VGG Net

参考文献：

1. https://arxiv.org/abs/1409.1556
2. VGG 论文阅读记录：https://zhuanlan.zhihu.com/p/42233779
3. VGG16学习笔记：https://www.cnblogs.com/lfri/p/10493408.html （tensorflow + keras 实现）
4. 一文读懂VGG网络：https://zhuanlan.zhihu.com/p/41423739

- **input**：224x224 RGB
- **filter**: 3x3 & 1x1
- **stride**: 1
- **padding**: 1 for 3x3 filter.
- **max-pooling**: 2x2 with stride 2
- **activation**: ReLU

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

3x3 卷积核

1x1 卷积核

- 上图列出了 VGG11 到 VGG19 不同的结构，其中的 11 或者 19指的是具有权重参数的层，如卷积层（conv layers）和全连接层（FC），不包括池化层，Dropout 和激活函数层（ReLU）。
- 所有卷积层后面都跟有非线性激活函数层，如 ReLU。每经过一个maxpool 层 filter 的个数就翻倍，如 ConvNet A：input → conv3-64 → max pool → conv3-128(翻倍) → conv3-256(两个) → max pool → conv3-521。
- 之后再接三个全连接层，其中前两个的结构为 FC → ReLU → Dropout，最后一个只有 FC 。
- 最后一个是 softmax 用于分类。

- VGG中根据卷积核大小和卷积层数目的不同，可分为A，A-LRN，B，C，D，E共6个配置(ConvNet Configuration)，其中以D，E两种配置较为常用，分别称为VGG16和VGG19。

总结：

VGG16 的结构如下图：

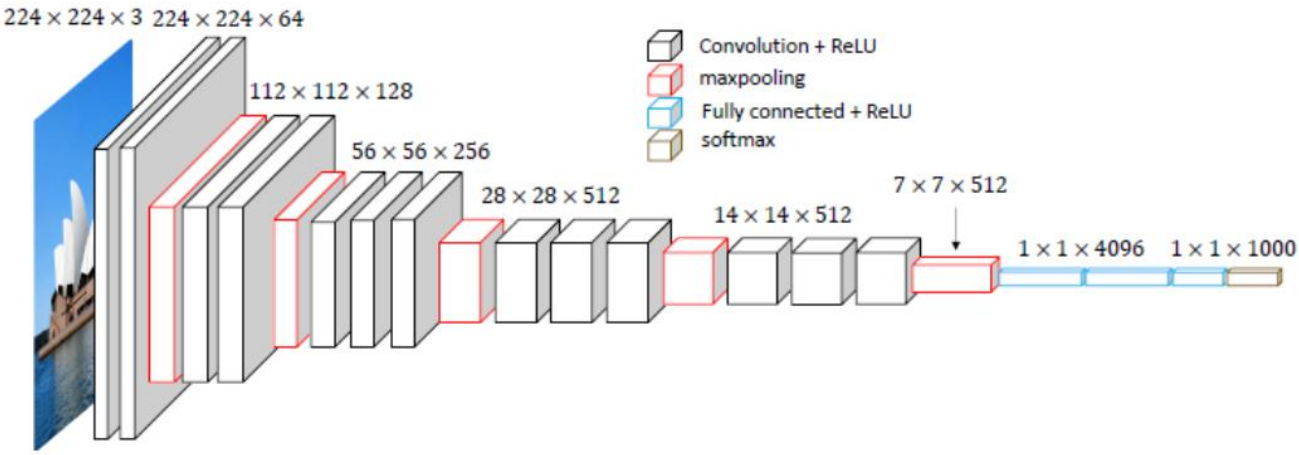
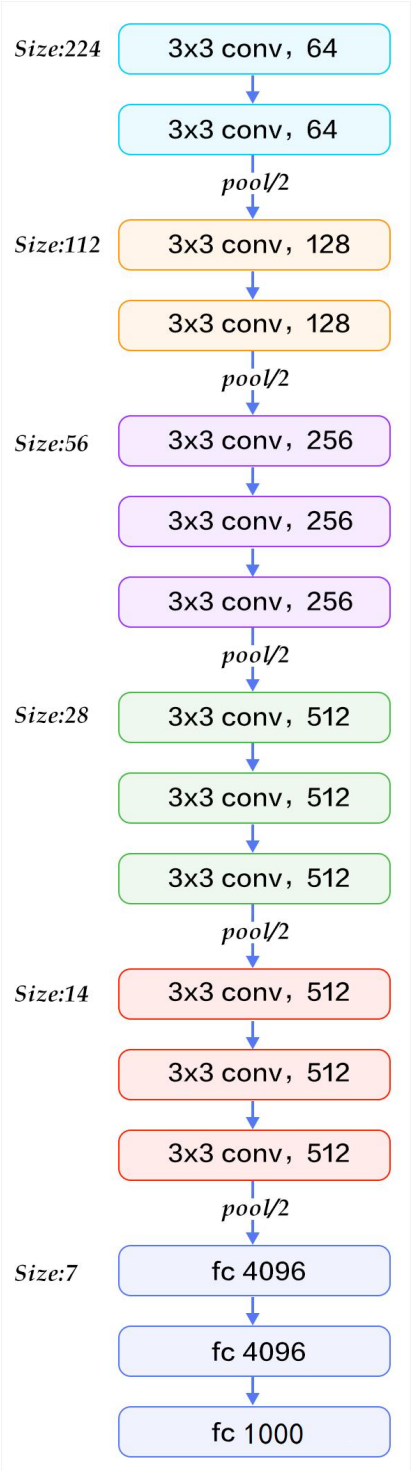


图2：VGG16 结构图



FeiFei Li 在CS231的课件中给出了整个网络的全部参数的计算过程（不考虑偏置）。图中蓝色是计算权重参数数量的部分；红色是计算所需存储容量的部分。如下图所示：

(not counting biases)

INPUT: [224x224x3] memory: 224*224*3=150K params: 0
CONV3-64: [224x224x64] memory: 224*224*64=3.2M params: (3*3*3)*64 = 1,728
CONV3-64: [224x224x64] memory: 224*224*64=3.2M params: (3*3*64)*64 = 36,864
POOL2: [112x112x64] memory: 112*112*64=800K params: 0
CONV3-128: [112x112x128] memory: 112*112*128=1.6M params: (3*3*64)*128 = 73,728
CONV3-128: [112x112x128] memory: 112*112*128=1.6M params: (3*3*128)*128 = 147,456
POOL2: [56x56x128] memory: 56*56*128=400K params: 0
CONV3-256: [56x56x256] memory: 56*56*256=800K params: (3*3*128)*256 = 294,912
CONV3-256: [56x56x256] memory: 56*56*256=800K params: (3*3*256)*256 = 589,824
CONV3-256: [56x56x256] memory: 56*56*256=800K params: (3*3*256)*256 = 589,824
POOL2: [28x28x256] memory: 28*28*256=200K params: 0
CONV3-512: [28x28x512] memory: 28*28*512=400K params: (3*3*256)*512 = 1,179,648
CONV3-512: [28x28x512] memory: 28*28*512=400K params: (3*3*512)*512 = 2,359,296
CONV3-512: [28x28x512] memory: 28*28*512=400K params: (3*3*512)*512 = 2,359,296
POOL2: [14x14x512] memory: 14*14*512=100K params: 0
CONV3-512: [14x14x512] memory: 14*14*512=100K params: (3*3*512)*512 = 2,359,296
CONV3-512: [14x14x512] memory: 14*14*512=100K params: (3*3*512)*512 = 2,359,296
CONV3-512: [14x14x512] memory: 14*14*512=100K params: (3*3*512)*512 = 2,359,296
POOL2: [7x7x512] memory: 7*7*512=25K params: 0
FC: [1x1x4096] memory: 4096 params: 7*7*512*4096 = 102,760,448
FC: [1x1x4096] memory: 4096 params: 4096*4096 = 16,777,216
FC: [1x1x1000] memory: 1000 params: 4096*1000 = 4,096,000

TOTAL memory: 24M * 4 bytes ~= 96MB / image (only forward! ~*2 for bwd)
TOTAL params: 138M parameters

VGG16具有如此之大的参数数目，可以预期它具有很高的拟合能力；但同时缺点也很明显：

- 即训练时间过长，调参难度大。
- 需要的存储容量大，不利于部署。例如存储VGG16权重值文件的大小为500多MB，不利于安装到嵌入式系统中。

总结：

VGG优点

- VGGNet的结构非常规整、简洁，整个网络都使用了同样大小的卷积核尺寸（3x3）和最大池化尺寸（2x2）。
- 几个小滤波器（3x3）卷积层的组合比一个大滤波器（5x5或7x7）卷积层好：对于给定的感受野（与输出有关的输入图片的局部大小），采用堆积的小卷积核是优于采用大的卷积核，因为多层非线性层可以增加网络深度来保证学习更复杂的模式，而且代价还比较小（参数更少）。

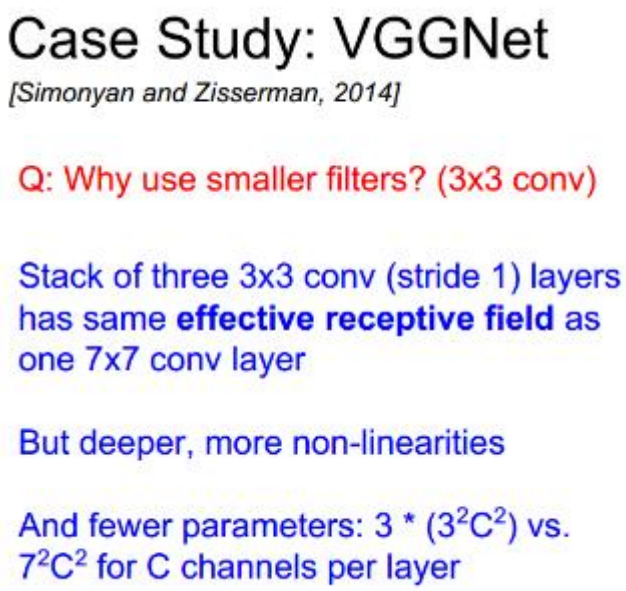
- 验证了通过不断加深网络结构可以提升性能。

VGG缺点

- VGG耗费更多计算资源，并且使用了更多的参数（主要不是3x3卷积的原因），导致更多的内存占用（140M）。其中绝大多数的参数都是来自于第一个全连接层。VGG可是有3个全连接层啊！

- PS：论文中有称道：后来发现这些全连接层即使被去除，对于性能也没有什么影响，这样就显著降低了参数数量。

- 注：很多pretrained的方法就是使用VGG的model（主要是16和19），VGG相对其他的方法，参数空间很大，最终的model有500多m，AlexNet只有200m，GoogLeNet更少，所以train一个vgg模型通常要花费更长的时间，所幸有公开的pretrained model让我们很方便的使用。



总结：