
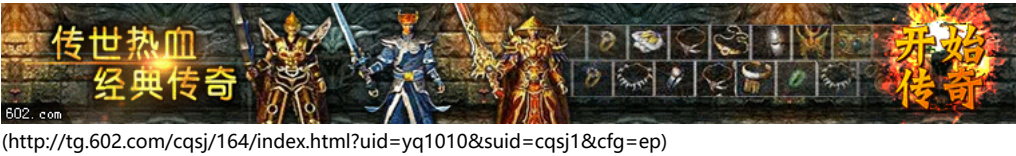


速度快3倍，大小仅1/4，多快好省搭建深度学习模型

 数据派THU 2017-07-05 16:21:49 (/w-DatapiTHU/)



DataPi THU, Share and Study

来源：AI科技大本营
作者：Jacob Gildenblat

本文长度为4300字，建议阅读6分钟
本文对各类剪枝技术进行详解，并以案例的形式来进行实验实操。

一般来说，神经网络层数越深、参数越多，所得出的结果就越精细。但与此同时，问题也来了：越精细，意味着所消耗的计算资源也就越多。这个问题怎么破？这就要靠剪枝技术了。言下之意，把那些对输出结果贡献不大的参数剪掉。这项技术可追溯至深度学习大神Yan LeCun在1990年的研究。

本文除了对各类剪枝技术进行详解，还会以案例的形式来进行实验实操：修剪一个基于VGG-16模型的猫狗分类器。这个案例结果证明，剪枝后的模型在速度上比原来快了近3倍，而文件大小只有原来的1/4。这对于移动设备，速度和大小都极其重要。

这个神奇操作的原理和细节究竟是怎样呢？等不及了吧，让我们开始。



对神经网络进行剪枝这个想法并不新奇，可追溯至1990年（Yan Lecun的工作-<http://yann.lecun.com/exdb/publis/pdf/lecun-90b.pdf>）。其基本的思想是：神经网络的参数众多，但其中有些参数对最终的输出结果贡献不大而显得冗余，剪枝顾名思义，就是要将这些冗余的参数剪掉。

首先，需要根据对最终输出结果的贡献大小来对模型的神经元们排序，然后，舍去那些贡献度低的神经元来，使得模型运行速度更快、模型文件更小。

对于移动设备而言，模型的运行速度和文件大小都是极其重要的。



小编推荐

- NO.1

你知道大肠有多脏吗？
(/Rinawale58/327066)
- NO.2

【观察】恩威道源商城
近年来互联网出现越来越多
(/fcx114-com/32968)
- NO.3

创客黄修源：我带了一
他造了一台互联网电动汽车
(/dongjingguizhuo/3)
- NO.4

【留学】都柏林理工大
都柏林理工大学酒店管理
(/ireland8com/32752)
- NO.5

株洲气温猛降16℃！一
虽然已经过了立秋可株洲
(/zzz4weixin/329125)
- NO.6

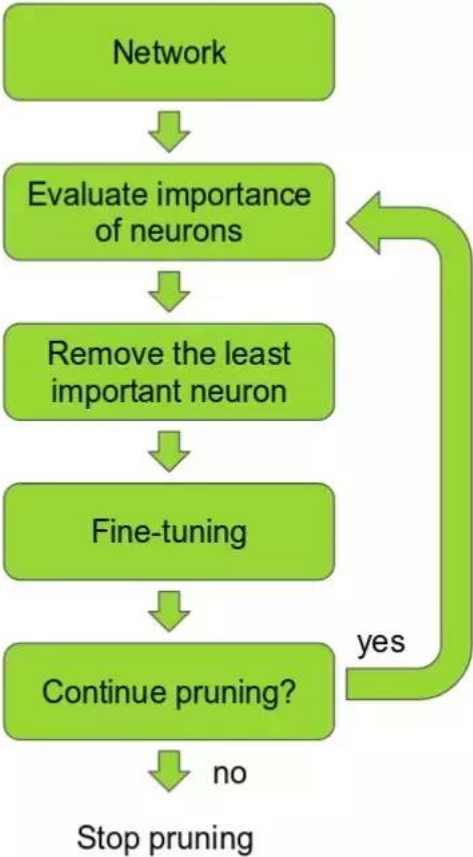
6月30日发生的这个事
6月30日，是2017年上半
(/yanyuelong2014/3)
- NO.7

这狗粮撒的，我猝不及
我先清清面前的狗粮
(/lengtoo/3301084.h)
- NO.8

荡气回肠！一篇文看完
苏州沦陷后，顾公硕曾冒险
(/minguohuashi/328)

贡献度的排序指标可以是神经元的权重参数L1/L2正则化的平均值、激活函数的平均输出值、在验证数据集上不为0的神经元个数或其他指标。剪掉这些贡献度低的神经元，模型的精确度会有子定的损失（当然我们希望损失的越小越好），因此，剪枝后的模型通常需要更多的训练来使其保证一定的性能。这里要注意的是，如果一次性剪枝的神经元过多，可能会导致模型“损坏”太严重而性能太差。

因此，模型的剪枝实际上是一个迭代的过程，这通常称为“迭代式剪枝”，迭代的过程就是剪枝和模型训练两者的交替重复。



via:《Pruning Convolutional Neural Networks for Resource Efficient Inference》 - <https://arxiv.org/abs/1611.06440>

剪枝的方法是不是听起来很棒？那为什么这种方法似乎并不那么为人所知呢？

实际上，目前已经有大量论文在研究神经网络模型的剪枝方法，然而在工业界我们至今没有听说有实际项目采用了这些剪枝方法，这或许会让人感到惊讶。我猜测主要的原因可能有以下几点：

目前对神经元的贡献度进行排序的方法还不够好，导致模型精度损失太多；

方法实现的难度较大；

那些使用了神经网络剪枝技术的人可能把它当作技术机密，而不为外人所知。

因此，我决定自己来实现神经网络的剪枝技术，并实验一下结果如何。

本篇博文将首先回顾一下之前的几种剪枝技术，然后对近期提出的一种方法的实现细节做深入的解析。这之后我们将对一个VGG模型进行调优（fine tune），以用于猫狗图片分类（数据集为Kaggle Dogs vs Cats dataset-<https://www.kaggle.com/c/dogs-vs-cats>），这是迁移学习的一种，在实际应用中也十分常见。

剪枝的目的：提升运行速度 vs 减小模型文件大小

在VGG16模型中，90%的权重参数都在全连接层中，但这些权重参数对模型的最终结果的提升仅为1%。

直到最近为止，大多数的工作都在研究如何对全连接层进行剪枝。对全连接层进行剪枝，这对于减小模型文件的大小非常有效。

接下来我们将重点研究在卷积神经网络中如何对卷积窗口进行剪枝。

剪枝其实还有个好处就是能够减小内存开销。

同时，《Pruning Convolutional Neural Networks for Resource Efficient Inference》(https://arxiv.org/abs/1611.06440)这篇论文指出，随着网络层越深，其剪枝的程度越高。这意味着最后的卷积层被剪枝得最多，导致后面的全连接层的神经元数量大大减少。

对卷积窗口进行剪枝的方式，也可以是减小卷积窗口中的权重参数，或是舍弃卷积窗口的某一维。你也可以丢弃那些稀疏的卷积窗口，但这并不会使模型运行速度有数量级的提升。近期的研究工作提出了对稀疏的卷积窗口“结构化”处理，而不是单纯地完全丢弃。

许多研究工作共同表明了一个重要的结论：尤其在迁移学习领域，通过训练一个更大的神经网络模型，再逐步剪枝得到的小模型取得的结果要比直接训练这样一个小模型的结果好得多。

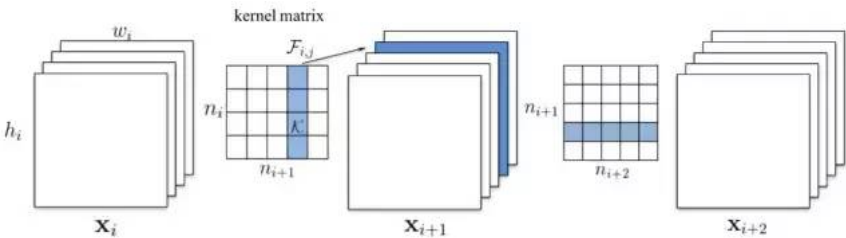
下面让我们简单了解一下几种剪枝技术。

几种剪枝技术

论文1:《Pruning filters for efficient convnets》

https://arxiv.org/abs/1608.08710

这篇论文中，作者提出对卷积层进行完全的剪枝。对第k个卷积输出层进行剪枝，不仅影响当前的卷积层输出，也会影响接下来的网络层，也就是对于之后的网络层没有了原始输入中的第k个。



via:《Pruning filters for efficient convnets》 - https://arxiv.org/abs/1608.08710



(/)

(/n- (/n- (/n- (/n- (/n- (/n- (/n- (/n- (/n- (/n- (/top/)

这篇论文对卷积窗口贡献度排序的方法很简单，所采用的的排序指标为卷积窗口经L1正则化的权重参数。

xinwen/caijing/meishijixingzuc yule/ lishi/ tiyumi/duanzi/yangshsifangh

对卷积窗口剪枝的迭代过程中，每一轮迭代会将全部的卷积窗口进行排序（排序指标为卷积核中L1正则化的权重参数），舍弃排序后指标最低的M个卷积窗口以达到剪枝的目的，然后用剪枝后的卷积窗口进行模型训练，再不断地重复这个过程。

论文2:《Structured Pruning of Deep Convolutional Neural Networks》

<https://arxiv.org/abs/1512.08571>



这篇论文与上篇类似，但在排序上用了更加复杂的方法。论文采用了N个卷积单元过滤器 (Particle Filters)来对相应的N个卷积层进行剪枝操作。

每一个卷积单元会根据其影响模型在验证数据集上的准确率程度而被分配一个分值，分值低的卷积单元会被过滤掉以达到剪枝的目的。卷积单元的剪枝与模型训练是迭代进行的。

由于这种剪枝过程非常耗时，因此实验中使用了很小的验证数据集用于给卷积单元打分。

论文3:《Pruning Convolutional Neural Networks for Resource Efficient Inference》

<https://arxiv.org/abs/1611.06440>

这篇论文非常不错，是来自Nvidia的研究工作。

首先，论文将剪枝问题当作是一个组合优化问题：从众多的权重参数中选择一个最优组合B，使得被剪枝的模型的代价函数损失最小。相应的公式如下：

$$\min_{W'} \left| C(D|W') - C(D|W) \right| \quad \text{s.t.} \quad ||W'||_0 \leq B$$

值得注意的是，论文用的是代价函数损失的绝对值，而不是单纯的差值。使用代价函数损失的绝对值作为优化目标，可以保证被剪枝的模型在性能上不会损失过多。论文给出的实验结果表明，被剪枝的模型能取得更好的结果，这可能是因为这种模型更加稳定。

现在所有的排序方法都可以用这种损失函数来评估。

Oracle剪枝方法

VG G16模型有4224个卷积层。理想情况下，最好的剪枝方法用暴力方法 (Brute Force) 实现的逻辑是：对所有的卷积层，不断地剪枝，观察剪枝后的结果对娱乐数据历史代价函数的变化。由于养生的私房菜，最新 更多+


作者来自Nvidia，他们用了大量的GPU来对此进行实现。以上这种方法被称为Oracle剪枝，它能够取得最佳排序结果使得模型的代价函数变化最小。论文提出了通过计算与Oracle剪枝方法的斯皮尔曼相关性来衡量各种剪枝方法的效果，出乎意料的是，他们接下来提出的方法与Oracle方法是最为接近的。

xinwen/caijing/meishijixingzuc yule/ lishi/ tiyumi/duanzi/yangshsifangh

他们提出了一种基于一阶泰勒展开的模型代价函数来对神经元排序的新方法。对卷积单元h做剪枝实际上就是对其赋值为0。

C(W, D)表示在数据集D上的模型代价函数的平均值，W为模型的权重参数。现在我们可以通过计算对卷积核中各个单元剪枝时的C(W, D, h = 0)来表示C(W, D)，这些值应该是非常接近的（因为仅将一个卷积单元赋值为0所带来的影响不大）。

对卷积单元h的排序指标为abs(C(W, D, h = 0) - C(W, D))。



(<http://tg.602.com/cqsj/164/iuid=yq1010&suid=cqsj1&cf>)

$$\Theta_{TE}(h_i) = \left| \Delta C(h_i) \right| = \left| C(\mathcal{D}, h_i) - \frac{\delta C}{\delta h_i} h_i - C(\mathcal{D}, h_i) \right| = \left| \frac{\delta C}{\delta h_i} h_i \right|.$$
$$\Theta_{TE}(z_l^{(k)}) = \left| \frac{1}{M} \sum_m \frac{\delta C}{\delta z_{l,m}^{(k)}} z_{l,m}^{(k)} \right|$$

卷积层的排序结果会经过L2正则化处理，个人猜测这是一种基于经验主义的方法，因为我也不清楚这样做是否必要，但实验结果就是表明这种做法能够大大增强剪枝的效果。

论文所提出的这种排序方法是直观、易于理解的，我们尝试过使用激活函数值、梯度值来作为排序指标，如果它们的值都很大，这表示对输出结果的影响很大。可以将这两者相乘作为排序指标，通过相乘结果来决定是否进行剪枝。

实验结果表明，论文提出的方法在性能（准确率）上远高于其他的方法，这样看来与Oracle方法进行斯皮尔曼相关性计算是一种很好的评测方法，能够评测剪枝方法的好坏。

不管怎样，我认为这篇论文提出的方法还是非常不错的，容易编码实现和测试，在之后的篇幅我会深入解析这种方法的实现。

利用泰勒排序方法对一个猫狗分类器进行剪枝

下面我们有一个迁移学习的任务，需要在一个相对较小的数据集上创建分类器模型（类似于这篇讲Keras的博文-<https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html>）。

如果我们用一个预训练好的VGG模型来进行迁移学习，之后再对其进行剪枝会怎么样呢？

如果VGG16模型已经学习出许多有关汽车、人和马的特征了，那么对于学习一个猫狗分类器而言，这些已学习的特征中有没有对识别美食、星座、娱乐、历史、体育、段子、养生、私房、最新、更多+有帮助的呢？

[yangqiu.cn](#)

请输入关键词

(/)

(/n- (/n- (/n- (/n- (/n- (/n- (/n- (/n- (/n- (/n- (/top/)

我想这类问题是普遍存在的。

xinwen/caijing/meishijixingzuc yule/) lishi/) tiyumi/duanzi/yangshsifangh

这里的训练数据包括1000张猫的图片 and 1000张狗的图片（数据来源于Kaggle Dogs vs Cats data set-
<https://jacobgil.github.io/deeplearning/%5BKaggle%20Dogs%20vs%20Cats%20dataset%5D%28https://www.kaggle.com/c/dogs-vs-cats%29>）；测试数据包括400张猫的图片 and 400张狗的图片。

首先展示一下实验结果：

模型的准确率从98.7%下降至97.5%；

模型文件大小从538MB缩小至150MB；

在i7 CPU设备上，模型对一张图片进行分类所消耗的时间从0.78s减小至0.277s，几乎快了3倍。

第一步：训练一个大型模型

我们使用VGG16模型，丢弃了原始的全连接层，然后加了3个新的全连接层。重新训练模型时，我们固定了卷积层的参数（不参与模型训练），只对新加的3个全连接层作训练。在PyTorch中，新加的全连接层表示如下：

```
self.classifier = nn.Sequential(  
    nn.Dropout(),  
    nn.Linear(25088, 4096),  
    nn.ReLU(inplace=True),  
    nn.Dropout(),  
    nn.Linear(4096, 4096),  
    nn.ReLU(inplace=True),  
    nn.Linear(4096, 2))
```

经过20轮的训练次数之后，重新训练的模型在测试数据上取得了98.7%的准确率。

第二步：对卷积核进行排序

为了计算泰勒展开，我们需要在数据集上做模型的前向传播和反向传播过程（如果数据集太大可以只取其中一部分，这里用到的数据集较小，仅有2000张图片）。

现在我们需要得到卷积层中的梯度和激活函数值，在PyTorch中，我们可以对梯度计算注册一个hook，完成时将会被调用的回调方法如下：

```
for layer, (name, module) in enumerate(self.model.features._modules.items()):  
    x = module(x)  
    if isinstance(module, torch.nn.modules.conv.Conv2d):  
        x.register_hook(self.compute_rank)  
        self.activations.append(x)  
        self.activation_to_layer[activation_index] = layer  
        activation_index += 1
```



(/)

```
def compute_rank(self, grad):  
    activation_index = len(self.activations) - self.grad_index - 1  
    activation = self.activations[activation_index]  
    values = torch.sum((activation * grad), dim = 0).\  
        sum(dim=2).sum(dim=3)[0, :, 0, 0].data  
  
    # Normalize the rank by the filter dimensions  
    values = \  
        values / (activation.size(0) * activation.size(2) * activation.size(3))  
  
    if activation_index not in self.filter_ranks:  
        self.filter_ranks[activation_index] = \  
            torch.FloatTensor(activation.size(1)).zero_().cuda()  
  
    self.filter_ranks[activation_index] += values  
    self.grad_index += 1
```



从中可以看出，在一个batch中，对每一个激活函数的输出和对应的梯度值做了点乘操作，然后对于每一个激活函数的输出（即一个卷积层的输出），我们对除了输出维度之外的所有维度进行求和。

举个例子，如果batch大小为32，激活函数输出的数目为256，大小为112x112，那么激活函数输出/梯度向量的大小则为32x256x112x112，所以最终的输出是一个大小为256的向量，该向量是这层256个卷积窗口的排序结果。

在得到排序结果之后，我们可以用最小堆来得到N个排名最低的卷积窗口。Nvidia论文中在每轮迭代中设置N=1，为了更快获得结果，我们设置N=512。这表示在每轮剪枝迭代中，我们将会从原始的4224个卷积核中去掉约12%。

排序较低的卷积窗口的分布很有意思，其中的大多数都在更深的网络层中被剪枝了。下面是第一轮迭代之后，卷积窗口被剪枝的数量分布：

Layer number	Number of pruned filters pruned
Layer 0	6
Layer 2	1
Layer 5	4
Layer 7	3
Layer 10	23
Layer 12	13
Layer 14	9
Layer 17	51
Layer 19	35
Layer 21	52
Layer 24	68
Layer 26	74
Layer 28	73

这一阶段，我们使模型中所有参数参与训练和学习，重新训练模型10个轮次就能够在这个数据集上取得很好的结果了。接下来我们回到第三步，进行剪枝。

这样做的代价是：完成一次模型的迭代所需要的训练轮数是过去训练的50%。在这个测试数据集上，由于数据规模较小所以问题不大。但如果你的数据集很大，那么你最好有很多的GPU来训练模型。

总结

目前对神经网络模型进行剪枝的方法并不流行，但我却认为该方法是值得更多的关注和适用实际的，我们的实验结果证实了在猫狗数据集上该方法的优越性。实际上，有许多解决深度学习中的难题的方法也与之类似，尤其对于迁移学习而言，在一个限定的数据集上对模型做剪枝是一件非常有意义的事情。

原文链接

<https://jacobgil.github.io/deeplearning/pruning-deep-learning>

参考论文

《Pruning Convolutional Neural Networks for Resource Efficient Inference》

<https://arxiv.org/abs/1611.06440>

对应的PyTorch实现代码地址

<https://github.com/jacobgil/pytorch-pruning>

后台回复关键词“剪枝”，即可下载本文涉及的4篇论文。



(<http://tg.602.com/cqsj/164/iuid=yq1010&suid=cqsj1&cf>)

数据军备竞赛：如何武装自己的数据团队？

Competing in a Data Race: How to Arm My Data Team

xinwen/caijing/meishijixingzuc yule/ lishi/ tiyumi/duanzi/yangshsifangh

清华-青岛数据科学研究院&大数据文摘 重磅发布《顶级数据团队建设报告》

7月11日 下午2点整，清华报告厅
带您一起揭秘数据团队建设图景！

扫码报名







602.com
(<http://tg.602.com/cqsj/164/iuid=yq1010&suid=cqsj1&cf>)

为保证发文质量、树立口碑，数据派现设立“错别字基金”，鼓励读者积极纠错。

若您在阅读文章过程中发现任何错误，请在文末留言，或到后台反馈，经小编确认后，数据派将向检举读者发8.8元红包。

同一位读者指出同一篇文章多处错误，奖金不变。不同读者指出同一处错误，奖励第一位读者。

感谢一直以来您的关注和支持，希望您能够监督数据派产出更加高质的内容。

公众号底部菜单有惊喜哦！

企业，个人加入组织请查看“联合会”

往期精彩内容请查看“号内搜”

加入志愿者或联系我们请查看“关于我们”

杨邱自媒体

yangqiu.cn

首页 (/) 新闻 财经 美食 星座 娱乐 历史 体育 段子 养生 私房 最新 更多+

请输入关键词

独家干货 | 优质内容 | 讲座快讯 | 行业资讯

Share And Study

xinwen/caijing/meishijixingzuc yule/ lishi/ tiyumi/duanzi/yangshsifangh

数据派THU是清华-青岛数据科学研究院的官方微信平台。

独家传播来自清华的数据科学知识。



数据派THU

(ID : DatapiTHU)

请扫描二维码关注

投稿、申请转载、商务合作，请发送邮件至：
datapi@tsingdata.com

密

神秘游戏 火爆

602.com

(<http://tg.602.com/cqsj/164/uid=yq1010&suid=cqsj1&cfg=>

标签 (/tag/) : 深度 (/tag/601/) 大小 (/tag/11716/) 速度快 (/tag/13041/) 模型 (/tag/17328/) 多快好省 (/tag/185070/)

下一篇：江西快3号码推荐6月30日 (/k3Superman/3302710.html) 返回列表 (/n-xinwen/)

热门文章

现场直击！原来，上海高考试卷是这样批出来的..... (/shanghaifabu/3269868.html)	2017-07-04
学霸宋仲基为学霸送福利，学霸的世界教授不懂 (/doctorand/3269771.html)	2017-07-04
健康青稞酒飘香，互助青稞酒节吸引远方客人 (/shichanglianxian/3300050.html)	2017-07-05
一部极简香港经济史：客途秋恨的今非昔比 (/goldenfleece21/3256499.html)	2017-07-04
越南菲律宾全傻了，中国在南海竟然是这样填岛的 (/jiuloog/3299601.html)	2017-07-05
【重要通知】2017年北京电子科技职业学院基础学院暑假安全提示 致家长的一封信 (/dkygtpy/32668.html)	2017-07-04
热热闹闹迎新年 北京中医药大学东方学院见习生在我院举办新年联欢会 (/dzsrmyy2015/32668.html)	2017-07-04
北京中医药大学东方学院定州临床教学基地新年联欢会在我院行政楼四楼会议室举办 (/dzsrmyy2015/32668.html)	2017-07-04



602.com

(<http://tg.602.com/cqsj/164/index.html?uid=yq1010&suid=cqsj1&cfg=ep>)