AlexNet 是 Hinton 在 ISVRC2012 中使用的神经网络模型，top5 测试错误率是 15.3%。

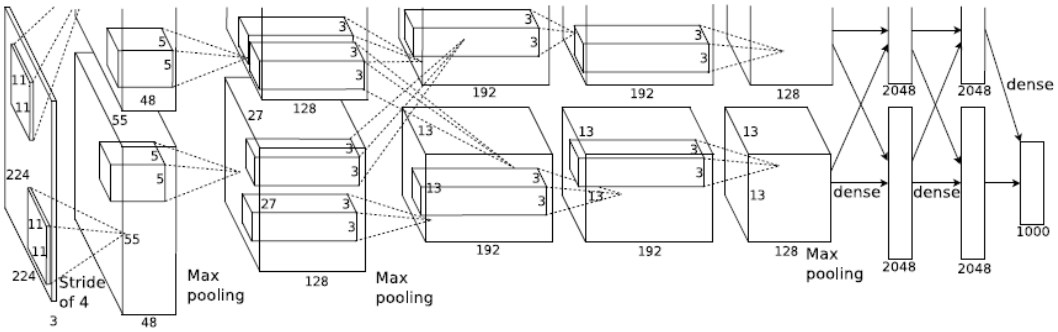


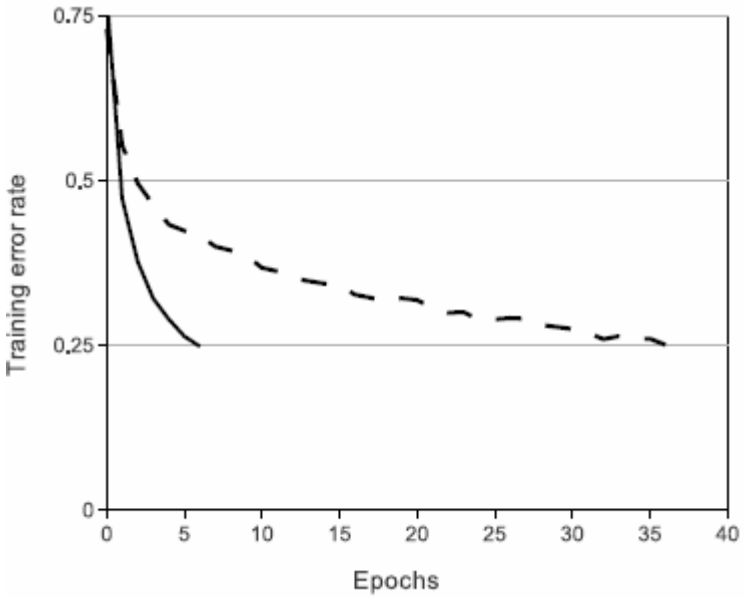
Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

这是论文中的模型结构。输入为 224*224*3，实际上输入为 227*227*3 会更好。AlexNet 包含 5 个卷积层，3 个全连接层，输出 1000*1 向量，输入到 1000 类的 softmax 分类器，得到分类结果。第 2,4,5 卷积层的卷积核只连接同一个 GPU 上的前一层的特征图,第 3 层则连接第 2 层所有特征图。全连接层中的神经元与前一层是全连接的。

下面主要说一下 Alexnet 的创新点：

1. 非线性激活函数 Relu 的使用

   Relu 应用在所有卷积层和全连接层



   在 CIFAR-10 的数据集上，在达到 25%训练误差时，Relu 比 tanh 快了 6 倍，说明其有更快的收敛速度。Rule 在大于 0 时是线性的，不存在过饱和区。

2. 局部相应归一化（Local Response Normalization）LRN

   在 Relu 激活后，使用 LRN，有利与增加泛化能力，局部归一化的目的是侧抑制，Relu 响应范围是无边界的，因此需要归一化。LRN 只应用在前俩个卷积层的 Relu 上。

$$b_{x,y}^{i}=a_{x,y}^{i}/(k+\alpha\sum_{j=max(0,i-n/2)}^{min(N-1,i+n/2)}(a_{x,y}^{j})^{2})$$

其中a代表在feature map中第i个卷积核(x,y)坐标经过了ReLU激活函数的输出，n表示相邻的几个卷积核。N表示这一层总的卷积核数量。k, n, α和β是hyper-parameters，他们的值是在验证集上实验得到的，其中k = 2，n = 5，α = 0.0001，β = 0.75。

这里的归一化采用的是通道上的归一化。也就是说每个点的平方和是沿着 a 的第三维 channel 方向的。总的来说 LRN 是在特征图的通道上做归一化操作

LRN 的优点是对局部神经元创建竞争机制，是的响应大的变的更大，响应小的变得更小，做了平滑处理增强了泛化能力。

但在 VGG 中提到了 LRN 并没有卵用，反而增加了计算内存。

3. 重叠池化

Overlapping Pooling 用在第一层、第二层和第五层之后。

首先，池化可以显著减少参数，还具有平移不变性。

之前传统的池化操作是 s=z，就是池化层的大小等于步长，Alexnet 则采用步长小于池化层大小 s=2，z=3，这减少了 top-5 和 top-1 错误率的 0.4%和 0.3%，在训练过程中，我们发现重叠池化可以避免过拟合

这里再说一下池化

Hinton 层说卷积层中加入池化操作是一个巨大的错误，他表现优异则是一场灾难。池化常常会丢失位置信息，这一定程度实现了平移不变性，但是也损失了空间的相对关系。但是如果卷积层之间有足够的重叠，那么通过稀疏编码可以保留一些位置信息。

4.减少过拟合

（1）增强数据集

（2）Dropout

Dropout 就是随机丢弃一定概率的神经元。因为训练时采用 mini-batch 的方法，Dropout 的有 ensemble 的功效，有利于减小误差。

具体看一下模型结构：（转载 CSDN--学思行仁）

首先总体概述下：

AlexNet 为 8 层结构，其中前 5 层为卷积层，后面 3 层为全连接层；学习参数有 6 千万个，神经元有 650,000 个

AlexNet 在两个 GPU 上运行；

AlexNet 在第 2,4,5 层均是前一层自己 GPU 内连接，第 3 层是与前面两层全连接，全连接是 2 个 GPU 全连接；

RPN 层第 1,2 个卷积层后；

Max pooling 层在 RPN 层以及第 5 个卷积层后。

ReLU 在每个卷积层以及全连接层后。

卷积核大小数量：

conv1:96 11*11*3(个数/长/宽/深度)

conv2:256 5*5*48
conv3:384 3*3*256
conv4: 384 3*3*192
conv5: 256 3*3*192
ReLU、双 GPU 运算：提高训练速度。(应用于所有卷积层和全连接层)
重叠 pool 池化层：提高精度，不容易产生过度拟合。(应用在第一层，第二层，第五层后面)
局部响应归一化层(LRN)：提高精度。(应用在第一层和第二层后面)
Dropout：减少过度拟合。(应用在前两个全连接层)

第一层

第一层输入数据为原始图像的 227*227*3 的图像（最开始是 224*224*3，为后续处理方便必须进行调整）,这个图像被 11*11*3（3 代表深度，例如 RGB 的 3 通道）的卷积核进行卷积运算，卷积核对原始图像的每次卷积都会生成一个新的像素。卷积核的步长为 4 个像素，朝着横向和纵向这两个方向进行卷积。由此，会生成新的像素；（227-11）/4+1=55 个像素(227 个像素减去 11，正好是 54，即生成 54 个像素，再加上被减去的 11 也对应生成一个像素)，由于第一层有 96 个卷积核，所以就会形成 55*55*96 个像素层，系统是采用双 GPU 处理，因此分为 2 组数据：55*55*48 的像素层数据。

重叠 pool 池化层：这些像素层还需要经过 pool 运算（池化运算）的处理，池化运算的尺度由预先设定为 3*3，运算的步长为 2，则池化后的图像的尺寸为：(55-3) /2+1=27。即经过池化处理过的规模为 27*27*96.

局部响应归一化层(LRN)：最后经过局部响应归一化处理，归一化运算的尺度为 5*5；第一层卷积层结束后形成的图像层的规模为 27*27*96.分别由 96 个卷积核对应生成，这 96 层数据氛围 2 组，每组 48 个像素层，每组在独立的 GPU 下运算。

第 2 层分析:

第二层输入数据为第一层输出的 27*27*96 的像素层（为方便后续处理，这对每幅像素层进行像素填充），分为 2 组像素数据，两组像素数据分别在两个不同的 GPU 中进行运算。每组像素数据被 5*5*48 的卷积核进行卷积运算，同理按照第一层的方式进行：(27-5+2*2) /1+1=27 个像素，一共有 256 个卷积核，这样也就有了 27*27*128 两组像素层。

重叠 pool 池化层：同样经过池化运算，池化后的图像尺寸为（27-3）/2+1=13，即池化后像素的规模为 2 组 13*13*128 的像素层。

局部响应归一化层(LRN)：最后经过归一化处理，分别对应 2 组 128 个卷积核所运算形成。每组在一个 GPU 上进行运算。即共 256 个卷积核，共 2 个 GPU 进行运算。

第 3 层分析

第三层输入数据为第二层输出的两组 13*13*128 的像素层（为方便后续处理，这对每幅

像素层进行像素填充），分为 2 组像素数据，两组像素数据分别在两个不同的 GPU 中进行运算。每组像素数据被 3*3*128 的卷积核（两组，一共也就有 3*3*256）进行卷积运算，同理按照第一层的方式进行：(13-3+1*2) /1+1=13 个像素，一共有 384 个卷积核，这样也就有了 13*13*192 两组像素层。

第 4 层分析:

第四层输入数据为第三层输出的两组 13*13*192 的像素层（为方便后续处理，这对每幅像素层进行像素填充），分为 2 组像素数据，两组像素数据分别在两个不同的 GPU 中进行运算。每组像素数据被 3*3*192 的卷积核进行卷积运算，同理按照第一层的方式进行：（13-3+1*2）/1+1=13 个像素，一共有 384 个卷积核，这样也就有了 13*13*192 两组像素层。

第 5 层分析:

第五层输入数据为第四层输出的两组 13*13*192 的像素层（为方便后续处理，这对每幅像素层进行像素填充），分为 2 组像素数据，两组像素数据分别在两个不同的 GPU 中进行运算。每组像素数据被 3*3*192 的卷积核进行卷积运算，同理按照第一层的方式进行：（13-3+1*2）/1+1=13 个像素，一共有 256 个卷积核，这样也就有了 13*13*128 两组像素层。

重叠 pool 池化层：进过池化运算，池化后像素的尺寸为（13-3）/2+1=6，即池化后像素的规模变成了两组 6*6*128 的像素层，共 6*6*256 规模的像素层。

第 6 层分析:

第 6 层输入数据的尺寸是 6*6*256，采用 6*6*256 尺寸的滤波器对第六层的输入数据进行卷积运算；每个 6*6*256 尺寸的滤波器对第六层的输入数据进行卷积运算生成一个运算结果，通过一个神经元输出这个运算结果；共有 4096 个 6*6*256 尺寸的滤波器对输入数据进行卷积，通过 4096 个神经元的输出运算结果；然后通过 ReLU 激活函数以及 dropout 运算输出 4096 个本层的输出结果值。

很明显在第 6 层中，采用的滤波器的尺寸（6*6*256）和待处理的 feature map 的尺寸（6*6*256）相同，即滤波器中的每个系数只与 feature map 中的一个像素值相乘；而采用的滤波器的尺寸和待处理的 feature map 的尺寸不相同，每个滤波器的系数都会与多个 feature map 中像素相乘。因此第 6 层被称为全连接层。

第 7 层分析:

第 6 层输出的 4096 个数据与第 7 层的 4096 个神经元进行全连接，然后经由 ReLU 和 Dropout 进行处理后生成 4096 个数据。

第 8 层分析:
第 7 层输入的 4096 个数据与第 8 层的 1000 个神经元进行全连接，经过训练后输出被

训练的数值。