shuzfan的专栏

DeepRebirth——通过融合加速网络

标签： 神经网络 压缩 深度学习

2016-11-12 12:19 2221人阅读 评论(0) 收藏 举报

分类：

神经网络压缩与加速（13）

目录(?) [+]

这里介绍2017ICLR OpenReview中的一篇有关网络加速的文章《DeepRebirth: A General Approach for Accelerating Deep Neural Network Execution on Mobile Devices》。 看文章标题觉得高大上，看方法细节觉得卧槽好水，看自己的验证结果好像还有点用。

附：2017ICLR openreview http://openreview.net/group?id=ICLR.cc/2017/conference

引言

纵观之前的大部分压缩和加速方法都是在打全连接层的注意，比如分解、量化、剪枝等，但是目前的主流网络比如GoogLeNet和ResNet等，都尽可能的用大量小核卷积层和pooling层来取代全连接层。因此，以往的很多方法都不是很适合。

首先，作者将层分为：non-tensor layers 和 tensor layers，前者指不带参数的层，比如pooling、LRN、BatchNorm、softmax层等，后者则指Convolution、InnerProduct这些带参数的层。

然后作者统计了几个流行网络中，这些non-tensor layers 所消耗的时间，如下图。这些层，没有参数还占用空间和时间，作者觉得应该搞掉他们。

GPU (10)
操作系统 (7)
杂谈 (1)
TensorFlow (6)

文章存档

2018年01月 (1)
2017年12月 (3)
2017年11月 (6)
2017年10月 (6)
2017年09月 (12)

展开

阅读排行

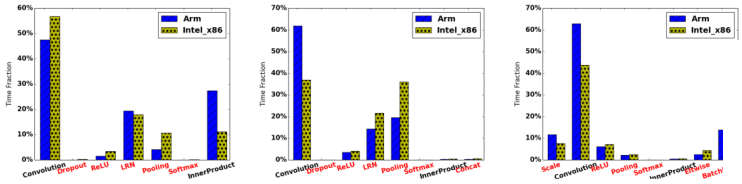
GoogLeNet系列解读 (41632)
人脸检测——MTCNN (31760)
深度学习——Xavier初始化方法 (28390)
caffe添加新层教程 (21109)
C++ Map常见用法说明 (19899)
mxnet学习记录【1】 (19885)
caffe层解读系列-softmax_loss (19848)
NDK各个版本链接 (16962)
人脸检测——DDFD (14882)
NMS——非极大值抑制 (14569)

评论排行

人脸检测——DDFD (178)
GoogLeNet系列解读 (56)
人脸检测——MTCNN (23)
caffe添加新层教程 (18)
mxnet学习记录【1】 (15)
神经网络压缩：Deep Compre... (15)
caffe层解读系列-softmax_loss (14)
Win10 如何以管理员身份设置... (13)
caffe层解读系列——Data以及... (12)
NMS——非极大值抑制 (12)

Table 1: Percentage of Forwarding Time on Non-tensor Layers

Network	Intel x86	Arm	Titan X
AlexNet	32.08%	25.08%	22.37%
GoogLeNet	62.03%	37.81%	26.14%
ResNet-50	55.66%	36.61%	47.87%
ResNet-152	49.77%	N/A	44.49%
Average	49.89%	33.17%	35.22%



(a) AlexNet (b) GoogLeNet (c) ResNet-50

Figure 1: Time Decomposition for each layer. Non-tensor layers (e.g., dropout, ReLU, LRN max, pooling, etc) shown in red color while non-tensor layers (e.g., convolution, inner-product) shown in black color.

方法：DeepRebirth

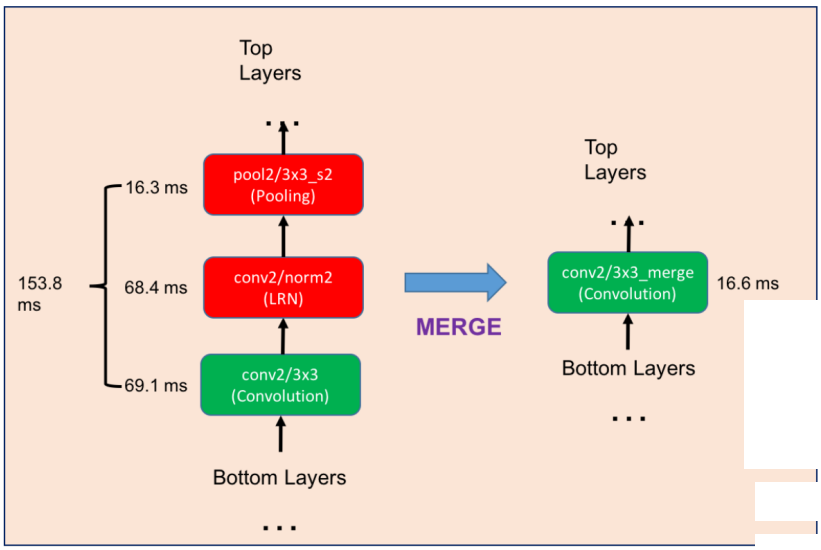
为了降低non-tensor layers的消耗，作者的方法就是融合，这里一共给出了2种融合的情况。

StreamLine Merging

思路非常简单，如下图：

左边是原始网络的一部分，我们用右边的等价结构来代替它，然后重新finetune网络。finetune的时候，新结构的学习率设为其他层的10倍。

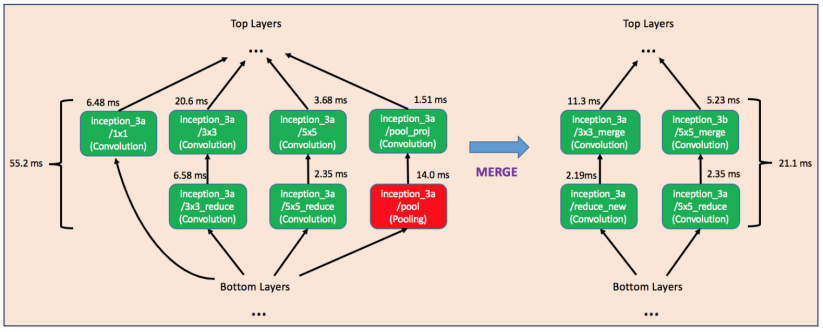
注意，原结构是 “3x3卷积 stride=1” + “LRN” + “3x3pooing stride=2”，替换后的结构则简化为 “3x3卷积 stride=2”



Branch Merging

这种融合主要针对GoogLeNet中的Inception结构。GoogLeNet虽然参数比较少，但由于层很多，所以速度并没有特别快。

如下图，作者融合掉了单独的1x1卷积分支以及pooling分支，同时为了保证融合后的结构可以和其他部分衔接，融合后的卷积层的num_output也要做出适当调整。



实验结果

这里只给出GoogLeNet的结果。下图是性能损失差异：

其中 Tucker Decomposition 是《Compression of deep convolutional neural networks for fast and low power mobile applications》 中提出的一种压缩分解方法。

Table 2: GoogLeNet Accuracy on each layer after merging

Step	Merged Layer(s)	Top-5 Accuracy
0	N/A	88.89%
1	conv1	88.73%
2	conv2	88.82%
3	inception_3a	88.50%
4	inception_3b	88.27%
5	inception_4a	88.60%
6	inception_4b-4d	88.61%
7	inception_4e	88.43%
8	inception_5a	88.41%
9	inception_5b	**88.43%**
Tucker Decomposition	N/A	**86.54%**

下面是速度提高对比：能加速2-3倍，我已经很满意了。T_T

Device	GoogLeNet	GoogLeNet -Tucker	GoogLeNet -Merge	GoogLeNet -Merge-Tucker
conv1	94.92 ms	87.85 ms	8.424 ms	6.038 ms
conv2	153.8 ms	179.4 ms	16.62 ms	9.259 ms
inception_3a	55.23 ms	85.62 ms	21.17 ms	9.459 ms
inception_3b	98.41 ms	66.51 ms	25.94 ms	11.74 ms
inception_4a	30.53 ms	36.91 ms	16.80 ms	8.966 ms
inception_4b	32.60 ms	41.82 ms	20.29 ms	11.65 ms
inception_4c	46.96 ms	30.46 ms	18.71 ms	9.102 ms
inception_4d	36.88 ms	21.05 ms	24.67 ms	10.05 ms
inception_4e	48.24 ms	32.19 ms	28.08 ms	14.08 ms
inception_5a	24.64 ms	14.43 ms	10.69 ms	5.36 ms
inception_5b	24.92 ms	15.87 ms	14.58 ms	6.65 ms
loss3	3.014 ms	2.81 ms	2.97 ms	2.902 ms
Total	**651.4 ms**	**614.9 ms (1.06x)**	**210.6 ms (3.09x)**	**106.3 ms (6.13x)**

- 上一篇 C++ Map常见用法说明
- 下一篇 模型压缩——将模型复杂度加入loss function

相关文章推荐

- 论文笔记：DeepRebirth——从非权重层入手来进...
- MySQL在微信支付下的高可用运营--莫晓东
- ImageNet中的LRN（Local Response Normalizati...
- 容器技术在58同城的实践--姚远
- LRN层的实现
- SDCC 2017之容器技术实战线上峰会
- 如何在Caffe中配置每一个层的结构
- SDCC 2017之数据库技术实战线上峰会
- 神经网络压缩：Deep Compression
- 腾讯云容器服务架构实现介绍--董晓杰
- Deep Learning（深度学习）学习笔记整理系列之...
- 微博热点事件背后的数据库运维心得--张冬洪
- 卷积神经网络简介（Convolutional Neural Networ...
- 深度学习——缩减+召回加速网络训
- 深度学习——缩减+召回加速网络训
- Batch Normalization —— 加速深度

查看评论

暂无评论

您还没有登录,请[登录]或[注册]

* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场