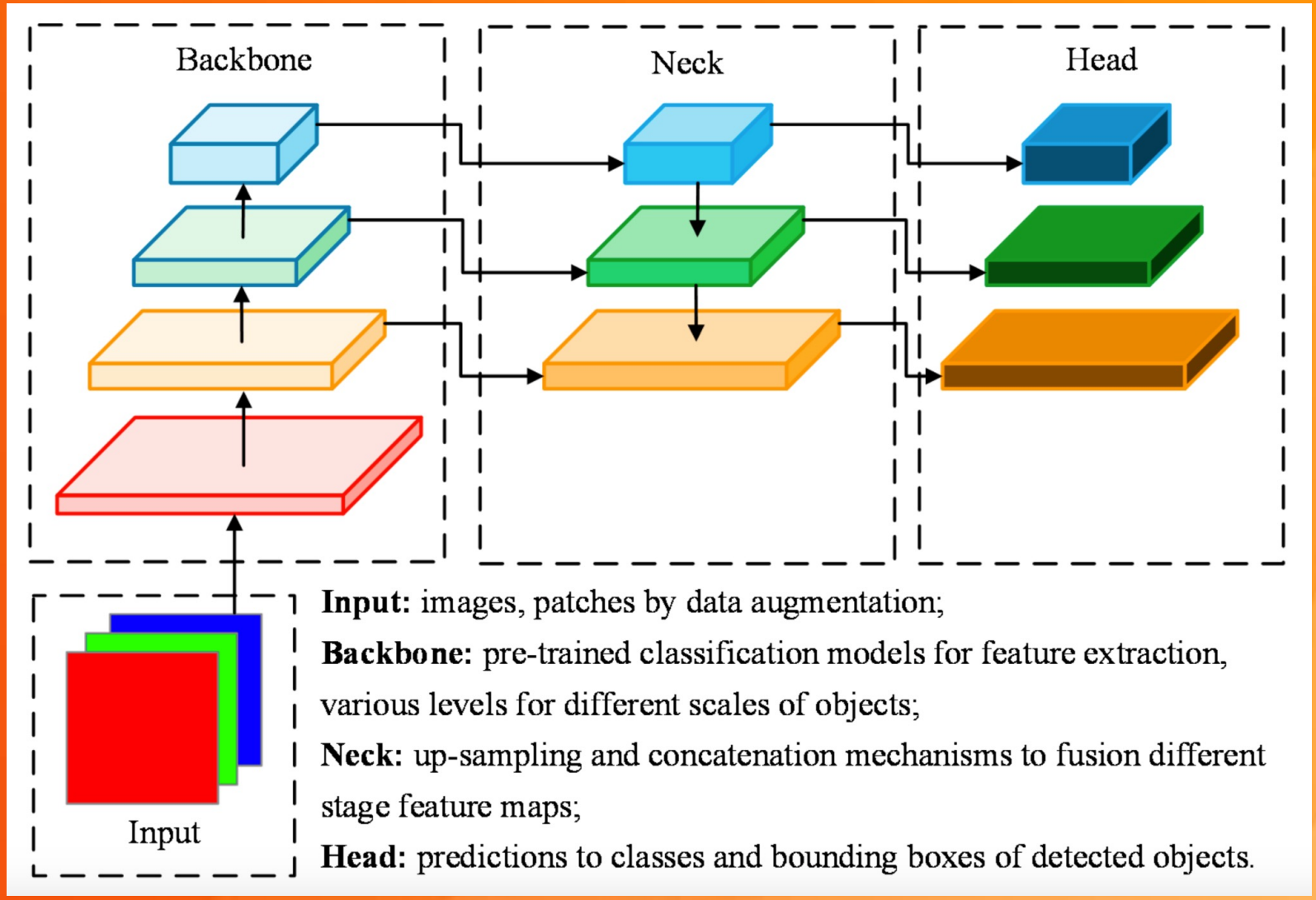
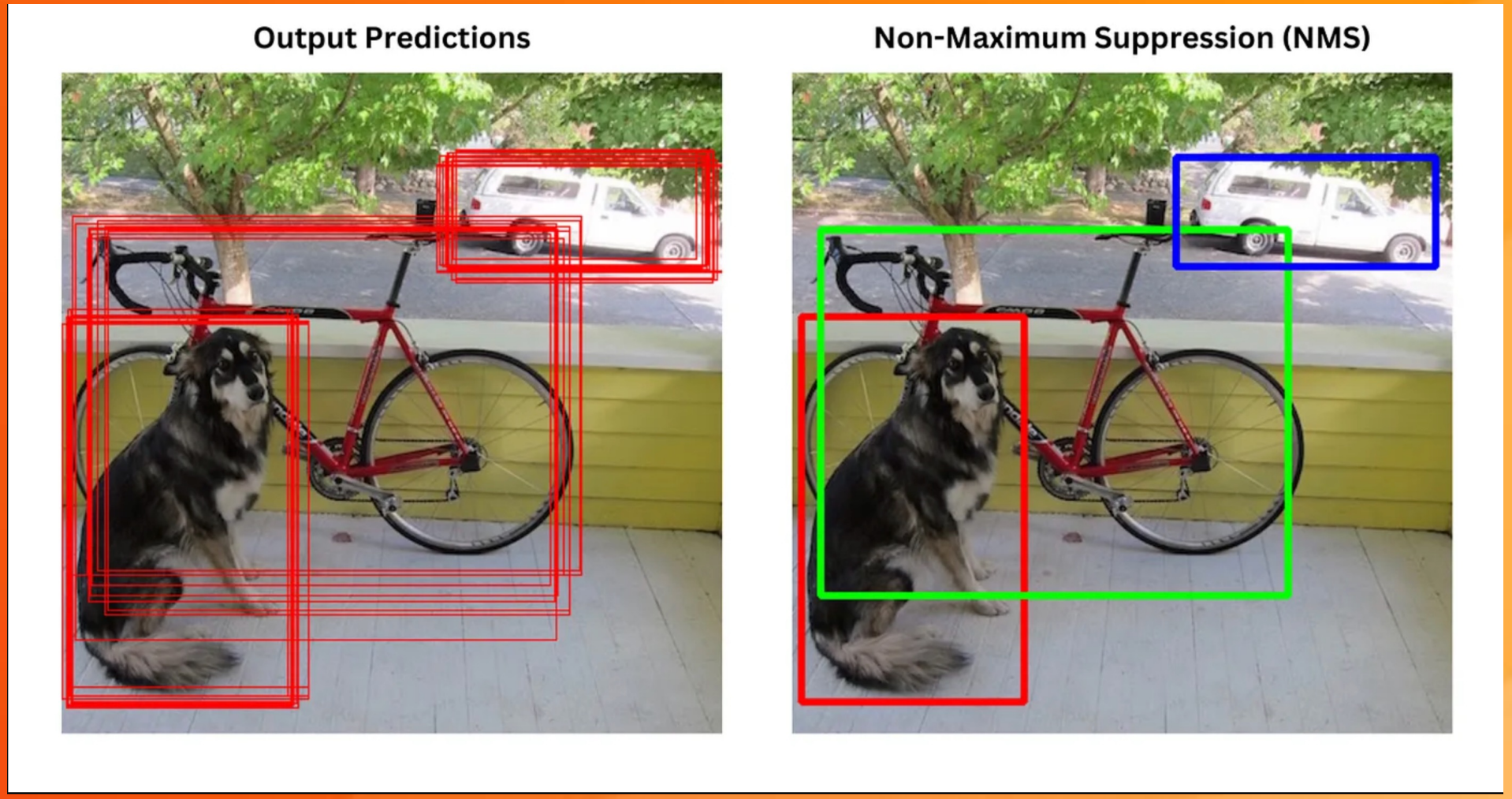

# The king has returned stronger

# YOLO Components



# Remove NMS – Non Max Suppression



# Remove NMS – Non Max Suppression

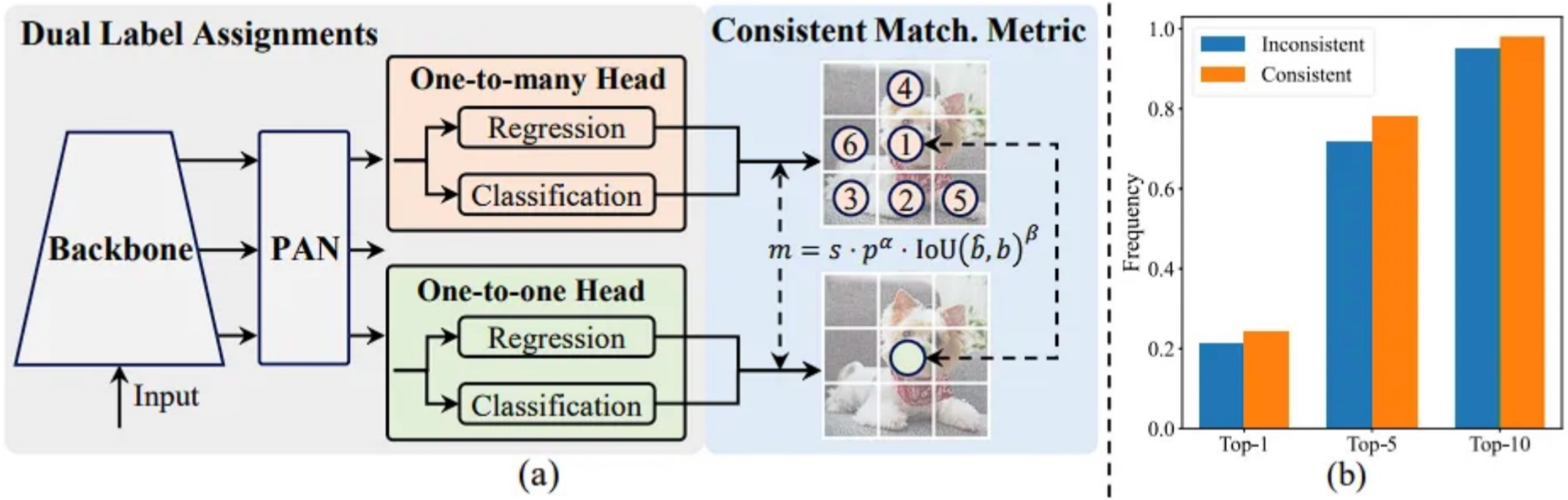
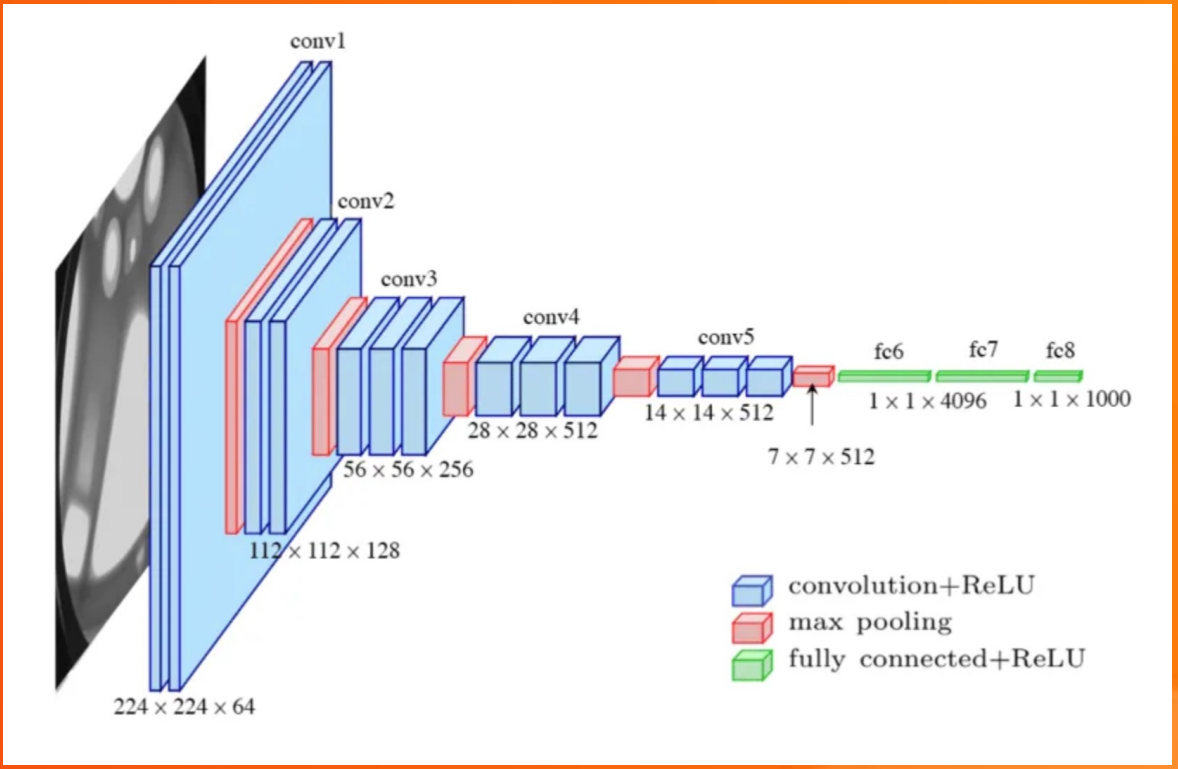



Figure 2: (a) Consistent dual assignments for NMS-free training. (b) Frequency of one-to-one assignments in Top-1/5/10 of one-to-many results for YOLOv8-S which employs $\alpha_{o2m}$=0.5 and $\beta_{o2m}$=6 by default [20]. For consistency, $\alpha_{o2o}$=0.5; $\beta_{o2o}$=6. For inconsistency, $\alpha_{o2o}$=0.5; $\beta_{o2o}$ =2.

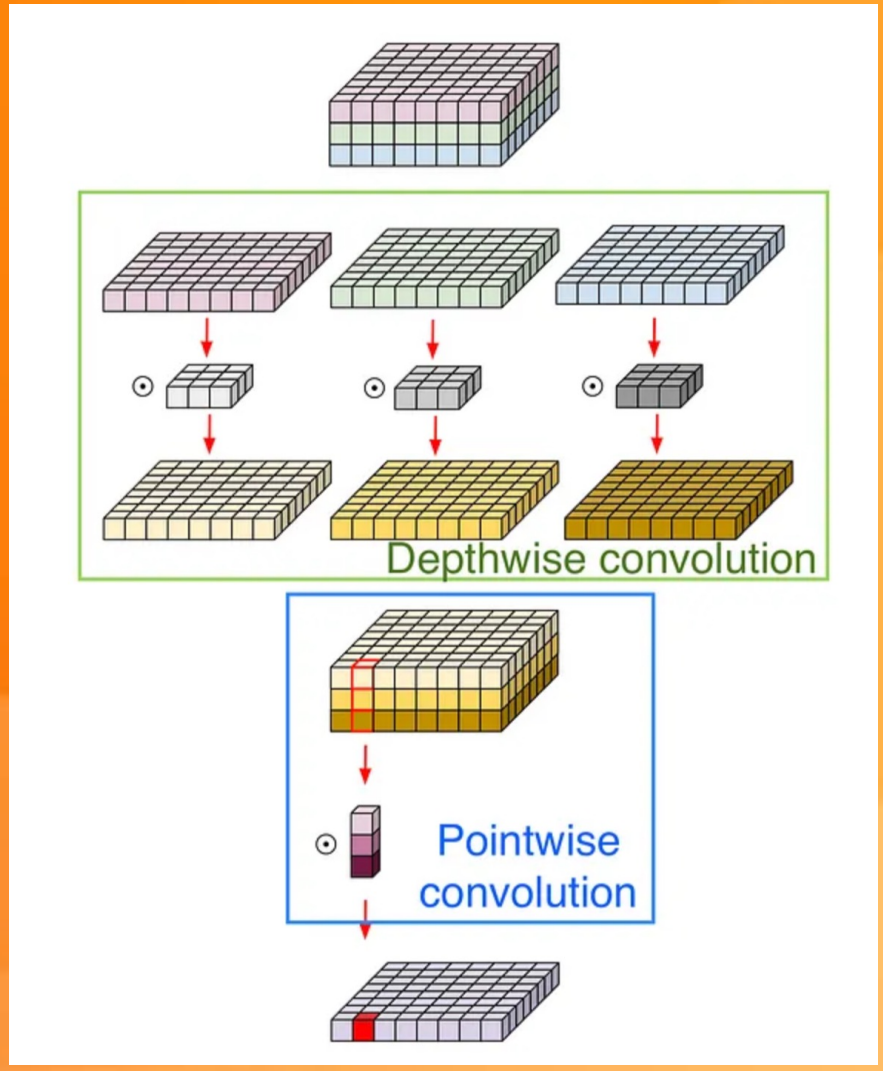
# Remove NMS – Non Max Suppression

- Both heads are trained simultaneously, allowing the backbone and neck of the model to leverage the comprehensive supervision from the one-to-many assignments, which improves the model’s learning and accuracy.
- And then during inference, the one-to-many head is discarded (Similar to deep supervision). Only the one-to-one head is used to make predictions. This approach ensures that the model can be deployed end-to-end without additional computational costs during inference.

# Spatial-Channel Decoupled Downsampling




Decoupling reduces the computational cost to O(2HW C^2 + 9/2(HW C)) and the parameter count to O(2C^2 + 18C). Meanwhile, it maximizes information retention during downsampling, leading to a competitive performance with latency reduction.



# Rank-Guided Block Design

- YOLOv10 proposes a rank-guided block design scheme that aims to decrease the complexity of stages that are shown to be redundant using compact architecture design.
- Sort all stages based on their intrinsic ranks in ascending order.
- Replace basic blocks in stages with higher redundancy (lower rank) with the more efficient CIB – Compact invert block.

# Rank-Guided Block Design

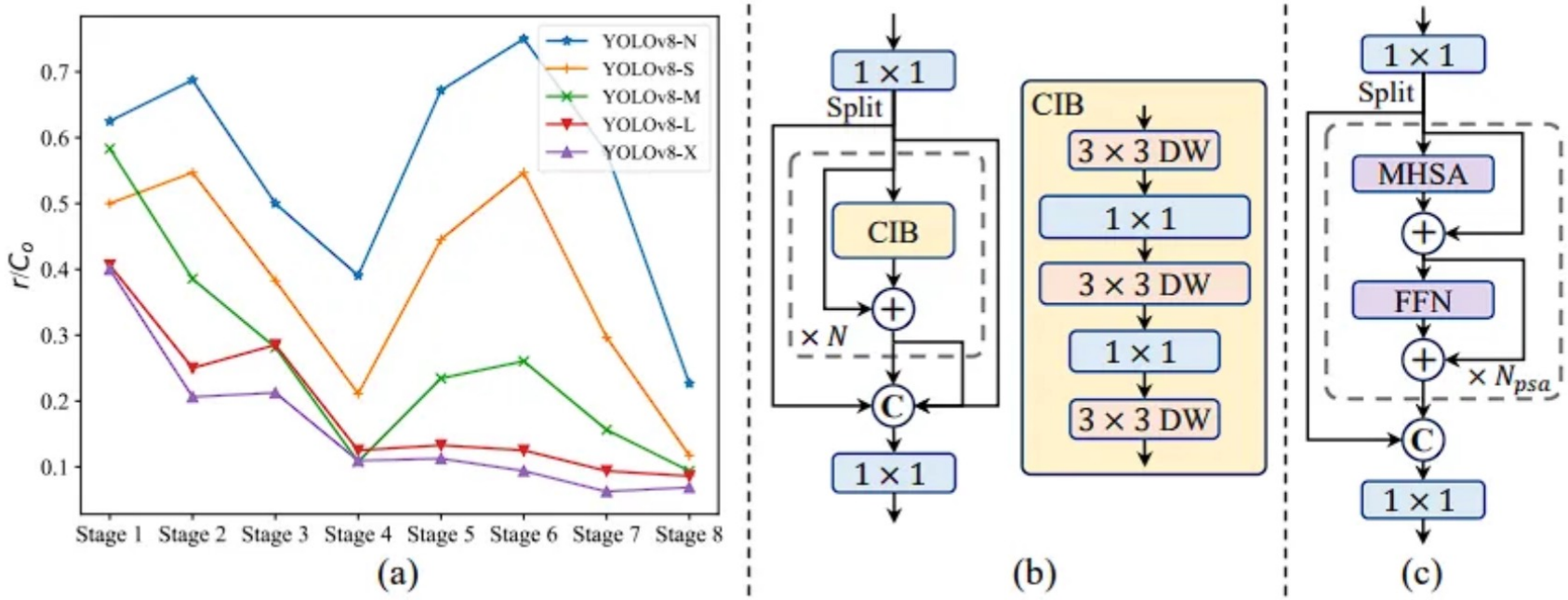
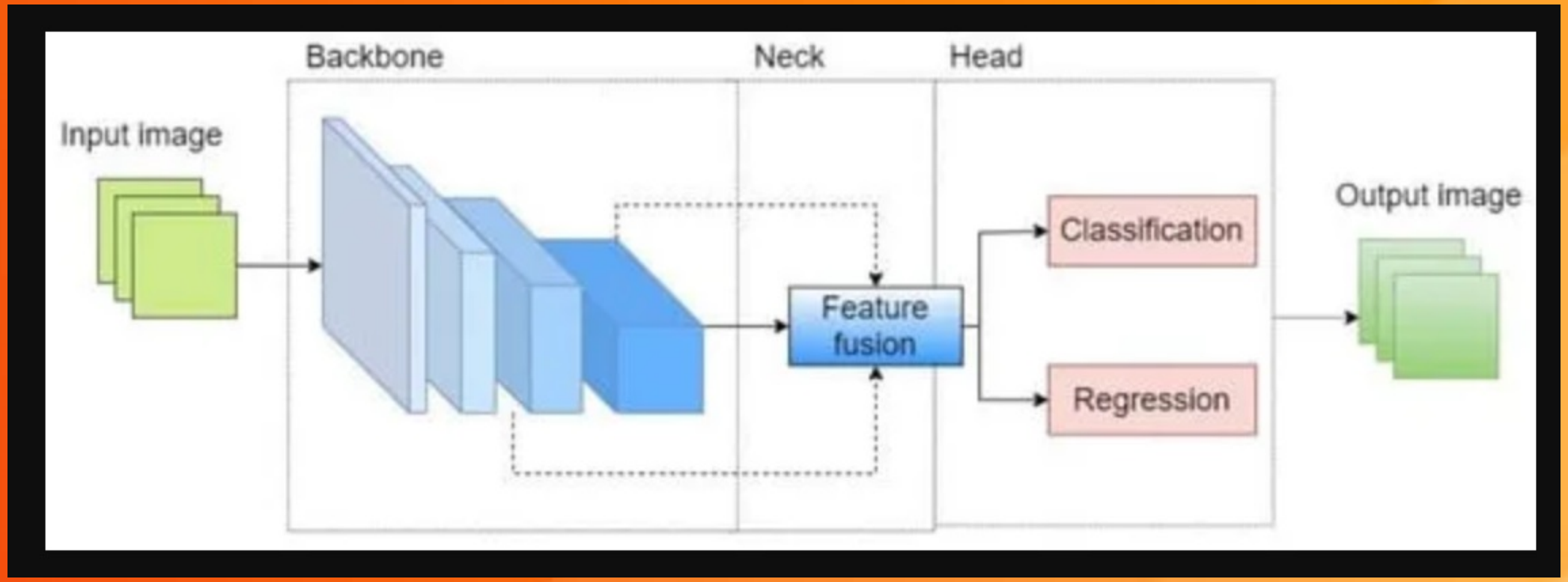



Figure 3: (a) The intrinsic ranks across stages and models in YOLOv8. The stage in the backbone and neck is numbered in the order of model forward process. The numerical rank $r$ is normalized to $r/C_o$ for y-axis and its threshold is set to $\lambda_{max}/2$, by default, where $C_o$ denotes the number of output channels and $\lambda_{max}$ is the largest singular value. It can be observed that deep stages and large models exhibit lower intrinsic rank values. (b) The compact inverted block (CIB). (c) The partial self-attention module (PSA).

# Rank-Guided Block Design

- Researchers further inspect the performance variation of replacing the basic block in the leading stage with CIB. If there is no performance degradation compared with the given model, they proceed with the replacement of the next stage and halt the process otherwise. Consequently, they implement adaptive compact block designs across stages and model scales, achieving higher efficiency without compromising performance.

# Lightweight Classification Head



# Lightweight Classification Head

## Classification Head

1. **Object Identification:** The classification head is responsible for determining the class of each detected object within a given bounding box. For example, it might classify objects as ‘person’, ‘car’, ‘dog’, etc.
2. **Class Probability Estimation:** It calculates the probability that a detected object belongs to each possible class. This involves applying a softmax function to output probabilities across multiple classes, ensuring the sum of probabilities is equal to one.

# Lightweight Classification Head

**Regression Head**

1. **Bounding Box Prediction:** The regression head is responsible for predicting the precise coordinates of bounding boxes that enclose detected objects. This includes predicting the center coordinates (x, y), width, and height of each bounding box.
2. **Box Confidence Score:** It also outputs a confidence score for each bounding box, indicating the likelihood that the box contains an object. This score helps in filtering out low-confidence predictions during post-processing.

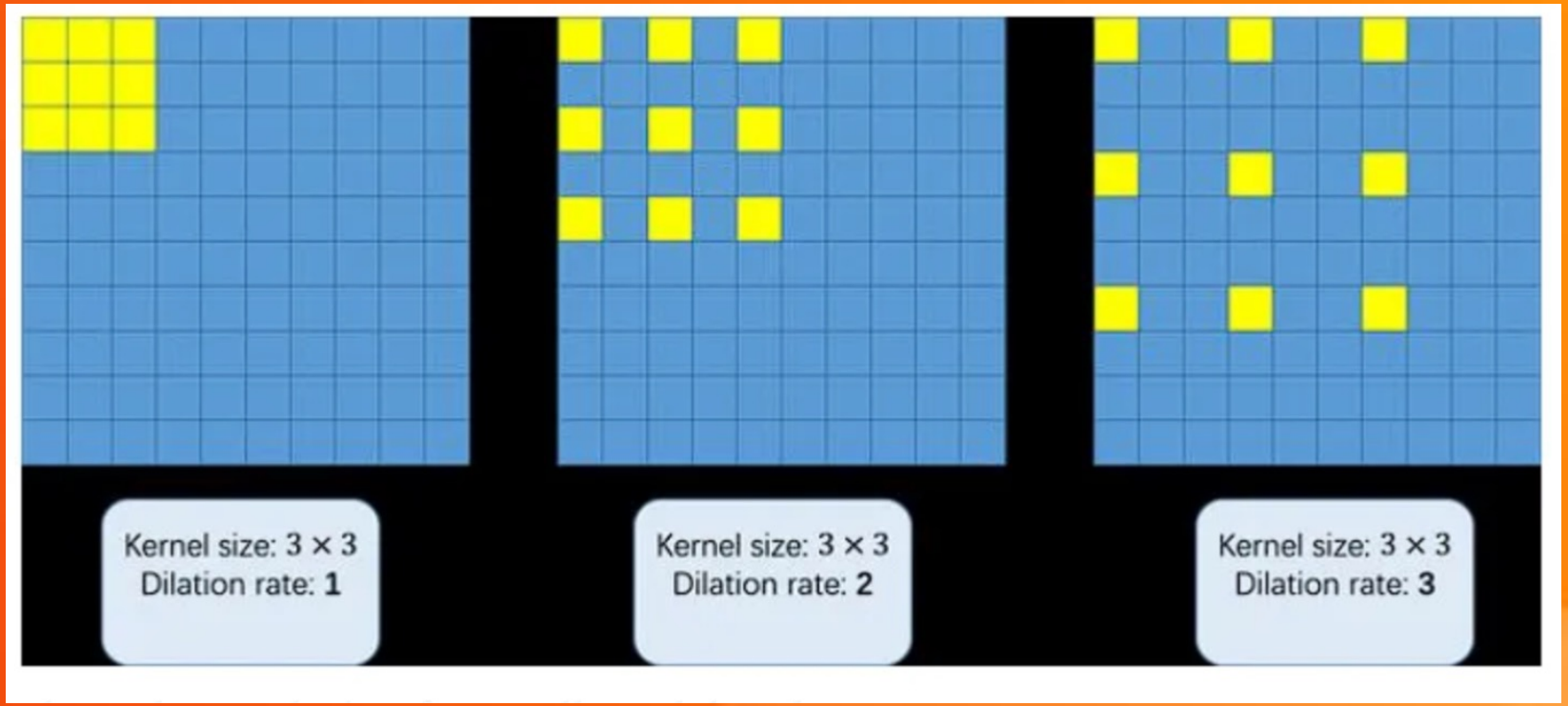
# Lightweight Classification Head

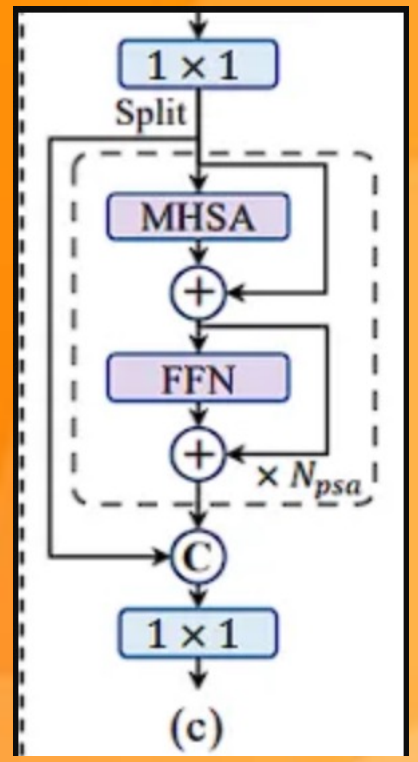
Researchers found that the **regression head** undertakes more significance for the performance of YOLOs. Consequently, we can reduce the overhead of the **classification head** without worrying about hurting the performance greatly. Therefore, they simply adopt a lightweight architecture for the classification head, which consists of two depthwise separable convolutions with a kernel size of 3×3 followed by a 1×1 convolution.

# Accuracy Driven Design

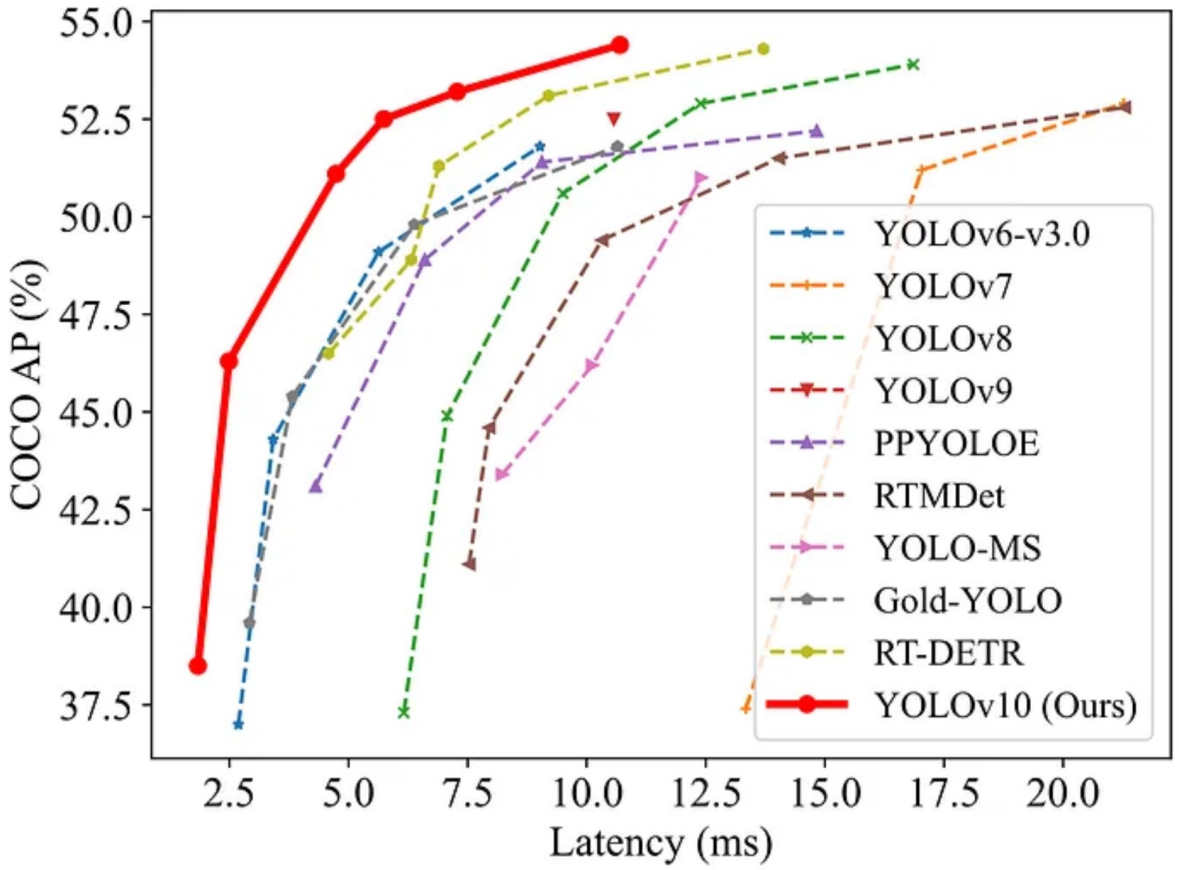
- **Large-Kernel Convolution:** YOLOv10 selectively employs larger convolution kernels in the model, increasing the receptive field and allowing the network to “see” and understand objects within a wider context. This is particularly beneficial for detecting small objects often missed by models with limited receptive fields.
- **Partial Self-Attention (PSA):** YOLOv10 incorporates a resource-efficient self-attention mechanism that enhances the model’s ability to capture long-range dependencies within the image, further improving object detection accuracy without significantly increasing computational cost.

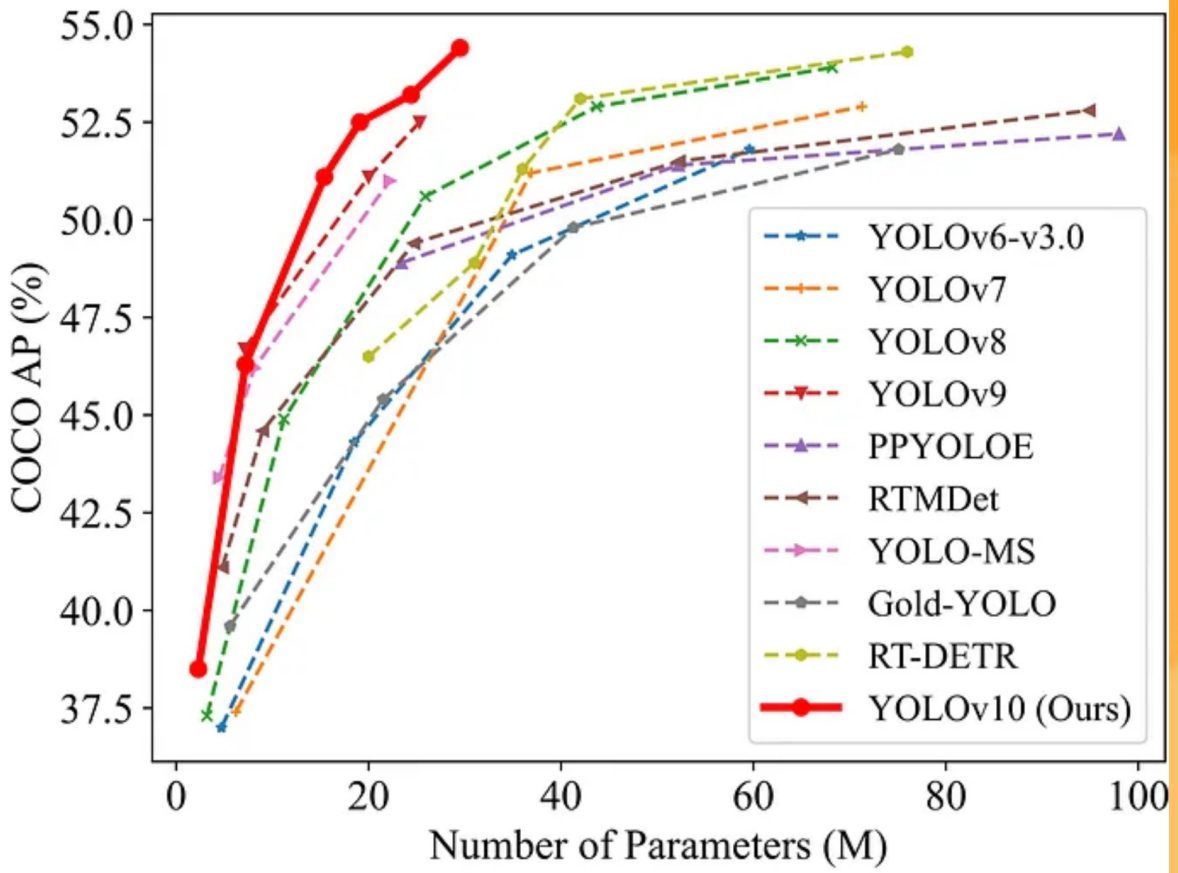
# Accuracy Driven Design



# Result


55.0
52.5
50.0
47.5
45.0
42.5
40.0
37.5
COCO AP (%)
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Latency (ms)
YOLOv6-v3.0
YOLOv7
YOLOv8
YOLOv9
PPYOLOE
RTMDet
YOLO-MS
Gold-YOLO
RT-DETR
YOLOv10 (Ours)


55.0
52.5
50.0
47.5
45.0
42.5
40.0
37.5
COCO AP (%)
0
20
40
60
80
100
Number of Parameters (M)
YOLOv6-v3.0
YOLOv7
YOLOv8
YOLOv9
PPYOLOE
RTMDet
YOLO-MS
Gold-YOLO
RT-DETR
YOLOv10 (Ours)

# YOLOv10 Family

| Model | Input Size | $AP^{val}$ | FLOPs (G) | Latency (ms) |
|---|---|---|---|---|
| YOLOv10-N | 640 | 38.5 | **6.7** | **1.84** |
| YOLOv10-S | 640 | 46.3 | 21.6 | 2.49 |
| YOLOv10-M | 640 | 51.1 | 59.1 | 4.74 |
| YOLOv10-B | 640 | 52.5 | 92.0 | 5.74 |
| YOLOv10-L | 640 | 53.2 | 120.3 | 7.28 |
| YOLOv10-X | 640 | **54.4** | 160.4 | 10.70 |

# Reference

https://vishal-ai.medium.com/yolov10-object-detection-king-is-back-739eaaab134d
https://ai.gopubby.com/yolov10-the-last-model-from-the-yolo-family-604c360d3fdb
https://medium.com/@batuhansenerr/yolov10-custom-object-detection-bd7298ddbfd3