YOLOv5与YOLOv8的分析比较（以车牌号码识别为例）

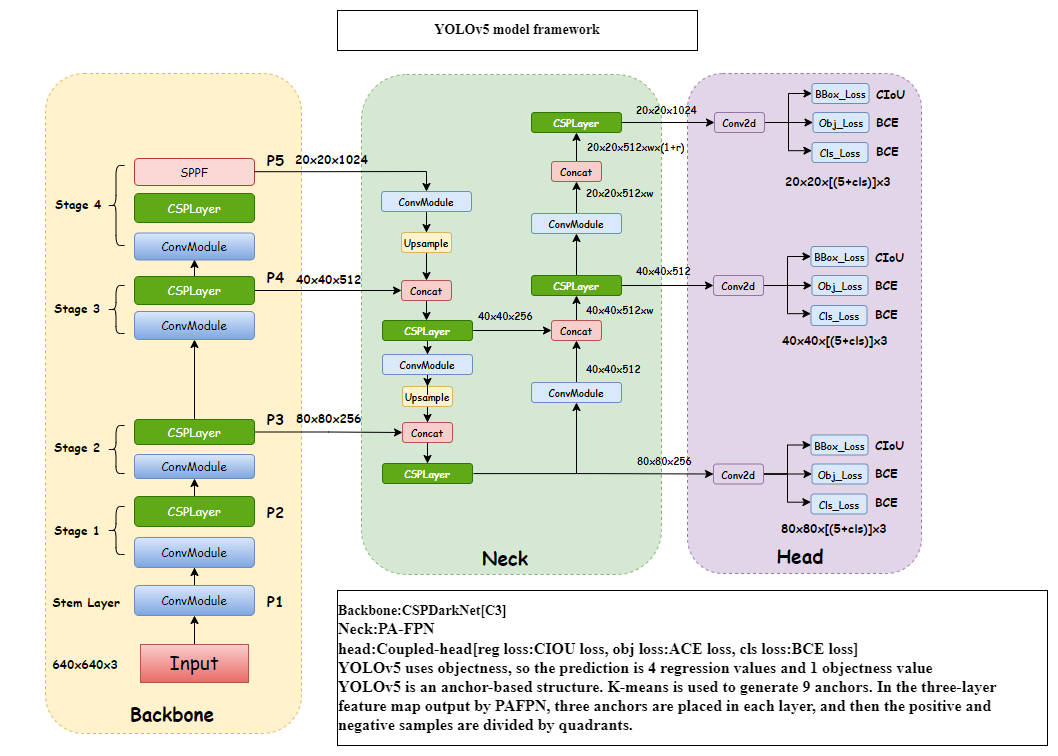
# 一：YOLOv8和YOLOv5有哪些改变

1.YOLOv8和YOLOv5的结构图

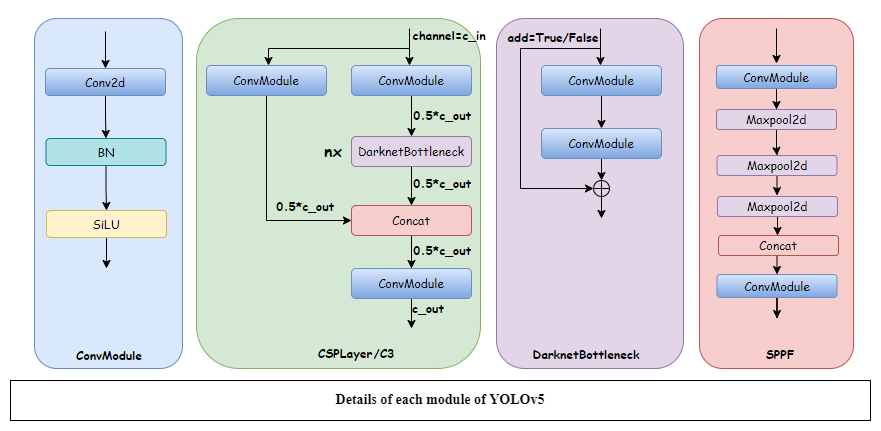
YOLOv8 和 YOLOv5 出自同一团队，故 YOLOv8 也可以看做在 YOLOv5 上的优化，结构也很类似，下面来对比一下 YOLOv5 和 YOLOv8 的异同点。

|  |  |  |
| --- | --- | --- |
| Module | YOLOv8 | YOLOv5 |
| Input size | 640\*640\*3 | 640\*640\*3 |
| Backbone | C2f(Added more information about gradient flow | C3 |
| Neck | PA-FPN | PA-FPN |
| Head | Decoupling head | Coupling head |
| Anchor | Anchor-free | Anchor-based |
| Label | TAL | gt center point |
| Objectness | No | Yes |
| output | 20\*20\*(4+cls)/40\*40\*(4+cls)/80\*80\*(4+cls) | 20\*20\*(5+cls)/40\*40\*(5+cls)/80\*80\*(5+cls) |

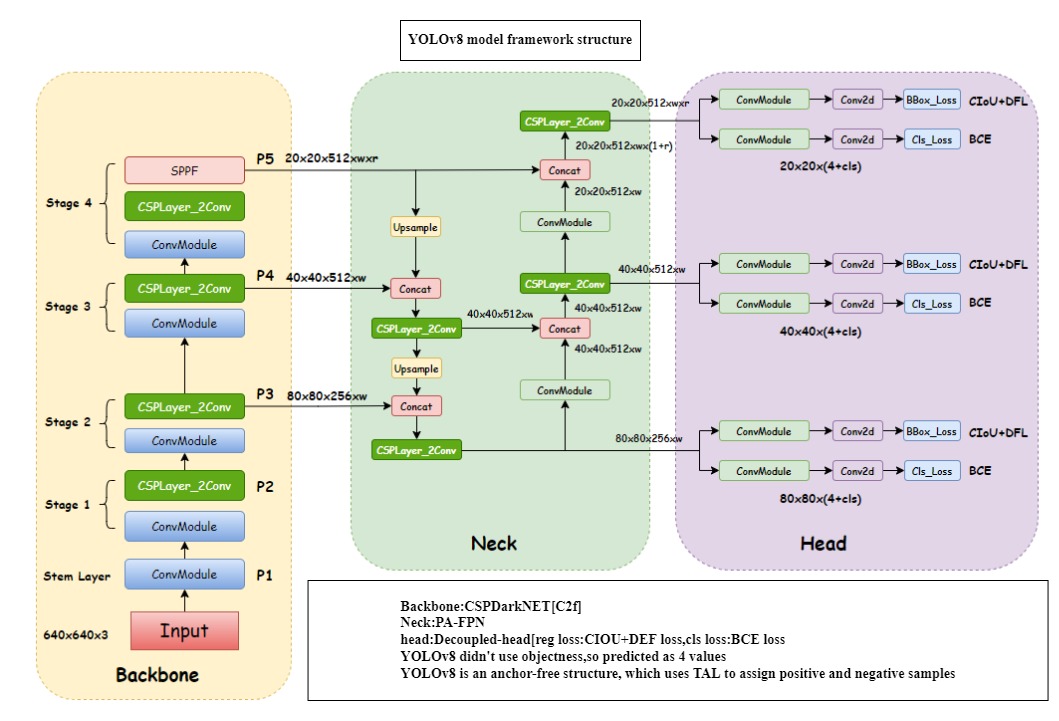
YOLOv5 模型框架如下：



YOLOv5 模块细节如下：



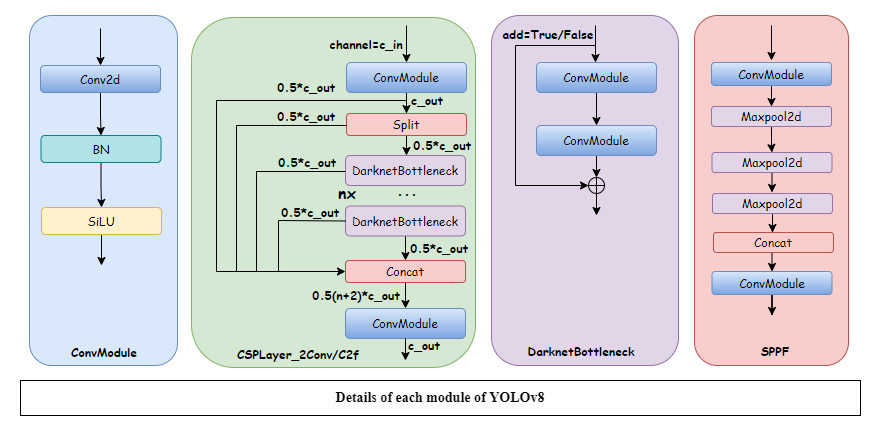
YOLOv8 模型框架结构如下：



YOLOv8 模块细节如下：

1.主要差别就在于 YOLOv8 中使用了梯度更丰富（源于 YOLOv7 中的 E-ELAN 模块）的 C2f 模块

2.YOLOv5 中使用的是只进行了一次梯度分流的 C3 模块



1. YOLOv8的主要改动

核心特性和改动可以归结为如下：

1. 提供了一个全新的 SOTA 模型，包括 P5 640 和 P6 1280 分辨率的**[目标检测](https://www.zhihu.com/search?q=%E7%9B%AE%E6%A0%87%E6%A3%80%E6%B5%8B&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra={"sourceType":"answer","sourceId":2842111496}" \t "_blank)**网络和基于 **[YOLACT](https://link.zhihu.com/?target=https://arxiv.org/abs/1904.02689" \t "_blank)** 的实例分割模型。和 YOLOv5 一样，基于缩放系数也提供了 N/S/M/L/X 尺度的不同大小模型，用于满足不同场景需求
2. 骨干网络和 Neck 部分可能参考了 YOLOv7 ELAN 设计思想，将 YOLOv5 的 C3 结构换成了梯度流更丰富的 C2f 结构，并对不同尺度模型调整了不同的通道数，属于对模型结构精心微调，不再是无脑一套参数应用所有模型，大幅提升了模型性能。不过这个 C2f 模块中存在 Split 等操作对特定硬件部署没有之前那么友好了
3. Head 部分相比 YOLOv5 改动较大，换成了目前主流的解耦头结构，将分类和检测头分离，同时也从 Anchor-Based 换成了 Anchor-Free
4. Loss 计算方面采用了 TaskAlignedAssigner 正样本分配策略，并引入了 Distribution Focal Loss
5. 训练的数据增强部分引入了 YOLOX 中的最后 10 epoch 关闭 Mosiac 增强的操作，可以有效地提升精度
6. 从上面可以看出，YOLOv8 主要参考了最近提出的诸如 YOLOX、YOLOv6、YOLOv7 和 PPYOLOE 等算法的相关设计，本身的创新点不多，偏向**[工程实践](https://www.zhihu.com/search?q=%E5%B7%A5%E7%A8%8B%E5%AE%9E%E8%B7%B5&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra={"sourceType":"answer","sourceId":2842111496}" \t "_blank)**，主推的还是 ultralytics 这个框架本身。

下面将按照Loss 计算、训练数据增强、训练策略和模型推理过程共 4 个部分详细介绍 YOLOv8 目标检测的各种改进，实例分割部分暂时不进行描述。

1. loss计算

Loss 计算过程包括 2 个部分： 正负样本分配策略和 Loss 计算。YOLOv8 算法中则直接引用了 TOOD 的 TaskAlignedAssigner。他的匹配策略简单总结为： 根据分类与回归的分数加权的分数选择正样本。

公式：

s 是标注类别对应的预测分值，u 是预测框和 gt 框的 iou，两者相乘就可以衡量对齐程度。

Loss 计算包括 2 个分支： 分类和回归分支，没有了之前的 objectness 分支。

分类分支依然采用 BCE Loss

回归分支需要和 Distribution Focal Loss 中提出的积分形式表示法绑定，因此使用了 Distribution Focal Loss， 同时还使用了 CIoU Loss，3 个 Loss 采用一定权重比例加权即可。

1. 训练数据增强

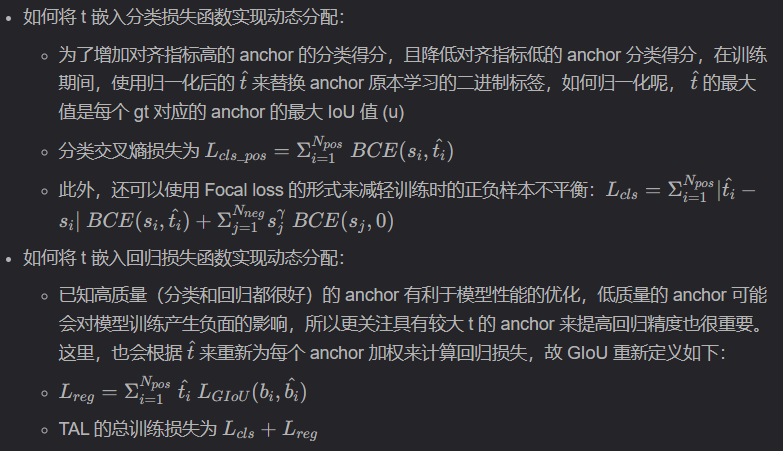
数据增强方面和 YOLOv5 差距不大，只不过引入了 YOLOX 中提出的最后 10 个 epoch 关闭 Mosaic 的操作。考虑到不同模型应该采用的数据增强强度不一样，因此对于不同大小模型，有部分超参会进行修改，典型的如大模型会开启 MixUp 和 CopyPaste。

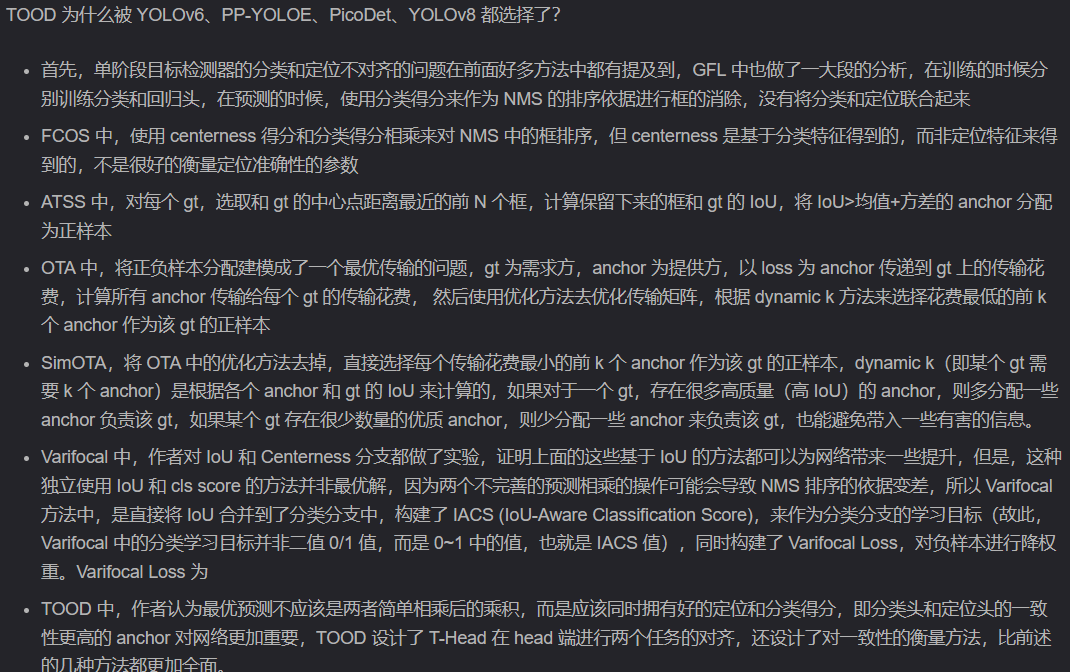
1. 训练策略

YOLOv8 的训练策略和 YOLOv5 没有啥区别，最大区别就是模型的训练总 epoch 数从 300 提升到了 500，这也导致训练时间急剧增加

1. 模型推理过程

YOLOv8 的推理过程和 YOLOv5 几乎一样，唯一差别在于前面需要对 Distribution Focal Loss 中的积分表示 bbox 形式进行解码，变成常规的 4 维度 bbox，后续计算过程就和 YOLOv5 一样了。





# 二：性能度量指标

对于机器学习模型的性能评判好坏（即泛化能力），比如对测试集的预测和分类结果好不好。但为了量化评判的标准，我们有一系列的性能度量，有了指标，我们在对比不同的模型时便可量化比较其好坏。同时，通过指标进一步调参不断优化我们的模型。

1. 混淆矩阵（confusion matrix）

如果我们有一个二分类的模型，那么把预测的情况与实际情况结果两两组合，就会有四种情况，如下表所示：

P(positive): 代表1 N(Negative):代表0

T(True):预测正确 F(False):预测错误

|  |  |  |  |
| --- | --- | --- | --- |
|  | | True | |
| 1 | 0 |
| predicted | 1 | TP | FP |
| 0 | FN | TN |

1. 精准率（Precision）

这个指标是针对预测结果计算，在所有被预测为正的样本中实际为正的样本的概率。也就是在预测为正的样本中，我们预测正确的概率。公式如下：

1. 准确率(accuracy)

定义：预测正确的结果占总样本的百分比

公式如下：

精准率与准确率的区别：精准率代表正样本结果中的预测准确程度。准确率代表整体的预测准确程度。

1. 召回率（Recall）

定义：在实际为正的样本中被预测为正样本的概率。

公式如下：

1. 查准率-查全率（PR\_curve）

我们希望查准率（precision）和查全率（recall）同时都非常高，但实际这两个指标是一对矛盾体，其中一个非常高，另一个肯定会非常低，选择合适的阈值点要根据实际需求。比如我们需要高的查全率，就会牺牲一些查准率，在保证查全率最高的情况下，查准率也不那么低。

1. F1分数（F1\_curve）

F1分数就是用来平衡精准率和召回率的，他同时考虑了两者，让两者同事达到最高，取一个平衡，公式如下：

1. 检测指标

mAP：average of AP for each class

AP: average of precision at different recalls, PR曲线下的面积

TP: 与ground truth的IOU > 0.5

FP: 与ground truth的IOU <= 0.5，同一个ground truth的多余检测框

FN: 没有检测到的ground truth

mAP@.5：IOU>0.5认为是TP

mAP@.5:.95：IOU分别取0.5,0.55,0.6,0.65 ... 0.95，得到多个mAP，然后取平均

三：实验对比分析（以识别车牌号码为例）

1. 数据集

通过roboflowx获取开源的车牌标注数据集，并按7:2:1划分成train，valid, test各1414,404,202张图片。

按照YOLOv5 text数据集格式标注，包含0-9数字，和A-Z共36个标签。

1. 训练模型参数设置

|  |  |  |
| --- | --- | --- |
| Model | YOLOv8 | YOLOv5 |
| Input img | 640\*640 | 640\*640 |
| Batch | 16 | 16 |
| epoch | 100 | 100 |
| model | Yolov8s | Yolov5s |
| weights | Yolov8s.pt | Yolov5s.pt |
| layers | 225 | 214 |
| parameters | 11149562 | 7116721 |

1. 训练结果比较

3.1 confusion\_matrix

从混淆矩阵中可以明显看出YOLOv5预测错误的次数明显多于YOLOv5.

|  |  |
| --- | --- |
| confusion_matrix | confusion_matrix |
| YOLOv8 | YOLOv5 |

3.2 results

|  |
| --- |
| results |
| YOLOv5 results |
| results |
| YOLOv8 results |

|  |  |  |
| --- | --- | --- |
| Model | YOLOv8 | YOLOv5 |
| Precision | 0.97604 | 0.97039 |
| recall | 0.96264 | 0.95416 |
| mAP50 | 0.97865 | 0.9749 |
| mAP50-90 | 0.80059 | 0.78609 |

从结果可以看出：在训练100个epoch的情况下，YOLOv8的mAP\_0.5:0.95最高达到0.80059, 而YOLOv5只达到0.78609。其他指标YOLOv8均优于YOLOv5。

3.3 F1-curve

|  |
| --- |
| F1_curve |
| YOLO v8 F1\_curve |
| F1_curve |
| YOLO v5 F1\_curve |

可以看出YOLOv8比YOLOv5更早达到峰值。

3.4 车牌检测结果比较

从识别结果可以看出，在车牌清晰的情况下，两者都能很好地识别准确，但YOLOv8相较于YOLOv5能获得更高的精准率，但在车牌模糊不清，有重影的情况下，均较难识别出来，但这只是特殊情况。在同样光线较弱的情况下，YOLOv8效果更好一些。

|  |
| --- |
| val_batch0_pred |
| YOLOv8\_batch0\_pred |
| val_batch0_pred |
| YOLOv5\_batch0\_pred |

|  |
| --- |
| val_batch1_pred |
| YOLOv8\_batch0\_pred |
| val_batch1_pred |
| YOLOv5\_batch1\_pred |

四：结论

本文详细分析和总结了最新的 YOLOv8 算法，并同YOLOv5进行比较，从整体设计到模型结构、Loss 计算、训练数据增强、训练策略和推理过程进行了详细的说明。并以车牌识别为例，对比两者在工程应用上的精准度，速度做一个比较。可以看出YOLOv8在各项指标上表现均优于YOLOv5。本文主要对比结果和代码以上传到https://github.com/Chaochao690480/YOLOv8\_licenseplate\_detection，供工程人员或学习者在模型选择上根据自身情况来做一个选择参考。

简单来说 YOLOv8 是一个包括了图像分类、Anchor-Free 物体检测和实例分割的高效算法，检测部分设计参考了目前大量优异的最新的 YOLO 改进算法，实现了新的 SOTA。不仅如此还推出了一个全新的框架。不过这个框架还处于早期阶段，还需要不断完善。

# Литература

1. YOLOv5, <https://github.com/ultralytics/yolov5>
2. YOLOX: Exceeding YOLO Series in 2021, <https://arxiv.org/abs/2107.08430>
3. PP-YOLOE: An evolved version of YOLO, <https://arxiv.org/abs/2203.16250>
4. CSPNet: A New Backbone that can Enhance Learning Capability of CNN, <https://arxiv.org/abs/1911.11929>
5. Path aggregation network for instance segmentation, <https://arxiv.org/abs/1803.01534>
6. OTA: Optimal Transport Assignment for Object Detection, <https://arxiv.org/abs/2103.14259>
7. Computer Architecture: A Quantitative Approach
8. SIoU Loss: More Powerful Learning for Bounding Box Regression, https://arxiv.org/abs/2205.12740