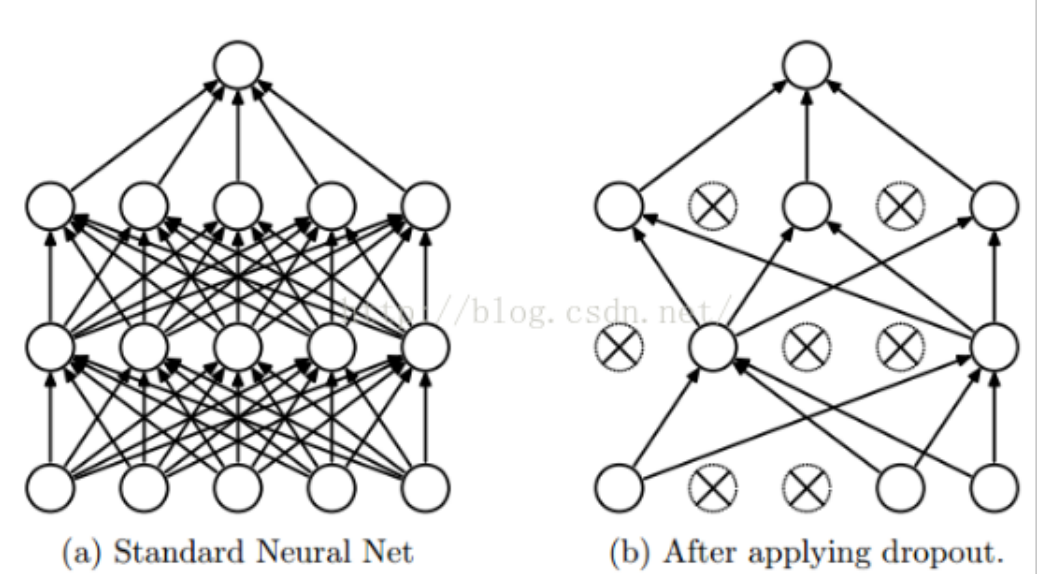
我们知道如果要训练一个大型的网络，训练数据很少的话，那么很容易引起过拟合(也就是在测试集上的精度很低)，可能我们会想到用L2正则化、或者减小网络规模。然而深度学习领域大神Hinton，在2012年文献：《Improving neural networks by preventing co-adaptation of feature detectors》提出了，在每次训练的时候，让一半的特征检测器停过工作，这样可以提高网络的泛化能力，Hinton又把它称之为dropout。

Hinton认为过拟合，可以通过阻止某些特征的协同作用来缓解。在每次训练的时候，每个神经元有百分之50的几率被移除，这样可以让一个神经元的出现不应该依赖于另外一个神经元。

另外，我们可以把dropout理解为 模型平均。 假

设我们要实现一个图片分类任务，我们设计出了100000个网络，这100000个网络，我们可以设计得各不相同，然后我们对这100000个网络进行训练，训练完后我们采用平均的方法，进行预测，这样肯定可以提高网络的泛化能力，或者说可以防止过拟合，因为这100000个网络，它们各不相同，可以提高网络的稳定性。而所谓的dropout我们可以这么理解，这n个网络，它们权值共享，并且具有相同的网络层数(这样可以大大减小计算量)。我们每次dropout后，网络模型都可以看成是整个网络的子网络。(需要注意的是如果采用dropout，训练时间大大延长，但是对测试阶段没影响)。

啰嗦了这么多，那么到底是怎么实现的？Dropout说的简单一点就是我们让在前向传导的时候，让某个神经元的激活值以一定的概率p，让其停止工作，示意图如下：



(a) Standard Neural Net　　(b) After applying dropout.

左边是原来的神经网络，右边是采用Dropout后的网络。这个说是这么说，但是具体代码层面是怎么实现的？怎么让某个神经元以一定的概率停止工作？这个我想很多人还不是很了解，代码层面

的实现方法，下面就讲解一下其代码层面的实现。以前我们网络的计算公式是：

$$
\begin{aligned}
z_i^{(l+1)} &= \mathbf{w}_i^{(l+1)}\mathbf{y}^l + b_i^{(l+1)}, \\
y_i^{(l+1)} &= f(z_i^{(l+1)}),
\end{aligned}
$$

采用dropout后计算公式就变成了：

$$
\begin{aligned}
r_j^{(l)} &\sim \text{Bernoulli}(p), \\
\widetilde{\mathbf{y}}^{(l)} &= \mathbf{r}^{(l)} * \mathbf{y}^{(l)}, \\
z_i^{(l+1)} &= \mathbf{w}_i^{(l+1)}\widetilde{\mathbf{y}}^l + b_i^{(l+1)}, \\
y_i^{(l+1)} &= f(z_i^{(l+1)}).
\end{aligned}
$$

上面公式中Bernoulli函数，是为了以概率p，随机生成一个0、1的向量。

算法实现概述：

1、其实Dropout很容易实现，源码只需要几句话就可以搞定了，让某个神经元以概率p，停止工作，其实就是让它的激活值以概率p变为0。比如我们某一层网络神经元的个数为1000个，其激活值为x1，x2……x1000，我们dropout比率选择0.4，那么这一层神经元经过drop后，x1……x1000神经元其中会有大约400个的值被置为0。

2、经过上面屏蔽掉某些神经元，使其激活值为0以后，我们还需要对向量x1……x1000进行rescale，也就是乘以1/(1-p)。如果你在训练的时候，经过置0后，没有对x1……x1000进行rescale，那么你在测试的时候，就需要对权重进行rescale：

$$W_{test}^{(l)} = pW^{(l)}$$

问题来了，上面为什么经过dropout需要进行rescale？查找了相关的文献，都没找到比较合理的解释，后面再结合源码说一下我对这个的见解。

所以在测试阶段：如果你既不想在训练的时候，对x进行放大，也不愿意在测试的时候，对权重进行缩小(乘以概率p)。那么你可以测试n次，这n次都采用了dropout，然后对预测结果取平均值，这样当n趋近于无穷大的时候，就是我们需要的结果了（也就是说你可以采用train阶段一模一样的代码，包含了dropout在里面，然后前向传导很多次，比如1000000次，然后对着1000000个结果取平均值）。

三、源码实现

下面我引用keras的dropout实现源码进行讲解，keras开源项目github地址为：

https://github.com/fchollet/keras/tree/master/keras。其dropout所在的文件为：

https://github.com/fchollet/keras/blob/master/keras/backend/theano_backend.py，dropout实现函数如下：

```
#dropout函数的实现
def dropout(x, level):
    if level < 0. or level >= 1:#level是概率值，必须在0~1之间
        raise Exception('Dropout level must be in interval [0, 1[.')
    retain_prob = 1. - level
  #我们通过binomial函数，生成与x一样的维数向量。binomial函数就像抛硬币一样，我们可以把每个神经元当做抛硬币一样
    #硬币 正面的概率为p，n表示每个神经元试验的次数
    #因为我们每个神经元只需要抛一次就可以了所以n=1，size参数是我们有多少个硬币。
```

```
sample=np.random.binomial(n=1,p=retain_prob,size=x.shape)#即将生成一个0、1分布的向量，0表示这个神经元被屏蔽，不工作了，也就是dropout了
        print sample
        x *=sample#0、1与x相乘，我们就可以屏蔽某些神经元，让它们的值变为0
        print x
        x /= retain_prob

        return x
#对dropout的测试，大家可以跑一下上面的函数，了解一个输入x向量，经过dropout的结果
x=np.asarray([1,2,3,4,5,6,7,8,9,10],dtype=np.float32)
dropout(x,0.4)</span>
```

函数中，x是本层网络的激活值。Level就是dropout就是每个神经元要被丢弃的概率。不过对于dropout后，为什么需要进行rescale：

```
x /= retain_prob
```

有的人解释有点像归一化一样，就是保证网络的每一层在训练阶段和测试阶段数据分布相同。我查找了很多文献，都没找到比较合理的解释，除了在文献《Regularization of Neural Networks using DropConnect》稍微解释了一下，其它好像都没看到相关的理论解释。

我们前面说过，其实Dropout是类似于平均网络模型。我们可以这么理解，我们在训练阶段训练了1000个网络，每个网络生成的概率为Pi，然后我们在测试阶段的时候，我们肯定要把这1000个网络的输出结果都计算一遍，然后用这1000个输出，乘以各自网络的概率Pi，求得的期望值就是我们最后的平均结果。我们假设，网络模型的输出如下：

$$f(x;\theta,M)$$

M是Dropout中所有的mask集合。所以当我们在

测试阶段的时候，我们就是对M中所有的元素网络，最后所得到的输出，做一个期望：

$$o = \mathbf{E}_M\left[f(x;\theta,M)\right] = \sum_M p(M) f(x;\theta,M)$$

P(M)表示网络各个子网络出现的概率。因为dropout过程中，所有的子网络出现的概率都是相同的，所以。

个人总结：个人感觉除非是大型网络，才采用dropout，不然我感觉自己在一些小型网络上，训练好像很是不爽。之前搞一个比较小的网络，搞人脸特征点定位的时候，因为训练数据不够，怕过拟合，于是就采用dropout，最后感觉好像训练速度好慢，从此就对dropout有了偏见，感觉训练过程一直在波动，很是不爽。

参考文献：

1、《Improving neural networks by preventing co-adaptation of feature detectors》

2、《Improving Neural Networks with Dropout》

3、《Dropout: A Simple Way to Prevent Neural Networks from Overtting》

4、《ImageNet Classification with Deep Convolutional》

**********作者：hjimce 时间：2015.12.20 联系QQ：1393852684 原创文章，转载请保留原文地址、作者等信息**********

上一篇 下一篇

评论（1）

sysu_cis 1楼

博主您好，有一个问题我有些不太理解，就是你在解释rescale的时候，说了这样一句话，“P(M)表示网络各个子网络出现的概率。因为dropout过程中，所有的子网络出现的概率都是相同的，所以。”但是我现在假设p=0.2，然后我的网络只有2个神经元。这个时候子网络有4种情况并且概率应该是不同的把，我认为概率不应该是0.64,0.16,0.16,0.04吗？

2016-08-22 16:35 回复

发表评论

我的热门文章

深度学习（二十九）Batch Normalization 学习笔记

深度学习（十八）基于R-CNN的物体检测

深度学习（十）keras学习笔记

深度学习（四十一）cuda8.0+ubuntu16.04+theano…

深度学习（五）caffe环境搭建

相关博文

Java的深度复制与浅层复制（一）

hjimce算法类博文目录

深度学习与计算机视觉系列(5)_反向传播与它的直观…

深度学习情感分析

深度学习和浅层学习 Deep Learning and Shallow Le…

机器学习的两次浪潮——浅层学习和深度学习

深度学习与计算机视觉系列(7)_神经网络数据预处理…

关于深度学习中Dropout的理解

深度学习与计算机视觉系列(2)_图像分类与KNN

CNN卷积神经网络