# 背景

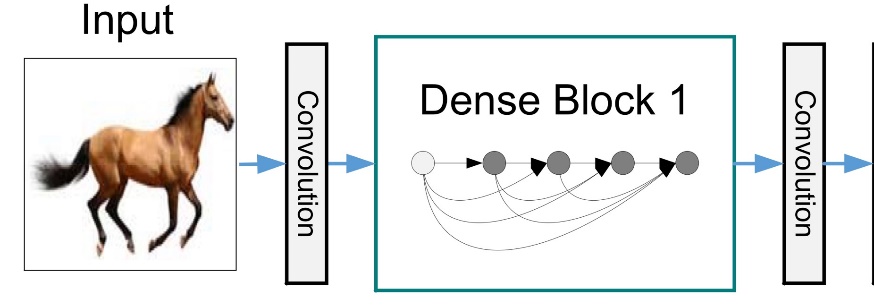
例如像ResNet、Highway Networks等这样的网络，它们的共同之处就在于都是将浅层特征和高层特征创建了一路连接。因此DenseNet也使用了这样的思想，同时受到《Deep networks with stochastic depth》的启发，神经网络其实并不一定要是一个递进层级结构，也就是说网络中的某一层可以不仅仅依赖于紧邻的上一层的特征，而可以依赖于更前面层学习的特征，因此DenseNet将之前所有的浅层特征都连接到当前的特征层上。

和ResNet相比，差别在于ResNet是将前一层和当前层的特征以相加的方式(通道数不变，只是特征值对应相加)融合在一起，而DenseNet是以concatenate(通道数的合并，也就是说描述图像本身的特征增加了，而每一特征下的信息是没有增加)的方式融合在一起。

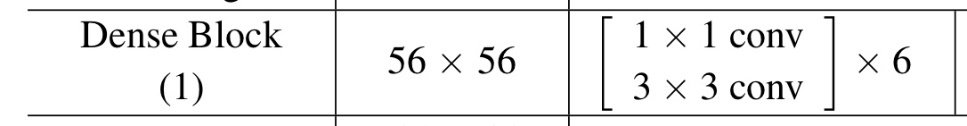
从参数量角度考虑，由于每一层所得到的特征图很大一部分是冗余的，因此DensNet限制了每一个模块输出特征图的个数，将之前所有层的特征图作为网络保留特征部分，当前得到的特征图作为新加入其中的部分，这样实现了对参数的控制，使得网络很窄。

# 方法

①模块内部采用密集连接方式



当前层接收之前所有层的输出。模块内部的特征图大小相同，通过concatenate操作即可融合。通过这种特征复用的方式使得模型在每一层上需要学习的参数很少，由此可以产生高效的模型，而不只是通过网络的深度和宽度刻画网络的特征表达能力。

②瓶颈结构

如上图所示，每一层都是复合操作，包括了BN、ReLU和卷积。并且考虑到计算效率，在每个3×3卷积前都加了1×1卷积用于降维（现在都用这种手段）。一般将1×1卷积的输出通道数设置为4k，k是每一层的输出通道数。

③过渡层

模块和模块之间由于各自输出的特征图大小不同，所以采用过渡层，主要包括BN、1×1卷积、2×2平均池化。以此实现下采样。

# 总结

DenseNet对标ResNet，和ResNet最大的不同就是密集的特征复用。

但是个人觉得将之前的每一层特征都concatenate起来，也会有冗余，所以可以将密集连接简化，比如说后面的block4，可以将其与网络浅层的特征连接断开。