YOLO and ShuffleNet

Rachel Huang, Jonathan Pedoeem

July 3, 2018

UNCW REU 2018

Table of contents

Proposal

Proposal

Goals:

- Run a real-time object detection architecture on a website.
- Target speed: 10 FPS

If there is time:

- age/gender/race classification.
- App development

You Only Look Once (YOLO)

What is YOLO?

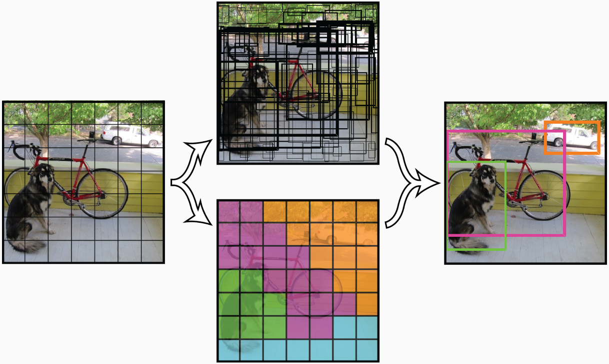


Figure 1: [Redmon et al.(2016)Redmon, Divvala, Girshick, and Farhadi]

Steps:

- Divide image into *S x S grid.*
- *Each cell predicts B bounding boxes with confidence scores.*
- *Each cell predicts C conditional class probability.*

Equations

Probability Equation:

$$Pr(Class_i|Object) * Pr(Object) * IOU_{pred}^{truth} = Pr(Class_i) * IOU_{pred}^{truth}. \quad (1)$$

$$\lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$
$$+ \lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$
$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$
$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$
$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \quad (3)$$

Figure 2: Loss function of YOLO
[Redmon et al.(2016)Redmon, Divvala, Girshick, and Farhadi]

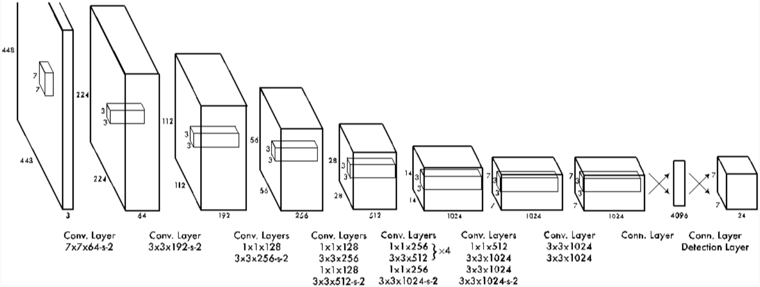
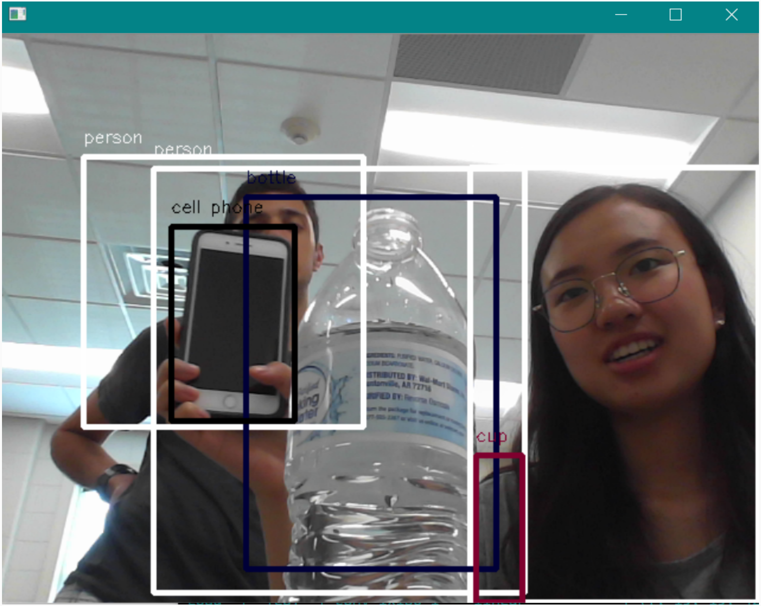


Figure 3: [Redmon et al.(2016)Redmon, Divvala, Girshick, and Farhadi]

- 24 convolutional layers and 2 fully connected layers
- Pretrained the layers on ImageNet

Preview



YOLOv3

Improvements:

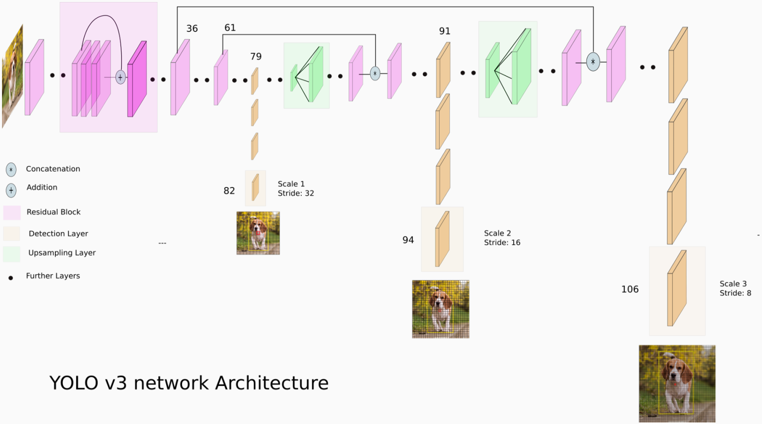
- Anchor boxes
- Multiple predictions (13x13, 26x26, 52x52)
- Object confidence
- Non-maximal suppression
- Routing
- Skip connections

Table 1: Comparison of YOLO Versions

Version	Layers	FLOPS (Bn)	FPS	mAP
YOLOv1	26	not reported	45	63.4 (VOC)
YOLOv1-Tiny	9	not reported	155	52.7 (VOC)
YOLOv2	32	62.94	40	48.1
YOLOv2-Tiny	16	5.41	244	23.7
YOLOv3	106	140.69	20	57.9
YOLOv3-Tiny	24	5.56	220	33.1

Version 1 is trained and tested on Pascal VOC, while all other versions are trained and tested on MS COCO

YOLOv3- Architecture



YOLO v3 network Architecture

ShuffleNet

ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices

- Reduce Flops to the Millions instead of Billions. 10-150 MFLOPS
- Authors claim *"ShuffleNet achieves ~13× actual speedup over AlexNet while maintaining comparable accuracy."*
- No mention on how quick
- Only 8 layers, now that's a low number
- *"In tiny networks, expensive pointwise convolutions result in limited number of channels to meet the complexity constraint, which might significantly damage the accuracy."*

ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices

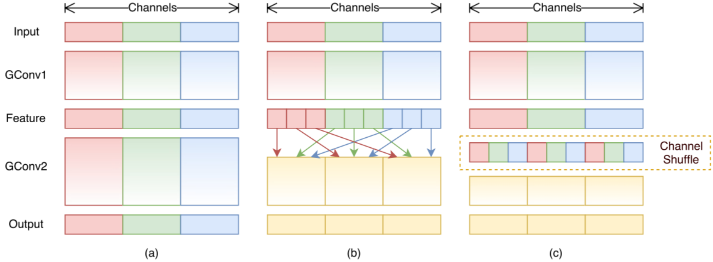


Figure 1: Channel shuffle with two stacked group convolutions. GConv stands for group convolution. a) two stacked convolution layers with the same number of groups. Each output channel only relates to the input channels within the group. No cross talk; b) input and output channels are fully related when GConv2 takes data from different groups after GConv1; c) an equivalent implementation to b) using channel shuffle.

Figure 4: [Zhang et al.(2017)Zhang, Zhou, Lin, and Sun]

ShuffleNet cont.

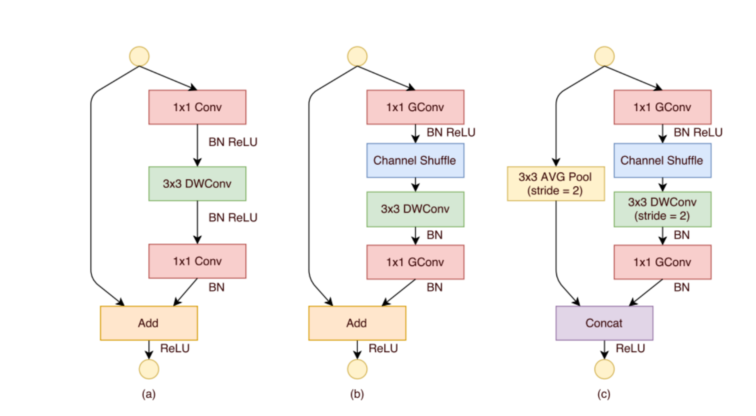


Figure 5: [Zhang et al.(2017)Zhang, Zhou, Lin, and Sun]

Conclusion

Conclusion

Next steps:

- Convert tiny-YOLOv2 to Javascript to run on a website.
- Implement tiny-YOLOv3.
- Get algorithm to run at 10 FPS.
  - Standard Neural Network Compression Techniques
  - Inspiration from ShuffleNet, SqueezeNet, and MobileNet

Live Demo

Questions?

References

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi.
You only look once: Unified, real-time object detection.
In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun.
Shufflenet: An extremely efficient convolutional neural network for mobile devices.
arXiv preprint arXiv:1707.01083, 2017.