【RT-DETR有效改进】利用MobileNetV3替换Backbone（轻量化网络结构，提点）

发布时间：2025-10-18 12:07:22

前言

大家好，这里是RT-DETR有效涨点专栏。

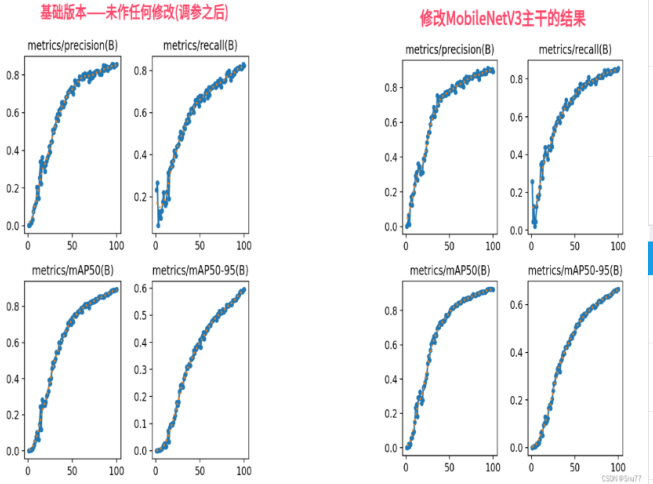
本专栏的内容为根据ultralytics版本的RT-DETR进行改进，内容持续更新，每周更新文章数量3-10篇。

专栏以ResNet18、ResNet50为基础修改版本，同时修改内容也支持ResNet32、ResNet101和PPHGNet版本，**其中ResNet为RT-DETR官方版本1：1移植过来的**，参数量基本保持一致(误差很小很小)，不同于ultralytics仓库版本的ResNet官方版本，同时ultralytics仓库的一些参数是和RT-DETR相冲的所以我也是会教大家调好一些参数和代码，**真正意义上的跑ultralytics的和RT-DETR官方版本的无区别。**

欢迎大家订阅本专栏，一起学习RT-DETR

一、本文介绍

本文给大家带来的改进机制是**MobileNetV3**，其主要改进思想集中在结合硬件感知的网络架构搜索（NAS）和NetAdapt算法，以优化移动设备CPU上的性能。它采用了新颖的架构设计，包括反转残差结构和线性瓶颈层，以及新的高效分割解码器Lite Reduced Atrous Spatial Pyramid Pooling（LR-ASPP），以提升在移动分类、检测和分割任务上的表现。实验表明，MobileNets在资源和准确性的权衡方面表现出色，并在多种应用（如对象检测、细粒度分类、面部属性识别和大规模地理定位）中展现了其有效性，**可以看出其mAP增加了大概三四个点，同时参数量却下降了百分之四十以上！**



专栏链接：RT-DETR剑指论文专栏，持续复现各种顶会内容——论文收割机RT-DETR

目录

最新文章
所有文章
Redis缓存：热点数据查询的数据库减压策略
《超级马里奥派对 空前盛会》NS2版本预告片发布 新功能演示!
2
古代数与现代方法的计算差别对比
日常运维：11G RAC环境GRID目录及文件权限被篡改的修复
看行情的网站有哪些 简单明了易操作
2025-07-03：使字符频率相等的最少操作次数。用go语言，给定一个字符串 s。 如果某个字符串 t 中所有字符的出现次数相
2025 年路沿石厂家 TOP 企业品牌推荐排行榜，五莲花 / 五莲红 / 五莲灰 / 芝麻灰 / 芝麻白 / 芝麻黑 / 黄锈石 / 黄金麻 / 浪淘沙 / 白麻路沿石公司推荐！
Mac玩《枪火重生》攻略，教你如何在苹果电脑上运行《枪火重生》！
ABoVE/ASCENDS：合并大气二氧化碳、甲烷和气象数据，2017
推荐阅读
c语言读写操作-C语言中的文件读写操作及其在实际项目中的应用
环境变量设置文件在软件开发中的应用与实践
仿真出激活码-利用深度学习技术实现高效仿真出激活码的方法
手机模拟鼠标-在手机上模拟鼠标操作的实用技巧与最佳实践
amd驱动程序-探索AMD驱动程序：优化性能与稳定性的关键
c和c 字符串-C与C++面向对象编程的对比与融合实践
阿里巴巴CRM系统：提高客户关系管理效率的关键工具
电子邮箱的格式-电子邮件格式与规范

sql语言是具有 功能-SQL语言在IT领域的重要功能与应用
html before after-使用JavaScript模块化提升应用程序性能与可维护性

二、MobileNetV3的框架原理

Searching for MobileNetV3

Andrew Howard[1] Mark Sandler[1] Grace Chu[1] Liang-Chieh Chen[1] Bo Chen[1] Mingxing Tan[2]
Weijun Wang[1] Yukun Zhu[1] Ruoming Pang[2] Vijay Vasudevan[2] Quoc V. Le[2] Hartwig Adam[1]
[1]Google AI, [2]Google Brain
{howarda, sandler, cxy, lcchen, bochen, tanmingxing, weijunw, yukun, rpang, vrv, qvl, hadam}@google.com

CSDN @Sxu77

官方论文地址： 官方论文地址点击即可跳转

官方代码地址： 官方代码地址

摘要: *我们提出了下一代 MobileNets，它基于一系列互补的搜索技术以及新颖的架构设计。MobileNetV3 通过结合硬件感知的网络架构搜索 (NAS) 以及 NetAdapt 算法进行优化，适应移动电话 CPU，并通过新颖的架构进步进行了改进。本文探索了自动化搜索算法和网络设计如何共同利用互补方法来提升整体技术水平。通过这一过程，我们发布了两个新的 MobileNet 模型：MobileNetV3-Large 和 MobileNetV3-Small，分别针对高资源和低资源使用场景。然后将这些模型适配并应用于目标检测和语义分割任务。对于语义分割（或任何密集像素预测）任务，我们提出了一种新的高效分割解码器 Lite Reduced Atrous Spatial Pyramid Pooling (LR-ASPP)。我们在移动分类、检测和分割方面实现了新的最佳成绩。与 MobileNetV2 相比，MobileNetV3-Large 在 ImageNet 分类的准确度上提高了 3.2%，同时减少了 20% 的延迟。MobileNetV3-Small 与具有相似延迟的 MobileNetV2 模型相比，准确度提高了 6.6%。MobileNetV3-Large 检测在 COCO 检测上的速度比 MobileNetV2 快 25% 以上，准确度大致相同。在 Cityscapes 分割任务中，MobileNetV3-Large LR-ASPP 比 MobileNetV2 R-ASPP 快 34%，准确度相似。*

CSDN @Sxu77

MobileNetV3的主要改进思想集中在结合硬件感知的网络架构搜索（NAS）和NetAdapt算法，以优化移动设备CPU上的性能。它采用了新颖的架构设计，包括反转残差结构和线性瓶颈层，以及新的高效分割解码器Lite Reduced Atrous Spatial Pyramid Pooling（LR-ASPP），以提升在移动分类、检测和分割任务上的表现。这些改进通过精心设计的轻量级架构，实现了更高的准确度、更低的延迟，并在不同的资源使用场景中实现了更好的性能。

MobileNetV3的主要创新点包括：

1. **结合了硬件感知的网络架构搜索（NAS）和NetAdapt算法，针对移动设备CPU进行优化。**
2. **引入了新颖的架构设计，包括反转残差结构和线性瓶颈层。**
3. **提出了高效的Lite Reduced Atrous Spatial Pyramid Pooling（LR-ASPP）作为新的分割解码器。**

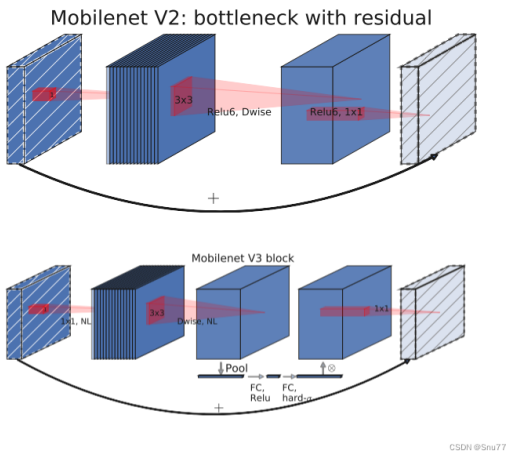
2.1 NAS和NetAdapt算法

MobileNetV3采用了硬件感知的网络架构搜索（NAS）和NetAdapt算法，这两种技术相互补充，可以结合起来有效地为特定硬件平台找到优化的模型。特别是，它采用了平台感知NAS进行块级搜索，类似于之前的MnasNet-A1方法，使用相同的基于RNN的控制器和相同的分解层次搜索空间，以便为大型移动模型找到全局网络结构，目标是大约80ms的延迟。然后在此基础上应用NetAdapt算法和其他优化措施。这种方法允许在顺序方式中对单个层进行微调，而不是尝试推断粗略但全局的架构。NetAdapt的第二个技术是用于层级搜索，它更适用于小型移动模型，因为对于小型模型来说，准确性随着延迟的变化更加显著，因此需要一个较小的权重因子w = -0.15来补偿不同延迟下的较大准确性变化。通过这个新的权重因子，我们从头开始一个新的架构搜索，以找到初始种子模型，然后应用NetAdapt和其他

优化来获得最终的MobileNetV3-Small模型

2.2 反转残差结构和线性瓶颈层

MobileNetV3在架构上进行了一些修改，以降低某些较慢层的延迟，同时保持准确性。这些修改超出了当前搜索空间的范围。第一项修改重新设计了网络的最后几层是如何相互作用以更有效地生成最终特征的。基于MobileNetV2的反转瓶颈结构的当前模型在最终层使用1x1卷积以扩展到更高维的特征空间。这一层对于预测中拥有丰富的特征至关重要。然而，这也增加了额外的延迟。为了减少延迟并保留高维特征，我们将这一层移到最终的平均池化之后



CSDN @Snu77

上图展示了MobileNetV2和MobileNetV3的网络结构层。

上侧 (MobileNetV2层): 展示了反转残差和线性瓶颈结构。每个块由狭窄的输入和输出层组成，这些层没有非线性操作，后面跟着扩展到更高维空间并投影到输出的操作。残差连接连接了瓶颈层，而不是扩展层。

下侧 (MobileNetV2 + Squeeze-and-Excite): 展示了与Squeeze-and-Excite层一起使用的MobileNetV3。与先前不同，在残差层中应用了挤压和激励操作。

三、MobileNetV3的核心代码

下面的代码是整个MobileNetV1的核心代码，大家如果想学习可以和上面的框架原理对比着看一看估计会有一定的收获，使用方式看章节四。

```
"""A from-scratch implementation of MobileNetV3 paper ( for educational purposes ).

Paper
    Searching for MobileNetV3 - https://arxiv.org/abs/1905.02244v5

author : shubham.aiengineer@gmail.com
"""


import torch
from torch import nn
from torchsummary import summary


class SqueezeExitationBlock(nn.Module):
    def __init__(self, in_channels: int):
        """Constructor for SqueezeExitationBlock.

        Args:
            in_channels (int): Number of input channels.
        """
        super().__init__()

        self.pool1 = nn.AdaptiveAvgPool2d(1)
        self.linear1 = nn.Linear(
            in_channels, in_channels // 4
        )  # divide by 4 is mentioned in the paper, 5.3. Large squeeze-and-excite
        self.act1 = nn.ReLU()
        self.linear2 = nn.Linear(in_channels // 4, in_channels)
        self.act2 = nn.Hardsigmoid()

    def forward(self, x):
        """Forward pass for SqueezeExitationBlock."""

        identity = x

        x = self.pool1(x)
        x = torch.flatten(x, 1)
        x = self.linear1(x)
        x = self.act1(x)
        x = self.linear2(x)
        x = self.act2(x)

        x = identity * x[:, :, None, None]

        return x


class ConvNormActivationBlock(nn.Module):
    def __init__(
```

```
        self,
        in_channels: int,
        out_channels: int,
        kernel_size: list,
        stride: int = 1,
        padding: int = 0,
        groups: int = 1,
        bias: bool = False,
        activation: torch.nn = nn.Hardswish,
    ):
        """Constructs a block containing a convolution, batch normalization and activation layer

        Args:
            in_channels (int): number of input channels
            out_channels (int): number of output channels
            kernel_size (list): size of the convolutional kernel
            stride (int, optional): stride of the convolutional kernel. Defaults to 1.
            padding (int, optional): padding of the convolutional kernel. Defaults to 0.
            groups (int, optional): number of groups for depthwise seperable convolution. Defaults to 1.
            bias (bool, optional): whether to use bias. Defaults to False.
            activation (torch.nn, optional): activation function. Defaults to nn.Hardswish.
        """
        super().__init__()

        self.conv = nn.Conv2d(
            in_channels,
            out_channels,
            kernel_size,
            stride=stride,
            padding=padding,
            groups=groups,
            bias=bias,
        )
        self.norm = nn.BatchNorm2d(out_channels)
        self.activation = activation()

    def forward(self, x):
        """Perform forward pass."""

        x = self.conv(x)
        x = self.norm(x)
        x = self.activation(x)

        return x


class InverseResidualBlock(nn.Module):
    def __init__(
        self,
        in_channels: int,
        out_channels: int,
        kernel_size: int,
        expansion_size: int = 6,
        stride: int = 1,
        squeeze_exitation: bool = True,
        activation: nn.Module = nn.Hardswish,
    ):

        """Constructs a inverse residual block

        Args:
            in_channels (int): number of input channels
            out_channels (int): number of output channels
            kernel_size (int): size of the convolutional kernel
            expansion_size (int, optional): size of the expansion factor. Defaults to 6.
            stride (int, optional): stride of the convolutional kernel. Defaults to 1.
            squeeze_exitation (bool, optional): whether to add squeeze and exitation block or not. Defaults to True.
            activation (nn.Module, optional): activation function. Defaults to nn.Hardswish.
        """

        super().__init__()

        self.residual = in_channels == out_channels and stride == 1
        self.squeeze_exitation = squeeze_exitation

        self.conv1 = (
            ConvNormActivationBlock(
                in_channels, expansion_size, (1, 1), activation=activation
            )
            if in_channels != expansion_size
            else nn.Identity()
        )  # If it's not the first layer, then we need to add a 1x1 convolutional layer to expand the number of channels
        self.depthwise_conv = ConvNormActivationBlock(
            expansion_size,
            expansion_size,
            (kernel_size, kernel_size),
            stride=stride,
            padding=kernel_size // 2,
            groups=expansion_size,
            activation=activation,
        )
        if self.squeeze_exitation:
            self.se = SqueezeExitationBlock(expansion_size)

        self.conv2 = nn.Conv2d(
            expansion_size, out_channels, (1, 1), bias=False
        )  # bias is false because we are using batch normalization, which already has bias
        self.norm = nn.BatchNorm2d(out_channels)

    def forward(self, x):
        """Perform forward pass."""

        identity = x

        x = self.conv1(x)
        x = self.depthwise_conv(x)

        if self.squeeze_exitation:
            x = self.se(x)

        x = self.conv2(x)
        x = self.norm(x)

        if self.residual:
            x = x + identity

        return x


class MobileNetV3(nn.Module):
    def __init__(
        self,
        n_classes: int = 1000,
        input_channel: int = 3,
        config: str = "large",
        dropout: float = 0.8,
    ):
        """Constructs MobileNetV3 architecture

        Args:
        `n_classes`: An integer count of output neuron in last layer, default 1000
        `input_channel`: An integer value input channels in first conv layer, default is 3.
        `config`: A string value indicating the configuration of MobileNetV3, either `large` or `small`, default is `large
        `dropout` [0, 1] : A float parameter for dropout in last layer, between 0 and 1, default is 0.8.
        """

        super().__init__()

        # The configuration of MobileNetv3.
```

```
        # input channels, kernel size, expension size, output channels, squeeze exitation, activation, stride
        RE = nn.ReLU
        HS = nn.Hardswish
        configs_dict = {
            "small": (
                (16, 3, 16, 16, True, RE, 2),
                (16, 3, 72, 24, False, RE, 2),
                (24, 3, 88, 24, False, RE, 1),
                (24, 5, 96, 40, True, HS, 2),
                (40, 5, 240, 40, True, HS, 1),
                (40, 5, 240, 40, True, HS, 1),
                (40, 5, 120, 48, True, HS, 1),
                (48, 5, 144, 48, True, HS, 1),
                (48, 5, 288, 96, True, HS, 2),
                (96, 5, 576, 96, True, HS, 1),
                (96, 5, 576, 96, True, HS, 1),
            ),
            "large": (
                (16, 3, 16, 16, False, RE, 1),
                (16, 3, 64, 24, False, RE, 2),
                (24, 3, 72, 24, False, RE, 1),
                (24, 5, 72, 40, True, RE, 2),
                (40, 5, 120, 40, True, RE, 1),
                (40, 5, 120, 40, True, RE, 1),
                (40, 3, 240, 80, False, HS, 2),
                (80, 3, 200, 80, False, HS, 1),
                (80, 3, 184, 80, False, HS, 1),
                (80, 3, 184, 80, False, HS, 1),
                (80, 3, 480, 112, True, HS, 1),
                (112, 3, 672, 112, True, HS, 1),
                (112, 5, 672, 160, True, HS, 2),
                (160, 5, 960, 160, True, HS, 1),
                (160, 5, 960, 160, True, HS, 1),
            ),
        }

        self.model = nn.Sequential(
            ConvNormActivationBlock(
                input_channel, 16, (3, 3), stride=2, padding=1, activation=nn.Hardswish
            ),
        )

        for (
            in_channels,
            kernel_size,
            expansion_size,
            out_channels,
            squeeze_exitation,
            activation,
            stride,
        ) in configs_dict[config]:
            self.model.append(
                InverseResidualBlock(
                    in_channels=in_channels,
                    out_channels=out_channels,
                    kernel_size=kernel_size,
                    expansion_size=expansion_size,
                    stride=stride,
                    squeeze_exitation=squeeze_exitation,
                    activation=activation,
                )
            )

        hidden_channels = 576 if config == "small" else 960
        _out_channel = 1024 if config == "small" else 1280

        self.model.append(
            ConvNormActivationBlock(
                out_channels,
                hidden_channels,
                (1, 1),
            if x.size(1) in self.index:
                position = self.index.index(x.size(1))  # Find the position in the index list
                results[position] = x

        return results


if __name__ == "__main__":

    # Generating Sample image
    image_size = (1, 3, 224, 224)
    image = torch.rand(*image_size)

    # Model
    mobilenet_v3 = MobileNetV3(config="small")

    # summary(
    #     mobilenet_v3,
    #     input_data=image,
    #     col_names=["input_size", "output_size", "num_params"],
    #     device="cpu",
    #     depth=2,
    # )

    out = mobilenet_v3(image)
    print("Output shape : ", out.shape)
```

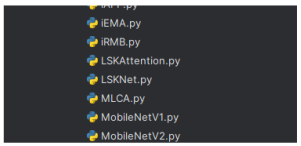
四、手把手教你添加MobileNetV3网络结构

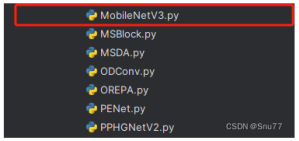
下面教大家如何修改该网络结构，主干网络结构的修改步骤比较复杂，我也会将task.py文件上传到CSDN的文件中，大家如果自己修改不正确，可以尝试用我的task.py文件替换你的，然后只需要修改其中的第1、2、3、5步即可。

⭐修改过程中大家一定要仔细⭐

4.1 修改一

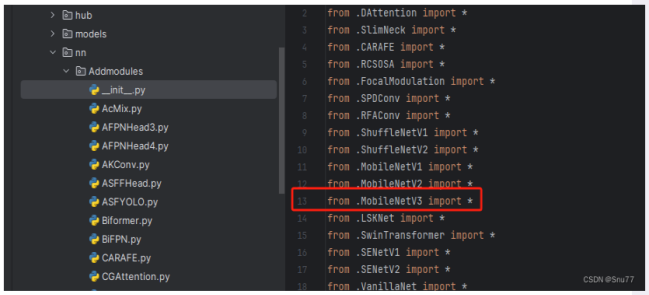
首先我门中到如下"ultralytics/nn"的目录，我们在这个目录下在创建一个新的目录，名字为'Addmodules'**（此文件之后就用于存放我们的所有改进机制）**，之后我们在创建的目录内创建一个新的py文件复制粘贴进去，可以根据文章改进机制来起，这里大家根据自己的习惯命名即可。





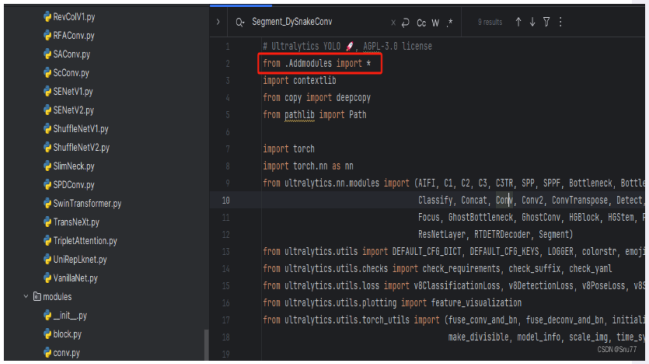
4.2 修改二

第二步我们在我们创建的目录内创建一个新的py文件名字为'__init__.py'**(只需要创建一个即可)**，然后在其内部导入我们本文的改进机制即可。



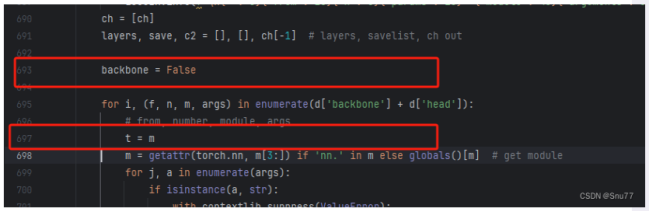
4.3 修改三

第三步我门中到如下文件'ultralytics/nn/tasks.py'然后在开头导入我们的所有改进机制 **(如果你用了我多个改进机制，这一步只需要修改一次即可)** 。



4.4 修改四

添加如下两行代码！！！



4.5 修改五

找到七百多行大概把具体看图片，按照图片来修改就行，添加红框内的部分，注意没有()只是函数名。

```
            c1, c2 = ch[f], args[0]
            if c2 != nc:  # if c2 not equal to number of classes (i.e. for Classify() output)
                c2 = make_divisible(min(c2, max_channels) * width, divisor=8)
            if m is Blocks:
                if args[1] in {'BottleNeck'}:
                    c1, c2 = ch[f], args[0] * 4
                    args = [c1, args[0], *args[1:]]
                else:
                    args = [c1, c2, *args[1:]]
            else:
                args = [c1, c2, *args[1:]]
            if m in (BottleneckCSP, C1, C2, C2f, C3, C3TR, C3Ghost, C3x, RepC3, Blocks):
                args.insert(_index=2, n)  # number of repeats
                n = 1

        elif m in {vanillanet_5, vanillanet_6, vanillanet_7, vanillanet_8, vanillanet_9, vanillanet_10,
                   repvit_m0_6, repvit_m0_9, repvit_m1_0, repvit_m1_1, repvit_m1_5, repvit_m2_3, LSKNet, LSKNET_Tiny,
                   LSKNET_base, SwinTransformer, MobileNetV1, MobileNetV2, MobileNetV3, shufflenet_v1_x0_5,
                   shufflenet_v1_x1_0, shufflenet_v1_x1_5, shufflenet_v1_x2_0, shufflenetv2, revcol_small, revcol_tiny,
                   revcol_base, revcol_xlarge, revcol_large, efficient, efficientnet_v2, FasterNet,
                   CSWin_64_12211_tiny_224, CSWin_64_24322_small_224, CSWin_96_24322_base_224,CSWin_144_24322_large_224,
                   convnextv2_atto, convnextv2_large, convnextv2_base, convnextv2_tiny,
                   transnext_micro, transnext_tiny, transnext_small, transnext_base,
                   unireplknet_a, unireplknet_f, unireplknet_p, unireplknet_n, unireplknet_t, unireplknet_s,
                   unireplknet_b, unireplknet_l, unireplknet_xl, efficientvit_backbone_b0, efficientvit_backbone_b1,
                   efficientvit_backbone_b2, efficientvit_backbone_b3,
                   EfficientViT_M0, EfficientViT_M1, EfficientViT_M2, EfficientViT_M3, EfficientViT_M4, EfficientViT_M5,
                   Ghostnetv1, Ghostnetv2}:

            m = m(*args)
            c2 = m.width_list  # 返回通道列表
            backbone = True

        elif m is AIFI:
            args = [ch[f], *args]
```

CSDN @Snu77

```
        elif m in {自行添加对应的模型即可，下面都是一样的}:
            m = m(*args)
            c2 = m.width_list  # 返回通道列表
            backbone = True
```

4.6 修改六

用下面的代码替换红框内的内容。

```
        elif m in {Detect, Segment, Pose, OBB}:
            args.append([ch[x] for x in f])
            if m is Segment:
                args[2] = make_divisible(min(args[2], max_channels) * width, divisor=8)
        elif m is RTDETRDecoder:  # special case, channels arg must be passed in index 1
            args.insert(1, [ch[x] for x in f])
        else:
            c2 = ch[f]

        m_ = nn.Sequential(*(m(*args) for _ in range(n))) if n > 1 else m(*args)  # module
        t = str(m)[8:-2].replace('__main__.', '')  # module type
        m.np = sum(x.numel() for x in m_.parameters())  # number params
        m_.i, m_.f, m_.type = i, f, t  # attach index, 'from' index, type
        if verbose:
            LOGGER.info(f'{i:>3}{str(f):>20}{n_:>3}{m.np:10.0f}  {t:<45}{str(args):<30}')  # print
        save.extend(x % i for x in ([f] if isinstance(f, int) else f) if x != -1)  # append to savelist
        layers.append(m_)
        if i == 0:
            ch = []
        ch.append(c2)
    return nn.Sequential(*layers), sorted(save)
```

CSDN @Snu77

```
if isinstance(c2, list):
    m_ = m
    m_.backbone = True
else:
    m_ = nn.Sequential(*(m(*args) for _ in range(n))) if n > 1 else m(*args)  # module
    t = str(m)[8:-2].replace('__main__.', '')  # module type
m.np = sum(x.numel() for x in m_.parameters())  # number params
m_.i, m_.f, m_.type = i + 4 if backbone else i, f, t  # attach index, 'from' index, type
if verbose:
    LOGGER.info(f'{i:>3}{str(f):>20}{n_:>3}{m.np:10.0f}  {t:<45}{str(args):<30}')  # print
save.extend(
    x % (i + 4 if backbone else i) for x in ([f] if isinstance(f, int) else f) if x != -1)  # append to savelist
layers.append(m_)
if i == 0:
    ch = []
if isinstance(c2, list):
    ch.extend(c2)
    if len(c2) != 5:
        ch.insert(0, 0)
else:
    ch.append(c2)
```

4.7 修改七

修改七我们需要来到文件的开头，然后将下面红框内的部分用我给的代码进行替换即可，同时大家主意好不要替换错了！！

```
        if augment:
            return self._predict_augment(x)
        return self._predict_once(x, profile, visualize, embed)

    2 usages
    def _predict_once(self, x, profile=False, visualize=False, embed=None):
        """
        Perform a forward pass through the network.

        Args:
            x (torch.Tensor): The input tensor to the model.
            profile (bool):  Print the computation time of each layer if True, defaults to False.
            visualize (bool): Save the feature maps of the model if True, defaults to False.
            embed (list, optional): A list of feature vectors/embeddings to return.

        Returns:
            (torch.Tensor): The last output of the model.
        """
        y, dt, embeddings = [], [], []  # outputs
        for m in self.model:
            if m.f != -1:  # if not from previous layer
                x = y[m.f] if isinstance(m.f, int) else [x if j == -1 else y[j] for j in m.f]  # from earlier layers
            if profile:
                self._profile_one_layer(m, x, dt)
            x = m(x)  # run
            y.append(x if m.i in self.save else None)  # save output
            if visualize:
                feature_visualization(x, m.type, m.i, save_dir=visualize)
            if embed and m.i in embed:
                embeddings.append(nn.functional.adaptive_avg_pool2d(x, output_size: (1, 1)).squeeze(-1).squeeze(-1))  # flatten
                if m.i == max(embed):
                    return torch.unbind(torch.cat(embeddings, dim: 1), dim=0)
        return x

    1 usage
    def _predict_augment(self, x):
        """Perform augmentations on input image x and return augmented inference."""
        LOGGER.warning(f"WARNING ⚠ {self.__class__.__name__} does not support augmented inference yet. "
```

CSDN @Snu77

代码如下->

```
def _predict_once(self, x, profile=False, visualize=False):
    """
    Perform a forward pass through the network.

    Args:
        x (torch.Tensor): The input tensor to the model.
        profile (bool):  Print the computation time of each layer if True, defaults to False.
        visualize (bool): Save the feature maps of the model if True, defaults to False.

    Returns:
        (torch.Tensor): The last output of the model.
    """
    y, dt = [], []  # outputs
    for m in self.model:
        if m.f != -1:  # if not from previous layer
            x = y[m.f] if isinstance(m.f, int) else [x if j == -1 else y[j] for j in m.f]  # from earlier layers
        if profile:
            self._profile_one_layer(m, x, dt)
        if hasattr(m, 'backbone'):
            x = m(x)
            if len(x) != 5: # 0 - 5
                x.insert(0, None)
            for index, i in enumerate(x):
                if index in self.save:
                    y.append(i)
                else:
                    y.append(None)
            x = x[-1] # 最后一个输出传给下一层
        else:
            x = m(x)  # run
            y.append(x if m.i in self.save else None)  # save output
        if visualize:
            feature_visualization(x, m.type, m.i, save_dir=visualize)
    return x
```

到这里就完成了修改部分，但是这里面细节很多，大家千万要注意不要替换多余的代码，导致报错，也不要拉下任何一部，都会导致运行失败，而且报错很难排查！！！很难排查！！！

4.8 修改八

我们将下面的s用640替换即可，这一步也是部分的主干可以不修改，但有的不修改就会报错，所以我们还是修改一下。

```
10 usages
class DetectionModel(BaseModel):
    """YOLOv8 detection model."""

    def __init__(self, cfg='yolov8n.yaml', ch=3, nc=None, verbose=True):  # model, input channels, number of classes
        """Initialize the YOLOv8 detection model with the given config and parameters."""
        super().__init__()
        self.yaml = cfg if isinstance(cfg, dict) else yaml_model_load(cfg)  # cfg dict

        # Define model
        ch = self.yaml['ch'] = self.yaml.get('ch', ch)  # input channels
```

```yaml
        if nc and nc != self.yaml['nc']:
            LOGGER.info(f"Overriding model.yaml nc={self.yaml['nc']} with nc={nc}")
            self.yaml['nc'] = nc  # override YAML value
        self.model, self.save = parse_model(deepcopy(self.yaml), ch=ch, verbose=verbose)  # model, savelist
        self.names = {i: f'{i}' for i in range(self.yaml['nc'])}  # default names dict
        self.inplace = self.yaml.get('inplace', True)

        # Build strides
        m = self.model[-1]  # Detect()
        if isinstance(m, (Detect, Segment, Pose, OBB)):
            s = 256  # 2x min stride
            m.inplace = self.inplace
            forward = lambda x: self.forward(x)[0] if isinstance(m, (Segment, Pose, OBB)) else self.forward(x)
            m.stride = torch.tensor([s / x.shape[-2] for x in forward(torch.zeros(1, ch, s, s))])  # forward
            self.stride = m.stride
            m.bias_init()  # only run once
        else:
            self.stride = torch.Tensor([32])  # default stride for i.e. RTDETR

        # Init weights, biases
        initialize_weights(self)
        if verbose:
            self.info()
            LOGGER.info('')
```

CSDN @Snu77

4.9 RT-DETR不能打印计算量问题的解决

我们找到如下文件'ultralytics/utils/torch_utils.py'按照如下的图片进行修改，来解决计算量打印不出来的问题。

```yaml
2 usages
def get_flops(model, imgsz=640):
    """Return a YOLO model's FLOPs."""
    try:
        model = de_parallel(model)
        p = next(model.parameters())
        stride = 640  # max stride
        im = torch.empty((1, p.shape[1], stride, stride), device=p.device)  # input image in BCHW format
        flops = thop.profile(deepcopy(model), inputs=[im], verbose=False)[0] / 1E9 * 2 if thop else 0  # stride GFLOPs
        imgsz = imgsz if isinstance(imgsz, list) else [imgsz, imgsz]  # expand if int/float
        return flops * imgsz[0] / stride * imgsz[1] / stride  # 640x640 GFLOPs
    except Exception:
        return 0
```

CSDN @Snu77

4.10 可选修改

有些读者的数据集部分图片比较特殊，在验证的时候会导致形状不匹配的报错，如果大家在验证的时候报错形状不匹配的错误可以固定验证集的图片尺寸，方法如下 ->

找到下面这个文件ultralytics/models/yolo/detect/train.py然后其中有一个类是DetectionTrainer class中的build_dataset函数中的一个参数rect=mode == 'val'改为rect=False

```yaml
    1 usage
    def build_dataset(self, img_path, mode='train', batch=None):
        """
        Build YOLO Dataset.

        Args:
            img_path (str): Path to the folder containing images.
            mode (str): `train` mode or `val` mode, users are able to customize different augmentations for each mode.
            batch (int, optional): Size of batches, this is for `rect`. Defaults to None.
        """
        gs = max(int(de_parallel(self.model).stride.max() if self.model else 0), 32)
        return build_yolo_dataset(self.args, img_path, batch, self.data, mode=mode, rect=False, stride=gs)

    def get_dataloader(self, dataset_path, batch_size=16, rank=0, mode='train'):
        """Construct and return dataloader."""
        assert mode in ['train', 'val']
        with torch_distributed_zero_first(rank):  # init dataset *.cache only once if DDP
            dataset = self.build_dataset(dataset_path, mode, batch_size)
        shuffle = mode == 'train'
        if getattr(dataset, 'rect', False) and shuffle:
            LOGGER.warning("WARNING ⚠️ 'rect=True' is incompatible with DataLoader shuffle, setting shuffle=False")
```

CSDN @Snu77

五、MobileNetV3的yaml文件

5.1 yaml文件

大家复制下面的yaml文件，然后通过我给大家的运行代码运行即可，RT-DETR的调参部分需要后面的文章给大家讲，现在目前免费给大家看这一部分不开放。

```yaml
# Ultralytics YOLO 🚀, AGPL-3.0 license
# RT-DETR-1 object detection model with P3-P5 outputs. For details see https://docs.ultralytics.com/models/rtdetr

# Parameters
nc: 80  # number of classes
scales: # model compound scaling constants, i.e. 'model=yolov8n-cls.yaml' will call yolov8-cls.yaml with scale 'n'
  # [depth, width, max_channels]
  l: [1.00, 1.00, 1024]

backbone:
  # [from, repeats, module, args]
  - [-1, 1, MobileNetV3, []]  # 4

head:
  - [-1, 1, Conv, [256, 1, 1, None, 1, 1, False]]  # 5 input_proj.2
  - [-1, 1, AIFI, [1024, 8]] # 6
  - [-1, 1, Conv, [256, 1, 1]]  # 7, Y5, lateral_convs.0

  - [-1, 1, nn.Upsample, [None, 2, 'nearest']] # 8
  - [3, 1, Conv, [256, 1, 1, None, 1, 1, False]]  # 9 input_proj.1
  - [[-2, -1], 1, Concat, [1]] # 10
  - [-1, 3, RepC3, [256, 0.5]]  # 11, fpn_blocks.0
  - [-1, 1, Conv, [256, 1, 1]]   # 12, Y4, lateral_convs.1

  - [-1, 1, nn.Upsample, [None, 2, 'nearest']] # 13
  - [2, 1, Conv, [256, 1, 1, None, 1, 1, False]]  # 14 input_proj.0
  - [[-2, -1], 1, Concat, [1]]  # 15 cat backbone P4
  - [-1, 3, RepC3, [256, 0.5]]    # X3 (16), fpn_blocks.1

  - [-1, 1, Conv, [256, 3, 2]]   # 17, downsample_convs.0
  - [[-1, 12], 1, Concat, [1]]  # 18 cat Y4
  - [-1, 3, RepC3, [256, 0.5]]    # F4 (19), pan_blocks.0

  - [-1, 1, Conv, [256, 3, 2]]   # 20, downsample_convs.1
  - [[-1, 7], 1, Concat, [1]]  # 21 cat Y5
  - [-1, 3, RepC3, [256, 0.5]]    # F5 (22), pan_blocks.1

  - [[16, 19, 22], 1, RTDETRDecoder, [nc, 256, 300, 4, 8, 3]]  # Detect(P3, P4, P5)
```

5.2 运行文件

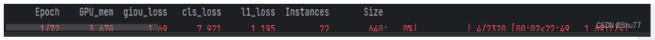
大家可以创建一个train.py文件将下面的代码粘贴进去然后替换你的文件运行即可开始训练。

```python
import warnings
from ultralytics import RTDETR
warnings.filterwarnings('ignore')

if __name__ == '__main__':
    model = RTDETR('替换你想要运行的yaml文件')
    # model.load('') # 可以加载你的版本预训练权重
    model.train(data=r'替换你的数据集地址即可',
                cache=False,
                imgsz=640,
                epochs=72,
                batch=4,
                workers=0,
                device='0',
                project='runs/RT-DETR-train',
                name='exp',
                # amp=True
                )
```

5.3 成功训练截图

下面是成功运行的截图（确保我的改进机制是可用的），已经完成了有1个epochs的训练，图片太大截不全第2个epochs了。

```
  0                -1  1   2970118  MobileNetV3                             []
  1                -1  1    246272  ultralytics.nn.modules.conv.Conv        [960, 256, 1, 1, None, 1, 1, False]
  2                -1  1    789760  ultralytics.nn.modules.transformer.AIFI [256, 1024, 8]
  3                -1  1     66048  ultralytics.nn.modules.conv.Conv        [256, 256, 1, 1]
  4                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
  5                 3  1     29184  ultralytics.nn.modules.conv.Conv        [112, 256, 1, 1, None, 1, 1, False]
  6           [-2, -1]  1         0  ultralytics.nn.modules.conv.Concat      [1]
  7                -1  3    657920  ultralytics.nn.modules.block.RepC3      [512, 256, 3, 0.5]
  8                -1  1     66048  ultralytics.nn.modules.conv.Conv        [256, 256, 1, 1]
  9                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 10                 2  1     10752  ultralytics.nn.modules.conv.Conv        [40, 256, 1, 1, None, 1, 1, False]
 11           [-2, -1]  1         0  ultralytics.nn.modules.conv.Concat      [1]
 12                -1  3    657920  ultralytics.nn.modules.block.RepC3      [512, 256, 3, 0.5]
 13                -1  1    590336  ultralytics.nn.modules.conv.Conv        [256, 256, 3, 2]
 14           [-1, 12]  1         0  ultralytics.nn.modules.conv.Concat      [1]
 15                -1  3    657920  ultralytics.nn.modules.block.RepC3      [512, 256, 3, 0.5]
 16                -1  1    590336  ultralytics.nn.modules.conv.Conv        [256, 256, 3, 2]
 17            [-1, 7]  1         0  ultralytics.nn.modules.conv.Concat      [1]
 18                -1  3    657920  ultralytics.nn.modules.block.RepC3      [512, 256, 3, 0.5]
 19       [16, 19, 22]  1   3917684  ultralytics.nn.modules.head.RTDETRDecoder    [1, [256, 256, 256], 256, 300, 4, 8, 3]
rtdetr-ShuffleNetV1 summary: 518 layers, 11908218 parameters, 11908218 gradients, 27.8 GFLOPs

TensorBoard: Start with 'tensorboard --logdir runs\train\exp10', view at http://localhost:6006/
train: Scanning C:\Users\Administrator\Desktop\RT-DETR\smoking detection.v1-smoking-detection-v1.yolov8\train\labels.cache... 9279 
train: WARNING ⚠ C:\Users\Administrator\Desktop\RT-DETR\smoking detection.v1-smoking-detection-v1.yolov8\train\images\FpEkcjragAAW
WARNING ⚠ Box and segment counts should be equal, but got len(segments) = 691, len(boxes) = 24489. To resolve this only boxes will
val: WARNING ⚠ C:\Users\Administrator\Desktop\RT-DETR\smoking detection.v1-smoking-detection-v1.yolov8\valid\images\EzsbUnTX0AE8OQ
WARNING ⚠ Box and segment counts should be equal, but got len(segments) = 30, len(boxes) = 913. To resolve this only boxes will be
val: Scanning C:\Users\Administrator\Desktop\RT-DETR\smoking detection.v1-smoking-detection-v1.yolov8\valid\labels.cache... 507 ima
Plotting labels to runs\train\exp10\labels.jpg...
optimizer: AdamW(lr=0.0001, momentum=0.937) with parameter groups 104 weight(decay=0.0), 156 weight(decay=0.0001), 167 bias(decay=0.
WARNING ⚠ TensorBoard graph visualization failure 'NoneType' object is not subscriptable
Image sizes 640 train, 640 val
Using 0 dataloader workers
Logging results to runs\train\exp10
Starting training for 72 epochs...
```



六、全文总结

从今天开始正式开始更新RT-DETR创指论文专栏，本专栏的内容会迅速铺开，在短期呢大量更新，价格也会乘阶梯性上涨，所以想要和我一起学习RT-DETR改进，可以在前期直接关注，**本文专栏旨在打造全网最好的RT-DETR专栏为想要发论文的家进行服务。**

专栏链接：RT-DETR创指论文专栏，持续复现各种顶会内容——论文收割机RT-DETR

出处：http://www.hzhcontrols.cn/
原文：http://www.hzhcontrols.cn/new-2008627.html
本文版权归原作者所有
欢迎转载，但未经作者同意必须保留此段声明，且在文章页面明显位置给出原文连接，否则保留追究法律责任的权利

相关：漏洞sql注入 linux怎么看本机数据库 linux怎么查看数据库端口 龙管家数据库被清除怎么恢复 linux创建db2数据库

上一篇：统计学-R语言-6.3

下一篇：免费开源OCR 软件Umi-OCR

Copyright © 2019 HZHControls All rights 豫ICP备19038928号-11