# CheapNet: Ultra-Lightweight Real-time Network for Semantic Segmentation and Defect Detection

Daitao Wang*, Wenjing Yu†, Hongjun Qiu‡, Ruyang Xiao§
†Software Engineering Institute of Guangzhou, Guangzhou, 510900, China
*†‡§¶Guangzhou Civil Aviation College, Guangzhou 510403, China
*1614282207@qq.com, †ywj@mail.seig.edu.cn,
‡qiuhongjun@gcac.edu.cn, §xx

**摘要—针对工业生产中的硬件设备存在资源限制的问题，本文提出了一种高性能，低内存占用的超轻量级语义分割网络-CheapNet，能在资源限制的硬件条件下实现高效缺陷部位的分割与定位，适用于环境稳定的工业场景产品缺陷检测. 本项目通过逐通道卷积与通道数不变的思想设计了两个基本单元 Encoding Block 与 Decoding Block，通过堆叠两个基本单元并结合跳跃连接组建了 CheapNet. 并结合图像形态学中的 Two-Pass 算法计算图像的连通域信息，可以得到缺陷部分的分割掩码，位置坐标与对应的像素面积. 本文在 DAGM 的十个子数据集对 CheapNet 进行验证测试. 实验表明:CheapNet 在内存占用，推理速度，分割和定位性能之间取得了较好的平衡. 具体来说，在没有任何预处理，后处理以及图像增广的情况下，它在 DAGM 数据集上仅以 0.026GFLOPs 和 871.7FPS 在单张 NVIDIA1 V100 GPU 上取得了 87.31% 的 mIoU，且参数量仅为 0.000277M，当 CheapNet 扩展到 13.4GFLOPs 时能够取得 94.12% 的 mIoU 分数. 结合 Two-Pass 算法能够以 252.3FPS 达到 79.85% 的 mAP.CheapNet 的性价比远远高于现有的具有可比性能的分割与定位算法. 适合应用于要求较为严苛的工业缺陷检测场景. 可有效地推动深度学习算法在工业质检领域的实施应用.Code and pre-trained models are available at https://github.com/hanknewbird/CheapNet.**

***Index Terms*—Real-time Semantic Segmentation,Real-time Object Detection,Convolutional Neural Network,Ultra-Lightweight Network,Industrial Quality Inspection.**

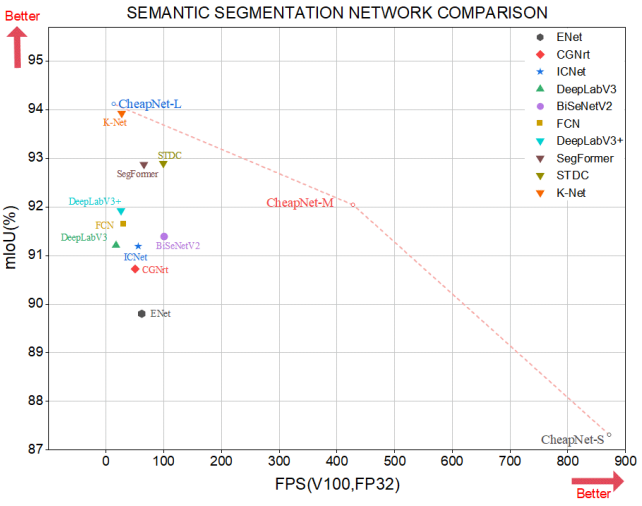


图 1. Comparsion with other semantic segmentation network.CheapNet achieve state-of-the-arts performance.

## I. Introduction

缺陷检测是工业质检中的一项基本任务. 比如在纺织与服装业的生产中，会产生诸如断纱，破洞，毛斑等缺陷. 在金属加工业，会产生诸如气孔，裂纹，夹杂等缺陷. 传统的产品检测主要由人工完成，存在着成本高，效率低，人为差错等诸多问题.

近年来随着人工智能，深度学习等技术的突飞猛进，具有人工智能的自动检测技术与手段在工业质检领域中也取得了许多不错的效果 [1], [2]. 大多数工业场景，对自动化检测技术所应用智能模型的推理速度，精准度，可靠性有较高的要求. 先前的很多工作 [3], [4] 已经在各种基准测试 [5], [6] 中获得了令人满意的效果，然而，许多工业场景受到算力，数据传输能力的约束，并要满足实时性，低成本部署的要求，有很多算法模型并不适合应用在此类工作场景，需要重新设计开发既能满足检测质量要求，又能满足低资源消耗要求的算法模型. 针对此问题，本文拟开发一种超轻量语义分割网

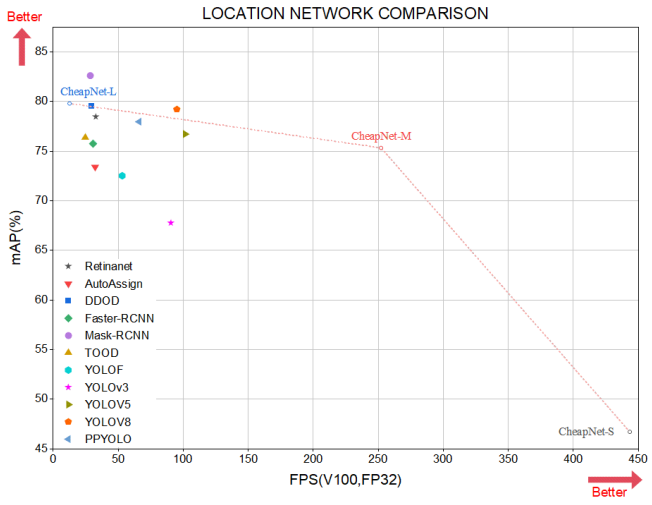


图 2. Comparsion with other object detection network.CheapNet achieve state-of-the-arts performance.

络-CheapNet, 以更好地适用到工业场景中.

通过对目前主流轻量化语义分割网络的深入研究[3], [7], 我们发现大部分算法都采用 VGG[8] 的卷积层构建思想, 即特征图通过每个卷积层后尺寸减半且通道数翻倍, 这样的构建方式会导致算法在编码阶段特征图通道数越来越大, 从而导致网络的参数量剧增. 为了解决这个问题, 我们使用逐通道卷积与逐通道转置卷积构建了 2 个部分, 其分别为 Encoding Block 与 Decoding Block, 通过堆叠这两个基本单元组建了一个 10 层的自编码器, 并使用 4 次跳跃连接将低维的全局信息与高维的局部信息进行融合, 最后输出区分缺陷与背景的二值图像, 我们将其称为 CheapNet. 值得注意的是, 不同于传统的卷积层构建方式 [8], 即每向前一层特征图尺寸减半并通道数翻倍,CheapNet 随着网络的加深, 其特征图尺寸减半但通道数不变, 这样的构建方式意味着每一个卷积层的参数量是恒定的, 不会随着网络的加深从而导致算法参数量剧增.

本文还尝试使用 Two-Pass 算法作为后处理, 计算 CheapNet 输出的分割二值图的连通域信息, 依托于 CheapNet 强大的降维能力–将复杂的原始图像降维为二值图,Two-Pass 能够将分割问题转化为定位问题, 由此可以得到每个缺陷部分的位置坐标以及对应的缺陷像素面积. 其整体流程如图5所示. 使用 CheapNet 进行分割与定位任务的结果示例如图6所示. 图1与2展示了其优于其他语义分割和目标检测网络的优势.

我们在 DAGM[9] 的 10 个不同纹理, 光照条件, 异常类型的纺织物子数据集中验证了 CheapNet 的有效性. 通过 512 × 512 的灰度图输入,CheapNet 在单张 NVIDA V100 GPU 上能够仅以 0.026GFLOPs 和 0.000277M 参数实现 871.7FPS 并取得 87.32% 的 mIoU, 当 CheapNet 扩展到 13G 时, 能够取得高达 94.12% 的 mIoU. 若结合 Two-Pass 算法, 能够在不增加模型参数的情况下实现 252.3FPS 达到 79.85% 的 mAP. 其性价比远远高于现有的具有可比性的其他网络. 可有效地推动智能检测算法在工业质检场景的落地.

Our main contributions are summarized as follows:

- 我们针对工业质检场景提出了一个超轻量级的语义分割网络-CheapNet, 并能够结合 Two-Pass 算法拓展做到缺陷的有效快速检测, 能够在 0.026GFLOPs 的限制下, 达到 871.7FPS, 并 mIoU 高达 87.31%.
- CheapNet 不同于大部分语义分割网络的搭建思想, 其特征图每经过一层卷积层, 其尺寸减半, 但通道数不变, 这一改变, 使得网络参数量大大降低.
- CheapNet 在 10 个具有挑战性的数据集上证明了它的有效性.CheapNet 的性价比远远高于具有可比分割与定位性能的现有算法.
- 结果表明, 使用深度学习算法做前处理, 再结合传统算法, 在大大减小计算量的同时, 其效果也能与主流算法媲美, 这也许是模型轻量化的另一条道路.

## II. Related Work

**Semantic Segmentation** 由于 CNN 的强大特征表示能力, 语义分割在近年来取得了很大的进展,FCN[10] 以全卷积方式开创性的实现了密集的语义预测和语义分割的端到端训练, 之后许多框架都在此基础上进行改进. 如 DeepLab[11] 与 PSPNet[12] 使用空洞卷积构建 ASPP 模块, 可以更好地描述多尺度的上下文信息,OCNet[13] 使用表示为稠密矩阵的对象上下文来强调对象信息,CCNet[14] 被改进以计算像素与行列中的所有像素之间的关联. 但这些努力都是为了实现高质量的分割指标, 并没有专门考虑分割的推理速度以及算法部署成本等问题.

**Lightweight Network** 轻量化语义分割模型需要在准确性, 模型参数和内存占用之间进行良好的权衡. 为了加速推理, 在模型压缩和加速的相关工作有剪枝与量化 [15], [16], [17], [18](parameter pruning and quantization), 低秩因子分解 (low-rank factorization)[19], [20], [21], [22], [23], 迁移/压缩卷积滤波器 (transferred/compact convolutional filters)[24], [25],

[26], 蒸馏学习 (knowledge distillation)[27], [28], [29] 等, 但它们均会降低模型的精度. 为了促进语义分割模型的部署与推理,ENet[30] 将池化与卷积操作并行, 并且降低了输入图像的尺寸, 最后将大的卷积核拆解为多个小的卷积核.ICNet[31] 将低分辨率图像经过一个 Heavy CNN, 得到较为粗糙的预测特征图, 然后提出了级联特征融合单元和级联标签指导的策略, 来引入中, 高分辨率的特征图.BiSeNet[32] 分别处理图像的空间细节和类别语义.DFANet[7] 设计了一种特征重用方法, 将多层次上下文结合到编码特征中,CGNet[33] 提出学习局部特征和周围环境的上下文特征联合, 并使用逐通道卷积对特征图的权重进行调整.BiSeNetv2[34] 减少通道容量并采用快速下采样策略, 使得网络变小.SegFormer[35] 在 Transformer 中引入层次结构, 提取不同尺度信息.STDC[36] 通过消除结构冗余缓解空间编码的耗时.K-Net[3] 通过一组可学习的内核一致地分割实例和语义类别. 但它们大部分采用的都是 VGG[8] 的网络搭建思想, 即每层的特征图尺寸是上层特征图的一半, 并且通道数翻倍, 这会导致随着网络的加深, 其参数量与复杂度将呈指数级增长.

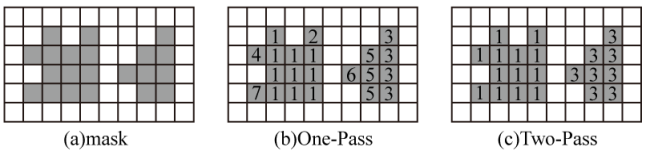


图 3. Two-Pass 算法会扫描两次 (a) 图像,(b) 第一次遍历图像时会给每一个非零像素赋予一个数字标签,(c) 第二次遍历图像时, 会将属于同一邻域的不同标签合并, 最后实现同一个邻域内的所有像素具有相同的标签

**Connected-Component Analysis** 连通域分析 [37] 在计算机视觉中用于分析图像的连通区域, 其主要作用是将图像相同像素值且相邻的像素找出来并标记, 常见的连通域分析算法有 Two-Pass[38] 与 Flood fill[39], 前者通过遍历 2 次图像进行赋值 (如图3), 后者通过递归都能够实现连通域分析. 其能够找出连通域坐标与面积. 它们的推理速度较快, 且原理巧妙, 但对输入图像要求较高.

**Summery** 尽管已经拥有了许多轻量级的语义分割与目标检测网络, 但我们认为其推理速度与准确性的平衡方法尚未得到充分探索, 本项目中的工作重点是开发超轻量级的分割与定位网络, 以提供工业部署性价比更高的质检算法.

## III. Methods

### A. Observations

常规的卷积神经网络通常由多个具有相同结构的基础单元构成. 我们注意到目前主流轻量化语义分割网络 [3], [40], [32], 特征图每经过卷积层后尺寸减半, 通道数翻倍, 根据公式(1)与(2)可知, 这会使得网络的参数量与复杂度剧增, 其主要原因是由于通道数多次倍增后导致的. 为了解决这一问题, 在本节中我们规定: 除初始输入与最终输出通道数外, 其余通道数均为 G. 通过这种思想, 本文构建了两个基本单元, 通过堆叠两个基本单元构建了 CheapNet 的基础形态.

$$Paras = (C_{out} \times K \times K \times C_{in}) + Bias \tag{1}$$

$$FLOPs = H_{out} \times W_{out} \times C_{out} \times K \times K \times C_{in} \tag{2}$$

In this context, the variable $C$ represents the number of channels, $K$ represents the size of the convolutional kernel, and $H$ and $W$ represent the height and width of the feature map.

### B. CheapNet Unit

卷积块是卷积神经网络的基本单元, 深度学习网络一般由多个基本单元堆叠构建而成. 因此, 基本单元的轻量化对于高效网络至关重要. 在深入研究深度可分离卷积 [41] 后, 我们舍弃了大部分结构. 并提出两种专门为超轻量化语义分割网络设计的基本单元, 它们分别为 Encoding Block(图4(a) 所示) 与 Decoding Block(图4(b) 所示). 值得一提的是, 与大部分网络 [8], [42], [10] 不同,CheapNet 的每个卷积层的输出通道数均为初始设定的 G.

*1) Encoding Block:* Encoding Block 主要由逐通道卷积,BatchNormal 和 LeakyReLu 堆叠构成. 其逐通道卷积的卷积核大小为 $3 \times 3$, 后接一个 BatchNorm[43] 优化数据的分布, 最后使用 LeakyReLu[44] 作为激活函数. 需要指出的是,Encoding Block 中的逐通道卷积使用的步长为 2, 填充为 1, 输出通道均为 G, 根据公式(3)可知, 特征图通过每层 Encoding Block 后, 通道数不发生变化, 但尺寸减半, 使其成为更低维的特征信息, 整个编码器部分的主要目的是尽可能的提取特征信息, 从而利用提取到的空间信息和全局信息进行精确分割.

$$S_{out} = \lfloor \frac{S_{in} - K + 2P}{Stride} \rfloor + 1 \tag{3}$$

In this context, the notation $S_{out}$ represents the output size of a feature map, while $S_{in}$ represents its input size. The symbol $P$ is used to denote the size of padding applied to the feature map.

*2) Decoding Block:* Decoding Block 主要由逐通道转置卷积与 BatchNorm 和 LeakyReLu 构成, 且输入数据由两个部分构成. 其与 Encoding Block 的主要区别在于输入的数据与卷积部分. 具体来说,Decoding Block 的输入来自上一层的输出与对应 Encoding Block 输出的横向拼接, 而 Encoding Block 的输入数据仅为上层的输出数据. 对于卷积部分,Encoding Block 为逐通道卷积, 表现为特征图通道数不变, 而尺寸减半, 是一个提取更低维特征的过程, 而 Decoding Block 为逐通道转置卷积, 其卷积核大小设置为 $3 \times 3$, 步长为 2, 填充与输出填充均为 1, 根据公式(4)可知, 特征图通过每层 Decoding Block 后, 通道数量不变, 而尺寸扩大一倍, 是一个在尽可能减少信息损失的前提下完成恢复特征的过程.

$$S_{out} = (S_{in} - 1) \times Stride - 2P + K + P_{out} \quad (4)$$

In this context,$S_{out}$ and $S_{in}$ represent the output and input dimensions of feature maps,respectively.$P$ and $P_{out}$ represent the padding and output padding sizes,respectively.

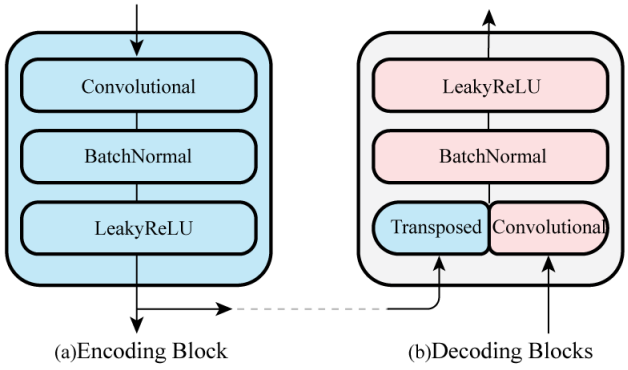


图 4. CheapNet Unit

*3) Skip Connection:* 高维特征的全局信息与低维的局部信息融合能够共同提高网络的精度, 因为浅层信息能够提供更多全局信息, 而深层特征拥有更丰富的局部信息, 为了提高模型的精度, 在我们的工作中 Decoding Block 的输入数据为上一层的输出数据与对应尺寸的 Encoding Block 的输出数据, 进行横向拼接操作的数据, 其目的是融合高低维信息达到提升精度的效果.

### C. Location Model

传统机器视觉算法拥有强力的理论支持, 并且相对更容易理解, 但对于输入图像要求较高, 在本项工作中, 我们将 Two-Pass 算法作为后处理, 这样能够将深度学习强大的特征学习能力与传统机器学习算法扎实的理论基础相结合. 具体来说, 我们使用 CheapNet 输出的二值图作为 Two-Pass 算法的输入, 利用 CheapNet 将复杂的输入图像降维为区分缺陷与背景的二值图, 再传入 Two-Pass 算法, 可计算出缺陷的像素面积与位置信息.

| Module | Layer | Input Size | Output Size |
|---|---|---|---|
| - | Input Image | $1 \times 512 \times 512$ | - |
| **CheapNet** | Encoding Block1 | $1 \times 512 \times 512$ | $G \times 512 \times 512$ |
| | Encoding Block2 | $G \times 512 \times 512$ | $G \times 256 \times 256$ |
| | Encoding Block3 | $G \times 256 \times 256$ | $G \times 128 \times 128$ |
| | Encoding Block4 | $G \times 128 \times 128$ | $G \times 64 \times 64$ |
| | Encoding Block5 | $G \times 64 \times 64$ | $G \times 32 \times 32$ |
| | Decoding Block1 | $G \times 32 \times 32$ | $G \times 64 \times 64$ |
| | Decoding Block2 | $2G \times 64 \times 64$ | $G \times 128 \times 128$ |
| | Decoding Block3 | $2G \times 128 \times 128$ | $G \times 256 \times 256$ |
| | Decoding Block4 | $2G \times 256 \times 256$ | $G \times 512 \times 512$ |
| | Decoding Block5 | $2G \times 512 \times 512$ | $1 \times 512 \times 512$ |
| - | Mask Output | - | $1 \times 512 \times 512$ |
| **Expand** | Two-Pass | $1 \times 512 \times 512$ | $N \times 5$ |
| - | Result Output | - | $N \times 5$ |

表 I
CheapNet 详细组成表. 其中 G 为通道数,N 为检测到的缺陷数量

### D. Network Architecture

图5展示了本项目的整体架构, 分别为 CheapNet(图5(left)) 与具有检测功能的 Two-Pass 算法(图5(right)).

CheapNet 的整体架构 (图5(left)) 类似于 UNet[45], 其编码器, 解码器部分分别由 5 个 Encoding Block 与 5 个 Decoding Block 堆叠而成, 为了更好地利用高低维度特征信息, 我们为 CheapNet 加入了 4 个跳跃连接.

在编码器部分, 特征图经过每个卷积层的通道数并不发生改变, 但特征图尺寸变为原来的一半, 在解码器部分, 数据的输入来源为上一层的 Decoding Block

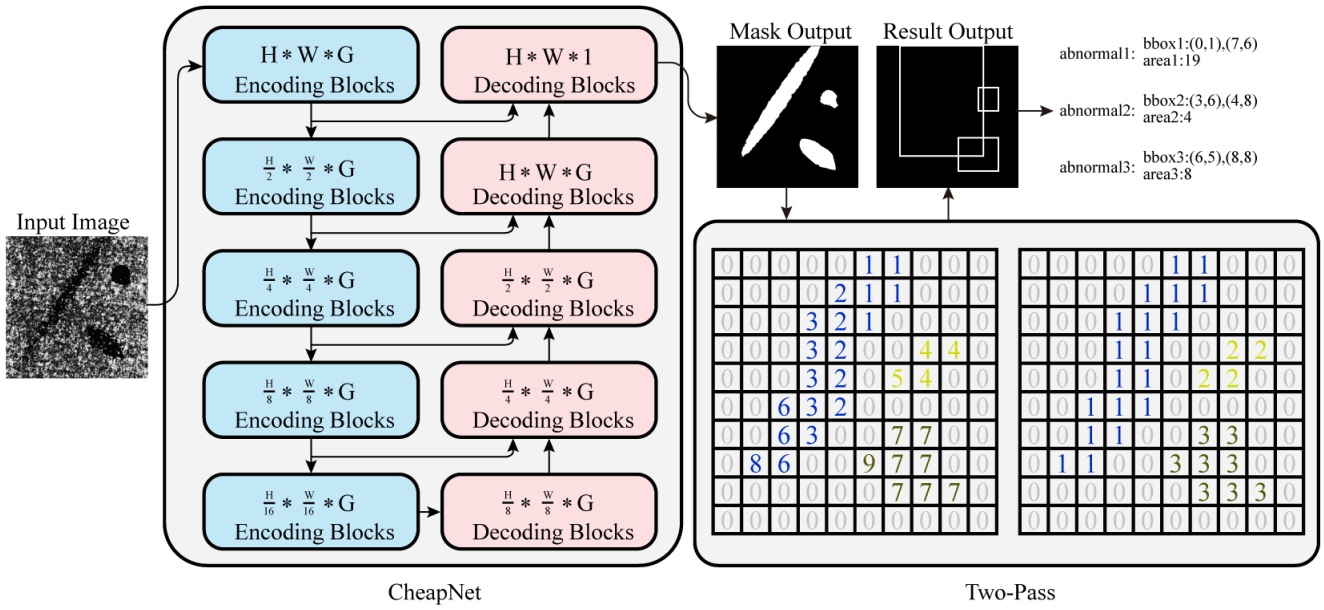


图 5. Model Architecture

与其对应的 Encoding Block(如 Decoding Block2 的输入数据为 Decoding Block1 与 Encoding Block4, 而 Decoding Block3 的输入数据为 Decoding Block2 与 Encoding Block3), 其特征图的通道数也不发生改变, 但尺寸变为原来的 2 倍. 最后, 将特征图经过一个 Sigmoid 函数 [46], 并规定输出大于 0.5 则标记为缺陷, 其余为背景. 故整个 CheapNet 的宏观展现形式是将输入图像降维为同等尺寸大小的分割二值图.

使用 Two-Pass 算法完成解析缺陷连通域信息的拓展部分如图5(right) 所示, 将 CheapNet 部分产生的分割二值图传入 Two-Pass, 经过 2 次图像遍历并逐像素赋值, 可计算得到图像的连通域信息, 从而得到关于缺陷部分的位置信息以及像素面积大小. 表I展示了更为详细的组成结构参数. 其中 G 代表通道数,$N \times 5$ 代表找到缺陷的数量与每个异常信息, 异常信息组成为 $[x_1, y_1, x_2, y_2, area]$, 其分别代表异常部分的左上角坐标. 右下角坐标以及异常部分的像素面积.

### *E. Comparison with Similar Works*

CheapNet 结构非常简洁, 与 Xception[41] 和 MobileNet[47], [48] 等采用的深度可分离卷积不同,CheapNet 主要使用了 $3 \times 3$ 的逐通道卷积构建了一个结构类似于 UNet 的网络. 并能够与 Two-Pass 算法结合完成解析缺陷信息的功能. 最重要的是, 不同于 VGG[8] 的卷积思想, 通过每个卷积层后特征图尺寸减半且通道数翻倍的方式, 除去最初的输入与最终的输出部分,CheapNet 的输出通道恒定为初期设定的 G, 其目的是尽可能的使得网络更加轻量与高效.

表II显示了使用标准卷积与逐通道卷积构建 CheapNet 时参数量的对比,G 代表每层网络的输出通道, 若使用标准卷积, 由公式(1)可算得参数为 $99G^2+38G+3$. 而使用逐通道卷积, 网络只有 $128G+12$ 参数, 仅占标准卷积约 $\frac{1.29}{G}$, 当通道数 G 越大时, 逐通道卷积对于 CheapNet 的轻量化改造效果越明显. 当通道数 G 设置为 1024 时, 其参数量将减少约 791 倍.

$$paras_s = C_{out} \times C_{in} \times k \times k \tag{5}$$

$$paras_g = \frac{C_{out}}{G} \times \frac{C_{in}}{G} \times k \times k \times G \tag{6}$$

In this context, $paras_s$ and $paras_g$ represent the parameters of standard and grouped convolutions, $C_{out}$ and $C_{in}$ represent the number of output and input channels, and $G$ represents the number of groups.

## IV. Experiments

### *A. Datasets*

**DAGM**: 该数据集由 10 个用于工业光学检测的综合数据集组成, 包含 10 种人工合成纺织物数据集, 它们(以下简称 DAGM1~10) 的组成纹理, 光照强度, 缺陷种类等都有较大的差异. 由于子数据集的图像数量不统一, 为了模型能够公平训练, 我们对部分子数据集的数量进

| Layer | Kernel | Channels | | Bias | BN | CheapNet Params | |
|---|---|---|---|---|---|---|---|
| | | Input | Output | | | Standard | CheapNet |
| Encoding Block1 | 3 | 1 | G | G | 2G | $12G$ | $9+3G$ |
| Encoding Block2 | 3 | G | G | G | 2G | $(9G+3)G$ | $12G$ |
| Encoding Block3 | 3 | G | G | G | 2G | $(9G+3)G$ | $12G$ |
| Encoding Block4 | 3 | G | G | G | 2G | $(9G+3)G$ | $12G$ |
| Encoding Block5 | 3 | G | G | G | 2G | $(9G+3)G$ | $12G$ |
| Decoding Block1 | 3 | G | G | G | 2G | $(9G+3)G$ | $12G$ |
| Decoding Block2 | 3 | 2G | G | G | 2G | $(18G+3)G$ | $21G$ |
| Decoding Block3 | 3 | 2G | G | G | 2G | $(18G+3)G$ | $21G$ |
| Decoding Block4 | 3 | 2G | G | G | 2G | $(18G+3)G$ | $21G$ |
| Decoding Block5 | 1 | 2G | 1 | 1 | 2 | $2G+3$ | $2G+3$ |
| sum | - | - | - | - | - | $99G^2+38G+3$ | $128G+12$ |

表 II
PARAMS COMPARE

行了调整, 最终数据集为 2100 张 $512\times512$ 像素大小并带粗略注释的灰度图像, 我们将其以 8:2 的比例分为训练集与验证集.

### *B. Training Details*

这项工作的所有训练过程都是在 8 张 NVIDIA V100 GPU 上进行的, 在推理阶段, 我们只使用了单张 Tesla V100 GPU 进行推导. 本项目在 PyTorch 框架 [49] 进行了包括消融实验, 模型对比等一系列实验. 除非另有说明, 否则默认使用 BCE loss[50] 作为损失函数, 使用 Adam 作为优化器 [51],epoch 为 120, 初始学习率为 0.01, 学习率策略采用 MultiStepLR, 其 milestones 为 $[60, 90]$,decay rate 为 0.1. 整个训练阶段我们未使用任何形式的数据增广操作. 得到的模型中, 我们只取训练过程中的最佳 mIoU 模型进行对比. 为了进行定量评估, 分别采用了 mean intersection over union(mIoU) 与 float-point operations(FLOPs) 对分割效果于计算复杂度进行定义. 使用 frame per second(fps) 测量运行速度.

### *C. Ablation Study*

为了评估逐通道卷积对于 CheapNet 轻量化的有效性, 本项目指定 G 为 2 到 1024 等 10 组不同的通道数, 并通过标准卷积与逐通道卷积分别构建了 CheapNet 的结构, 详细数据如表III所示.

**Paramters And FLOPs**: 通过对比各组不同方式构建的 CheapNet 的 FLOPs 与 Paramters, 使用逐通道卷积对 CheapNet 的轻量化作用是明显的. 其 FLOPs 与 Paramters 仅占使用标准卷积构建时的 65.3% ∼ 0.176% 与 58.3% ∼ 0.135%, 也就是说, 相比于标准卷积使用逐通道卷积构建 CheapNet 减少了 0.5 ∼ 565.5 倍的 FLOPs 和 0.7 ∼ 739.2 倍 paramters.

**Inference Time**: 推理速度是模型的一个重要指标, 使用逐通道卷积构建的 CheapNet 推理速度非常快, 其最高能够达到 871.7FPS. 远远高于同等性能的其他语义分割网络.

**mIoU**: 从结果来看, 使用标准卷积构建的 CheapNet 的 mIoU 指标始终优于使用逐通道卷积构建的 CheapNet, 但我们发现前者的 mIoU 并不是随着通道数的增加而增加的, 其在通道数为 4 时达到最高, 之后慢慢下降, 又继续提升, 我们暂未找到答案. 但后者随着通道数的增加, 拥有了更好的拟合能力, 其 mIoU 也随之提升. 相比于前者, 我们认为逐通道卷积使得 CheapNet 变得更加稀疏, 相当于每层网络做了一次正则处理. 所以才使得后者能够随着通道数的增加, 其 mIoU 也不断提高.

**BaseLine**: 为了后续与其他网络的对比研究, 我们选择通道数为 2,32,1024 的网络作为 CheapNet 的小型, 中等和大型版本, 我们分别命名为 CheapNet-S,CheapNet-M 和 CheapNet-L.

## V. Comparison on Other Frameworks

表IV与表V报告了 CheapNet 在语义分割和目标检测两个方向与一些主流模型进行比较, 并在图1与图2进行了可视化比较. 所有对比模型均基于 mmlab[61], [62], [63] 提供的基类实现. 我们在图6展示了 CheapNet 在 DAGM 的 10 个子数据集上的分割与定位的结果.

| Convolution | Channel | mIoU(%) | Paramter | mAcc(%) | 512*512 - NVIDIA V100 32G | | |
|---|---|---|---|---|---|---|---|
| | | | | | FLOPs | fps | ms |
| Standard | 2 | 93.92 | **0.000475M** | 96.98 | **0.040448000G** | **458.6** | **2.18** |
| Depthwise | | 87.31 | **0.000277M** | 91.73 | **0.026421248G** | **871.7** | **1.15** |
| Standard | 4 | **94.80** | 0.001739M | 96.68 | 0.135954432G | 381.0 | 2.62 |
| Depthwise | | 88.19 | 0.000551M | 96.29 | 0.051793920G | 837.3 | 1.19 |
| Standard | 8 | 94.06 | 0.006643M | 97.00 | 0.495288320G | 296.8 | 3.37 |
| Depthwise | | 90.76 | 0.001099M | 94.01 | 0.102539264G | 733.4 | 1.36 |
| Standard | 16 | 93.98 | 0.025955M | 96.33 | 1.887240192G | 222.3 | 4.50 |
| Depthwise | | 91.62 | 0.002195M | 97.05 | 0.204029952G | 703.0 | 1.42 |
| Standard | 32 | 93.78 | 0.102595M | **98.11** | 7.364280320G | 141.4 | 7.07 |
| Depthwise | | 92.04 | 0.004387M | **97.75** | 0.407011328G | 427.8 | 2.34 |
| Standard | 64 | 93.71 | 0.407939M | 96.48 | 29.090906112G | 117.6 | 8.50 |
| Depthwise | | 92.84 | 0.008771M | 96.59 | 0.812974080G | 234.4 | 4.27 |
| Standard | 128 | 93.97 | 1.626883M | 96.33 | 115.634339840G | 33.6 | 29.76 |
| Depthwise | | 93.19 | 0.017539M | 95.43 | 1.624899584G | 188.6 | 8.43 |
| Standard | 256 | 94.48 | 6.497795M | 96.60 | 461.081935872G | 10.7 | 93.71 |
| Depthwise | | 93.24 | 0.035075M | 95.37 | 3.248750592G | 56.8 | 17.62 |
| Standard | 512 | 94.50 | 25.9717515M | 96.76 | 1841.420042240G | 2.9 | 341.02 |
| Depthwise | | 93.31 | 0.070147M | 95.36 | 6.496452608G | 27.5 | 36.35 |
| Standard | 1024 | - | 103.847939M | - | 7359.867912192G | - | - |
| Depthwise | | **94.12** | 0.140291M | 95.39 | 12.991856641G | 13.4 | 74.81 |

表 III

COMPARISON NROMAL, DASHES INDICATE THAT WE COULD NOT OBTAIN A MEASUREMENT,DUE TO LACK OF MEMORY

| model | mIoU(%)⇑ | Paramters | mAcc(%) | 512*512 - NVIDIA V100 32G | | |
|---|---|---|---|---|---|---|
| | | | | GFLOPs | fps | ms |
| ENet[30] | 89.80 | **0.37M** | 92.26 | 4.36 | 62.3 | 16.05 |
| CGNet[33] | 90.72 | 0.49M | 93.63 | **3.41** | 50.8 | 19.69 |
| ICNet[31] | 91.19 | 47.82M | 93.13 | 15.39 | 56.2 | 17.79 |
| DeepLabV3[11] | 91.21 | 68.10M | 93.19 | 269.64 | 17.8 | 56.18 |
| BiSeNetV2[34] | 91.39 | 14.76M | 93.84 | 12.30 | **100.8** | **9.92** |
| FCN[10] | 91.64 | 49.48M | 94.00 | 197.69 | 30.8 | 32.47 |
| DeepLabV3+[40] | 91.92 | 43.58M | 93.86 | 176.22 | 26.5 | 37.74 |
| SegFormer[35] | 92.87 | 3.72M | 94.71 | 6.36 | 65.9 | 15.17 |
| STDC[36] | 92.89 | 8.57M | **97.12** | 8.46 | 99.9 | 10.01 |
| K-Net[3] | **93.92** | 62.17M | 96.23 | 181.69 | 27.3 | 36.63 |
| CheapNet-S | 87.31 | **0.000277M** | 91.73 | **0.026421248** | **871.7** | **1.15** |
| CheapNet-M | 92.04 | 0.004387M | **97.75** | 0.407011328 | 427.8 | 2.34 |
| CheapNet-L | **94.12** | 0.140291M | 95.39 | 12.991856641 | 13.4 | 74.81 |

表 IV

SEMANTIC SEGMENTATION NETWORK COMPARISON

| model | mAP(%)⇑ | Paramters | 512*512 - NVIDIA V100 32G | | |
|---|---|---|---|---|---|
| | | | GFLOPs | fps | ms |
| YOLOv3[52] | 67.73 | **3.67M** | **4.21** | 90.8 | 11.0 |
| YOLOF[53] | 72.49 | 42.06M | 25.13 | 53.3 | 18.8 |
| AutoAssign[54] | 73.41 | 35.97M | 50.60 | 32.4 | 30.9 |
| Faster-RCNN[55] | 75.74 | 41.12M | 63.25 | 31.0 | 32.2 |
| TOOD[56] | 76.33 | 31.79M | 46.19 | 25.0 | 40.0 |
| YOLOV5[57] | 76.70 | 7.02M | 5.07 | **101.5** | **9.9** |
| PPYOLO[58] | 77.95 | 7.61M | 5.17 | 66.4 | 15.1 |
| Retinanet[59] | 78.45 | 36.10M | 52.28 | 33.1 | 30.2 |
| YOLOV8[42] | 79.20 | 11.17M | 7.48 | 95.4 | 10.5 |
| DDOD[4] | 79.51 | 31.97M | 45.55 | 30.1 | 33.3 |
| Mask-RCNN[60] | **82.60** | 43.75M | 114.73 | 28.8 | 34.7 |
| CheapNet-S | 46.72 | **0.000277M** | **0.026421248** | **443.4** | **2.31** |
| CheapNet-M | 75.33 | 0.004387M | 0.407011328 | 252.3 | 3.96 |
| CheapNet-L | **79.85** | 0.140291M | 12.991856641 | 12.8 | 78.41 |

表 V
LOCATION NETWORK COMPARISON

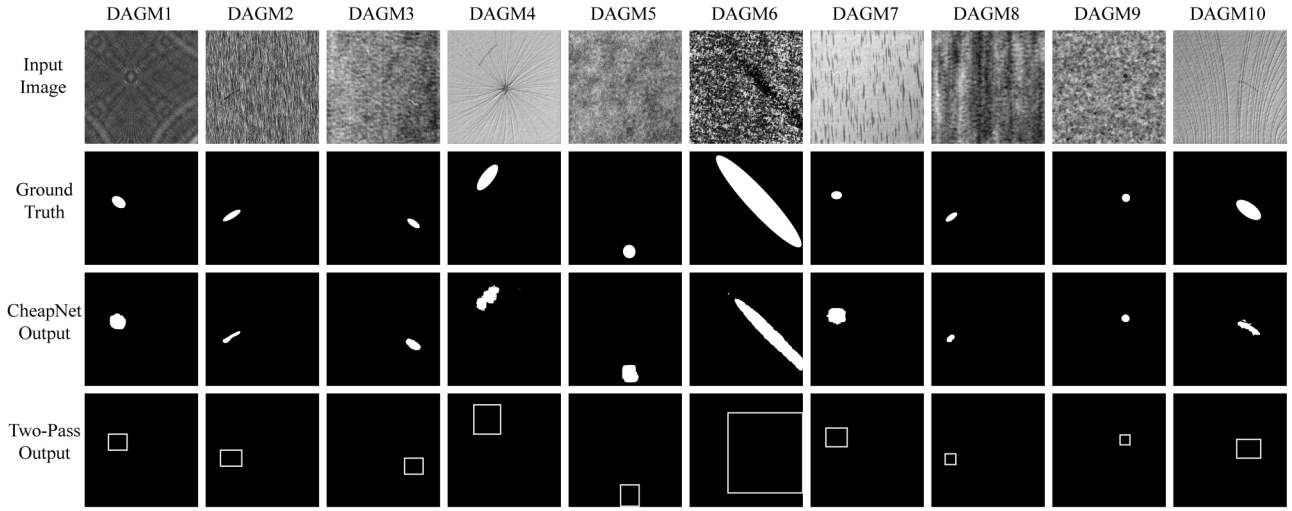


图 6. CheapNet predictions on different datasets

### A. Segmentation

表IV报告了 CheapNet 在 mIoU,paramters,mAcc,FLOPs,FPS 等参数与一些主流语义分割模型的比较. 总体而言,ChepNet 显著高于其他方法, 包括 ENet[30],CGNet[33] 在内的轻量化网络与 DeepLab[11],FCN[10] 在内的大型网络.

**CheapNet-S** 在 FLOPs 仅为 0.026G 时,mIoU 也能够达到 87.31%, 并且推理速度更是高达 871.7FPS, 比当前 paramters 最小的 ENet 与 CGNet 的 FLOPs 要小 2 个数量级. 此外推理速度也是 ENet 与 CGNet 的 13 和 16 倍.

**CheapNet-M** 是本项工作中各指标最均衡的网络. 与之 mIoU 分数相当的其他最新网络相比, 如轻量化的 SegFormer[35] 与 STDC[36],CheapNet-M 的速度更快 (427.8FPS), 参数更小 (0.004387M). 像素分类正确率更高 (97.75%).

**CheapNet-L** 是本项工作中 FLOPs 最大的网络, 即使如此, 其相比其他大型网络也要小 2 个数量级, 但其 mIoU 却高达 94.12, 它比当前最先进的语义分割网络 K-Net 还要高 0.2%mIou. 这证实了 CheapNet 的有效性.

### B. Location

在本项目中还结合 Two-Pass 拓展了解析缺陷信息的功能, 其中包含对于缺陷的位置信息. 表V报告了拓展后的 CheapNet 与一些主流目标检测网络的对比.

我们观测到,CheapNet-S 的 mAP 较低, 这是由于其在分割部分 mIoU 精度不够导致的. 但 CheapNet-M 与 CheapNet-L 性能优异, 其中 CheapNet-M 与 Faster-RCNN[55] 为同水平的 mIoU 分数. 但速度却高达 252.3FPS, 性价比更高.CheapNet-L 的 mAP 能够达到 79.85%, 达到了与 YoloV8[42] 和 DDOD[4] 同等水平, 虽体积较小, 却没有明显优势.

## VI. Conclusion And Future Work

在本项工作中, 我们定义网络每层通道数一致并提出一种新的超轻量级语义分割网络-CheapNet, 该架构专门为工业质检场景而设计, 并结合 Two-Pass 算法得到缺陷部分的位置坐标与缺陷的面积. 与其他优秀的工作相比, 我们的主要目标是尽可能的使得网络轻量化, 并能够同时完成分割与定位的功能. 本文工作在这项任务中获得了很大的进展. 对比现有的网络,CheapNet 的体积更小, 速度更快, 其性价比远远高于具有可比分割与定位性能的现有算法, 尽管本项工作的主要目标是使得网络能够运行在更加经济的设备中, 但我们发现 CheapNet 在高端 GPU 上的表现也非常优秀. 若将其部署在数据中心,CheapNet 速度更快, 更有效, 能够更大规模的完成质检任务, 这将为企业节省更多的费用.

即使 CheapNet 拥有推理速度快, 模型体积小等优点, 但我们的工作还存在一些限制. 一个主要的限制是 CheapNet 的训练方式是全监督学习, 需要较多的含有缺陷的图像作为数据集, 但实际工业场景中难以收集大量的异常数据集. 未来, 我们计划探索使用半监督或非监督学习的训练方式来解决这一限制, 以减少对于异常图像的依赖, 我们还计划使用生成对抗网络来生成异常图像作为数据集, 以进一步减少模型在工业场景落地的成本. 总的来说, 我们认为 CheapNet 对于环境较为稳定的工业质检场景具有重要的潜力, 并期待在这个领域进行未来的工作.

## 参考文献


[1] H. Y. Ngan, G. K. Pang, and N. H. Yung, "Automated fabric defect detection—a review," *Image and Vision Computing*, vol. 29, no. 7, pp. 442–458, 2011.

[2] A. Kumar, "Computer-vision-based fabric defect detection: A survey," *IEEE Transactions on Industrial Electronics*, vol. 55, no. 1, pp. 348–363, 2008.

[3] W. Zhang, J. Pang, K. Chen, and C. C. Loy, "K-Net: Towards unified image segmentation," in *NeurIPS*, 2021.

[4] Z. Chen, C. Yang, Q. Li, F. Zhao, Z.-J. Zha, and F. Wu, "Disentangle your dense object detector," in *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 4939–4948, 2021.

[5] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[6] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," *http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html*, 2012.

[7] H. Li, P. Xiong, H. Fan, and J. Sun, "Dfanet: Deep feature aggregation for real-time semantic segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9522–9531, 2019.

[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[9] F. A. H. . Matthias Wieler, Tobias Hahn, "Weakly supervised learning for indus-trial optical inspection." https://hci.iwr.uni-heidelberg.de/content/weakly-supervised-learning-industrial-optical-inspection.

[10] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 4, pp. 640–651, 2017.

[11] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[12] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2881–2890, 2017.

[13] Y. Yuan, L. Huang, J. Guo, C. Zhang, X. Chen, and J. Wang, "Ocnet: Object context network for scene parsing," *arXiv preprint arXiv:1809.00916*, 2018.

[14] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu, "Ccnet: Criss-cross attention for semantic segmentation," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 603–612, 2019.

[15] Y. Gong, L. Liu, M. Yang, and L. Bourdev, "Compressing deep convolutional networks using vector quantization," *arXiv preprint arXiv:1412.6115*, 2014.

[16] J. Wu, C. Leng, Y. Wang, Q. Hu, and J. Cheng, "Quantized convolutional neural networks for mobile devices," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4820–4828, 2016.

[17] V. Vanhoucke, A. Senior, and M. Z. Mao, "Improving the speed of neural networks on cpus," in *Deep Learning and Unsupervised Feature Learning Workshop, NIPS 2011*, 2011.

[18] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep learning with limited numerical precision," in *International conference on machine learning*, pp. 1737–1746, PMLR, 2015.

[19] R. Rigamonti, A. Sironi, V. Lepetit, and P. Fua, "Learning separable filters," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2754–2761, 2013.

[20] E. L. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus, "Exploiting linear structure within convolutional networks for efficient evaluation," *Advances in neural information processing systems*, vol. 27, 2014.

[21] M. Jaderberg, A. Vedaldi, and A. Zisserman, "Speeding up convolutional neural networks with low rank expansions," *arXiv preprint arXiv:1405.3866*, 2014.

[22] V. Lebedev, Y. Ganin, M. Rakhuba, I. Oseledets, and V. Lempitsky, "Speeding-up convolutional neural networks using fine-tuned cp-decomposition," *arXiv preprint arXiv:1412.6553*, 2014.

[23] C. Tai, T. Xiao, Y. Zhang, X. Wang, *et al.*, "Convolutional neural networks with low-rank regularization," *arXiv preprint arXiv:1511.06067*, 2015.

[24] T. Cohen and M. Welling, "Group equivariant convolutional networks," in *International conference on machine learning*, pp. 2990–2999, PMLR, 2016.

[25] S. Zhai, Y. Cheng, Z. M. Zhang, and W. Lu, "Doubly convolutional neural networks," *Advances in neural information processing systems*, vol. 29, 2016.

[26] W. Shang, K. Sohn, D. Almeida, and H. Lee, "Understanding and improving convolutional neural networks via concatenated rectified linear units," in *international conference on machine learning*, pp. 2217–2225, PMLR, 2016.

[27] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[28] C. Buciluǎ, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 535–541, 2006.

[29] J. Ba and R. Caruana, "Do deep nets really need to be deep," *Advances in neural information processing systems*, vol. 27, 2014.

[30] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "Enet: A deep neural network architecture for real-time semantic segmentation," *arXiv preprint arXiv:1606.02147*, 2016.

[31] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, "Icnet for real-time semantic segmentation on high-resolution images," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 405–420, 2018.

[32] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "Bisenet: Bilateral segmentation network for real-time semantic segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 325–341, 2018.

[33] T. Wu, S. Tang, R. Zhang, J. Cao, and Y. Zhang, "Cgnet: A light-weight context guided network for semantic segmentation," *IEEE Transactions on Image Processing*, vol. 30, pp. 1169–1179, 2020.

[34] C. Yu, C. Gao, J. Wang, G. Yu, C. Shen, and N. Sang, "Bisenet v2: Bilateral network with guided aggregation for real-time semantic segmentation," *International Journal of Computer Vision*, pp. 1–18, 2021.

[35] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," *arXiv preprint arXiv:2105.15203*, 2021.

[36] M. Fan, S. Lai, J. Huang, X. Wei, Z. Chai, J. Luo, and X. Wei, "Rethinking bisenet for real-time semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9716–9725, 2021.

[37] wiki, "Connected component labeling." https://en.wikipedia.org/wiki/Connected-component_labeling.

[38] G. Shapiro, L.; Stockman, *computer vision*. Prentice Hall, 2002.

[39] A. R. Smith, "Tint fill," in *Proceedings of the 6th annual conference on Computer graphics and interactive techniques*, pp. 276–283, 1979.

[40] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *ECCV*, 2018.

[41] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1251–1258, 2017.

[42] YOLOv8, "Yolov8." https://github.com/ultralytics/ultralytics.

[43] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, pp. 448–456, 2015.

[44] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. ICML*, vol. 30, p. 3, 2013.

[45] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015.

[46] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Neurocomputing*, vol. 1, no. 4, pp. 541–551, 1986.

[47] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 484–492, 2017.

[48] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018.

[49] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," *NIPS-W*, 2017.

[50] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2014.

[51] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[52] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," 2018.

[53] Q. Chen, Y. Wang, T. Yang, X. Zhang, J. Cheng, and J. Sun, "You only look one-level feature," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[54] B. Zhu, J. Wang, Z. Jiang, F. Zong, S. Liu, Z. Li, and J. Sun, "Autoassign: Differentiable label assignment for dense object detection," *arXiv preprint arXiv:2007.03496*, 2020.

[55] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Jun 2017.

[56] C. Feng, Y. Zhong, Y. Gao, M. R. Scott, and W. Huang, "Tood: Task-aligned one-stage object detection," in *ICCV*, 2021.

[57] G. Jocher, A. Chaurasia, A. Stoken, J. Borovec, Y. Kwon, K. Michael, and J. Fang, "ultralytics/yolov5: v6. 2-yolov5 classification models, apple m1, reproducibility, clearml and deci. ai integrations," *Zenodo. org*, 2022.

[58] S. Xu, X. Wang, W. Lv, Q. Chang, C. Cui, K. Deng, G. Wang, Q. Dang, S. Wei, Y. Du, and B. Lai, "Pp-yoloe: An evolved version of yolo," *ArXiv*, vol. abs/2203.16250, 2022.

[59] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017.

[60] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask r-cnn," *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[61] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu, Z. Zhang, D. Cheng, C. Zhu, T. Cheng, Q. Zhao, B. Li, Y. Lu, R. Zhu, Y. Wu, J. Dai, J. Wang, J. Shi, W. Ouyang, C. C. Loy, and D. Lin, "Mmcv: Openmmlab's open-source computer vision toolbox," *arXiv preprint arXiv:2007.13171*, 2020.

[62] Z. Dai, Y. Chen, S. Liu, D. Zhang, J. Yi, S. Pu, J. Jia, H. Li, G. Li, Y. Zhou, *et al.*, "Openmmlab semantic segmentation toolbox and benchmark," *arXiv preprint arXiv:2012.15712*, 2020.

[63] K. Chen, J. Wang, S. Li, W. Wu, T. Zhang, B. Zhou, and X. Lin, "MMDetection: Open mmlab detection toolbox and benchmark," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7122–7131, 2019.