**Yolov8 – yolo11**

The paper tackles a practical pain‑point in the YOLO community: the four newest versions (v8–v11) ship code before peer‑reviewed papers and often without official architecture diagrams. By triangulating **literature, docs and source code (.yaml & Python)**, the authors reconstruct full network graphs, compare backbone/neck/head designs, and trace the introduction of attention mechanisms from v10 onward. They also provide a step‑by‑step visual walkthrough of how YOLO11‑l processes a 640×640 image. The study concludes that although each release brings incremental efficiency—e.g., **SCDown, C3k2, C2PSA, DWConv**—many blocks remain unchanged, and the lack of official publications hampers future innovation.

**该综述关注一个长期困扰 YOLO 社区的问题：最新四版 YOLO（v8–v11）往往代码先行、缺乏官方论文与结构示意，导致研究者难以把握真实网络结构。作者通过“文献＋官方文档＋源码（YAML 与 Python 模块）”三路交叉验证，首次重建并并排绘制 v8‑v11 的完整网络图，逐块比较 Backbone‑Neck‑Head 设计，追踪 v10 起出现的注意力模块（PSA / C2PSA）演变。研究发现：尽管每一版都引入效率改进（例如 SCDown、C3k2、DWConv 等），仍有大量模块沿用旧版，而没有正式论文亦阻碍了社区创新。**

**关键贡献**

1. **首套 v8‑v11 统一结构图** — 基于 Ultralytics 与 Tsinghua 源码自动还原。
2. **模块级分类表** — 将 GELAN、CIB、PSA、C3k2 等新块映射回 C2f/CSP 谱系。
3. **经验性发现** — v10 开始注意力层成为标配，显示 YOLO 正向 Transformer 化迈进。
4. **YOLO11‑l 图像流示例** — 逐层展示 640×640 输入的特征流向，便于教学。

Yolov8

backbone, neck, and head

the backbone is in charge of extracting features from the input image at multiple scales(Convolutional layers and specialized blocks)

Features at different scales are integrated into the neck and transmitted to the head for prediction.

YOLOv8 variant is determined by three parameters: depth\_multiple, width\_multiple, and max\_channels.

The depth\_multiple parameters determine how many bottleneck blocks are in the C2f block.

The width\_multiple and max\_channels parameters define the output channels.

class C2f(nn.Module):

def \_\_init\_\_(self, c1, c2, n=1, shortcut=True):

super().\_\_init\_\_()

self.cv1 = Conv(c1, c2//2, 1, 1) # 分割特征

self.cv2 = Conv(c1, c2//2, 1, 1) # 旁路

self.m = nn.Sequential(\*[Bottleneck(c2//2, c2//2, shortcut) for \_ in range(n)])

self.cv3 = Conv(c2, c2, 1, 1) # 融合

def forward(self, x):

y1 = self.cv1(x)

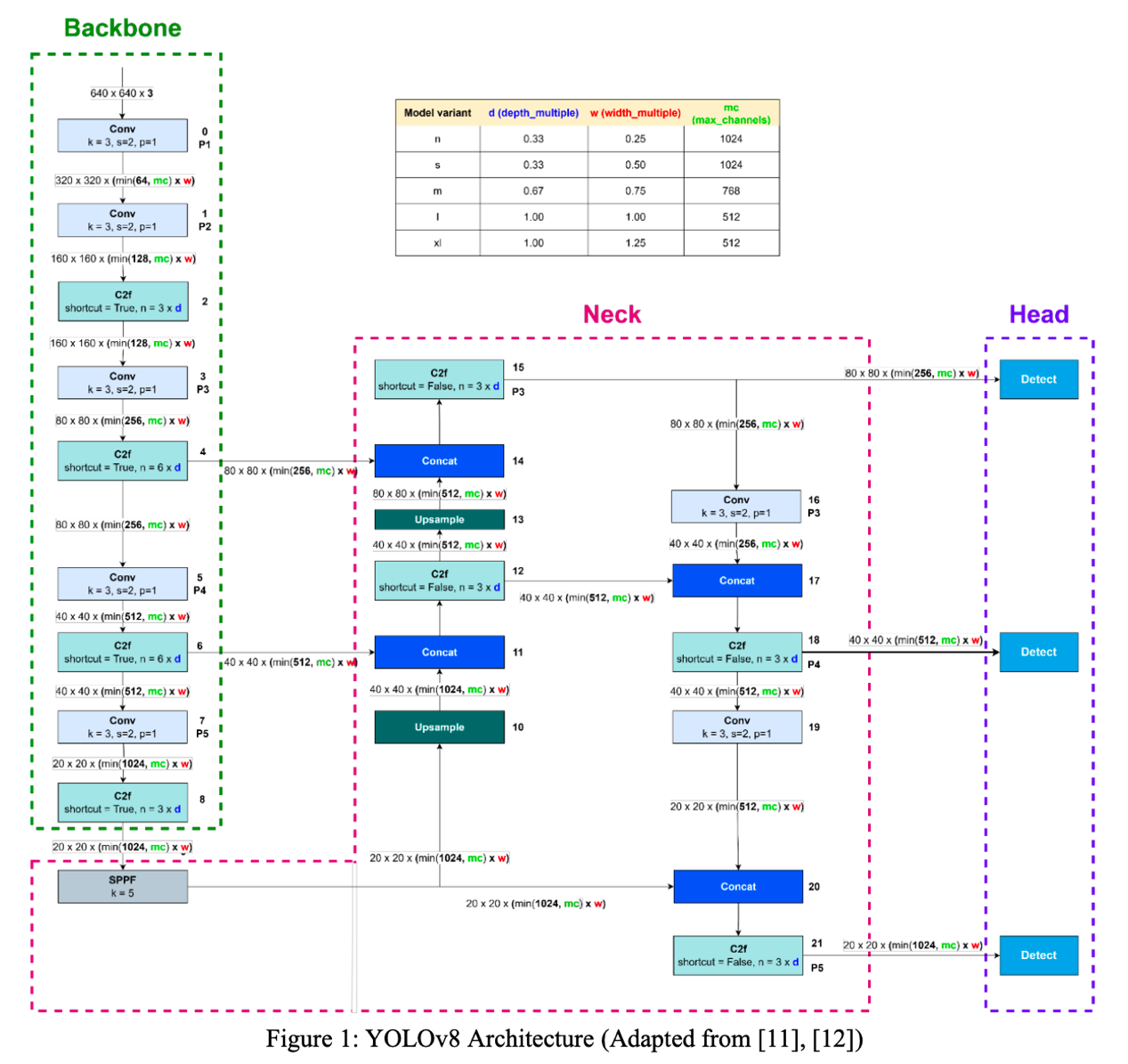
y2 = self.m(y1)

y3 = self.cv2(x)

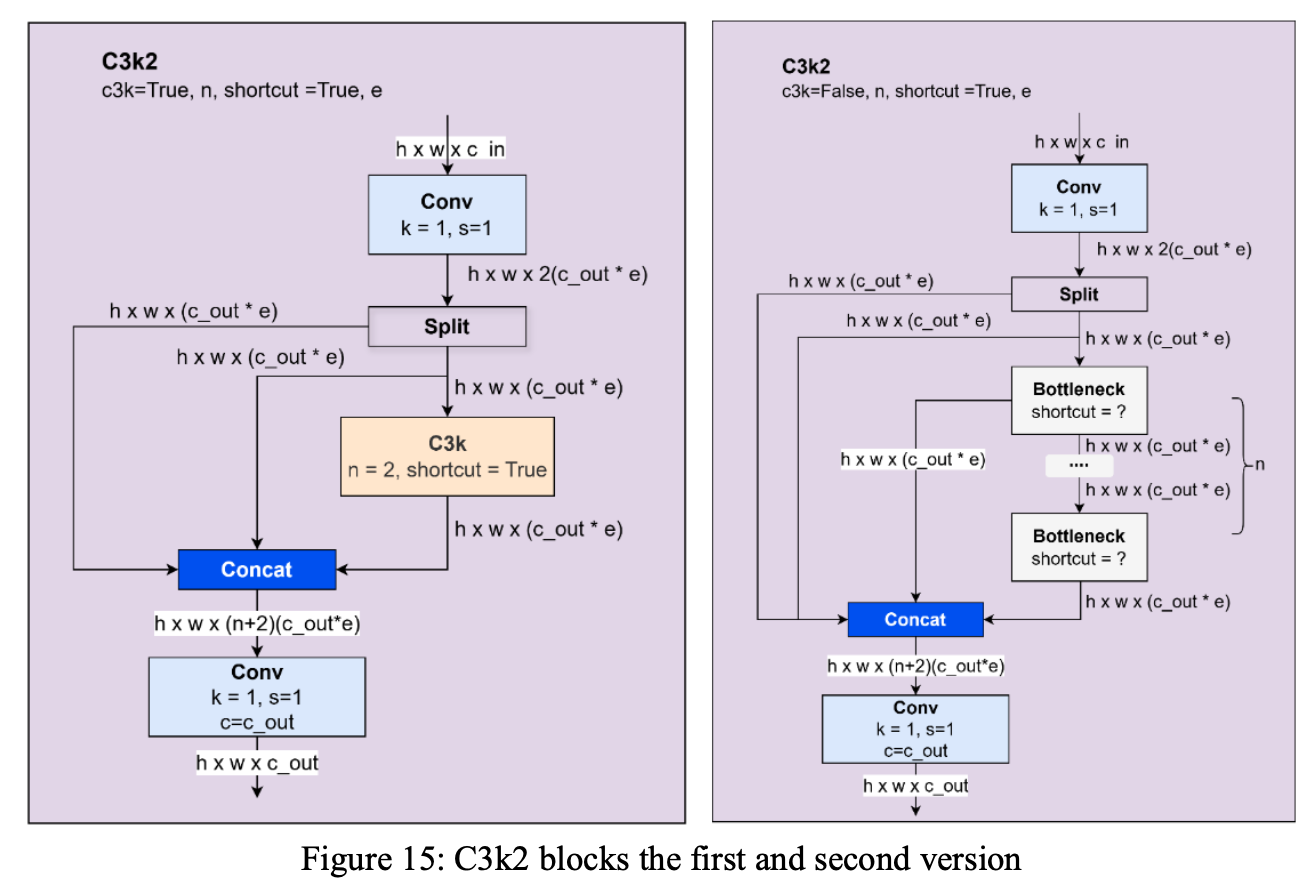
return self.cv3(torch.cat((y1, y2, y3), 1))

The stem component of YOLOv8 consists of two convolution blocks with stride 2 and kernel size 3.

These two blocks convert data into initial features and reduce input resolution.



The stage component in YOLO11 employs the C3k2 block. C3k2 represents an advancement of C2f in YOLOv8



## 1 什么是「自注意力」？——一分钟概念扫盲

* **想象**一幅图 里有很多像素／特征格子；我们希望每个格子都能“看看”图上其它格子，再决定自己该关注谁。
* **做法**：给每个格子做三份 1×1 卷积，得到 **Q（查询）‑K（键）‑V（值）**；
  + 用 Q 和所有 K 做点积 → 得到一排相似度分数；
  + Softmax 把分数变概率；
  + 概率 × V，加权求和 → 得到“看过别人之后”的新特征。
* **难点**：如果直接对整幅特征图做点积，复杂度是 O(H²W²)，在检测网络里会爆显存

 **形状**：与输入完全一致 (N, C, H, W)，因而可以无缝接到后续卷积或检测头。

 **语义**：每个像素的新特征 =「自己原本的信息 *混入* 其他位置的加权信息」。

* 高权重像素（经 Softmax 得到的权重大）会把自己的 V “广播”给别的像素。
* 结果是 **全局上下文** 被融合到局部特征里，利于分辨同类目标、抑制背景噪声。

| **#** | **一级** | **二级** | **三级（一句话要点）** |
| --- | --- | --- | --- |
| 1 | **引言** | 1.1 背景 | YOLO 版本众多但缺乏统一架构资料，社区理解受限。 |
|  |  | 1.2 痛点 | v8‑v11 无官方图/论文，需源码考证。 |
| 2 | **相关工作** | 2.1 既有综述 | 现有评测多停留在 v8 或 v10，缺乏块级细节。 |
|  |  | 2.2 空白分析 | 注意力模块、NMS‑free 等新特性未被系统梳理。 |
| 3 | **研究方法** | 3.1 数据源 | 克隆各版本 GitHub，解析 *.yaml* 和 modules/\*.py。 |
|  |  | 3.2 绘图流程 | 先自动生成网络图，再人工校正关键块。 |
| 4 | **结果与讨论** | 4.1 整体架构对比 |  |
|  |  | ‑ 4.1.1 YOLOv8 | Anchor‑free，C2f + SPPF + 3 Head。 |
|  |  | ‑ 4.1.2 YOLOv9 | GELAN Backbone、ADown、PGI 辅助分支。 |
|  |  | ‑ 4.1.3 YOLOv10 | SCDown、C2fCIB、PSA，NMS‑free 双分配。 |
|  |  | ‑ 4.1.4 YOLO11 | C3k2、C2PSA、DWConv，回归 NMS。 |
|  |  | 4.2 架构模块库 | 列举 Conv、ADown、SCDown、C3k2、PSA/C2PSA、Detect 等 10 类模块，说明来源与作用。 |
|  |  | 4.3 图像流示例 | 逐层追踪 YOLO11‑l 处理 640×640 图像的尺寸/通道变化。 |
|  |  | 4.4 综合对照表 | 汇总 v8‑v11 在 NMS、Anchor、关键新块等维度差异。 |
| 5 | **结论** | 5.1 发现 | 注意力层自 v10 起常态化；多数块仍沿用 CSP 思路。 |
|  |  | 5.2 建议 | 发布官方图与论文；统一基准测试。 |
| 6 | **参考文献** | — | 共 21 篇，含官方仓库与 arXiv 预印本。 |

| **组件** | **主要结构** | **关键细节与设计动机** |
| --- | --- | --- |
| **Stem** | 两个 Conv (3×3, stride 2) | 在输入 640×640 时依次降到 320×320 → 160×160，并把通道扩到 64 → 128 |
| **Backbone** | 4 个 Stage，均用 **C2f** 块 | 每个 C2f 由两层卷积＋分支拼接组成，可视作轻量版 CSP，既保持梯度流又减少参数；下采样仍靠 Conv (3×3,s2) |
| **Neck** | **SPPF + FPN** | SPPF（5×5 池化堆叠）提供多尺度上下文，再三次 *Upsample → Concat → C2f* 构成 FPN/HFPN |
| **Head** | 三路 **Detect** (80×80/40×40/20×20) | Anchor‑free + NMS；每一路含分类 head＋回归 head，各 2 个 Conv |
| **可调参数** | depth\_multiple 决定 C2f 内部 Bottleneck 数；width\_multiple/max\_channels 控制通道 | 与 v11 共用同一套伸缩策略 |

| **组件** | **主要结构** | **关键细节与设计动机** |
| --- | --- | --- |
| **Stem** | 与 v8 相同的 2×Conv(3,s2) | 便于复用预处理权重 |
| **Backbone** | 4 个 Stage，核心块升级为 **C3k2** | C3k2≈“三卷积＋可选 Bottleneck/C3k”；参数少于 C2f，c3k=True 时再用 **C3k 大核** 提升感受野 |
| **Neck** | **SPPF → C2PSA → 两次 Upsample/FPN** | - **C2PSA**：在最低分辨率(20×20)插入自注意力；由多层 PSA + FFN 构成，首/末 Conv 无激活以降低信息损失 |

**1. SpatialTransformer 类**

空间变换网络，用于特征对齐：

* 根据预测的偏移量对特征图进行空间变形
* 通过网格采样技术实现特征在空间上的重新定位
* 解决不同模态之间可能存在的空间不对齐问题

**2. CrossAttention 类**

跨模态注意力机制，实现两个模态之间的特征交互：

* 采用标准的多头注意力机制
* 一个模态的特征作为查询(query)，另一个模态的特征作为键(key)和值(value)
* 通过注意力权重计算实现跨模态信息交流

**3. DeformableCrossTransformer 类**

变形跨模态Transformer模块，结合空间变换和注意力机制实现特征融合：

* 使用卷积网络预测空间对齐所需的偏移量
* 利用SpatialTransformer进行特征空间对齐
* 对齐后的特征通过CrossAttention进行信息交互
* 包含规范化层和MLP层以增强表达能力

**4. CrossModalFusion 类**

跨模态特征融合模块，支持三种融合策略：

* add: 简单的加权求和融合
* concat: 通道维度拼接后通过1×1卷积融合
* transformer: 使用双向DeformableCrossTransformer进行特征增强和融合

**5. DualModalYOLO 类**

双模态YOLO主模型，整合了上述组件：

* 包含两个backbone分别处理白光和蓝光输入
* 在多个尺度特征层上使用CrossModalFusion进行特征融合
* 融合后的特征被送入neck和head部分生成最终预测
* 支持权重加载和backbone冻结等功能

这个模型能够同时处理白光和蓝光图像，通过复杂的特征对齐和融合机制，充分挖掘两种模态的互补信息，提高血液分层检测的准确性。核心创新点是变形注意力机制与跨模态特征融合的结合，能够解决不同光照条件下图像特征的空间对应关系问题。

下面是根据您的新要求生成的更为详细的报告，并融入了您提供的新的组件和功能描述：

## 双模态YOLO血液分层检测算法设计思路与工作总结

### 1. 核心思想

本算法旨在利用双模态YOLO架构结合白光和蓝光图像的互补信息，实现血液分层的检测与分割。白光和蓝光图像分别提供了不同的优点：白光图像在边界细节上的清晰度高，而蓝光图像在血液透光性方面具有优势。通过变形注意力机制与跨模态Transformer技术，我们能够有效地融合这两种模态的信息，以提升模型的性能，尤其是在血液分层检测任务中。

### 2. 架构总览

整个模型的架构设计包含以下几个关键部分：

* **双backbone结构**：两个并行的特征提取网络，分别处理白光图像和蓝光图像。
* **多尺度特征融合模块**：在不同尺度上应用变形跨模态Transformer进行特征融合。
* **共享的neck和head部分**：处理融合后的特征，生成最终的检测和分割结果。

### 3. 核心处理模块及功能

#### 3.1 **Spatial Transformer**

* **输入输出**：
  + 输入：源特征图 (B,C,H,W)(B, C, H, W) 和偏移场 (B,2,H,W)(B, 2, H, W)
  + 输出：变形后的特征图 (B,C,H,W)(B, C, H, W)
* **作用**：
  + **空间变换**：该模块通过预测的偏移量对输入的特征图进行空间重采样，从而实现不同模态之间的特征对齐。它类似于视觉中的光流计算，可以使得一个模态的特征图与另一个模态的特征图在空间上对齐，从而解决模态间可能存在的空间不对齐问题。
* **实现细节**：
  + 使用**网格采样技术**对特征图进行空间上的重新定位，根据预测的偏移场来对每个像素进行重采样，使得图像内容的空间位置对齐，避免了由于模态差异（如折射、光学特性）导致的对齐误差。

#### 3.2 **Cross Attention**

* **输入输出**：
  + 输入：主特征 (B,N,C)(B, N, C) 和上下文特征 (B,M,C)(B, M, C)
  + 输出：增强后的主特征 (B,N,C)(B, N, C)
* **作用**：
  + **跨模态注意力机制**：该模块采用标准的多头注意力机制，其中一个模态的特征作为查询（Query），另一个模态的特征作为键（Key）和值（Value）。通过计算查询与键的相似度，生成注意力权重，再加权值（Value），最终将信息从一个模态传递到另一个模态，增强主模态的特征表示。
* **实现细节**：
  + 通过多头注意力机制，Cross Attention模块能够在白光和蓝光图像之间进行信息交互，强化各自特征的表达能力。

#### 3.3 **Deformable Cross Transformer**

* **输入输出**：
  + 输入：两个模态的特征图（各为 B,C,H,WB, C, H, W）
  + 输出：融合后的特征图 (B,C,H,W)(B, C, H, W)
* **作用**：
  + **变形跨模态Transformer模块**是本模型的核心模块，它结合了**空间变形**和**跨模态注意力机制**，用以实现特征的高效融合。
  + 首先，使用卷积网络预测空间对齐所需的偏移量，并通过**Spatial Transformer**模块进行特征的空间对齐。
  + 然后，利用**Cross Attention**模块进行特征信息的交互和增强。
  + 最终，经过规范化层（如LayerNorm）和多层感知机（MLP层），提升特征的表达能力。
* **实现细节**：
  + 该模块能够有效地解决不同模态之间由于光照和血液光学特性导致的空间位置不一致问题，确保信息在多个模态之间准确流动和融合。

#### 3.4 **Cross Modal Fusion**

* **输入输出**：
  + 输入：两个模态的特征图 (B,C,H,W)(B, C, H, W)
  + 输出：融合后的特征图 (B,C,H,W)(B, C, H, W)
* **作用**：
  + **跨模态特征融合模块**提供了三种不同的特征融合策略，确保特征融合的灵活性和高效性：
    1. **加权融合**（add）：对两个模态的特征进行加权求和，得到融合后的特征图。
    2. **拼接融合**（concat）：将两个模态的特征在通道维度上拼接，并通过1×1卷积进行融合。
    3. **Transformer融合**（transformer）：利用双向Deformable Cross Transformer进行特征增强和融合。

### 4. 特征提取与融合流程

#### 4.1 多尺度特征提取

我们基于YOLO架构中的关键点，选择了三个主要特征尺度进行融合：

* **P3/8 特征层**（backbone第4层）
* **P4/16 特征层**（backbone第6层）
* **P5/32 特征层**（backbone第9层）

#### 4.2 内容感知形变与空间对齐

##### 4.2.1 内容感知形变的实现机制

内容感知形变通过以下步骤实现：

1. **特征拼接**：将两个模态的特征图在通道维度上拼接。
2. **偏移量预测**：使用卷积网络预测像素级的偏移场。
3. **空间变换应用**：将预测的偏移场应用到一个模态上进行对齐，从而实现空间对齐。

##### 4.2.2 空间对齐的必要性分析

* **支持对齐的理由**：白光和蓝光图像捕捉的物理特性不同，导致同一语义区域空间位置偏移。通过空间对齐，我们能够消除由光线折射、血液光学特性等因素带来的空间误差。
* **质疑对齐的必要性**：尽管强大的注意力机制理论上可以处理非局部关系，但变形操作增加了计算复杂度，且偏移场预测的精度取决于网络的能力。

#### 4.3 特征融合机制

在每个特征尺度上，我们应用了特定的融合策略，主要包括：

* **变形空间对齐**：通过Deformable Cross Transformer模块实现不同模态的空间对齐。
* **跨模态注意力交互**：通过Cross Attention模块进行信息交流和特征增强。
* **双向特征增强**：在每个特征尺度上，对两个模态的特征进行双向增强，确保信息充分融合。

### 5. 技术创新点

#### 5.1 变形跨模态Transformer

这是我们算法的核心创新之一，结合了两项关键技术：

* **空间变形网络**：使用Deformable Cross Transformer进行像素级的空间对齐，确保模态间特征的精确对齐。
* **Cross Attention机制**：通过跨模态注意力机制进行特征信息的交互，增强每个模态的表示能力。

#### 5.2 内容感知的空间对齐

与传统配准方法的区别：

* **数据驱动**：特征提取和匹配是通过学习得出的，而不是通过预定义的方式。
* **密集映射**：生成像素级的偏移场，而不是稀疏的控制点。
* **任务优化**：端到端训练，直接优化下游任务的性能。

#### 5.3 多级融合策略

我们提供了三种不同的融合策略，可根据具体需求选择：

* **加权融合**：对不同特征层采用不同的权重进行融合。
* **拼接融合**：将两个模态的特征直接拼接，并通过卷积或Transformer进行融合。
* **Transformer融合**：利用Transformer进行多层次的特征增强和融合。

### 6. 实现细节与优化

#### 6.1 多尺度特征处理

在forward过程中动态收集多尺度特征，并建立特征索引到融合模块的映射，确保正确的跳跃连接。

#### 6.2 对齐机制设计考量

* **适应性**：对不同区域预测不同的偏移量，以应对模态间的差异。
* **特征关联学习**：网络学习识别两个特征图中的对应模式，增强特征对齐效果。
* **端到端优化**：通过反向传播优化偏移预测，确保训练过程中对齐模块的优化。

### 7. 应用优势

针对血液分层检测的具体优势：

* **充分利用蓝光透光性**：蓝光能够穿透血液，提供更清晰的内部结构信息。
* **结合白光边界清晰度**：白光对外部轮廓的捕捉能力更强。
* **解决空间不对齐问题**：变形机制处理由光线折射等导致的空间差异。
* **多尺度特征整合**：同时关注局部细节和全局语义信息。

### 8. 算法工作流程

1. **输入处理**：接收白光和蓝光图像输入。
2. **特征提取**：通过各自的backbone网络提取多尺度特征。
3. **特征对齐与融合**：
   * 计算内容感知的空间偏移。
   * 应用空间变换进行特征对齐。
   * 通过跨模态注意力机制交换信息。
4. **多尺度特征整合**：融合后的特征送入neck网络。
5. **预测生成**：生成边界框、类别概率和分割掩码。

### 9. 未来改进方向

#### 9.1 对齐机制的改进

* 进行消融实验，比较不同对齐策略的效果。
* 探索更轻量级的对齐方法，降低计算复杂度。
* 研究自适应对齐强度，根据模态差异程度动态调整。

#### 9.2 其他潜在改进

* **更高效的注意力机制**：改进现有的跨模态注意力机制，提升信息交互效率。
* **模态特定的特征提取器**：为白光和蓝光图像设计专用的特征提取器，提高特征提取的质量。
* **动态融合策略**：根据任务需求动态选择最适合的特征融合策略，提升模型的灵活性和鲁棒性。

通过这种综合设计，我们的双模态YOLO架构充分发挥白光和蓝光图像的互补优势，为血液分层检测提供了高效、准确的解决方案，特别是通过内容感知的空间对齐和跨模态特征融合解决了模态间的异质性问题。

本项目聚焦于医院检验科在庞大 医疗数据 与高通量样本周转压力下暴露的痛点：传统人工目检血清、血浆与凝胶样本既费时又易出错，难以满足现代检验流水线的效率与精度要求。为此，需要将 计算机视觉 融入采血管成像环节，以可靠识别分层边界和样本类型，为后续自动移液、检测等环节提供结构化数据支撑。同时，项目面向整条 血液检测流水线平台 的落地，必须搭建具备精准位移、旋转与夹取能力的 机械平台 ，确保采血管在拍摄、搬运、进出料等环节的连续流转，实现“拍—判—取”一体化的自动化闭环，最终提升实验室处理速度、结果一致性与数据可追溯性。