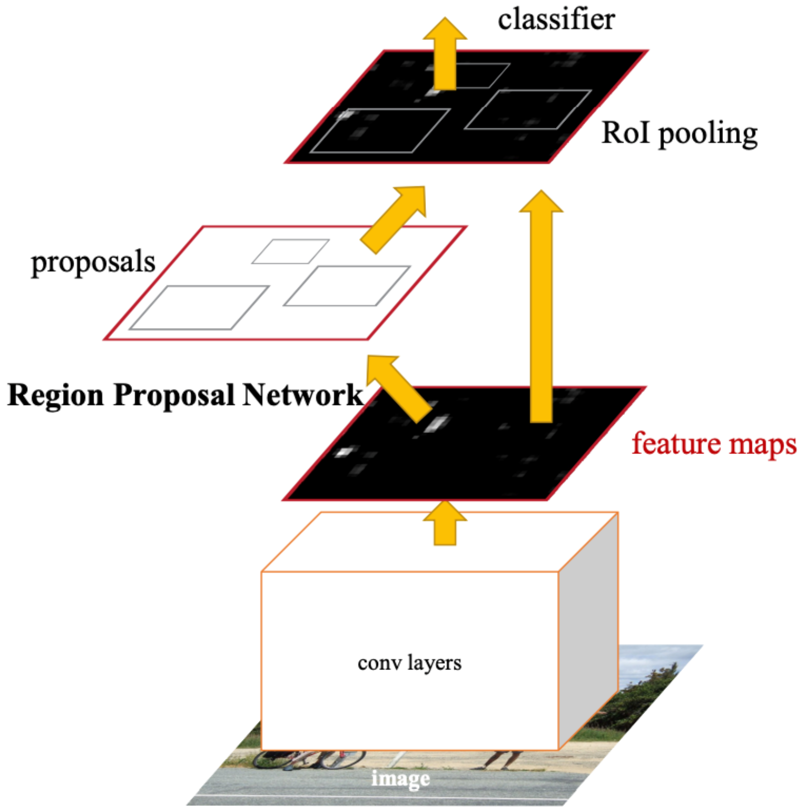
Real-time Instance Segmentation with the YOLACT Family

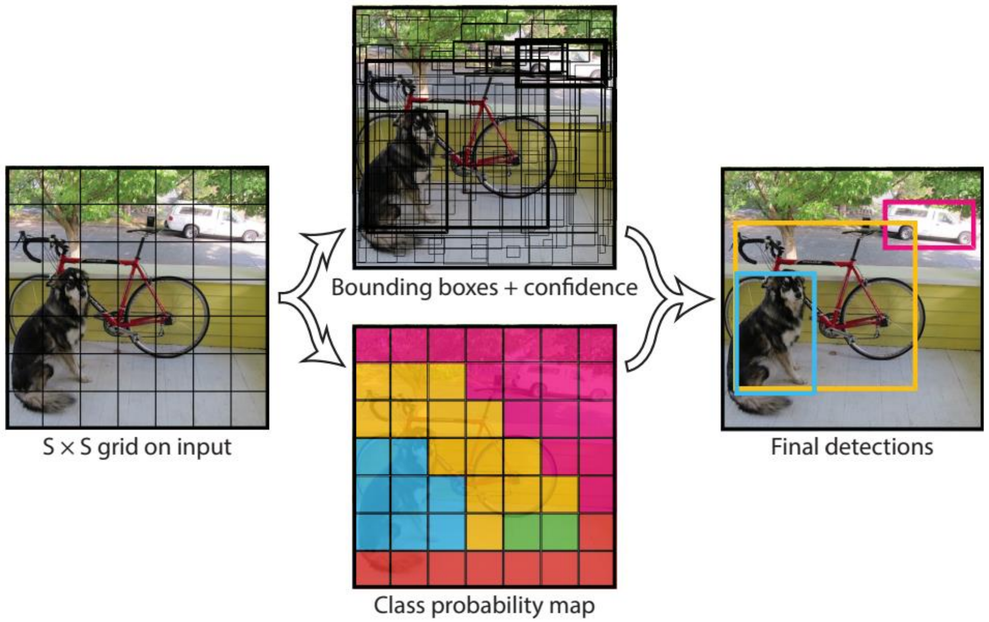
Yong Jae Lee
UC Davis / Cruise AI

Back in 2018 ...



2-stage object detection:

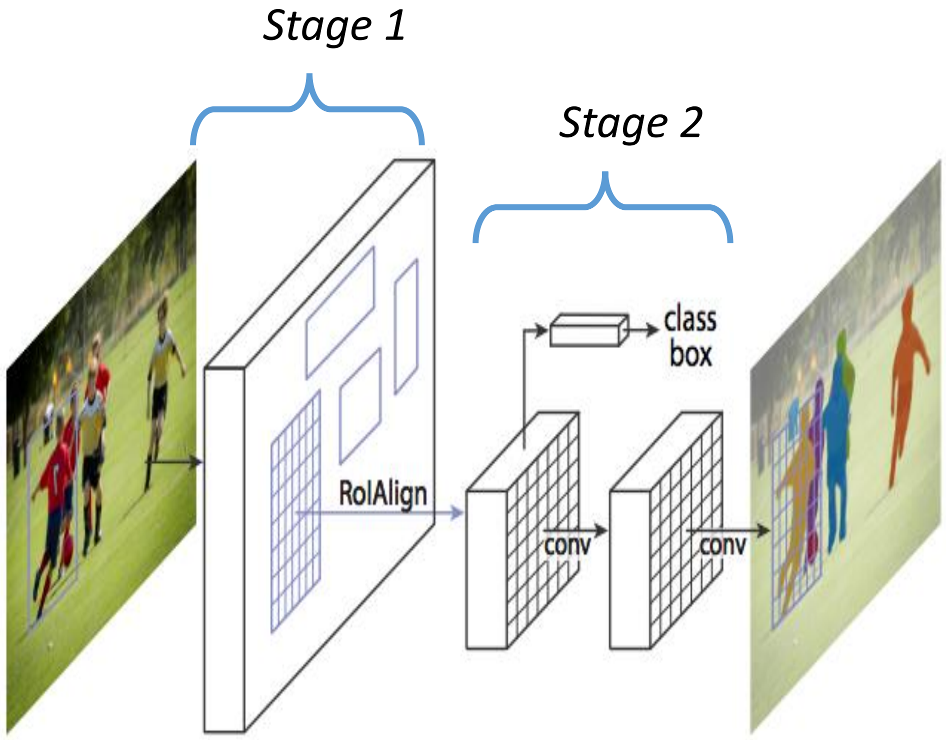
Faster-RCNN



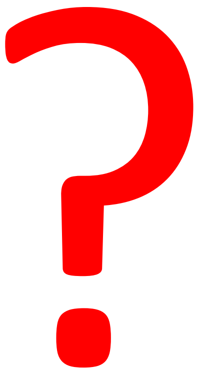
1-stage object detection:

YOLO, SSD

Back in 2018 ...



"Boxes are stupid anyway though, I'm probably a true believer in masks except I can't get YOLO to learn them."
- Joseph Redmon, YOLOv3



2-stage instance segmentation:

Mask-RCNN

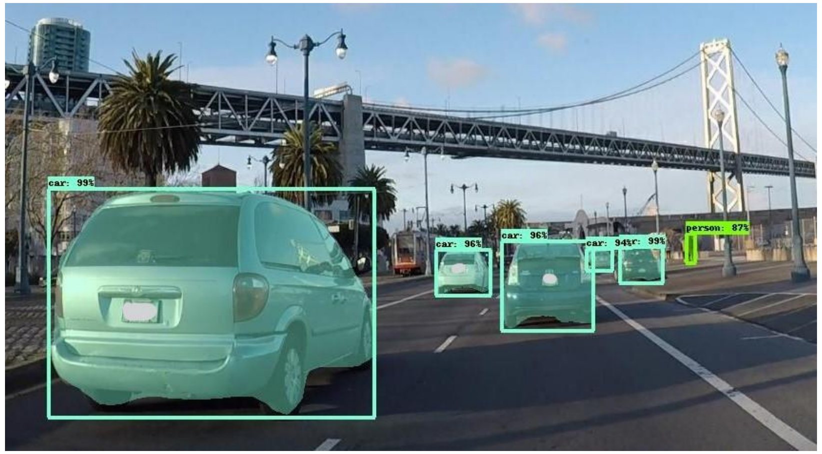
1-stage instance segmentation:

FCIS ... *but no real-time method*

Real-time instance segmentation applications

Robotics

Self-driving

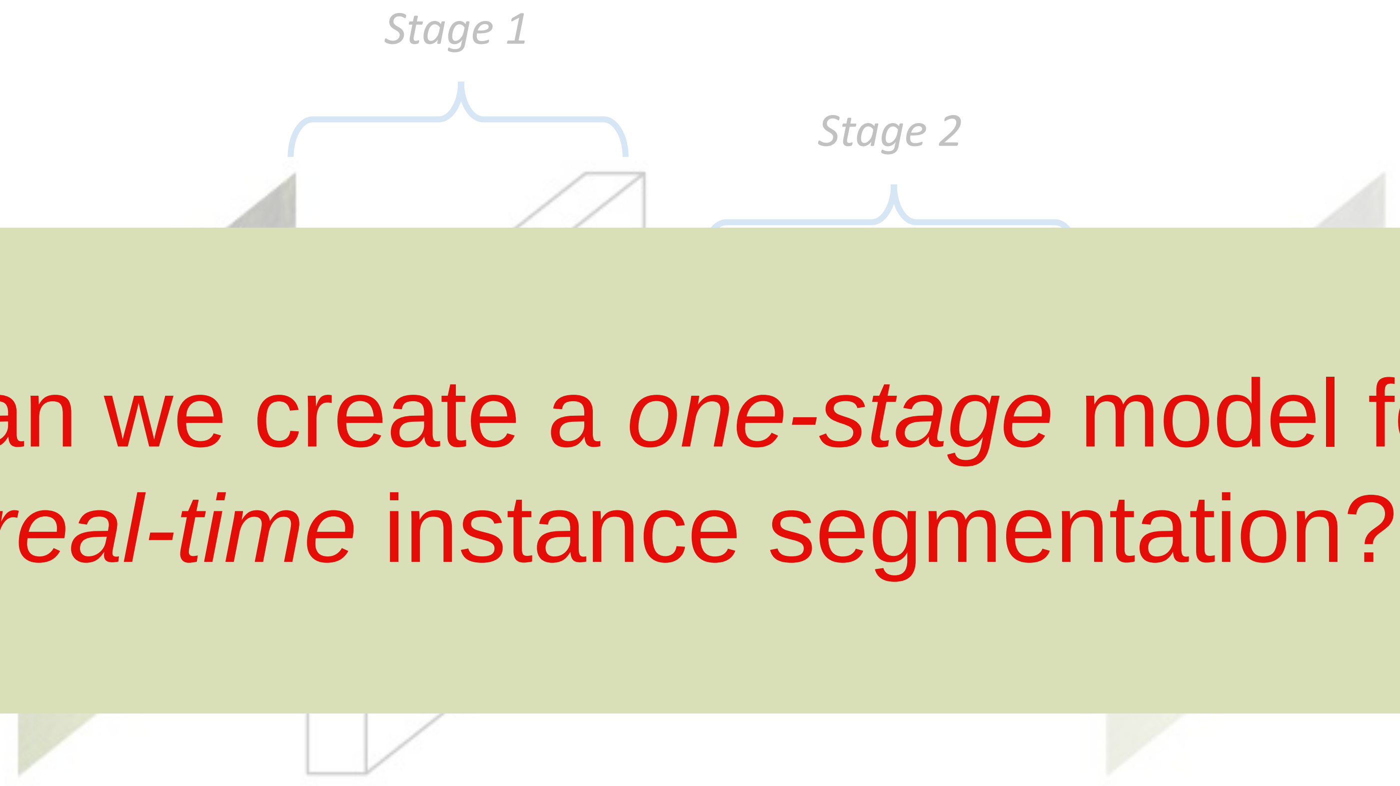


Drones

Instance segmentation is challenging

- Requires *different outputs* for same class instances in *different locations* (i.e. translation variance)

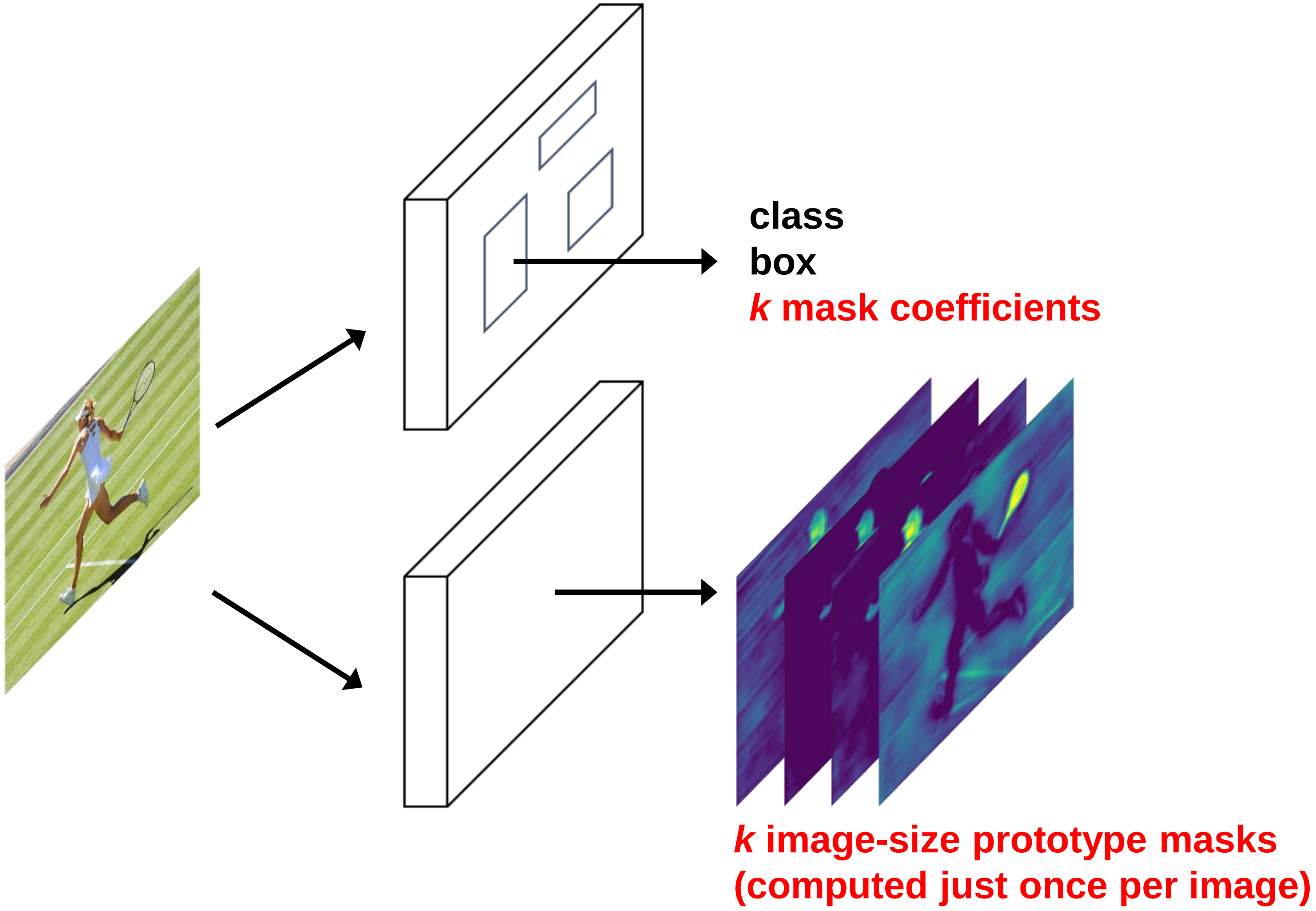
2-stage instance segmentation (e.g. Mask-RCNN)



Can we create a *one-stage* model for *real-time* instance segmentation?

- *Stage 1*: use Region Proposal Network to generate region proposals
- *Stage 2*: pool features for each proposal (via ROI-align) and classify

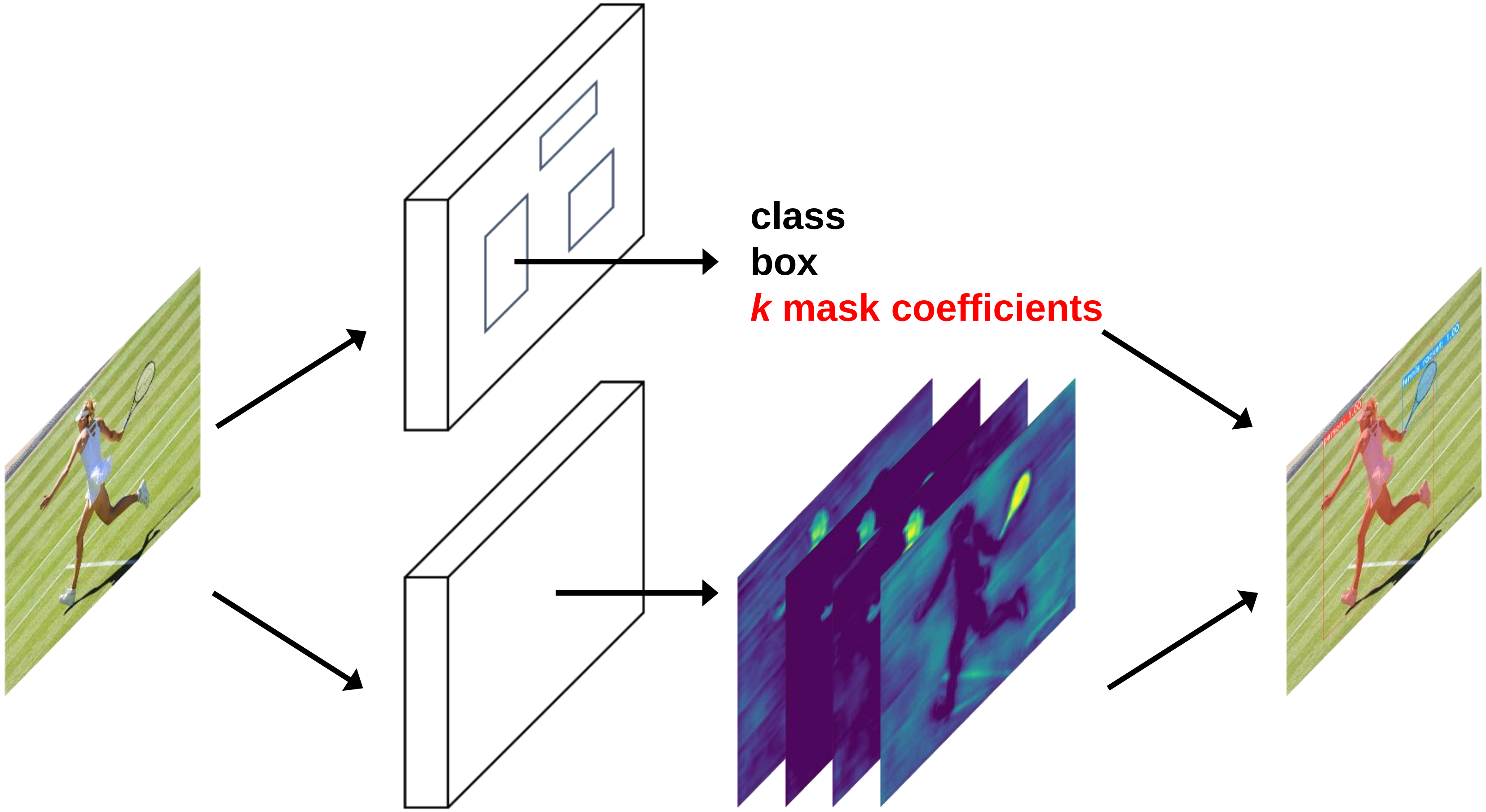
YOLACT Main Idea: Predict masks via two parallel tasks



- Linearly combine image-size prototype masks with mask coefficients

YOLACT: Real-time Instance Segmentation D. Bolya et al. ICCV 2019

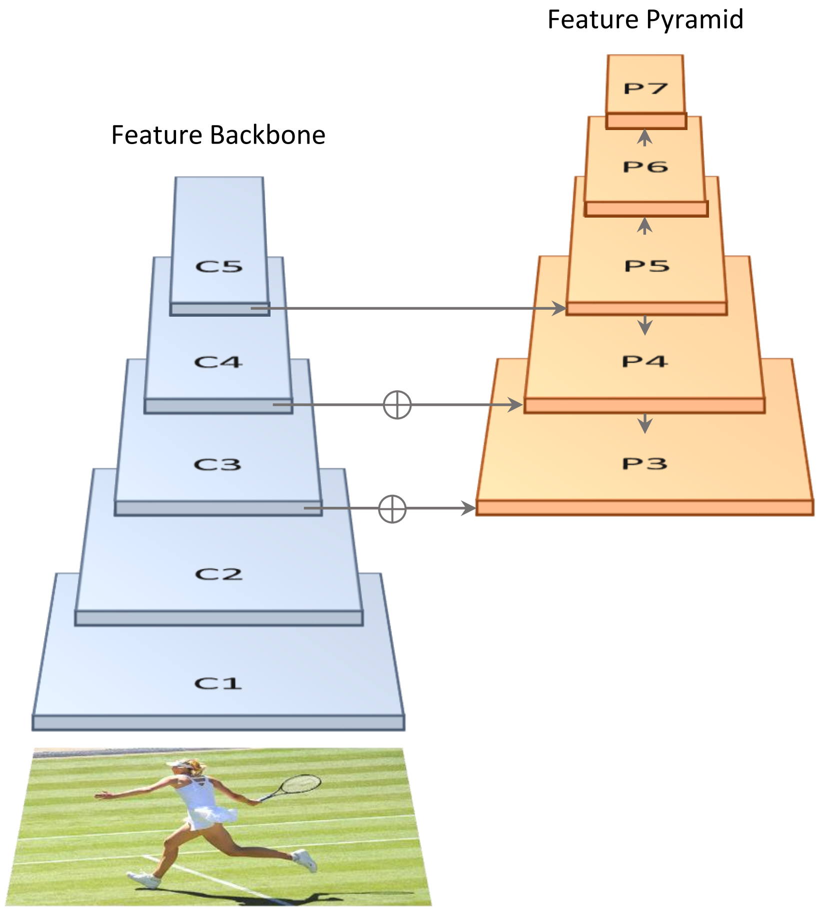
YOLACT Main Idea: Predict masks via two parallel tasks



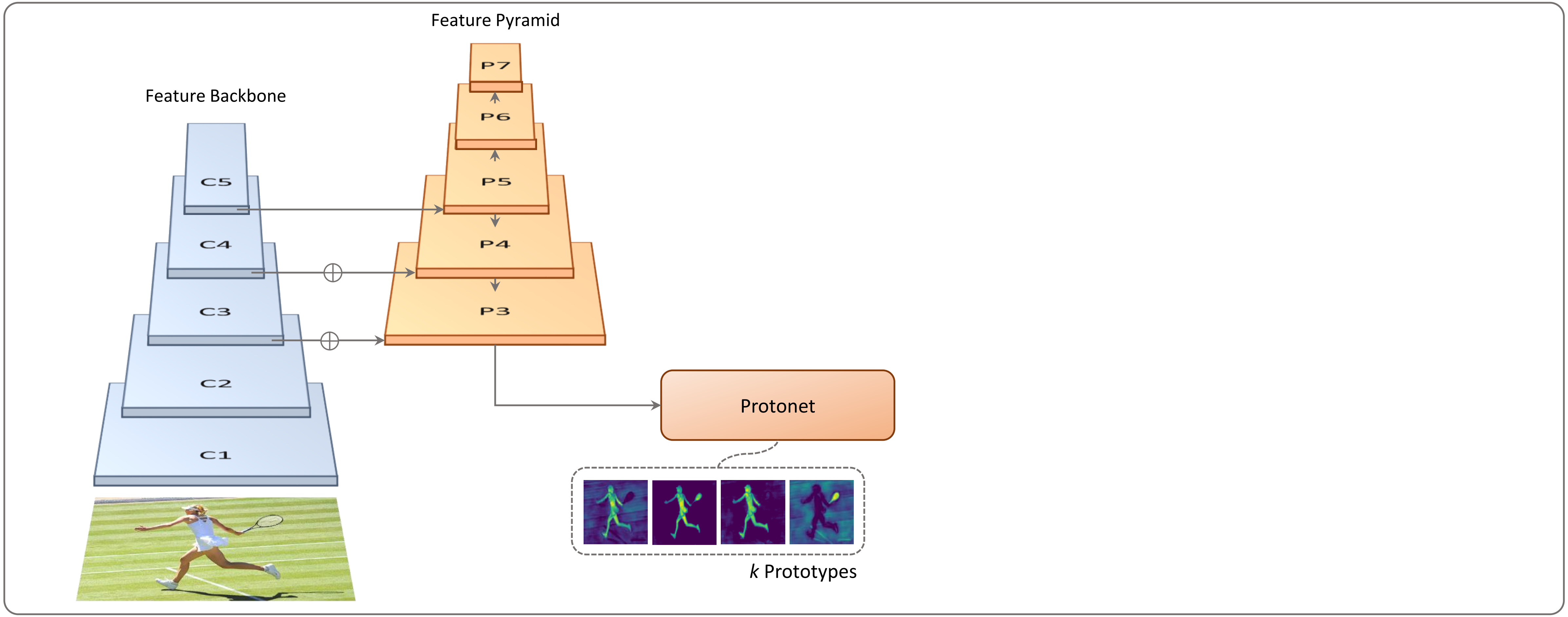
- Linearly combine image-size prototype masks with mask coefficients

YOLACT: Real-time Instance Segmentation D. Bolya et al. ICCV 2019

YOLACT architecture

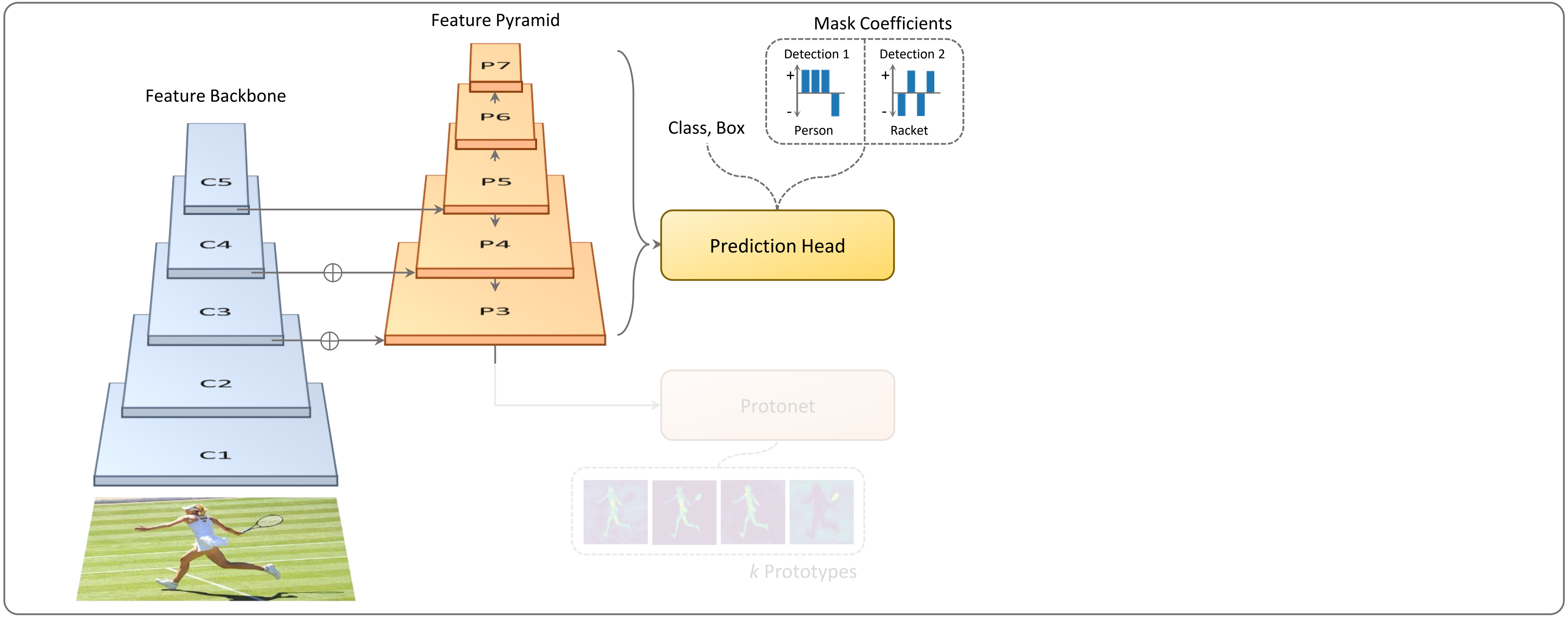


YOLACT architecture



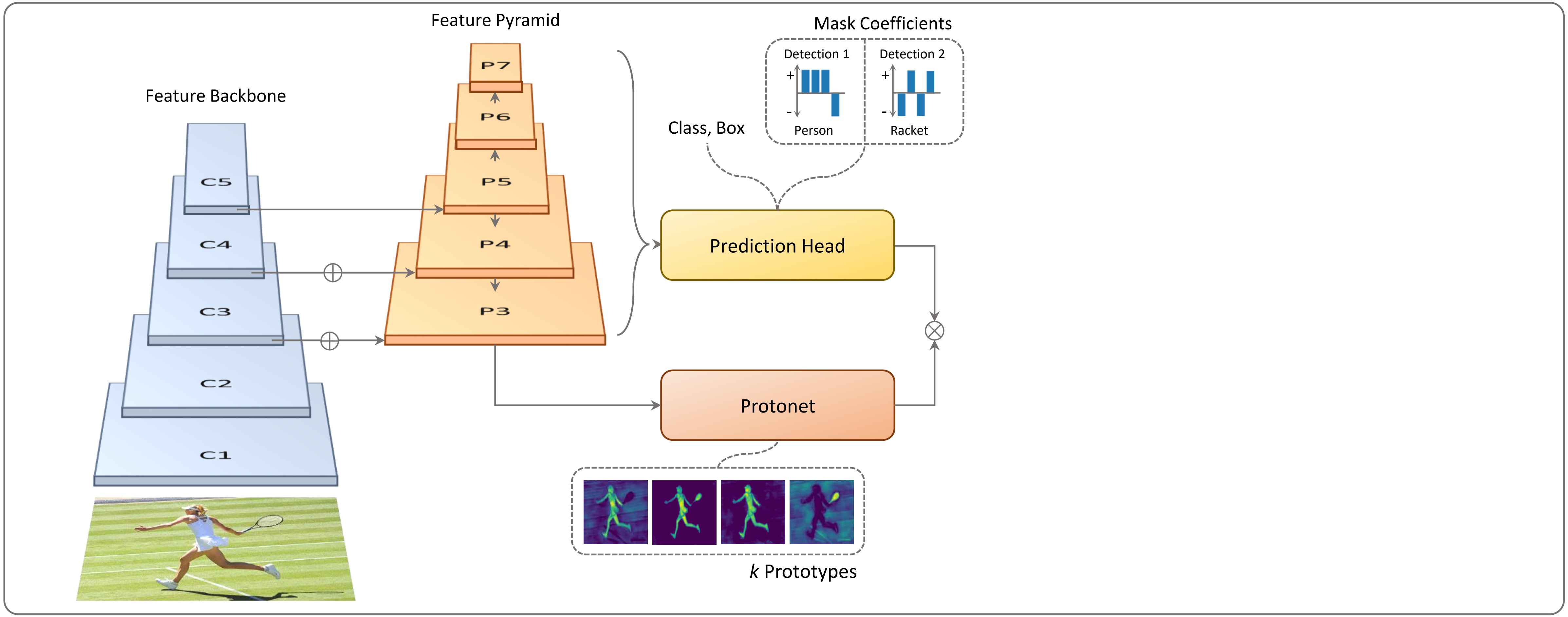
- Attach an FCN ("ProtoNet") to the largest feature layer (P3) to produce *k* image-resolution prototype masks

YOLACT architecture



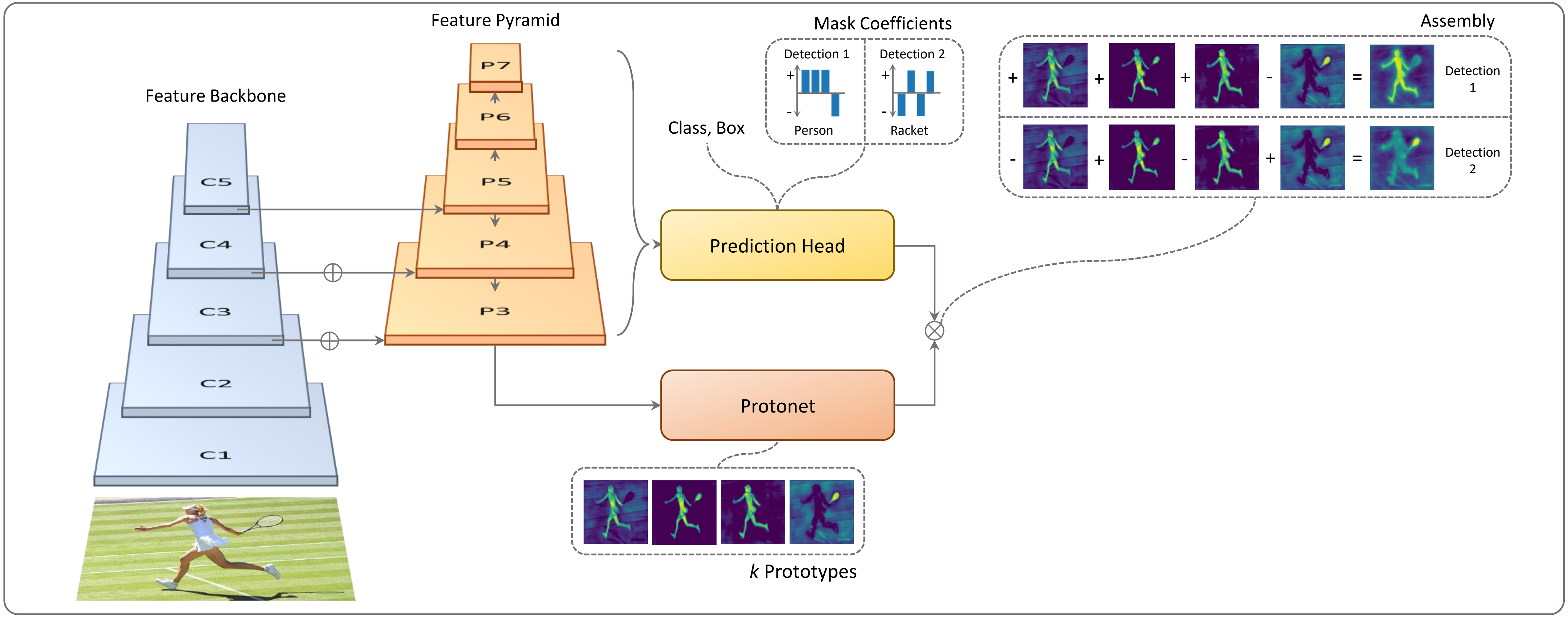
- In parallel, predict *k* mask coefficients for each anchor box (in addition to class confidences and box coefficients)

YOLACT architecture



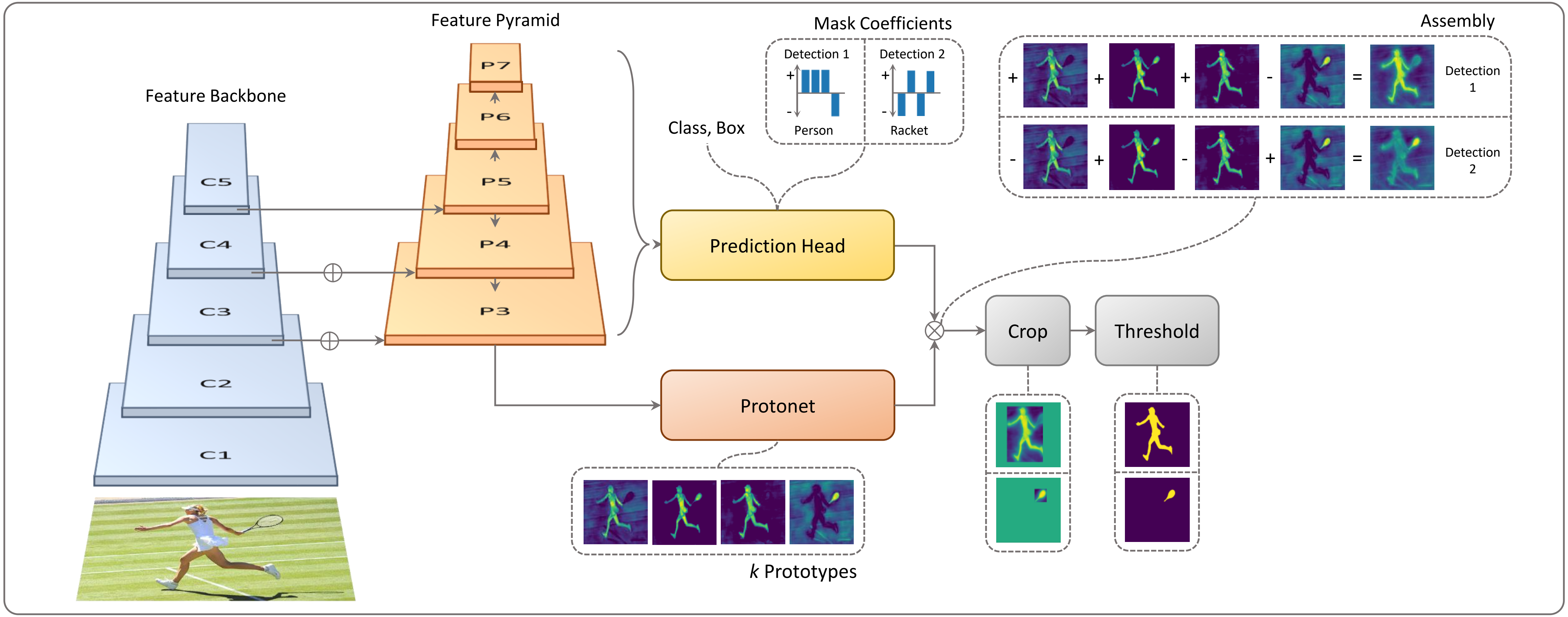
- For each instance, linearly combine prototypes using corresponding predicted coefficients

YOLACT architecture



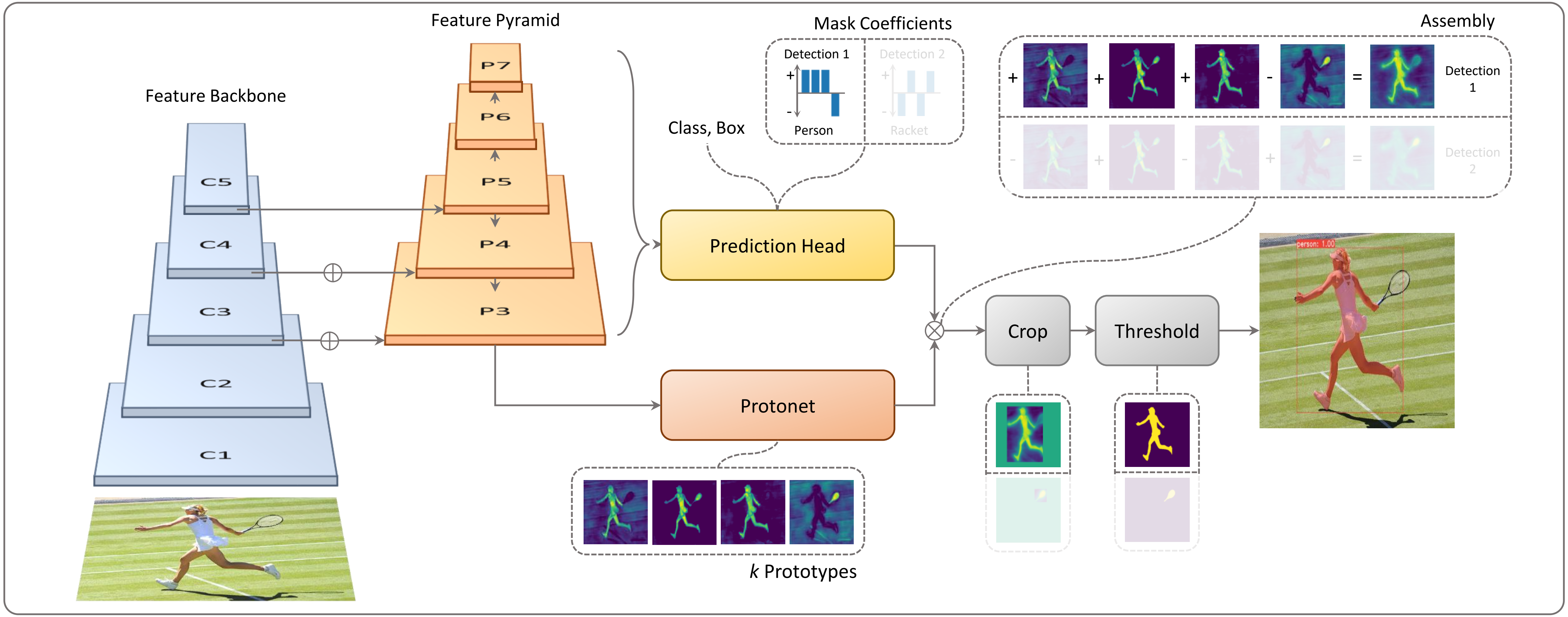
- For each instance, linearly combine prototypes using corresponding predicted coefficients

YOLACT architecture



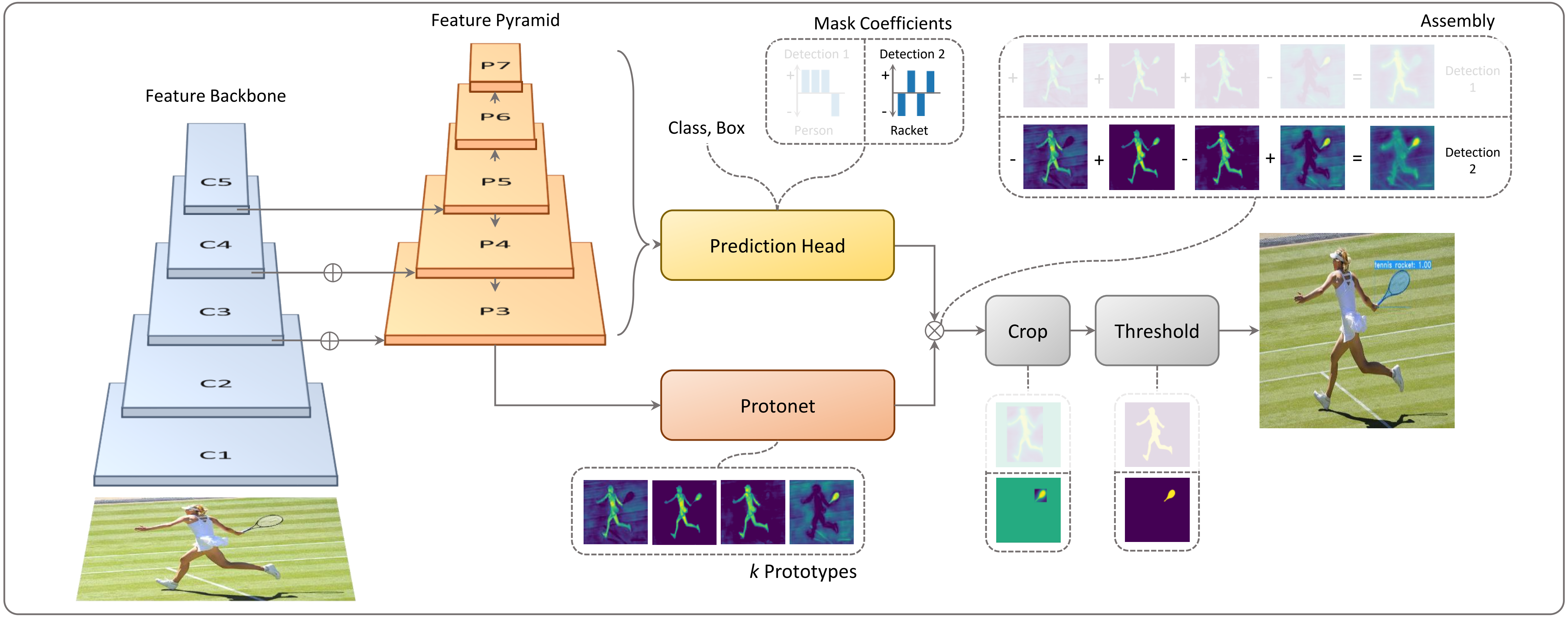
- Finally, crop with the predicted bounding box and threshold

YOLACT architecture



Example 1: Person

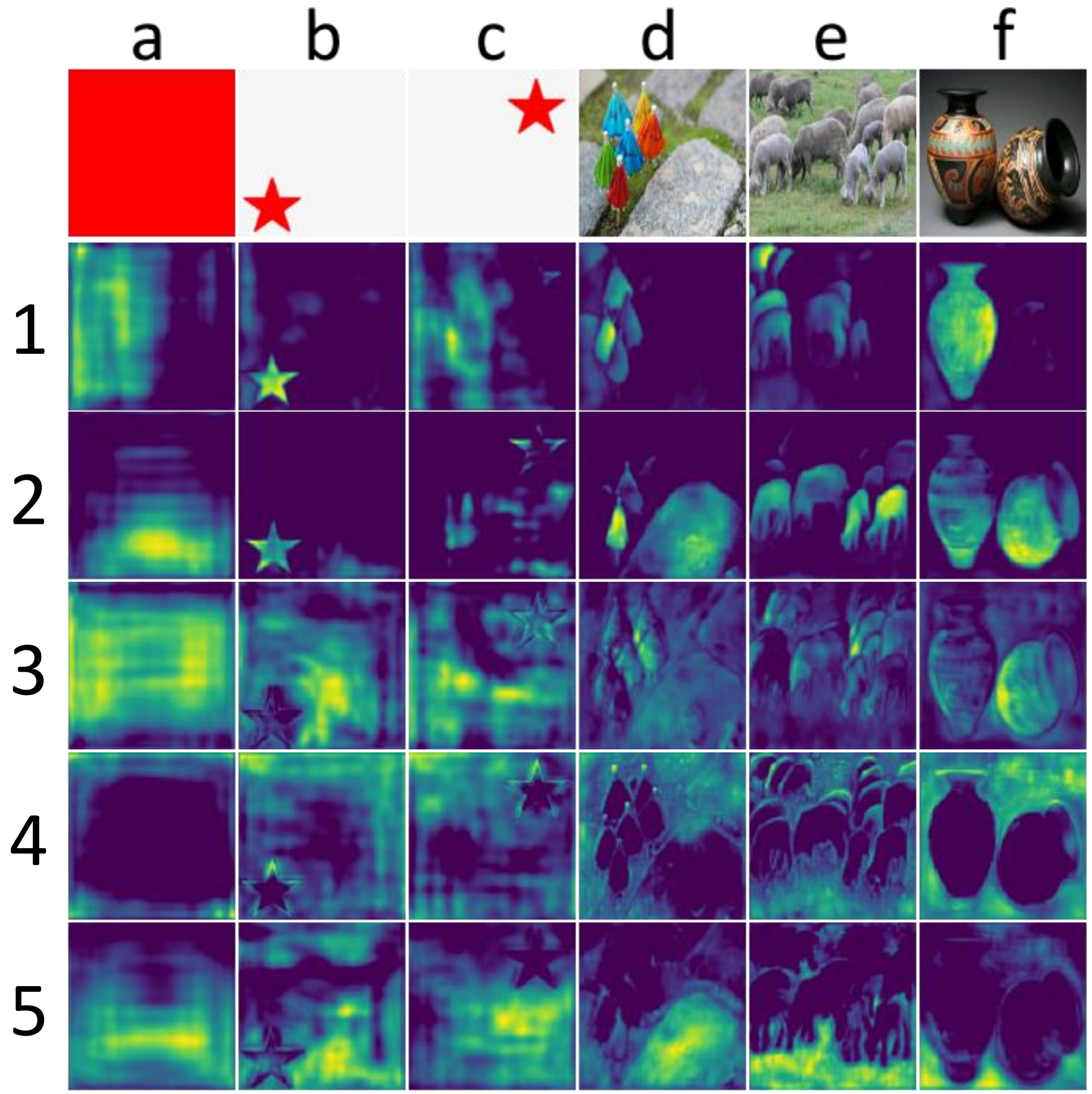
YOLACT architecture



Example 2: Tennis Racket

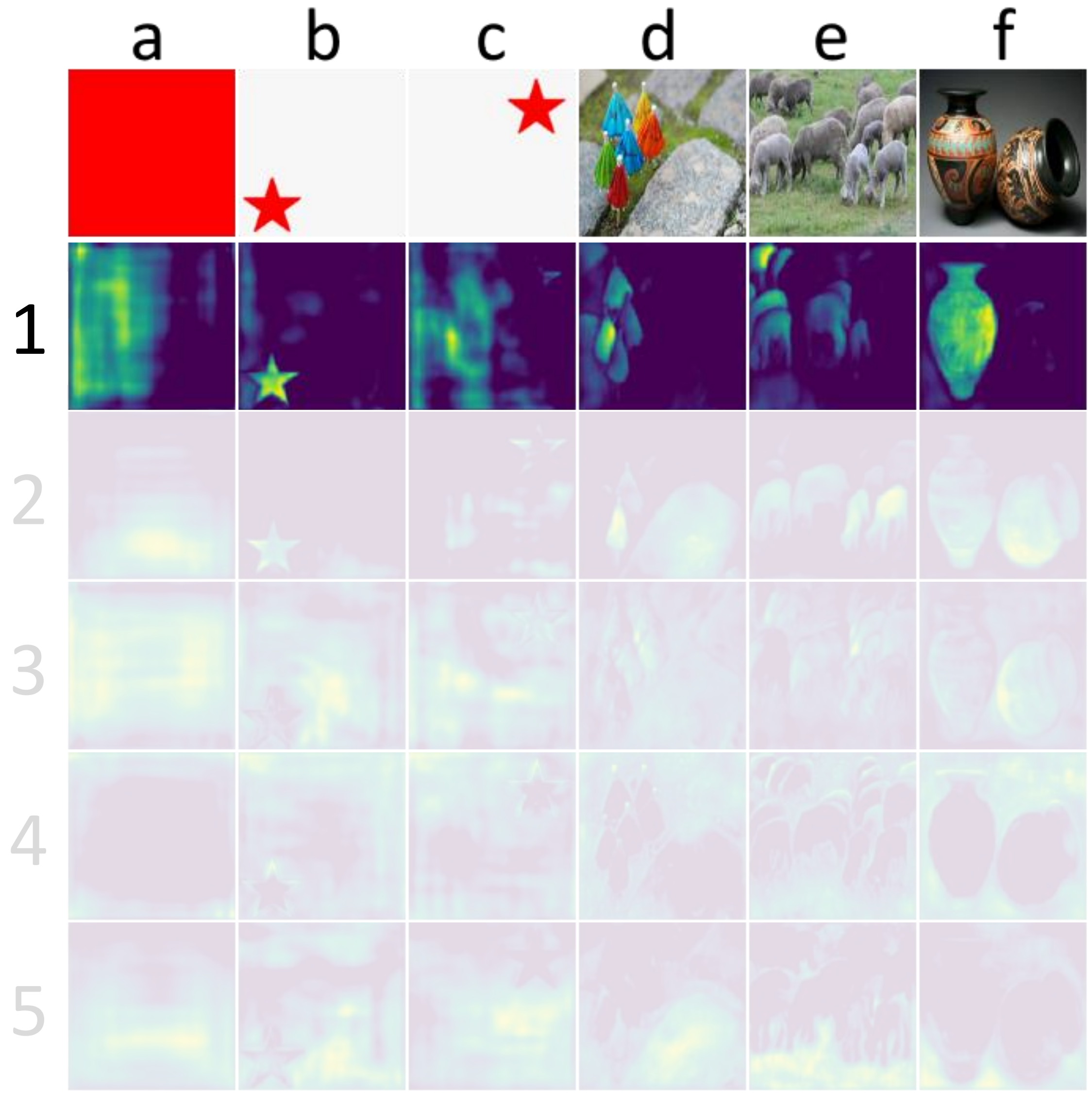
Prototype behavior

- *Supervision only comes from final mask loss*
- Spatially partition the image
- Segment background
- Detect instance contours
- Encode position-sensitive directional maps
- Most do a combination



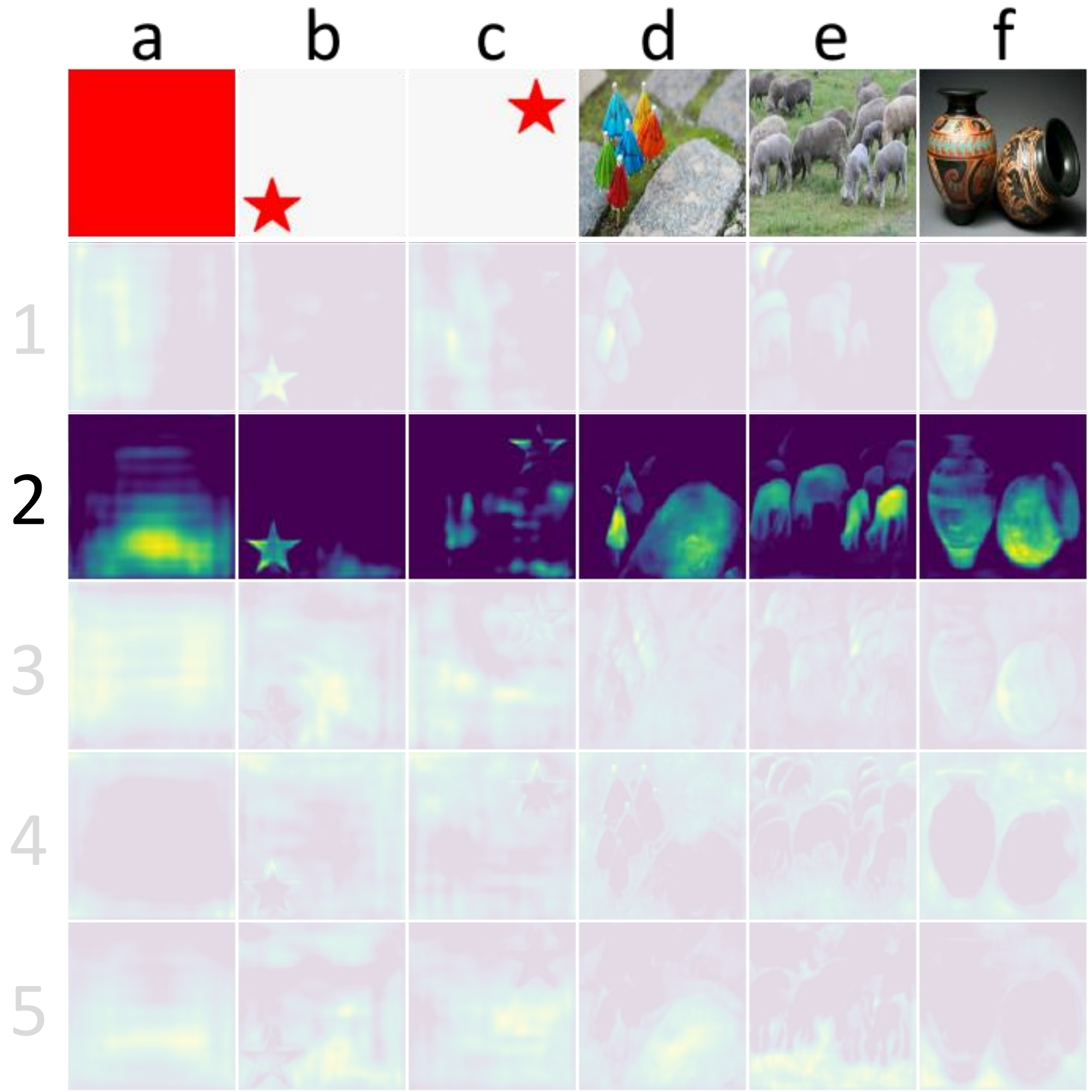
Prototype behavior

- *Supervision only comes from final mask loss*
- Spatially partition the image
- Segment background
- Detect instance contours
- Encode position-sensitive directional maps
- Most do a combination



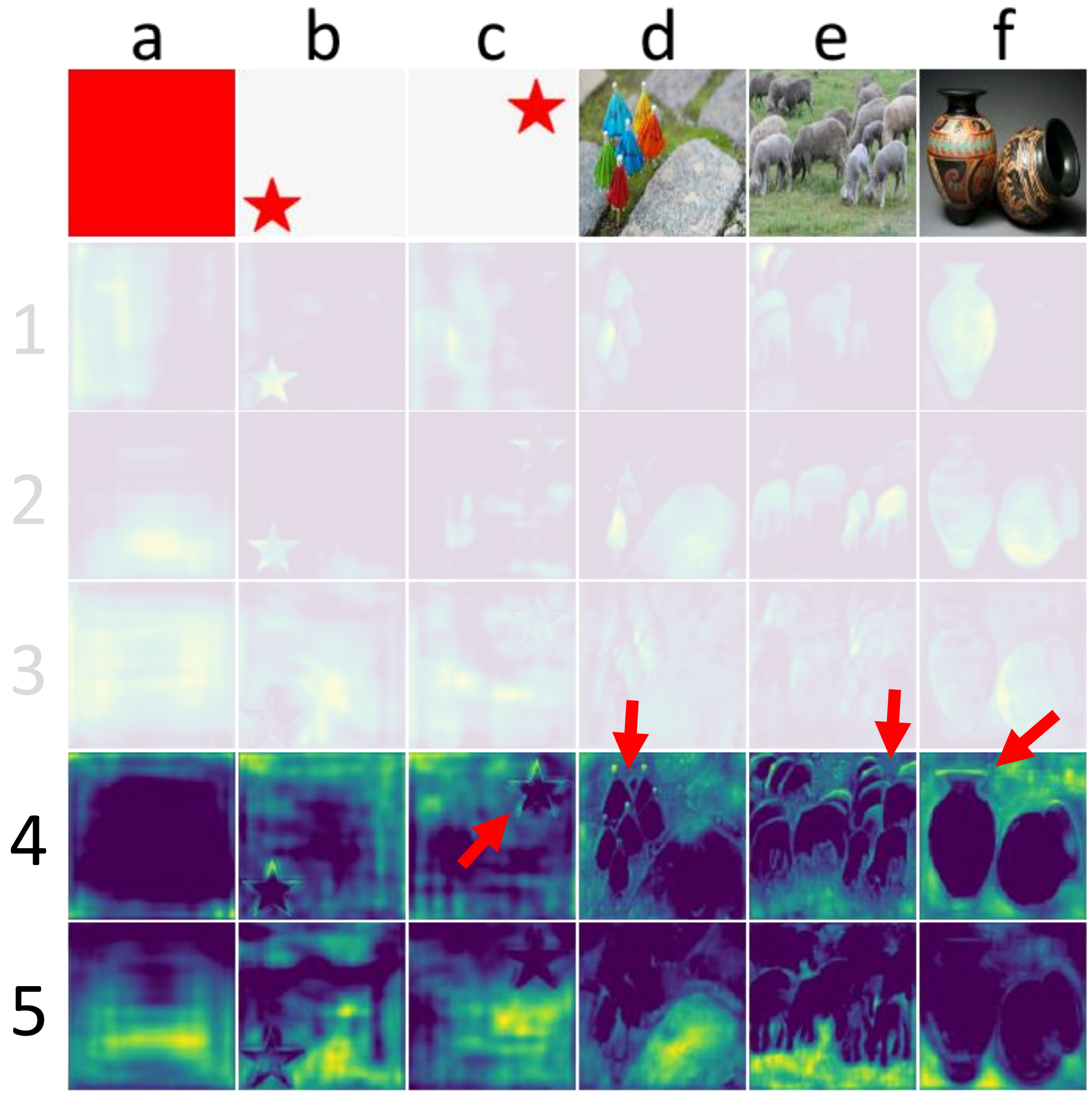
Prototype behavior

- *Supervision only comes from final mask loss*
- Spatially partition the image
- Segment background
- Detect instance contours
- Encode position-sensitive directional maps
- Most do a combination



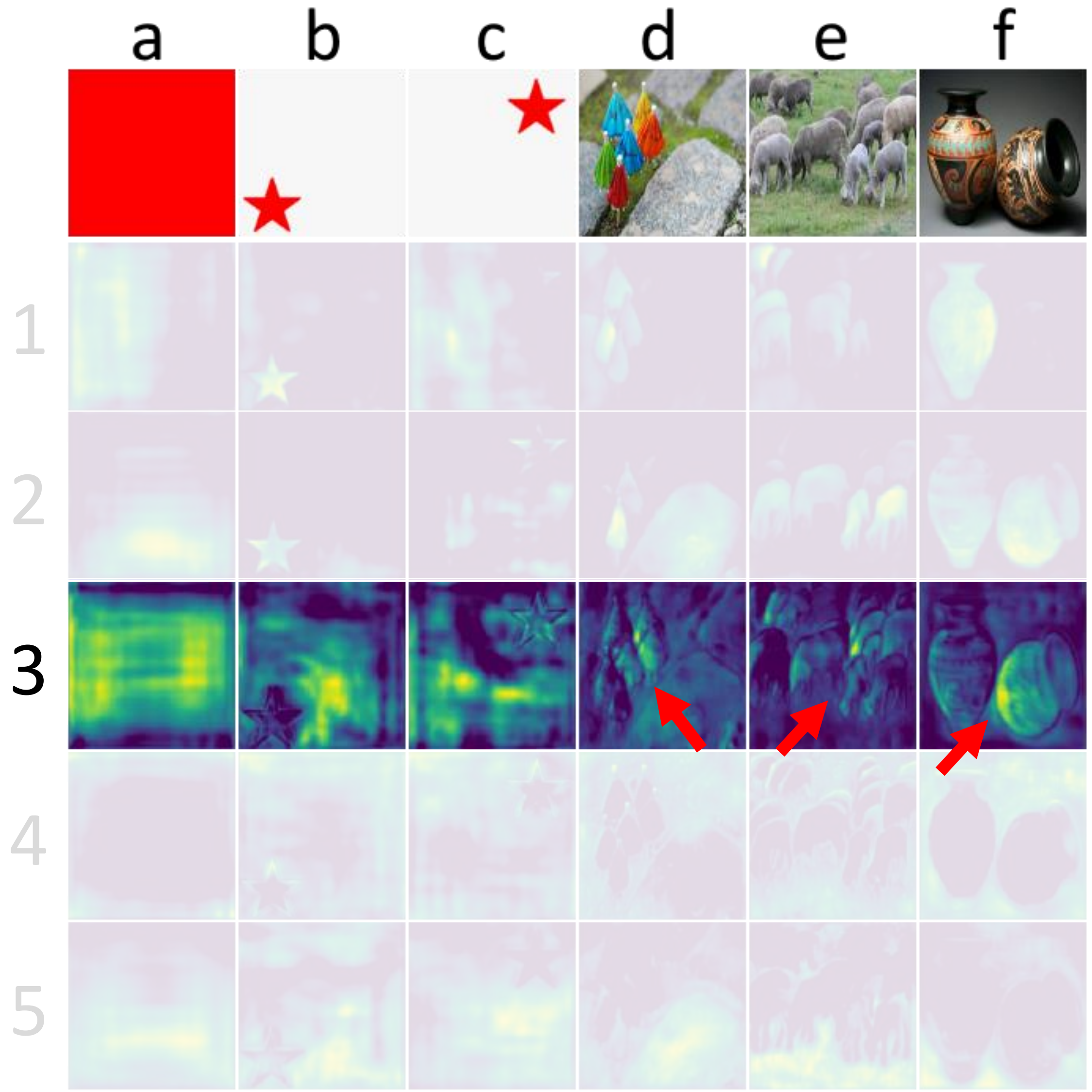
Prototype behavior

- *Supervision only comes from final mask loss*
- Spatially partition the image
- Segment background
- Detect instance contours
- Encode position-sensitive directional maps
- Most do a combination



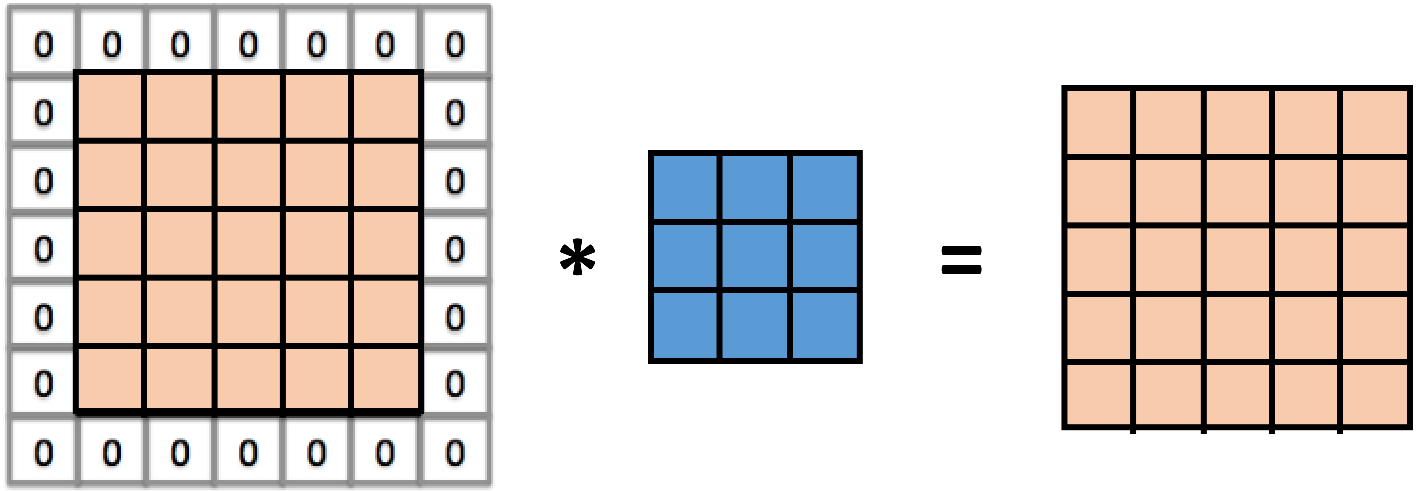
Prototype behavior

- *Supervision only comes from final mask loss*
- Spatially partition the image
- Segment background
- Detect instance contours
- Encode position-sensitive directional maps
- Most do a combination



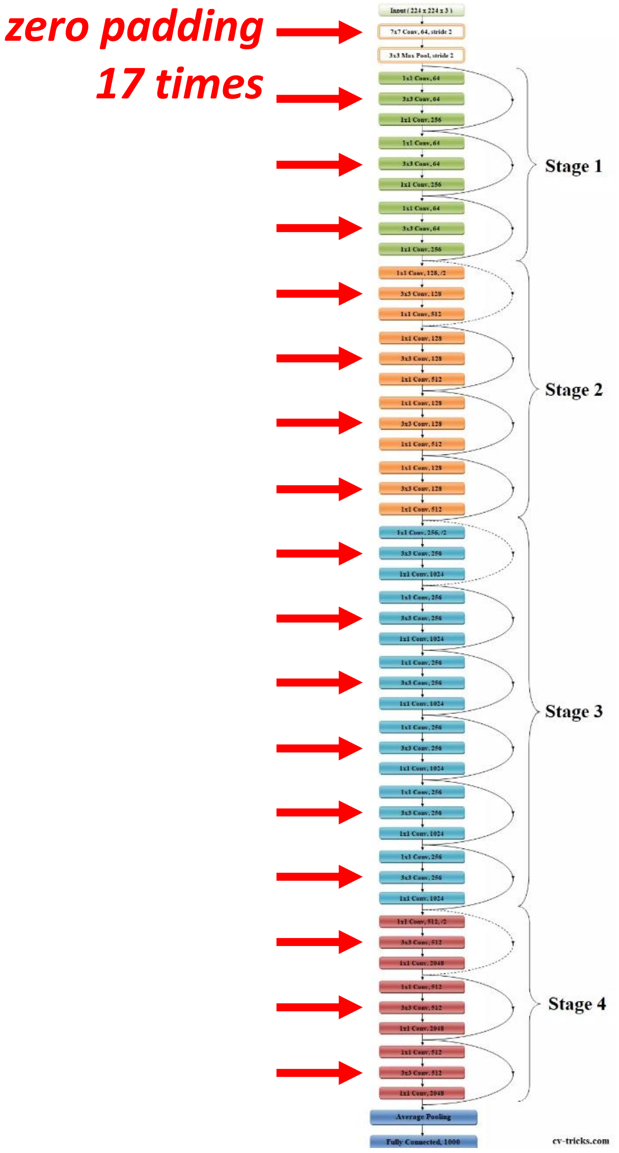
Zero-padding in ResNets

- Needed to keep input and output spatial resolution same



5x5 Input *3x3 Filter* *5x5 Output*

ResNet50



*Same finding in *How Much Position Information Do Convolutional Neural Networks Encode?* Islam et al. ICLR 2020

YOLACT++

1) Fast Mask-Rescoring (inspired by “Mask Scoring R-CNN”)
2) Deformable conv in backbone
3) Optimized prediction heads

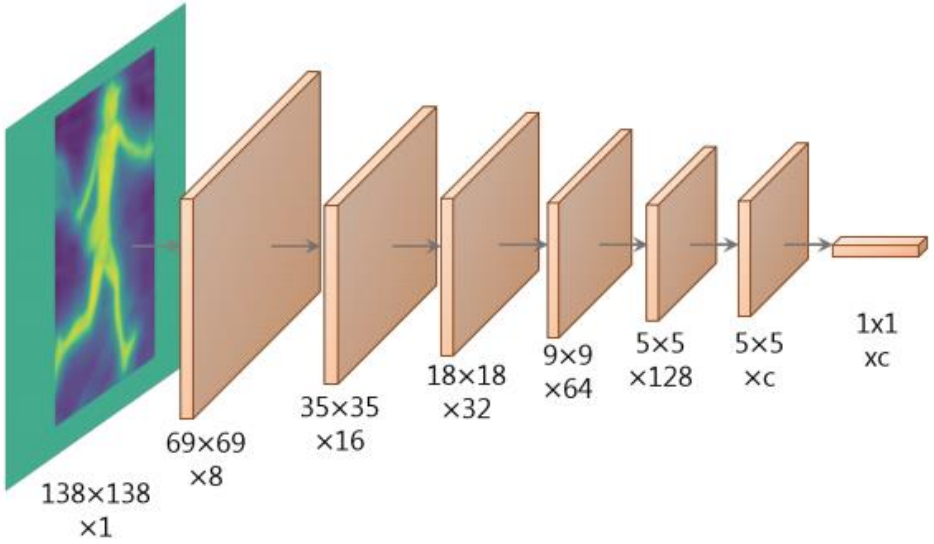


Fig. 6: **Fast Mask Re-scoring Network Architecture** Our mask scoring branch consists of 6 conv layers with ReLU non-linearity and 1 global pooling layer. Since there is no feature concatenation nor any fc layers, the speed overhead is only ~1 ms.

YOLACT++: Better real-time Instance Segmentation Bolya, Zhou et al. TPAMI 2020

Results

- First *real-time* **(> 30 fps)** instance segmentation algorithm with competitive results on the challenging MS COCO dataset

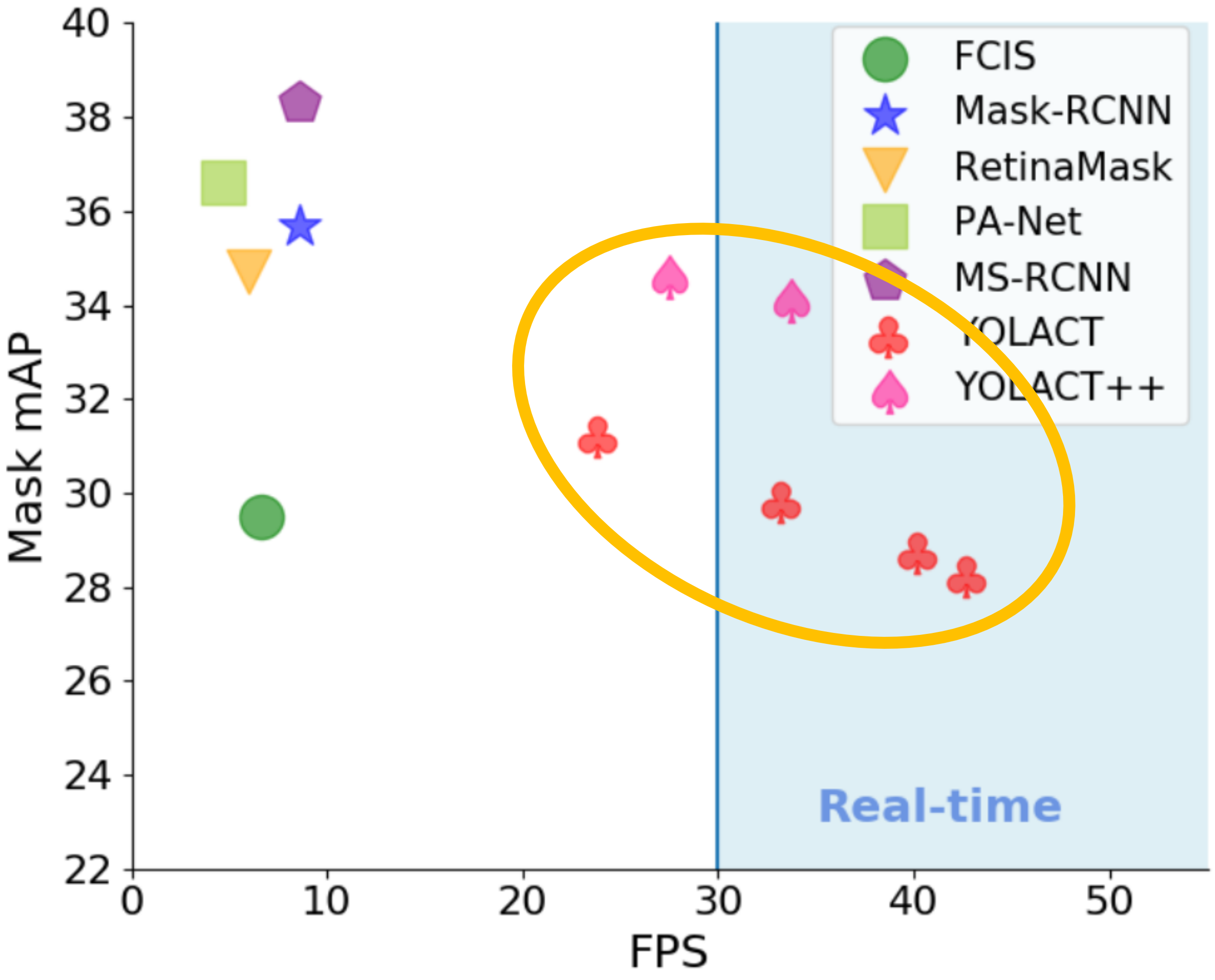


Figure 1: Speed-performance trade-off for various instance segmentation methods on COCO.

couch: 0.98
potted plant: 0.54
chair: 0.44
motorcycle: 1.00
motorcycle: 0.90
person: 0.95
frisbee: 0.96
person: 1.00
car: 0.93
person: 0.79

One-stage instance segmentation is an active area

BlendMask: Top-down meets bottom-up for instance segmentation, Chen et al., CVPR 2020

Centermask: Real-time anchor-free instance segmentation, Lee & Park, CVPR 2020

SOLO: Segmenting Objects by Locations, Wang et al., ECCV 2020

Conditional Convolutions for Instance Segmentation, Tian et al., ECCV 2020

SOLOv2: Dynamic and fast instance segmentation, Wang et al., NeurIPS 2020

…

Real-time instance segmentation on Edge devices

- YOLACT not fast enough for real-time inference on *edge devices* (Jetson Xavier AGX: ~6 FPS)

YolactEdge Main Idea: Exploit temporal redundancy in video

- Compute features on *keyframes* and re-use on *non-keyframes*

*Video frames from YouTube-VIS dataset

YolactEdge Main Idea: Exploit temporal redundancy in video

- Compute features on *keyframes* and re-use on *non-keyframes*
- Need both warped features and computed features
- Inspired by [Zhu CVPR'17; CVPR'18] but our focus is real-time instance segmentation on edge devices

*Video frames from YouTube-VIS dataset

Where is the computational bottleneck?

	C_1	C_2	C_3	C_4	C_5
# of convs	1	9	12	**69**	9
TFLOPS	0.1	0.7	1.0	**5.2**	0.8
%	1.5	8.7	13.2	**66.2**	10.3

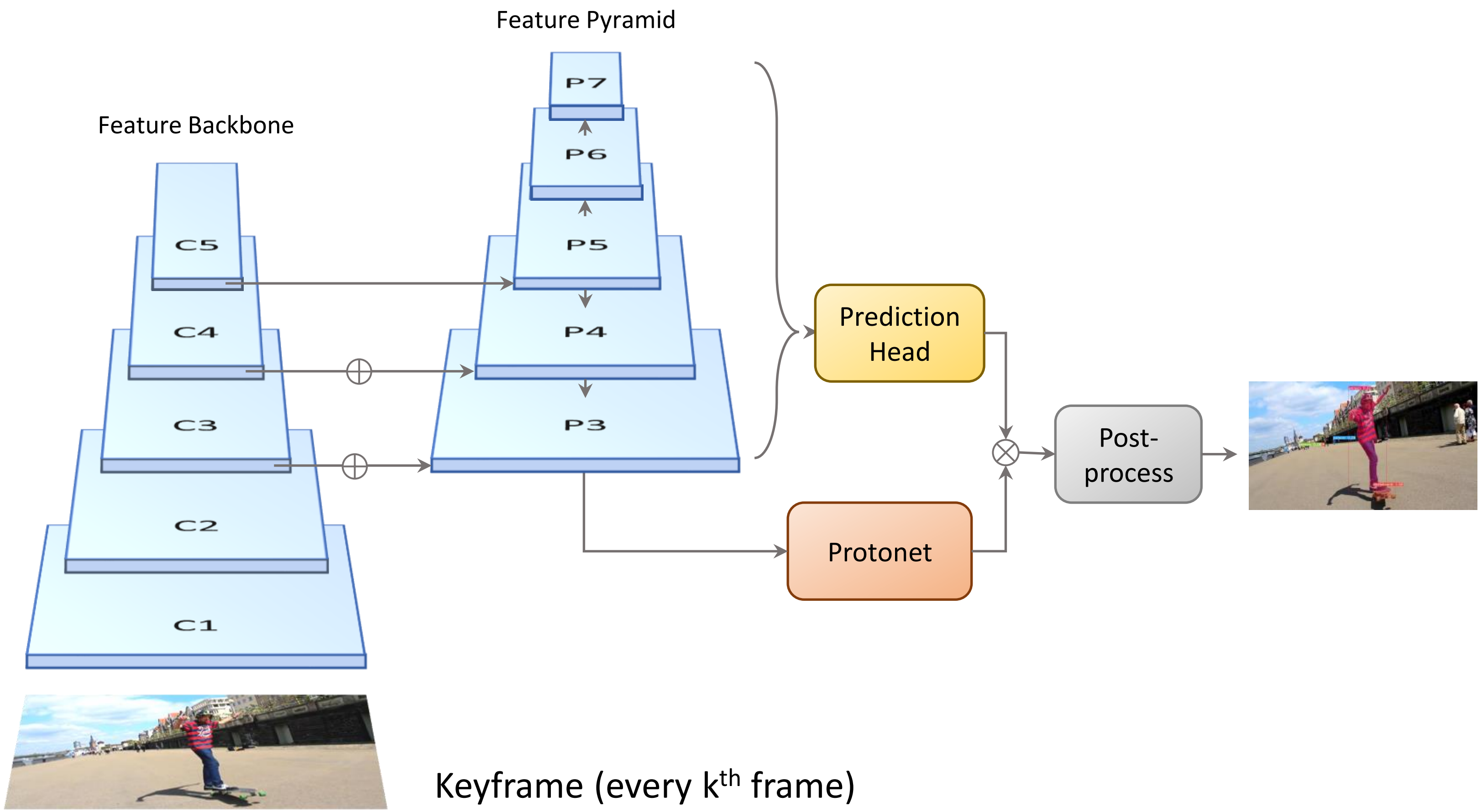
(a) **ResNet-101 Backbone**

Stage	%	Stage	%
Backbone	**54.7**	FPN	6.4
ProtoNet	7.8	Pred	10.6
Detect	6.6	Other	13.1

(b) **YOLACT**

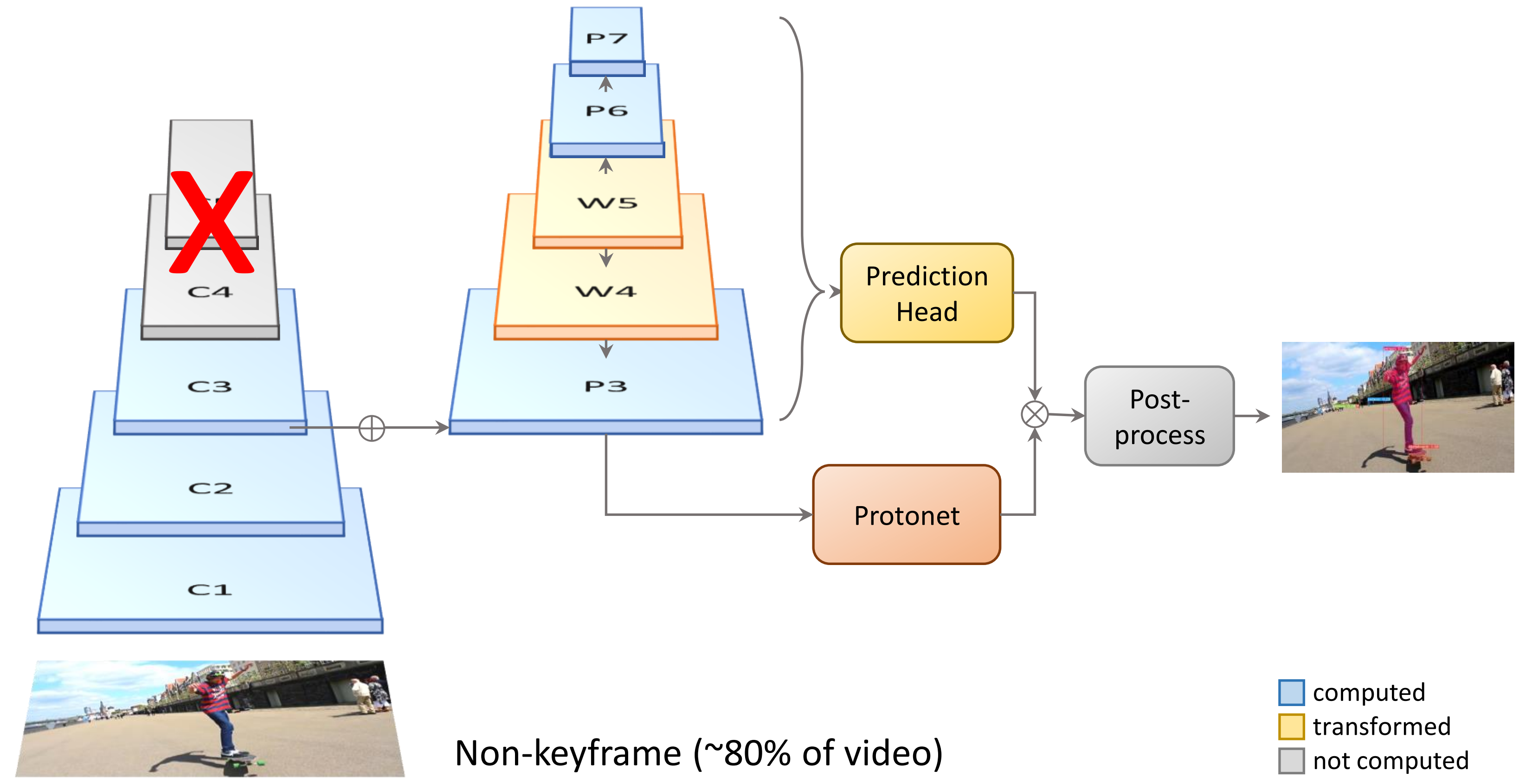
- Compute C1-C3 features on all frames
- Only compute C4 features on *keyframes,* warp to *non-keyframes*

YolactEdge architecture



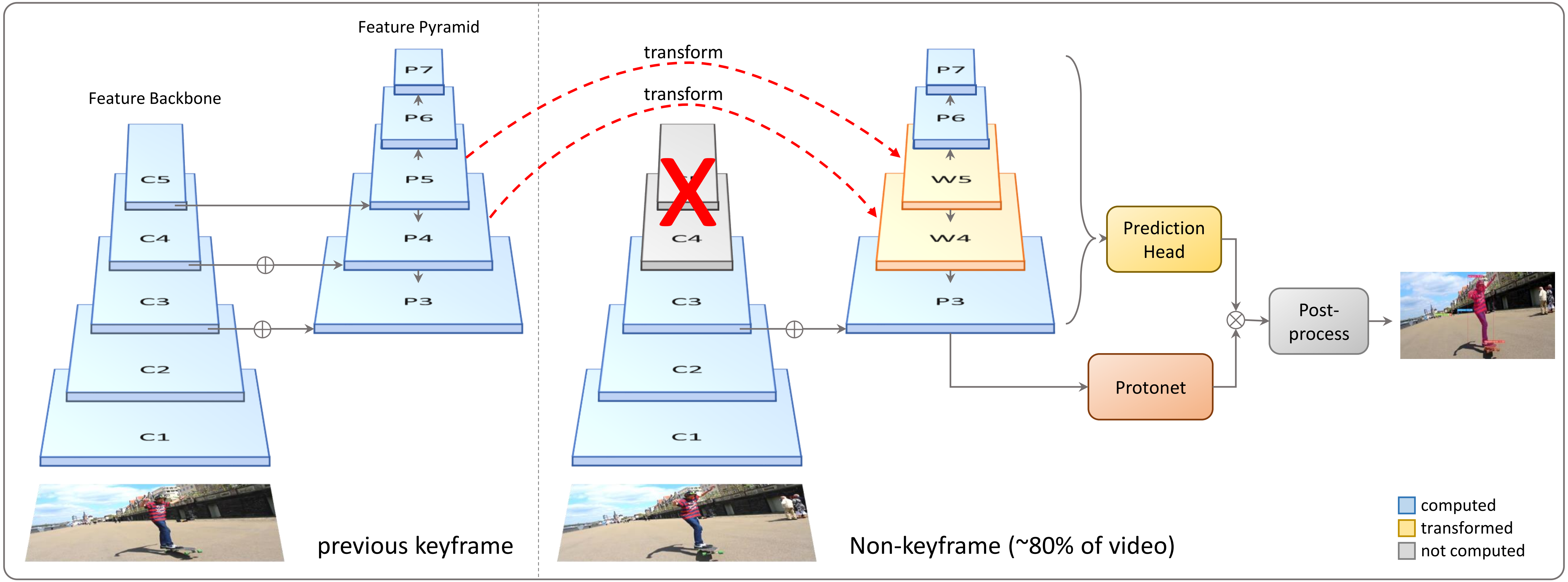
YolactEdge: Real-time Instance Segmentation on the Edge Liu, Rivera-Soto et al. ICRA 2021

YolactEdge architecture



YolactEdge: Real-time Instance Segmentation on the Edge Liu, Rivera-Soto et al. ICRA 2021

YolactEdge architecture



- *Partial* feature transform to preserve high-res details in the prototype masks

YolactEdge: Real-time Instance Segmentation on the Edge Liu, Rivera-Soto et al. ICRA 2021

Real-time on NVIDIA Jetson Xavier AGX

Method	Backbone	mask AP	box AP	AGX FPS	RTX FPS
YOLACT [1]	R-101-FPN	**47.3**	**48.9**	5.9	42.6
YolactEdge (w/o TRT)	R-101-FPN	46.9	47.8	9.5	61.2
YolactEdge (w/o video)	R-101-FPN	46.9	48.4	27.9	158.2
YolactEdge	R-101-FPN	46.2	47.1	**30.8**	**172.7**

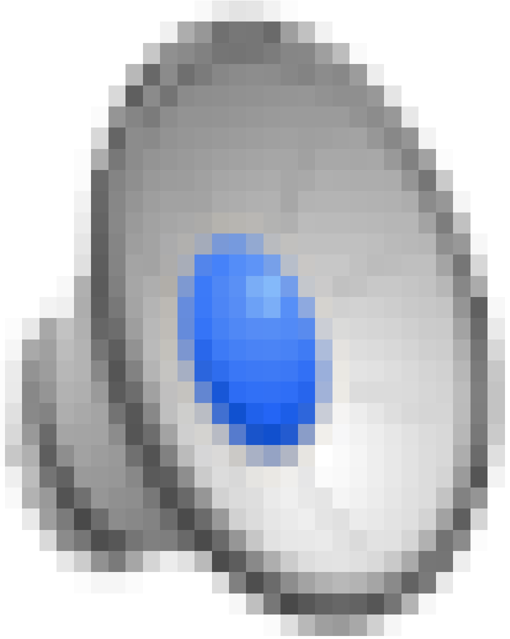
- Exploiting temporal redundancy leads to ~1.6x speedup

Real-time on NVIDIA Jetson Xavier AGX

Method	Backbone	mask AP	box AP	AGX FPS	RTX FPS
YOLACT [1]	R-101-FPN	**47.3**	**48.9**	5.9	42.6
YolactEdge (w/o TRT)	R-101-FPN	46.9	47.8	9.5	61.2
YolactEdge (w/o video)	R-101-FPN	46.9	48.4	27.9	158.2
YolactEdge	R-101-FPN	46.2	47.1	**30.8**	**172.7**

- Exploiting temporal redundancy leads to ~1.6x speedup
- TensorRT optimization enables *real-time (30 fps) speeds* on Xavier AGX

YolactEdge webcam demo



Conclusions

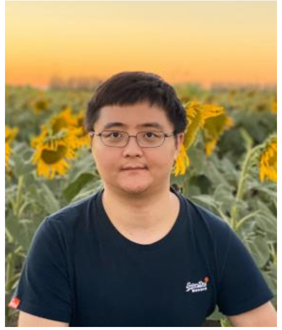
- YOLACT: First competitive real-time instance segmentation algorithm
- **Limitations:** small objects, crowds
- **Future work:** on-device learning

Daniel Bolya

Chong Zhou

Haotian Liu

Rafael A. Rivera Soto



Fanyi Xiao

OpenMMLab

github.com/dbolya/yolact
github.com/haotian-liu/yolact_edge