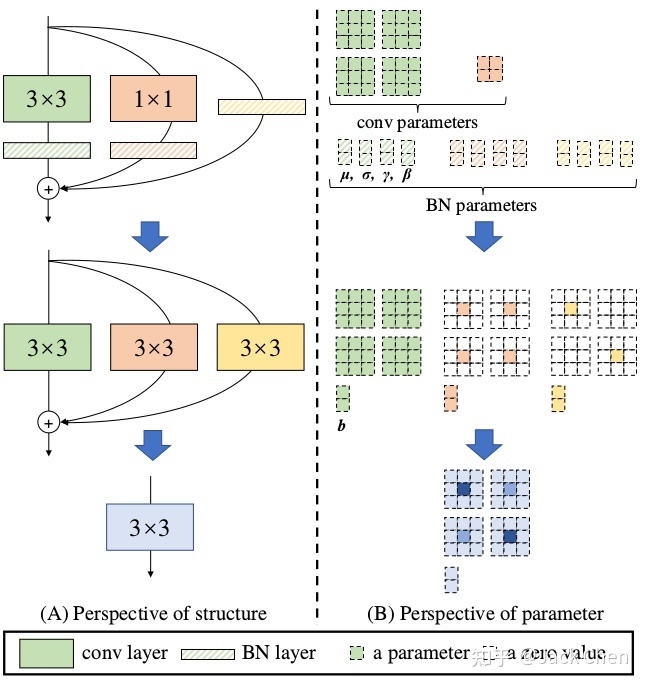
RepVGG

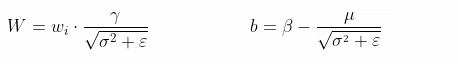


上面训练过程用到的模型涉及到三路：常规的conv\_3x3，conv\_1x1，identity，且这三路每一路都跟着BN层

conv\_3x3:fuse conv and bn

conv:weight,bias=false(usual)

bn:running\_mean,running\_var,weight,bias,eps(防止分母为0)



code:

std = (running\_var + eps).sqrt()

t = (gamma / std).reshape(-1, 1, 1, 1)

return kernel \* t, beta - running\_mean \* gamma / std

conv\_1x1:操作同3x3，先fuse conv and bn，然后padding 0

Identity:用conv\_1x1卷积且卷积核权重为1,只要令当前通道的卷积核参数为1，其余的卷积核参数为0

mutiple path to single path:

conv.weight = conv\_3x3.weight+conv\_1x1.weight+identity.weight

conv.bias = conv\_3x3.bias+conv\_1x1.bias+identity.bias

   warmup顾名思义就是热身，在刚刚开始训练时以很小的学习率进行训练，使得网络熟悉数据，随着训练的进行学习率慢慢变大，到了一定程度，以设置的初始学习率进行训练，接着过了一些inter后，学习率再慢慢变小；学习率变化：上升——平稳——下降；

具体步骤：

        启用warm up，设置warm up setp（一般等于epoch\*inter\_per\_epoch），当step小于warm up setp时，学习率等于基础学习率×(当前step/warmup\_step)，由于后者是一个小于1的数值，因此在整个warm up的过程中，学习率是一个递增的过程！当warm up结束后，学习率以基础学习率进行训练，再学习率开始递减

add concat and elementwise:

add:

****先将对应的特征图相加，再进行卷积操作，相当于加了一个先验****：对应通道的[特征图](https://so.csdn.net/so/search?q=%E7%89%B9%E5%BE%81%E5%9B%BE&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/yangjinyi1314/article/details/_blank)语义类似，从而****对应的特征图共享一个卷积核****

concat:

在channel维度上进行拼接，在channel维度上的拼接分无BN层和有BN层。

（1）无BN层：直接将deconvolution layer 和convolution layer concat。实验结果表明，该方式取得的结果精度较低，低于原有的VGG模型，分析主要的原因是漏检非常严重，原因应该是concat连接的两层参数不在同一个层级，类似BN层用在eltwise层上。

（2）有BN层：在deconvolution layer 和convolution layer 后面加batchnorm和scale层（BN）后再concat。实验结果表明，该方式取得了比原有VGG模型更好的检测效果（表中的迭代次数还没有完哦），增加了2%的精度，但是速度上慢了一些。