OverFeat Notes

1. 两种结构的模型

Layer	1	2	3	4	5	6	7	Output 8
Stage	conv + max	conv + max	conv	conv	conv + max	full	full	full
# channels	96	256	512	1024	1024	3072	4096	1000
Filter size	11x11	5x5	3x3	3x3	3x3	-	-	-
Conv. stride	4x4	1x1	1x1	1x1	1x1	-	-	-
Pooling size	2x2	2x2	-	-	2x2	-	-	-
Pooling stride	2x2	2x2	-	-	2x2	-	-	-
Zero-Padding size	-	-	1x1x1x1	1x1x1x1	1x1x1x1	-	-	-
Spatial input size	231x231	24x24	12x12	12x12	12x12	6x6	1x1	1x1

Table 1: **Architecture specifics for *fast* model.** The spatial size of the feature maps depends on the input image size, which varies during our inference step (see Table 5 in the Appendix). Here we show training spatial sizes. Layer 5 is the top convolutional layer. Subsequent layers are fully connected, and applied in sliding window fashion at test time. The fully-connected layers can also be seen as 1x1 convolutions in a spatial setting. Similar sizes for *accurate* model can be found in the Appendix.

Layer	1	2	3	4	5	6	7	8	Output 9
Stage	conv + max	conv + max	conv	conv	conv	conv + max	full	full	full
# channels	96	256	512	512	1024	1024	4096	4096	1000
Filter size	7x7	7x7	3x3	3x3	3x3	3x3	-	-	-
Conv. stride	2x2	1x1	1x1	1x1	1x1	1x1	-	-	-
Pooling size	3x3	2x2	-	-	-	3x3	-	-	-
Pooling stride	3x3	2x2	-	-	-	3x3	-	-	-
Zero-Padding size	-	-	1x1x1x1	1x1x1x1	1x1x1x1	1x1x1x1	-	-	-
Spatial input size	221x221	36x36	15x15	15x15	15x15	15x15	5x5	1x1	1x1

Table 3: **Architecture specifics for *accurate* model.** It differs from the *fast* model mainly in the stride of the first convolution, the number of stages and the number of feature maps.

2. Alexnet 图片分类回顾

- **训练阶段**

  每张训练图片 256*256，然后我们随机裁剪出 224*224 大小的图片，作为 CNN 的输入进行训练。

- **测试阶段**

  输入 256*256 大小的图片，我们从图片的 5 个指定的方位(上下左右+中间)进行裁剪出 5 张 224*224 大小的图片，然后水平镜像一下再裁剪 5 张，这样总共有 10 张；然后我们把这 10 张裁剪图片分别送入已经训练好的 CNN 中，分别预测结果，最后用这 10 个结果的平均作为最后的输出。

3. FCN（Fully Convolutional Network）

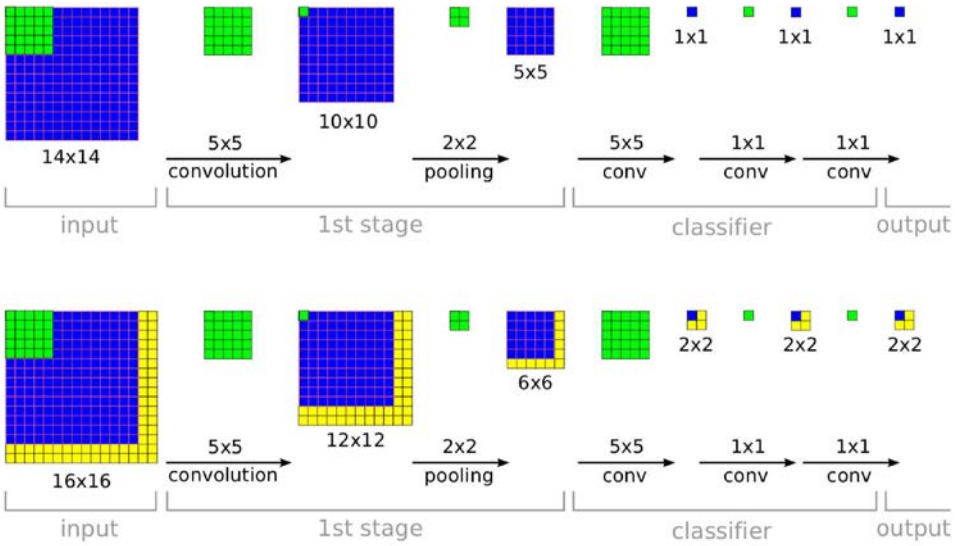
- **卷积层→全连接层**

  直接对 feature map 进行卷积，如输入大小是 5*5*4096，传统做法是将其拉成一个一维向量，然后通过矩阵乘法直接做全连接操作（W 的大小为 4096 by 5*5*4096），FCN 的做法是使用 5*5 大小的卷积核直接对特征图做卷积操作，最后都能得到 1*1*4096 大小的输出。

- **全连接层→全连接层**

  把它看成是用 1*1 大小的卷积核进行卷积操作。

- 示意图



如上图所示，上面图中绿色部分表示卷积核大小。假设我们设计了一个 CNN 模型，输入图片大小是 14*14，通过第一层卷积后我们得到 10*10 大小的图片，然后通过池化得到了 5*5 大小的图片。OK，关键部分来了，接着要从 5*5 大小的图片→1*1 大小的图片：

（1）传统的 CNN：如果从以前的角度进行理解的话，那么这个过程就是全连接层，我们会把这个 5*5 大小的图片，展平成为一个一维的向量，进行计算(写 CNN 代码的时候，经常会在这里加一个 flatten 函数，就是为了展平成一维向量)。

（2）FCN：FCN 并不是把 5*5 的图片展平成一维向量，再进行计算，而是直接采用 5*5 的卷积核，对一整张图片进行卷积运算。

AlexNet 在测试阶段采用了对输入图片的四个角落进行裁剪、预测，分别得到结果，最后的结果就是类似对应于上面 2*2 的预测图。这个 2*2 的每个像素点，就类似于对应于一个角落裁剪下来的图片预测分类结果。只不过 AlexNet 把这 4 个像素点相加在一起，求取平均值，作为该类别的概率值。

4. Offset max-pooling

- 示意图

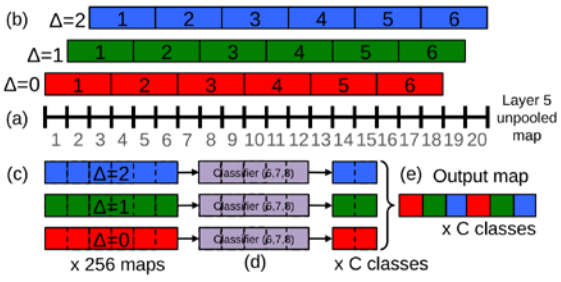


Figure 3: 1D illustration (to scale) of output map computation for classification, using y-dimension from scale 2 as an example (see Table 5). (a): 20 pixel unpooled layer 5 feature map. (b): max pooling over non-overlapping 3 pixel groups, using offsets of $\Delta = \{0, 1, 2\}$ pixels (red, green, blue respectively). (c): The resulting 6 pixel pooled maps, for different Δ. (d): 5 pixel classifier (layers 6,7) is applied in sliding window fashion to pooled maps, yielding 2 pixel by C maps for each Δ. (e): reshaped into 6 pixel by C output maps.

以往的 CNN 中，一般我们只用了△=0 得到池化结果后，就送入一层。文献的方法是，把上面的△=0、△=1、△=2 的三种组合方式的池化结果，分别送入网络的下一层。这样的话，我们网络在最后输出的时候，就会出现 3 种预测结果了。

我们前面说的是一维的情况，如果是 2 维图片的话，那么(△x, △y)就会有 9 种取值情况(3*3)；如果我们在做图片分类的时候，在网络的某一个池化层加入了这种 offset 池化方法，然后把这 9 种池化结果，分别送入后面的网络层，最后我们的图片分类输出结果就可以得到 9 个预测结果(每个类别都可以得到 9 种概率值，然后我们对每个类别的 9 种概率，取其最大值，作为此类别的预测概率值)。

5. OverFeat 算法

- 训练阶段与 AlexNet 基本相同。
- 测试阶段

在测试阶段，我们不再是用一张 221*221 大小的图片作为网络的输入，而是用了 6 张大小都不相同的图片，也就是所谓的多尺度输入预测，如下表格所示：

256 channels

Scale	Input size	Layer 5 pre-pool	Layer 5 post-pool	Classifier map (pre-reshape)	Classifier map size
1	245x245	17x17	(5x5)x(3x3)	(1x1)x(3x3)xC	3x3xC
2	281x317	20x23	(6x7)x(3x3)	(2x3)x(3x3)xC	6x9xC
3	317x389	23x29	(7x9)x(3x3)	(3x5)x(3x3)xC	9x15xC
4	389x461	29x35	(9x11)x(3x3)	(5x7)x(3x3)xC	15x21xC
5	425x497	32x35	(10x11)x(3x3)	(6x7)x(3x3)xC	18x24xC
6	461x569	35x44	(11x14)x(3x3)	(7x10)x(3x3)xC	21x30xC

从 Layer-5 pre-pool 到 Layer-5 post-pool：这一步的实现是通过池化大小为 3*3 进行池化，然后△x=0、1、2,△y=0、1、2，这样对于每一张特征图，我们都可以得到 9 幅池化结果图。以上面表格中的 sacle1 为例，Layer-5 pre-pool 大小是 17*17，经过池化后，大小就是 5*5，然后有 3*3 张结果图(不同 offset 得到的结果)。

从 Layer-5 post-pool 到 Classifier map(pre-reshape)：我们知道在训练的时候，从卷积层到全连接层，输入的大小是 4096*(5*5)，然后进行全连接，得到 4096*(1*1)。但是我们现在输入的是各种不同大小的图片，因此接着就采用 FCN 的招式，让网络继续前向传导。我们从 Layer-5 post-pool 到第六层的时候，如果把全连接看成是卷积，那么其实这个时候卷积核的大小为 5*5，因为训练的时候，Layer-5 post-pool 得到的结果是 5*5。因此在预测分类的时候，假设 Layer-5 post-pool 得到的是 7*9(上面表格中的 scale 3)，经过 5*5 的卷积核进行卷积后，它将得到(7-5+1)*(9-5+1)=3*5 的输出。

然后我们就只需要在后面把它们拉成一维向量摆放就 ok 了，这样在一个尺度上，我们可以得到一个 C*N 的预测值矩阵，每一列就表示图片属于某一类别的概率值，然后我们求取每一列的最大值，作为本尺度的每个类别的概率值。我们一共用了 6 种不同尺度做了预测，然后把这六种尺度结果再做一个平均，作为最后的结果。

从上面过程，我们可以看到整个网络分成两部分：layer 1~5 这五层我们把它称之为特征提取层；layer 6~output 我们把它们称之为分类层。