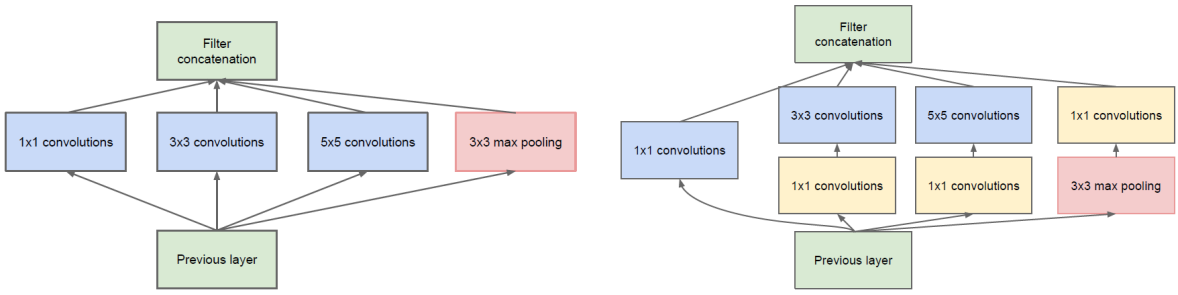
4.GoogLeNet

2014年由Google团队提出，论文地址[Going deeper with convolutions]



(a) Inception module, naïve version　　(b) Inception module with dimension reductions

图4 Inception Module

Filter concatenation是指按深度进行拼接，所以每个分支所得的特征矩阵高和宽必须相同。

（b）相对于（a）多了3个1×1的convolutions进行降维
例如：假设feature map depth=512，不使用1×1conv进行降维与使用1×1conv进行降维的参数对比

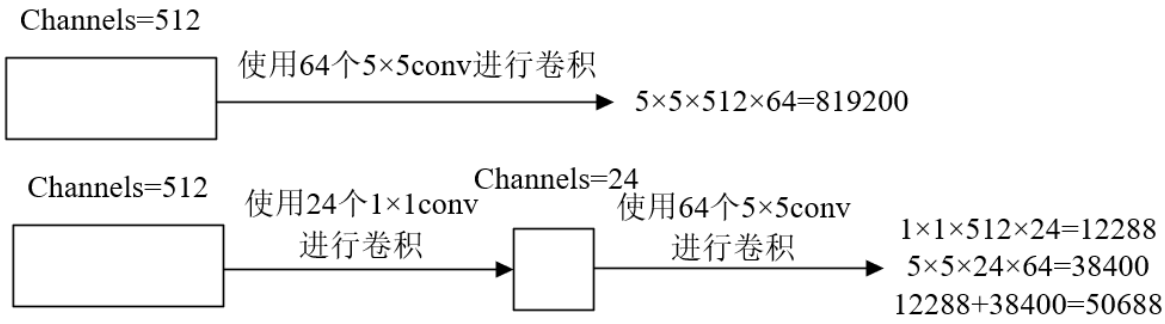


图5 使用1×1卷积参数对比

type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

图6 GoogLeNet网络结构

本文亮点：

1.引入了Inception结构（融合不同尺度的特征信息）

2.使用1×1conv进行降维以及映射处理

3.添加两个辅助分类器帮助训练

4.丢弃全连接层，使用平均池化层（大大减少模型参数）

辅助分类器：加在了4a，4d后面

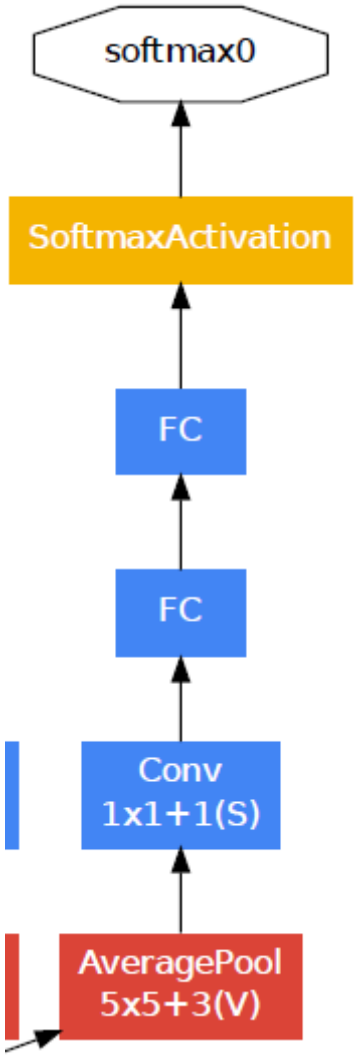


图7 辅助分类器

pytorch实现

@torch.jit.unused：此装饰器向编译器指示应忽略函数或方法，并用引发异常的方法代替。 这样，您就可以在尚不兼容TorchScript的模型中保留代码，并仍然可以导出模型。

namedtuple：具名元组。