6.ResNet

2015年由何恺明提出，论文地址[Deep Residual Learning for Image Recognition]

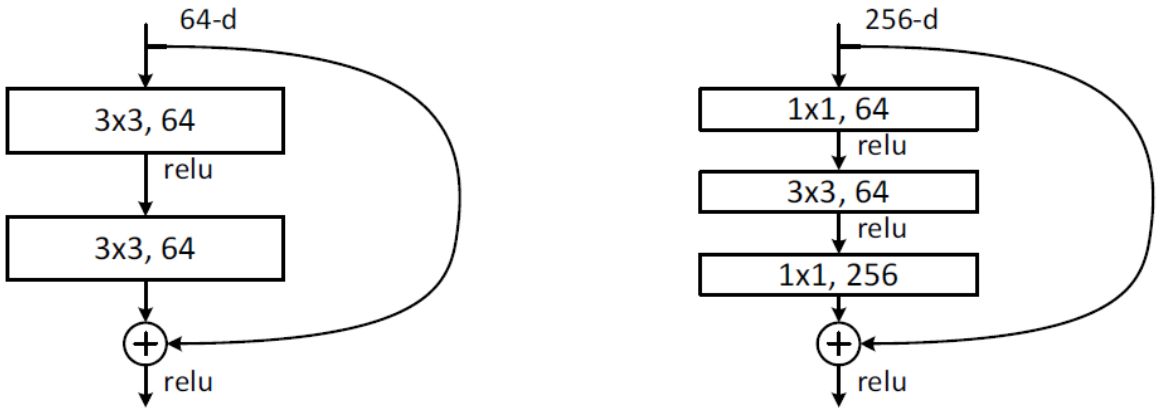


图19 Basicblock，Bottleneck block

注意：主分支与shortcut的输出特征矩阵shape必须相同，1×1的卷积核用来降维和升维

参数个数对比，假设输入为256d：
左边256×256×3×3+256×256×3×3=1179648
右边256×64×1×1+64×64×3×3+256×64×1×1=69632

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

图20 ResNet网络结构

实线与虚线残差结构：

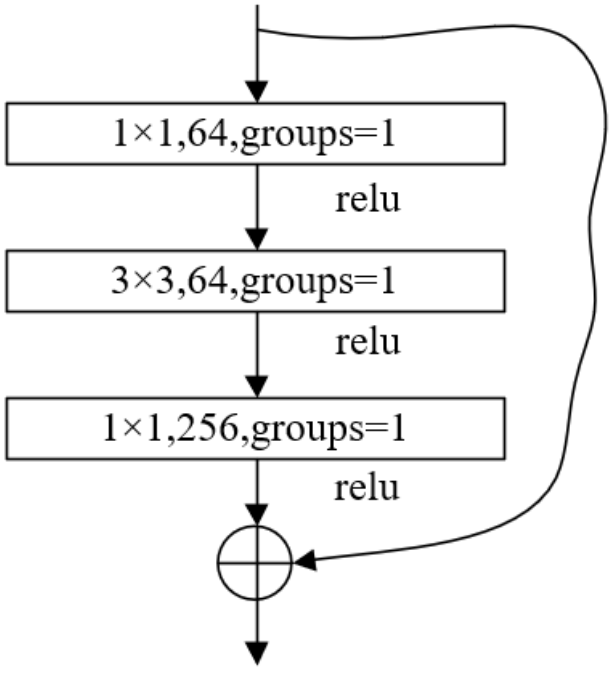
实线：输入与输出特征维度一样
虚线：例如conv3_x第一层，输入[56, 56, 64]，输出[28, 28, 128]，同时虚线有个1×1卷积层进行升维

注意：虚线结构是用来改变尺寸与升降维，对于34层，只有conv3_1, conv4_1, conv5_1有虚线结构，但对于深层的，conv2_1也是虚线，只不过此虚线结构只改变深度。conv1之后池化的输出为[56, 56, 64], 但conv2_1期望的输入是[56, 56, 256], 所以虚线只改变深度。

在torchvision中，bottleneck把下采样放在了3×3位置而不是原先的第一层1×1卷积层。

nn.Conv2d中groups参数理解，例如[64,96,1,1], 默认groups=1，那么输出的每个channel的计算，所有输入的channel都参与其中，如果groups=2，那么输入channels被分成2组，每组32个channels，输出channels被分成2组，每组48个channels。对于输出的2个48的channels，第1个48channels与输入的第一个16channels进行全卷积，第2个48channels与输入的第二个16channels进行全卷积。
当groups=in_channels=out_channels，那么输出的每个channel只与输入对应的channel进行卷积运算，groups决定将输入输出分成几组，所以groups要能被in_channels与out_channels同时整除。

三种残差结构比较：



resnet50
ridth_per_group=64

图21 resnet50

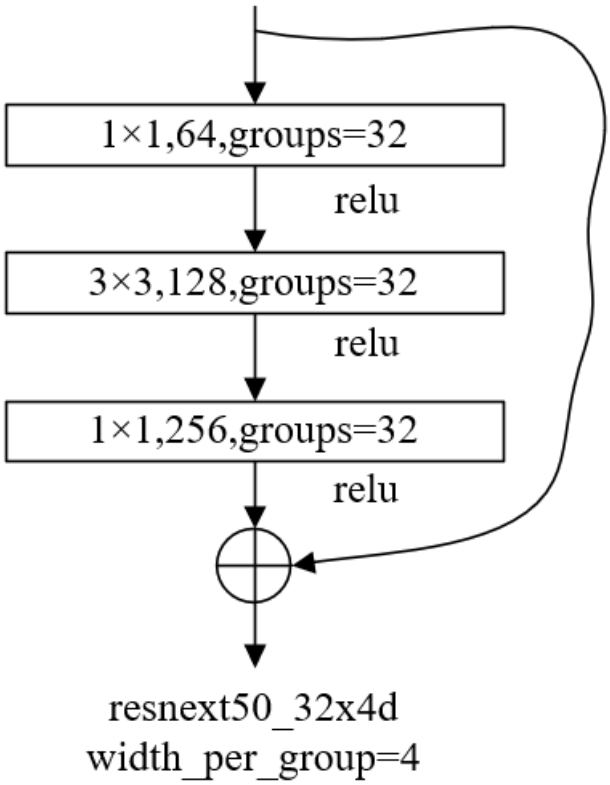


图22 resnext50_32x4d

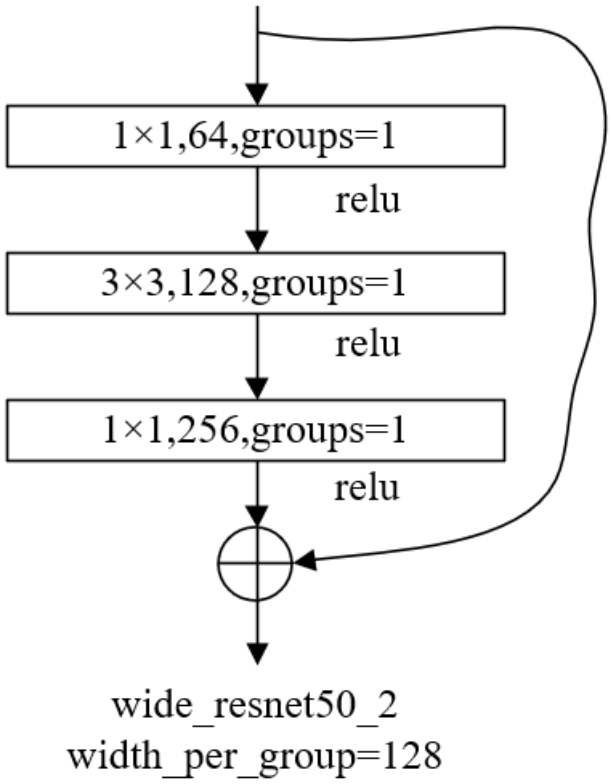


图23 wide_resnet50_2

$$width = int(planes \times \frac{width\_per\_group}{64.} \times groups$$

pytorch实现