**YOLOv8网络结构**

**1. Backbone**

YOLOv8的Backbone同样**参考**了CSPDarkNet-53网络，我们可以称之为CSPDarkNet结构吧，与YOLOv5不同的是，YOLOv8使用C2f(CSPLayer\_2Conv)代替了C3模块(如果你比较熟悉YOLOv5的网络结构，那YOLOv8的网络结构理解起来就easy了)。如图1所示为YOLOv8网络结构图(引用自MMYOLO)，对比图2的YOLOv5结构图，可以看到基本的架构是类似的。

这里值得注意的是，很多博文中写到YOLOv8使用了CSPDarkNet53作为backbone，当然是可以用的，但是官方代码中明显不是套用的CSPDarkNet53网络结构。事实上，YOLOv5的主干也并非是CSPDarkNet53网络。

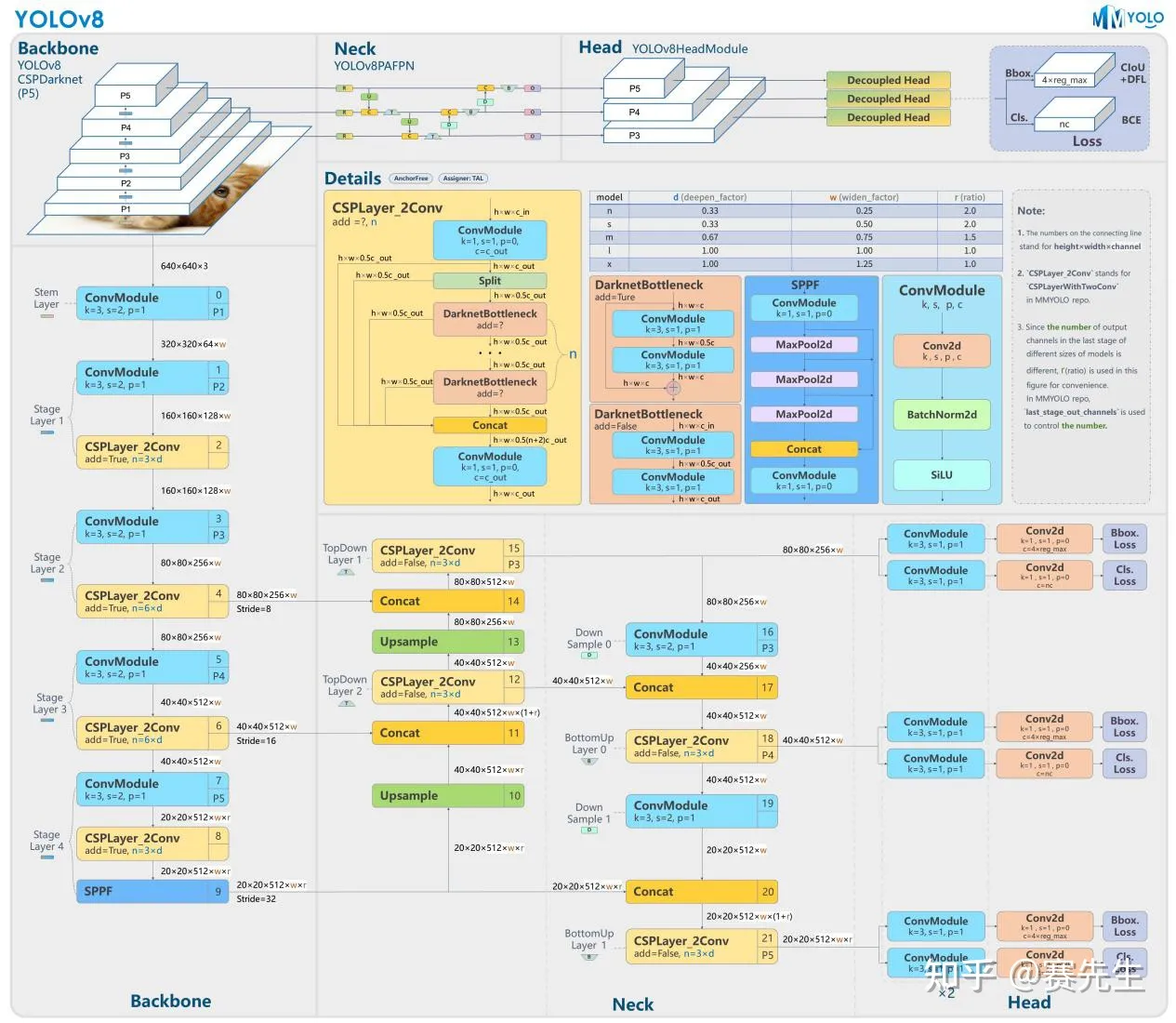


图1

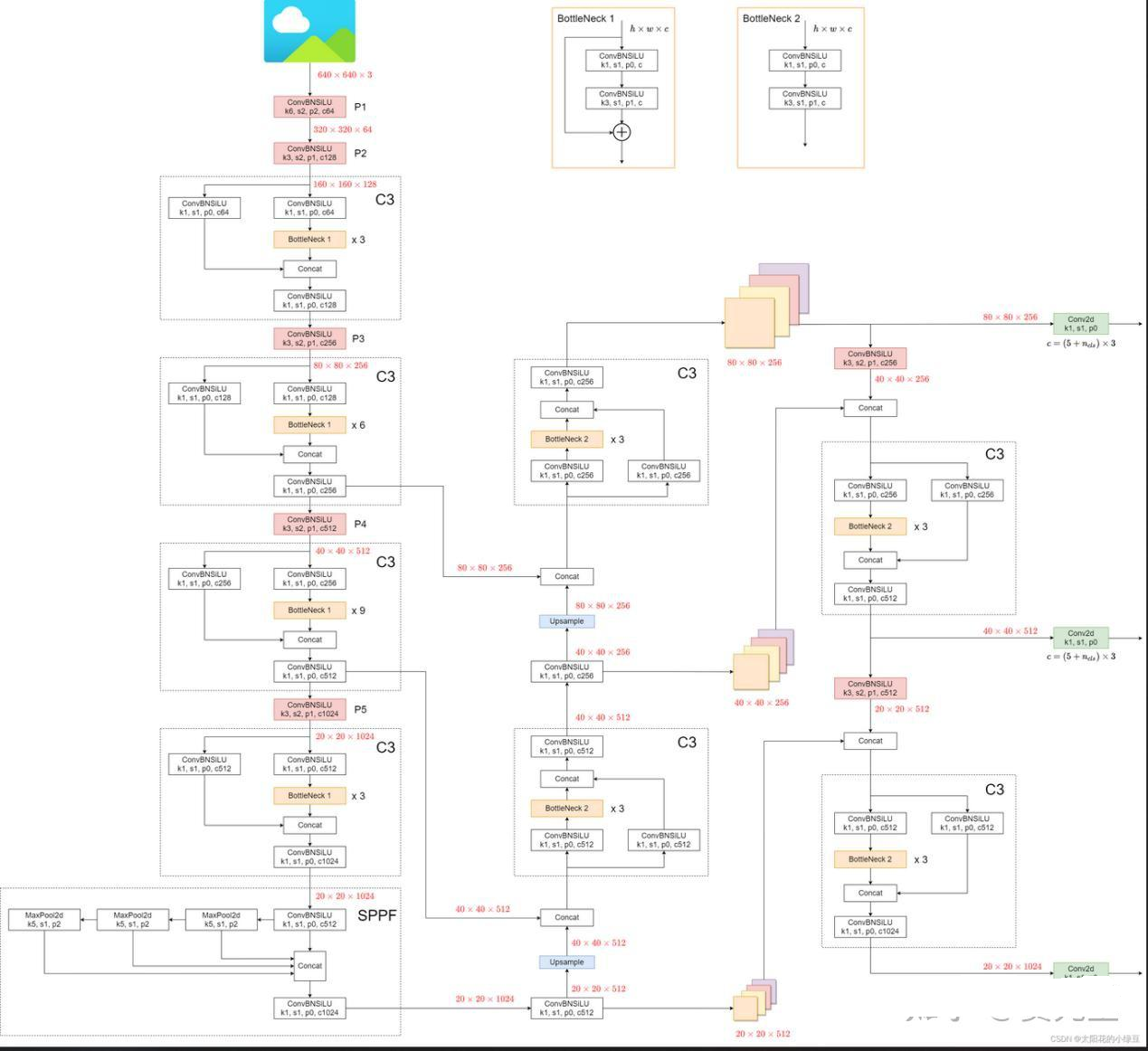


图2

**2. Neck**

YOLOv8的Neck使用的也是类似于YOLOv5的PAN-FPN，称作双流FPN，高效，速度快。

**3. Head**

与之前的YOLOv6，YOLOX类似，使用了Decoupled Head，YOLOv3、YOLOv4、YOLOv5均使用Coupled Head。

YOLOv8也使用3个输出分支，但是每一个输出分支又分为2部分，分别来分类和回归边框(参照图1的Decoupled Head)。

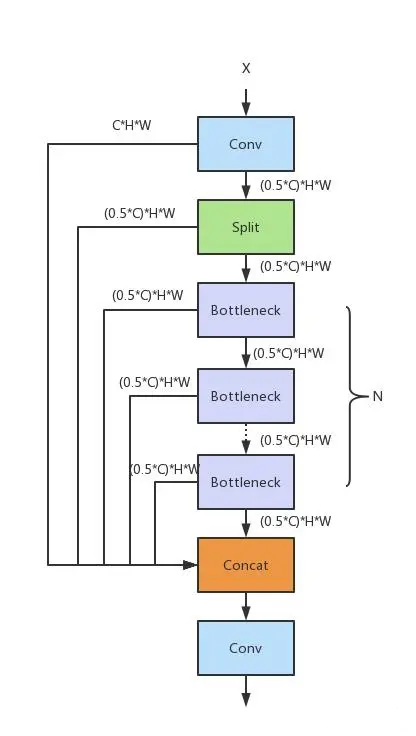
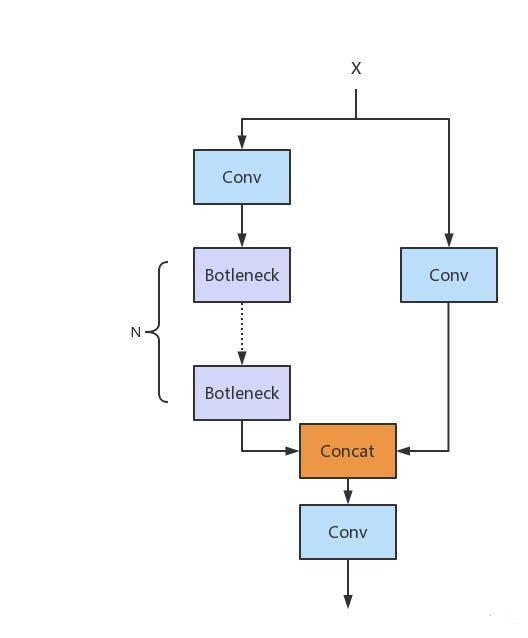
**细说Backbone**

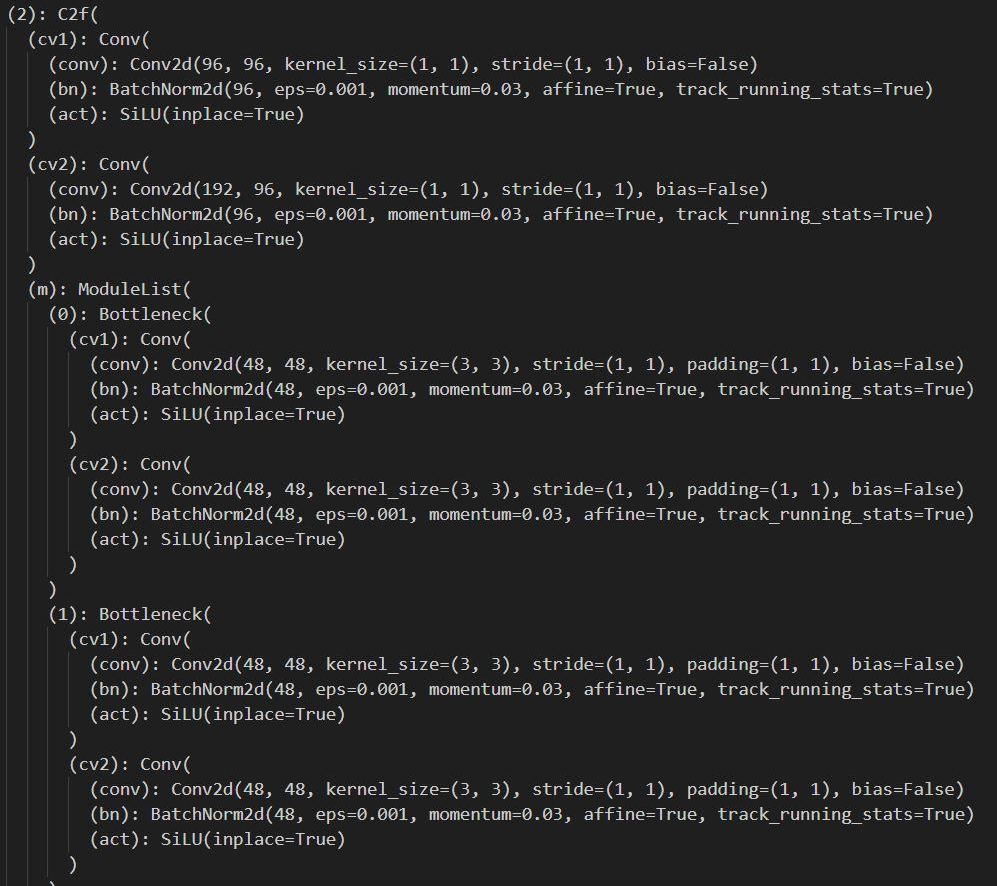
前面讲到，YOLOv8的Backbone类似于YOLOv5的Backbone，不同点是将C3换成了C2F，以及将第一个Convolution层设置为kernel size等于3，stride为2（YOLOv5的Kernel Size为6，padding为2）。

**C2F与C3对比**

那么C2F与C3单元相比，有什么优势呢？我们先上各自的网络结构图。我们可以参考图3的C3结构图，图4-1的C2F结构图，以及图4-2的网络构建进行理解。

图4中，进入到Bottleneck计算序列的输入Tensor的Channel都只有C2F输入通道的0.5倍，因此计算量明显降低。从另一方面讲，梯度流的增加，也能够明显提升收敛速度和收敛效果。

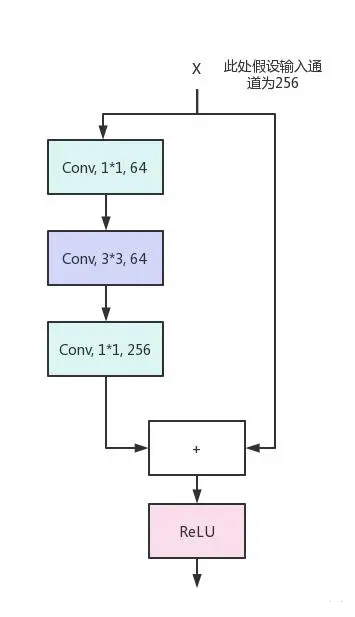
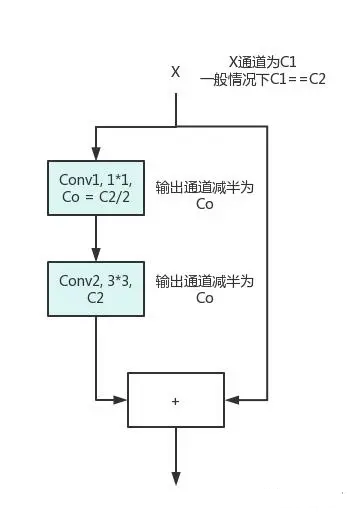




**Bottleneck**

YOLOv8的C2F使用了Bottleneck单元，但需要注意的是，Darknet所引入的Bottleneck不同于ResNet的Bottleneck。如图5和图6分别为Darknet的Bottleneck和ResNet的Bottelneck。

由图5和图6可以看出，Darknet的Bottleneck单元并未使用最后的1\*1卷积进行通道的恢复，而是直接在中间的3\*3卷积中进行了恢复。*Bottleneck可以大大减少参数，降低计算量。*



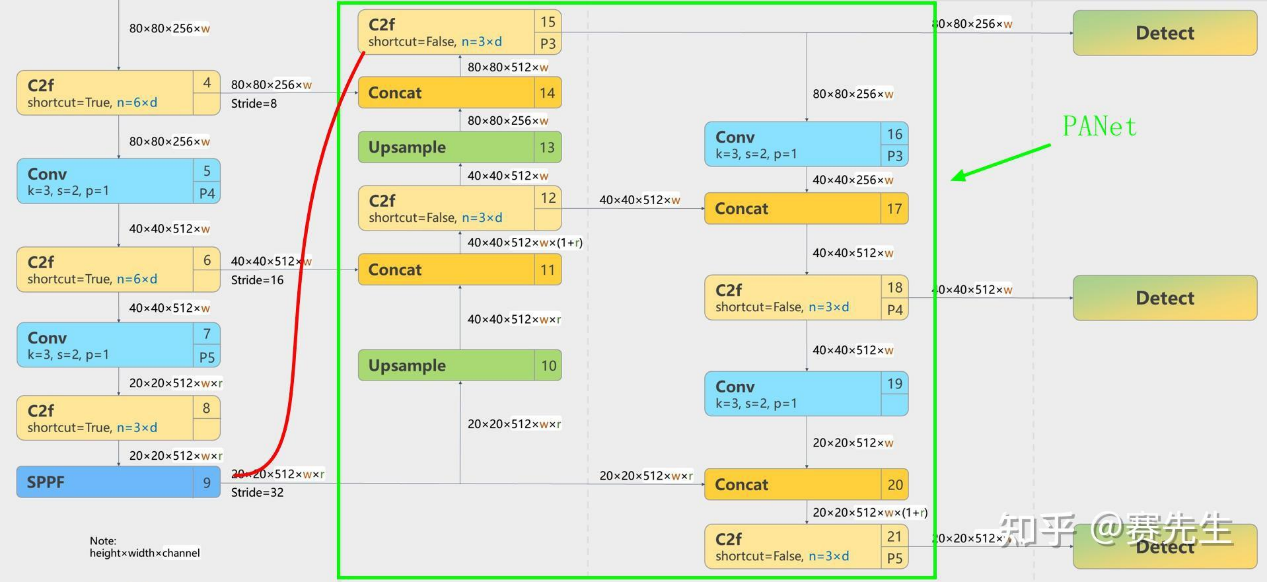
**Neck**

YOLOv8的Neck采用了PANet结构。如图7为网络局部图。

由图7可以看出，Backbone最后经过了一个SPPF(SPP Fast，图示Layer9)，之后H和W已经经过了32被的下采样。对应的，Layer4经过了8被下采样，Layer6经过了16背的下采样。设定输入为640\*640，得到Layer4、Layer6、Layer9的分辨率分别为80\*80、40\*40和20\*20。

Layer4、Layer6、Layer9作为PANnet结构的输入，经过上采样，通道融合，最终将PANet的三个输出分支送入到Detect head中进行Loss的计算或结果解算。

与FPN(单向，自上而下)不同，PANet是一个双向通路网络。与FPN相比，PANet引入了自下向上的路径，使得底层信息更容易传递到高层顶部(红色曲线标注路线)。



**Detect Head**

YOLOv8采用了类似于YOLOX的Decoupled Head，将回归分支和预测分支进行分离。Decoupled Head的有点可以参考YOLOX的论文中提到的，收敛更快，效果更好。

需要特别提及的是，YOLOv8的Detect Head中，针对回归分支使用了DFL策略，之前的目标检测网络将回归坐标作为一个确定性单值进行预测，DFL将坐标转变成了一个分布。

DFL理论主要用来解决边界模糊的问题。