SQUEEZENET: ALEXNET-LEVEL ACCURACY WITH 50X FEWER PARAMETERS AND <0.5MB MODEL SIZE

摘要

最近在深度卷积神经网络(CNNs)上的研究都主要集中在提高精度。对于以给定的精度等级，通常可能有多个CNN结构可以达到这个精度等级。在相同的精度下，更小的CNN结构至少有如下三个优势：(1)在分布式训练中更小的CNNs需要更少的服务器间的交流。(2)更小的CNNs需要更少的带宽来从云端下载一个新的模型到一个自动驾驶汽车。(3)更小的CNNs更容易部署到FPGA和其他内存有限的硬件上。为了证明这些优势，我们提出来一个小的CNN结构叫做SqueezenNet。SqueezenNet在ImageNet上达到了AlexNet等级的精度同时参数数量只是AlexNet的五十分之一。并且，使用模型压缩技术，我们能够压缩SqueezeNet比0.5M还要小(比AlexNet的五百一十分之一还要小)。SqueezeNet结构开放下载在：http//:github.com/DeepScale/SqueezeNet

1 介绍和动机

最近许多深度卷积神经网络(CNNs)上的研究都主要集中在提高在计算机视觉数据集上精度。对于以给定的精度等级，通常存在多个CNN结构可以达到这个精度等级。在相同的精度下，有更少参数的CNN结构有一些优势：

**更高效的分布式训练：**在服务器之间的通信是分布式CNN训练可扩展性的一个限制因素。对于分布式并行数据训练，通信开销直接与模型中的参数数量成比例(Iandola等人2016)。简而言之，小的模型因为需要更少的通信所以训练更快。

**当给客户更新模型时更小的开销。**对于自动驾驶汽车，例如特斯拉公司定期惊新的模型从他们的服务器拷贝到客户的汽车。这个行为通常为在线更新。客户调查发现随着最近的无线更新，teals的自动驾驶仪半自动驾驶功能的安全性得到了逐步提高(2016年客户调查)。但是今天的标准CNN/DNN模型的在线更新需要大量的数据传输。AlexNet需要服务器到汽车的240M的通信。更小的模型需要更少的通信，使得频繁的更新更加可行。

**可行的FPGA和嵌入式部署。**FPGA通常只有小于10MB的片上内存，没有片外存储器。所以一个高效的更小的模型可以被直接存储在FPGA上而不被存储带宽限制(Qiu等人2016)，同时视频帧流通过FPGA实时传输。而且，当将CNNs部署到专用集成电路(ASICs)上时，一个小的模型可以直接被储存在片上，更小的模型可以使得ASIC适合更小的模具。

如你所见，更小的CNN结构有这些好处，考虑到这些，我们直接集中精力在找到具有较少参数但相比于已知的较好的模型具有相同的精度的CNN结构。我们已经发现这种结构，称之为SqueezeNet。除此之外我们尝试采用更有纪律的方法来搜索新颖的cnn架构的设计空间。

本文的其他部分的按如下所说组织。在第二部分我们回顾了相关工作。然后在第三部分和第四部分我们描述和评估了SqueezeNet结构。然后我们将我们的注意力转换到理解CNN结构设计的选择如何影响模型尺寸和精度。我们通过探索类似SqueezeNet结构的设计空间来获得结论。在第五部分，我们做了在CNN微结构设计空间探索，我们定义微结构为独立层和模型的维度和组织结构。在第六部分我们做了在CNN宏结构设计空间探索，我们定义宏结构为CNN中的高等级的层组织结构。最后我们在第七部分做了总结。简言之，第三部分和第四部分对CNN研究者和志向简单的将SqueezeNet应用到新的领域的实践者有帮助。其余部分的目标是想要设计自己的CNN结构的先进的研究者。

2 相关工作

2.1 模型压缩

我们工作最终的目标视为了证实一个有着很少参数的模型同时保持着精度。为了解决这个问题，一个明智的方法是使用一个现存的CNN模型然后以一个有损的方式将其压缩。事实上，已经出现了一个围绕模型压缩的主题的研究社区，一些方法已经被提出。Denton等人提出了一个非常直接的方法是对一个预训练CNN模型应用singular value decomposition(单数值分解SVD)方法(Denton等人 2014)。Han等人研究出了Network pruning网络修剪方法,它以一个预训练模型开始，然后用零值替代将低于一个确定阈值的参数来形成一个稀疏矩阵，最后在稀疏CNN(Han等人 2015b)上进行多次迭代。最近Han等人通过将有量化(到8bit甚至更少)的网络修剪的霍夫曼编码结合起来创造了一个叫深度压缩(Han等人 2015a)的方法扩展了他们的工作，进一步设计了一个能直接在被压缩的模型上操作的硬件加速器叫做EIE(Han等人 2016a)，达到了实质的加速和能量的节省。

2.2 CNN微结构

人工神经网络中使用卷积至少有25年，LeCun等人帮助推广CNNs来做数字识别应用刚在1980s晚期(LeCun等人 1989)。在神经网络中，卷积神经网络通常是3D的，有高，宽和通道数作为关键维度。当应用到图片上，CNN滤波器通常在第一层有三个通道(例如 RGB)，每一个下一层的Li的滤波器的通道数与上一层的滤波器数目相同。LeCun等人做的早期的工作(LeCun等人 1989)使用5\*5\*通道数平方的滤波器，最近的VGG(Simonyan和Zisserman，2014)结构广泛使用3\*3的滤波器，例如网络中的网络模型(Lin等人 2013)和GoogLeNet家族的结构(Szegedy等人 2014；Ioffe和Szegedy2015；Szegedy等人2015；2016)在一些层中是用来1\*1的滤波器。

在设计非常深的CNNs的趋势下，为每层的滤波器手动选择维度变得非常笨重。为了解决这个问题，不同的高等级的建造块或者模型被提出，它由特殊的固定组织的多个卷积层组成。例如GoogLeNet提出的inception模块，它由一定数量的不同维度的滤波器组成，通常包括1\*1和3\*3的，有时候会有5\*5的滤波器(Szegedy等人 2014)，有时会有1\*3和3\*1的滤波器(Szegedy等人 2015)，许多这样的模型被结合在一起，可能加上一些特设层形成一个完整的网络。我们使用术语微结构来指独立模块的特别的组织和维度。

2.3 CNN宏结构

CNN微结构指的是独立的层和模块。我们定义CNN宏结构为在端对端地CNN结构中系统级别的多模块的组织结构。

可能在最近文章中最广泛研究的CNN宏结构话题是网络中深度的影响(例如层的数量)。Simoyan和Zisserman提出了12到19层CNNs的VGG(Simoyan和Zisserman 2014)家族，报告在ImageNet上越深的网络产生越高的精度(Deng等人 2009)。K.He等人提出了更深30层的CNNs产生了更高的ImageNet精度。

在多层或者模块上连接的选择是CNN宏结构研究中新型的区域。残差网络(ResNet)(He等人2015b)和Highway网络(Srivastava等人 2015)都提出了使用跳跃多个层的连接，例如增加从第三个激活层到第六个激活层的连接。我们称这些连接为旁路连接。残差网络的做着提供了有无旁路连接的三十四层的CNN的A/B对照。增加旁路连接产生了2%的top-5ImageNet精度的提升。

2.4 神经网络设计空间探索

神经网络(包括深度和卷积神经网络)有一个大的设计空间，它有微结构，宏结构，求解器和其他参数的许多选择。科学社区自然想要得到关于这些因素怎样影响一个网络的性能的直觉(例如 设计空间的形状)。许多神经网络设计空间探索(DSE)的研究已经专注于开发自动寻找产生更高精度的网络结构的自动化方法。这些自动化的DSE方法包括贝叶斯优化(Snoek等人 2012)，模拟退火方法(Ludermir等人 2006)，随机搜索(Bergstra和Bengio 2012)，进化算法(Stanley和Miikkulainen 2002)。多亏他们的功劳，他们的每篇文章提供了一个使用DSE方法产生一个相对于有代表性的基准达到很高精度的神经网络结构的例子。然而，这些文章没有提供关于神经网络设计空间的直觉。在本篇文章后面，我们不开了自动化方法，取而代之的是我们能做有原则的A/B比较的方式来调查CNN的结构决定怎样影响模型尺寸和精度重构了CNNs。

在接下来的部分，我们首先提出和评估了使用和未使用模型压缩的SqueezeNet结构。然后我们探索了微结构和宏结构设计的选择对SqueezeNet相似结构的影响。

3 SQUEEZENET：保持精度的同时有更少的参数

在这个部分，我们以列出我们带有很少参数的CNN结构的设计策略开始。然后我们引入了Fire模块，我们的新的建造模块，用于构建CNN结构。最终我们使用我们的设计策略来建造SqueezeNet,它主要由Fire模块组成。

3.1 结构设计策略

这篇论文中我们的要达到的目标是证实有很少参数的CNN结构同时也可以保持很好的精度。为了达到这个目的，当设计CNN时我们引进了三个主要的策略：

**策略1：使用1\*1的滤波器来代替3\*3的滤波器。**给定一个确定数目的卷积滤波器的负担，我们将会选择大多数的滤波器为1\*1的，因为1\*1的滤波器是3\*3的滤波器的参数九分之一。

**策略2：减少3\*3的滤波器的输入通道数目。减少3\*3滤波器的输入通道数目。**考虑一个全部由3\*3滤波器组成的卷积层。这一层总参数的数量是(输入的通道数)\*(滤波器数量)\*(3\*3)。所以为了保持CNN中总参数量较小，不仅减少3\*3滤波器的数量是重要的(如策略一所说)，而且减少3\*3的滤波器的输入通道数也很重要。我们使用squeeze层减少了3\*3滤波器的输入通道数，在下一个部分将要描述squeeze层。

**策略3：在网络的后面下采样这样卷积层可以获得大的激活特征图。**在一个卷积网络中，每个卷积层都会产生一个输出激活特征图，它的空间分辨率至少是1\*1的通常远大于1\*1。这些激活图的高度和宽度通过以下控制：(1)输入数据的尺寸(例如256\*256的图片) 。(2)在CNN结构中下采样层的选择。通常CNN结构中的下采样通过在一些卷积层中设置步进值(>1)或者池化层(例如(Szegedy等人 2014；Simonyan&Zisserman 2014；Krizhevsky等人 2012))来实现的。如果在网络较早的层中使用大的步进值，大多数网络将会有小的激活图。相反的，如果在大多数网络中使用步进值为一，在网络的最后使用步进值大于一，那么网络中的许多层将会有大的激活图。我们的直觉是较大的激活图(由于较晚的下采样)可以产生较高的分类精度。在其他保持一致的情况下，K.He和H.Sun对不同的CNN结构应用了延迟下采样，在每一种情况下延迟下采样都产生了较高的分类精度(He&Sun 2015)

策略1和2是关于减少CNN中参数的数量同时想要保持精度。策略3是关于在一个有参数负担限制的情况下最大化精度。接下来我们描述Fire模型，它是我们CNN结构的构造块。它使得我们成功地实现策略1，2和3。

3.2 Fire模块

我们按如下定义Fire模块。Fire模块组成为：一个squeeze卷积层(只包含1\*1的滤波器)，其输出给一个有着混合的1\*1和3\*3滤波器的expand层。如图一所示。在Fire模块中自由使用1\*1的滤波器是3.1部分策略1的应用。我们在Fire模块中使用三个可调的维度(超参数)：s1x1，e1x1，e3x3。在Fire模块中，s1x1是指在squeeze层中1\*1滤波器的数量，e1x1和e3x3分别是expand层中1\*1和3\*3滤波器的数量。当我们使用Fire模块时，我们通常设置s1x1小于(e1x1+e3x3)，这样squeeze层帮助我们限制了3\*3滤波器层的输入通道数目，如同3.1部分的策略2所说。

3.3 SQUEEZENET结构

现在我们描述SqueezeNet结构，我们在图二中描述了SqueezeNet以一个单独的卷积层(conv1)开始，后面跟着8个Fire模块(fire2-9)，最后以一个最终卷积层结束(conv10)。SqueezeNet在conv1,fire4,fire4,fire8和conv10后跟着一个步进为2的最大池化；这些相对较晚的池化的放置是遵循了3.1部分的策略3.我们在表一中提供了完整的SqueezeNet结构。

3.3.1 其他细节

简言之，我们从表1和图2中忽略了关于SqueezeNet的一些细节和设计选择。我们在下面提供这些射界选择，这些选择后的根据可能在下面引用的论文中可以找到。

我们在expand模块中3\*3滤波器的输入数据中增加了1像素的边界零填充，所以来自于1\*1和3\*3滤波器的输出激活具有相同的高度和宽度。

RELU(Nair&Hinton 2010)被用于从squeeze和expand层中激活。

Fire9模块后应用了0.5的Dropout(Srivastava等人 2014)。

在SqueezeNet中没有全连接层，这个设计选择是受启发与NIN(Lin等人 2013)结构。

训练SqueezeNet时，我们以0.04的学习率开始，然后在训练中线性减少学习率，如(Mishkin等人 2016)描述的一样。训练协议的细节(例如 批尺寸，学习率，参数初始化)请参考我们的兼容caffe的设置文件，位于：<http://github.com/DeepScale/SqueezeNet>。

Caffe框架不是天然支持同一个包含有不同分辨率的滤波器(例如 1\*1 3\*3)(Jia等人 2014)的卷积层，因此我们使用两个分离的卷积层来实现我们的expand层；一个带有1\*1滤波器的层和一个带有3\*3滤波器的层。然后我们将这两个层的输出以频道这个维度连接在一起。这与以一个包含1\*1和3\*3滤波器的层来实现在数量上是等价的。

我们将SqueezeNet设置文件以通过Caffe CNN框架定义的格式发布。然而处理Caffe一些其他的CNN框架已经出现，包括MXNet(Chen 等人 2015a)，Chainer(Tokui等人 2015)，Kears(Choolet 2016)和Torch(Collobert等人 2011)。这些每一个都有自己的格式来代替CNN结构。也是说，大部分这些苦使用相同的潜在计算后端力促cuDNN(Chetlur等人 2014)和MKL-DNN(Das等人 2016)。研究社区已经移植Squeezenet cnn架构以与许多其他cnn软件框架兼容:

SqueezeNet的MXNet接口：(Haria 2016)

SqueezeNet的Chainer接口：(Bell 2016)

SqueezeNet的Keras接口：(DT42 2016)

SqueezeNet中Fire模块的Torch接口：(Waghmare 2016)

4 SQUEEZENET的评估