
بسم الله الرحمن الرحيم

مبانی بینایی کامپیوتر

مدرس: محمدرضا محمدی

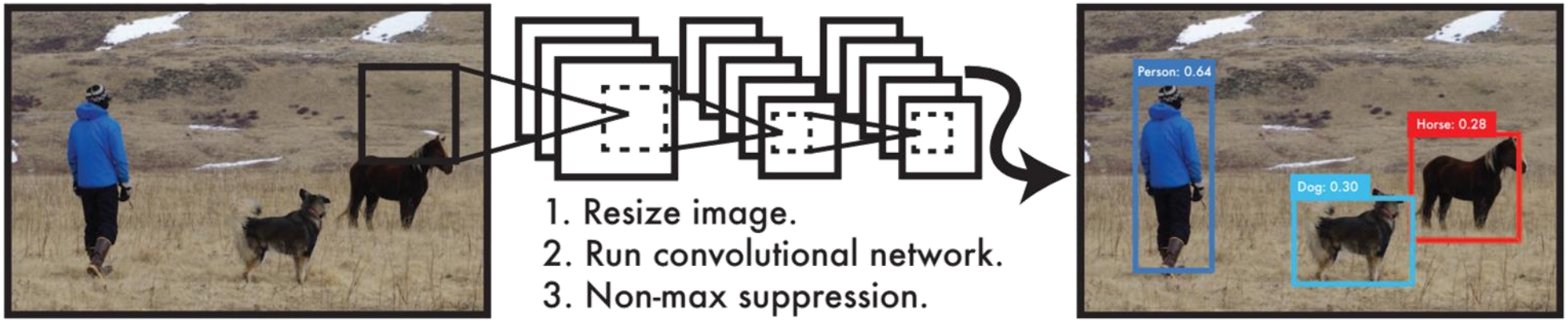
بهار ۱۴۰۳

تشخیص اشیاء

Object Detection

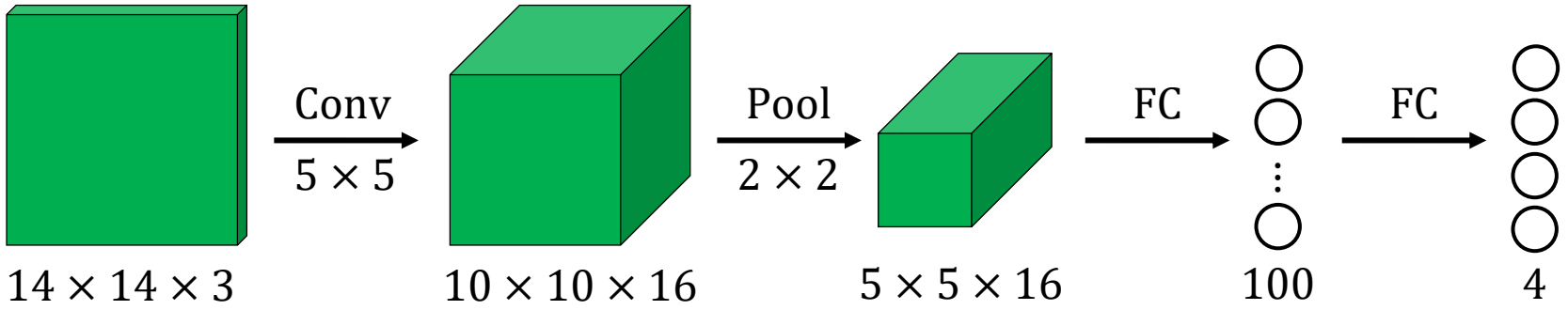
تشخیص اشیاء یک مرحله‌ای

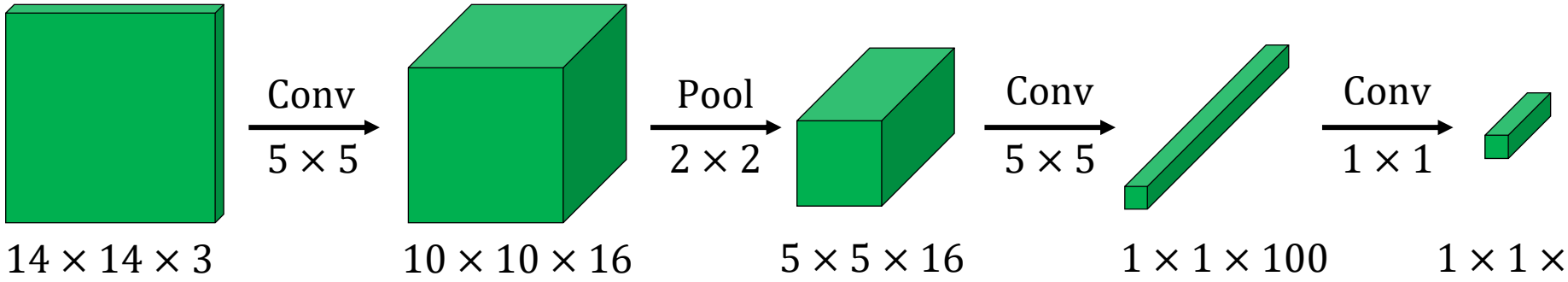
- برخی از الگوریتم‌های جدیدتر مانند YOLO، SSD و RetinaNet بدون بخش مستقیمی برای تولید ناحیه‌های پیشنهادی توسعه یافته‌اند
- این الگوریتم‌ها مبتنی بر پنجره لغزان هستند
  - به نوعی، توسعه روش دسته‌بندی و مکان‌یابی هستند



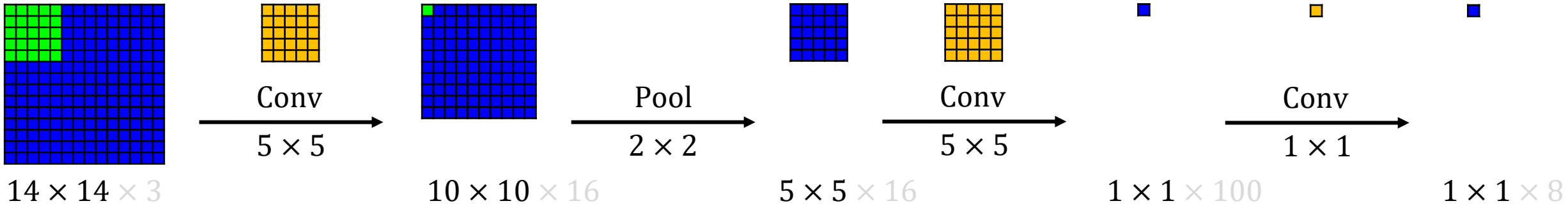
Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

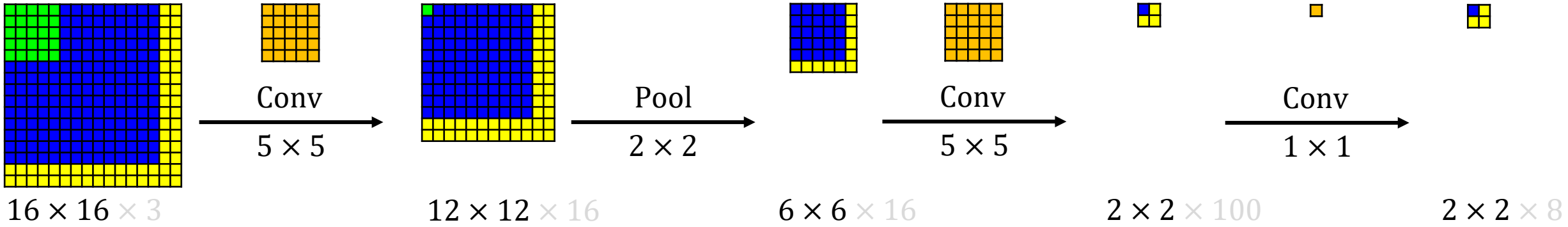
تبدیل لایه‌های FC به Conv



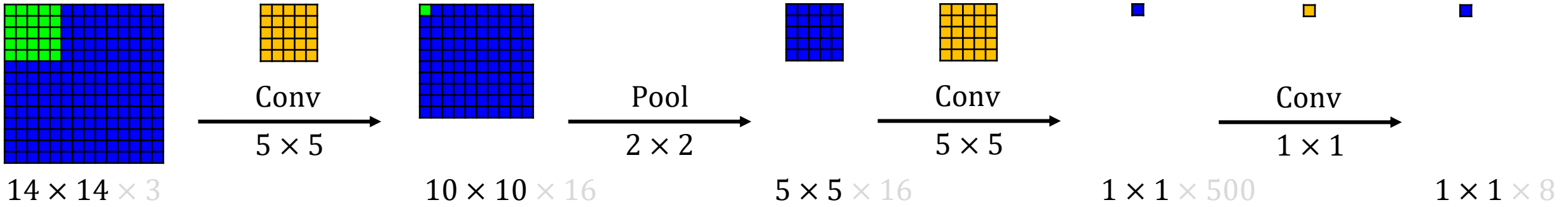


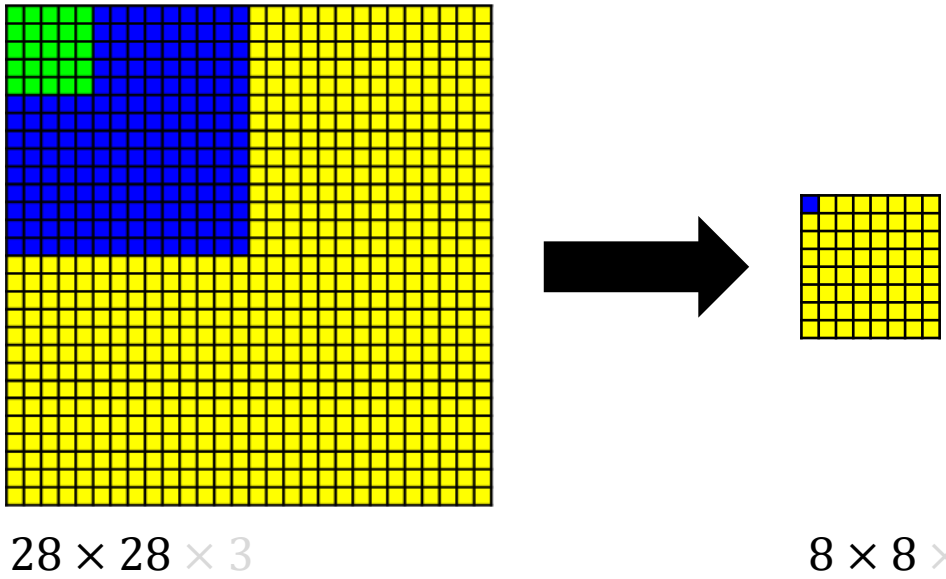
پیاده‌سازی کانولوشنی پنجره لغزان

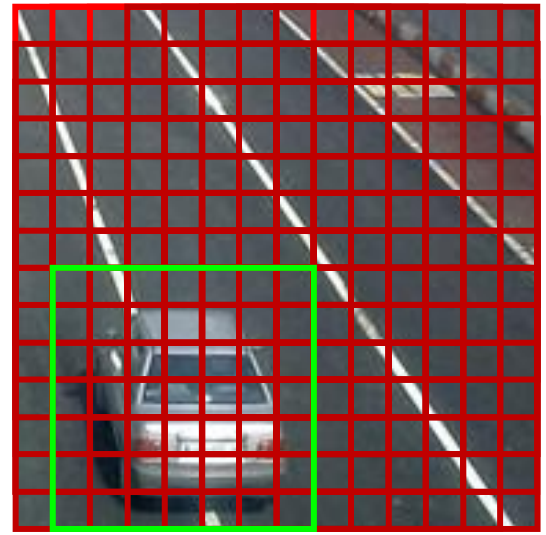




پیاده‌سازی کانولوشنی پنجره لغزان





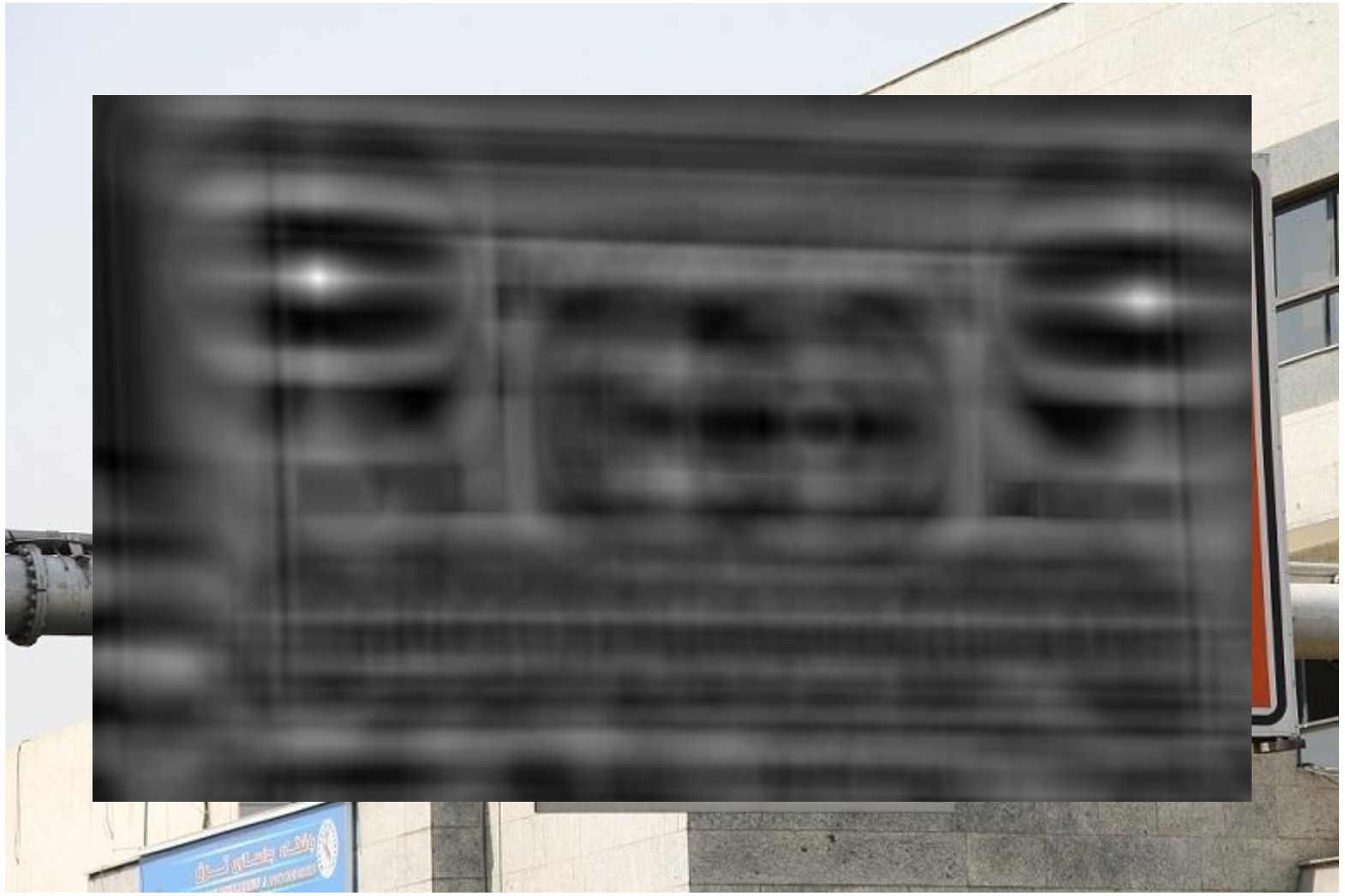


$$\begin{bmatrix} p_c \\ b_x \\ b_y \\ b_w \\ b_h \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

تطبیق کلیشه

- روش تطبیق کلیشه ساده‌ترین روش برای یافتن یک شیء در تصویر است
- در این روش، یک کلیشه (template) از شیء مورد نظر ساخته شده و مشابهت (یا فاصله) هر ناحیه از تصویر نسبت به آن سنجیده می‌شود

تطبیق کلیشه



تطبیق کلیشه


MARIO
001200
×00
WORLD
1-1
TIME
348


MARIO
001200
×00
WORLD
1-1
TIME
348

تطبیق کلیشه

- method
  - TM_SQDIFF

$$R(x,y) = \sum_{x',y'} (T(x',y') - I(x+x',y+y'))^2$$

  - TM_SQDIFF_NORMED

$$R(x,y) = \frac{\sum_{x',y'} (T(x',y') - I(x+x',y+y'))^2}{\sqrt{\sum_{x',y'} T(x',y')^2 \cdot \sum_{x',y'} I(x+x',y+y')^2}}$$

  - TM_CCORR

$$R(x,y) = \sum_{x',y'} (T(x',y') \cdot I(x+x',y+y'))$$

  - TM_CCORR_NORMED

$$R(x,y) = \frac{\sum_{x',y'} (T(x',y') \cdot I(x+x',y+y'))}{\sqrt{\sum_{x',y'} T(x',y')^2 \cdot \sum_{x',y'} I(x+x',y+y')^2}}$$

  - TM_CCOEFF

$$R(x,y) = \sum_{x',y'} (T'(x',y') \cdot I'(x+x',y+y'))$$

  - TM_CCOEFF_NORMED

$$R(x,y) = \frac{\sum_{x',y'} (T'(x',y') \cdot I'(x+x',y+y'))}{\sqrt{\sum_{x',y'} T'(x',y')^2 \cdot \sum_{x',y'} I'(x+x',y+y')^2}}$$

$$T'(x',y') = T(x',y') - 1/(w \cdot h) \cdot \sum_{x'',y''} T(x'',y'')$$

$$I'(x+x',y+y') = I(x+x',y+y') - 1/(w \cdot h) \cdot \sum_{x'',y''} I(x+x'',y+y'')$$

خواندن پلاک

- برای خواندن کاراکترهای پلاک، می‌توان تطبیق کلیشه هر کاراکتر با تصویر را محاسبه کرد

خواندن پلاک

- برای خواندن کاراکترهای پلاک، می‌توان تطبیق کلیشه هر کاراکتر با تصویر را محاسبه کرد

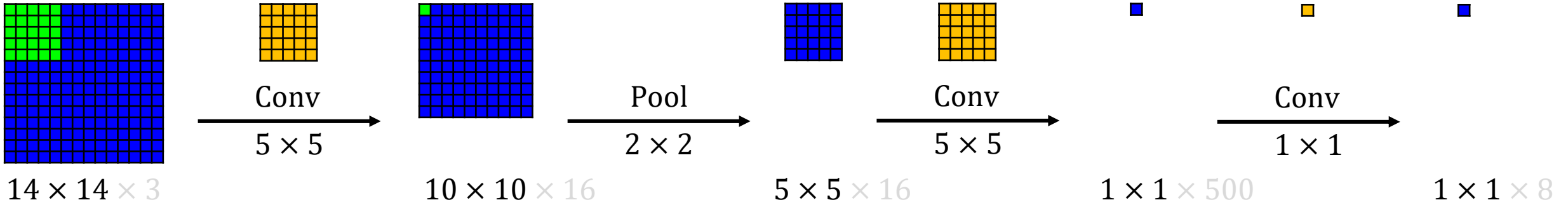
خواندن پلاک

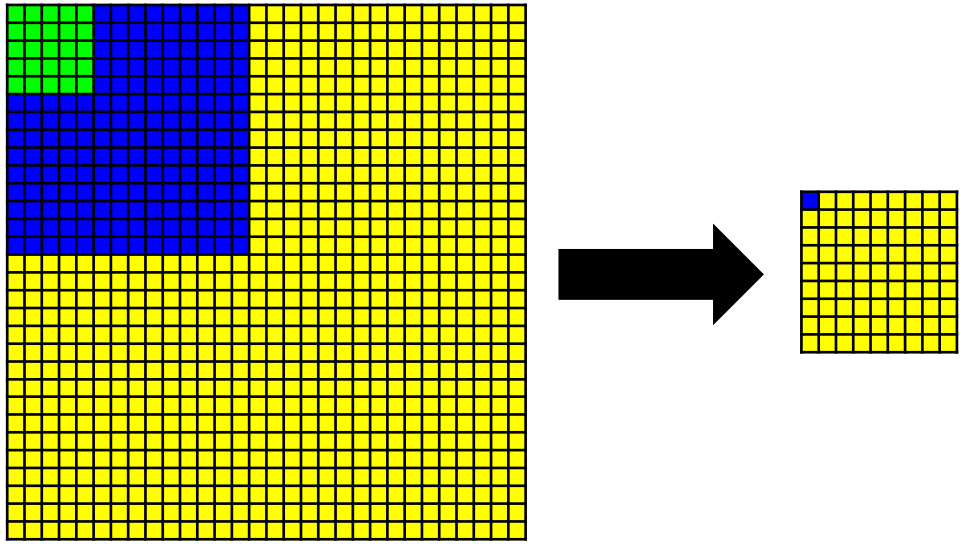
- برای خواندن کاراکترهای پلاک، می‌توان تطبیق کلیشه هر کاراکتر با تصویر را محاسبه کرد
- سپس، مکان‌هایی که نتیجه تطبیق کلیشه برای آنها بیش از حدی باشد را به عنوان مکان آن کاراکتر در نظر می‌گیریم

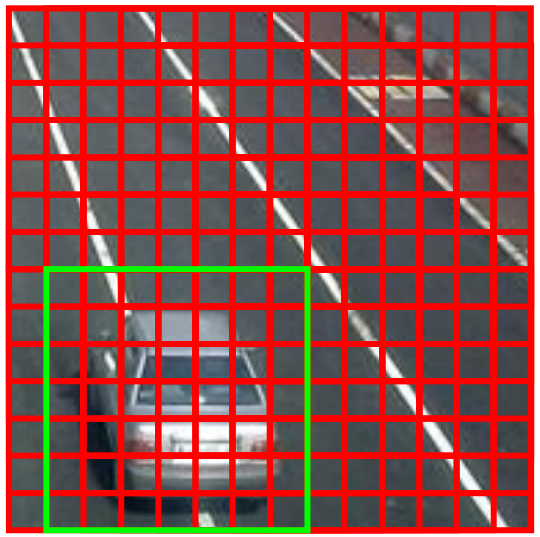
خواندن پلاک

- برای خواندن کاراکترهای پلاک، می‌توان تطبیق کلیشه هر کاراکتر با تصویر را محاسبه کرد
- سپس، مکان‌هایی که نتیجه تطبیق کلیشه برای آنها بیش از حدی باشد را به عنوان مکان آن کاراکتر در نظر می‌گیریم
- برای تشخیص کاراکترهای با ابعاد مختلف می‌توان از کلیشه‌های با ابعاد مختلف استفاده نمود

پیاده‌سازی کانولوشنی پنجره لغزان



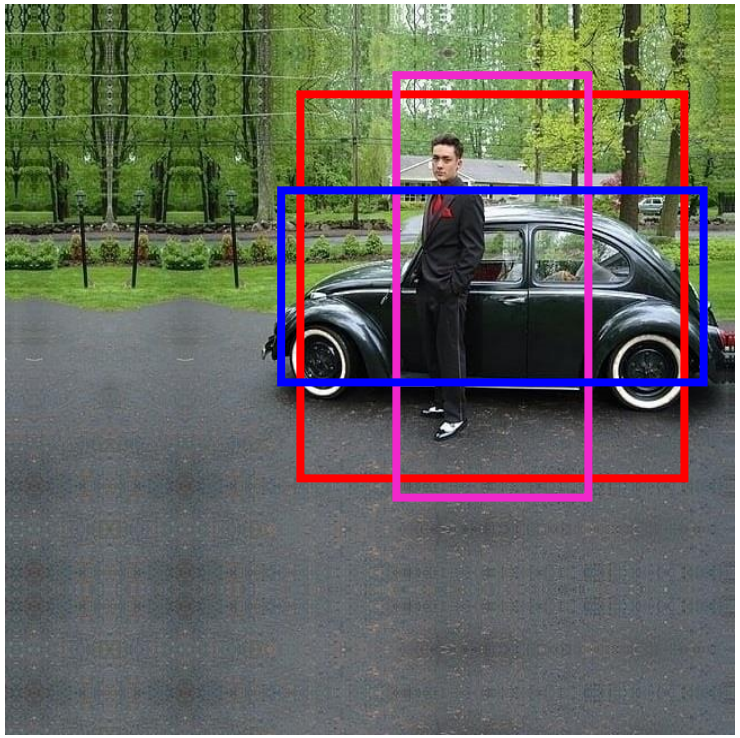




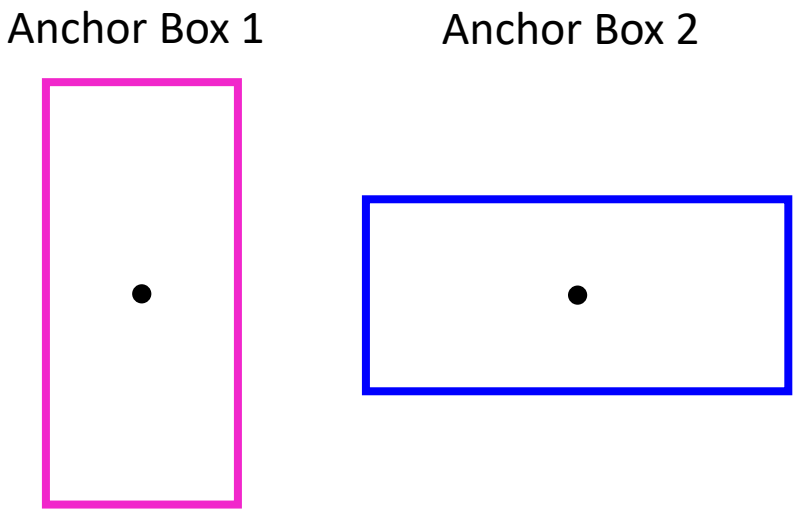
$$\begin{bmatrix} p_c \\ b_x \\ b_y \\ b_w \\ b_h \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

Anchor Boxes

- در هر ناحیه تنها یک شیء قابل تشخیص است
- برای اضافه کردن امکان تشخیص چند شیء، می‌توان در هر ناحیه چند خروجی قرار داد

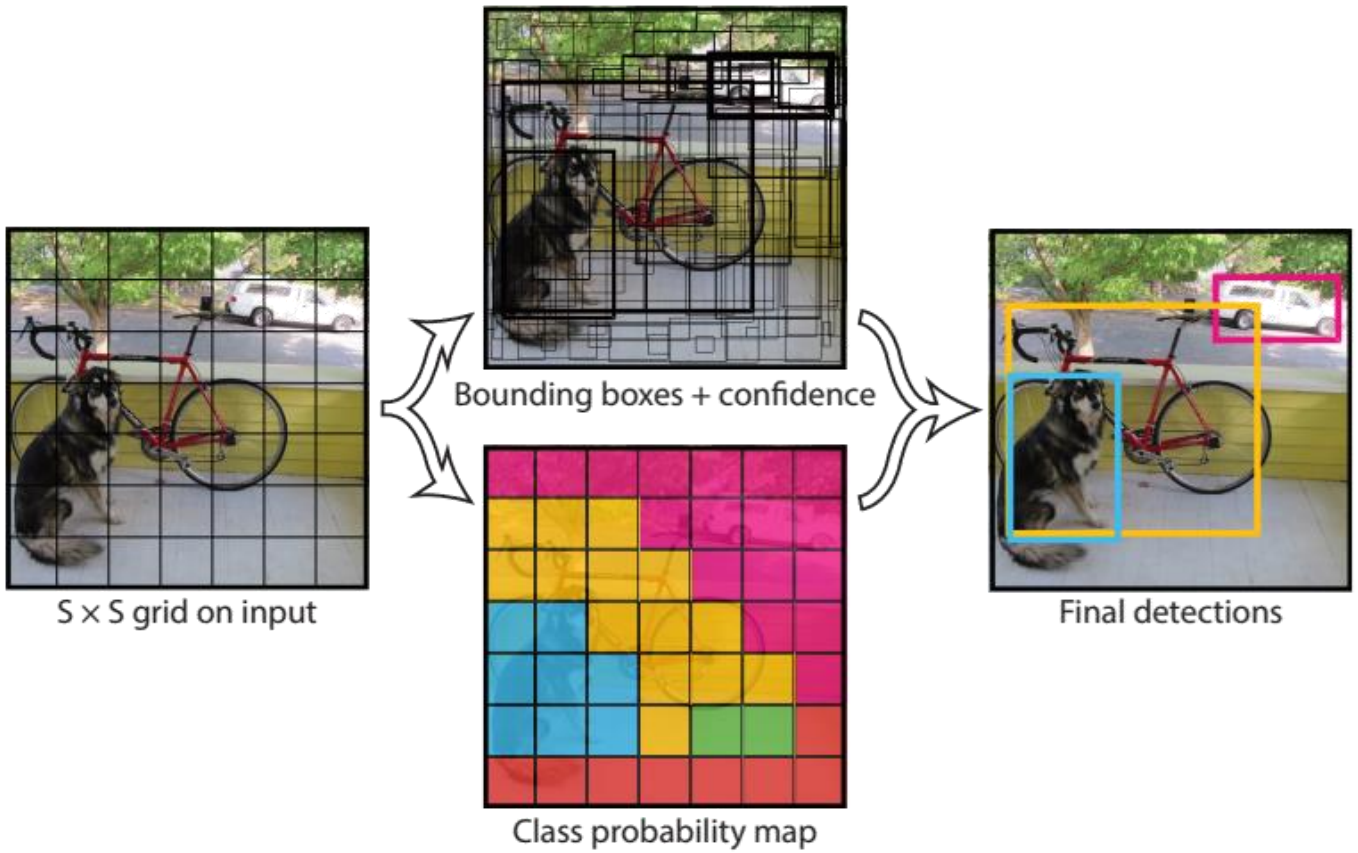


$$\begin{bmatrix} p_c \\ b_x \\ b_y \\ b_w \\ b_h \\ c_1 \\ c_2 \\ c_2 \\ p_c \\ b_x \\ b_y \\ b_w \\ b_h \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$



YOLO: You Only Look Once

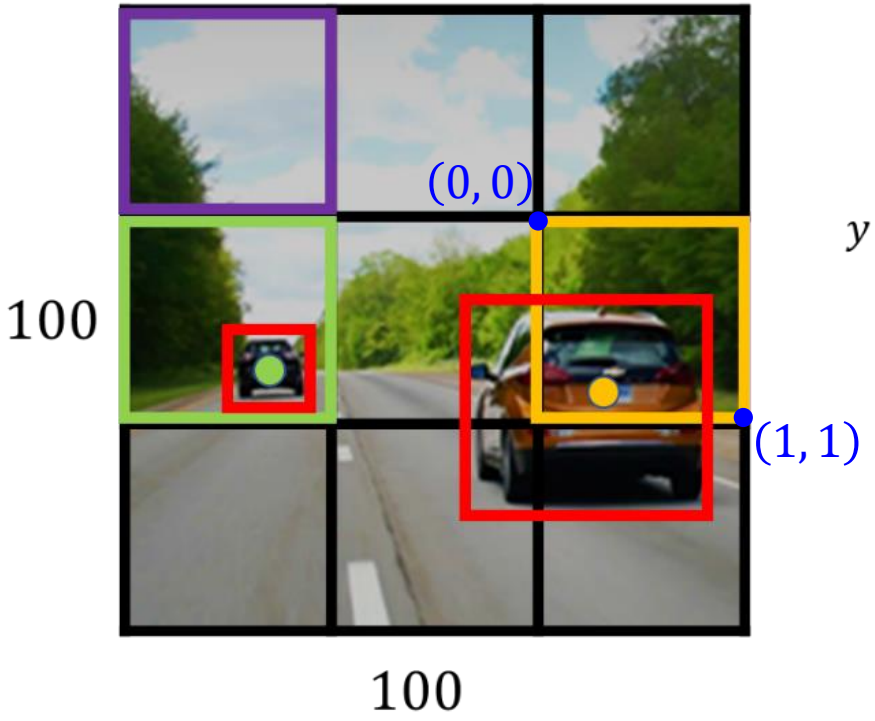
- در روش YOLO، تصویر ورودی به تعدادی ناحیه کوچک تقسیم می‌شود و برای هر ناحیه یک دسته‌بند و یک تابع رگرسیون طراحی می‌شود



Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." *Proceedings of the IEEE conference on computer vision and pattern recognition*. (2016).

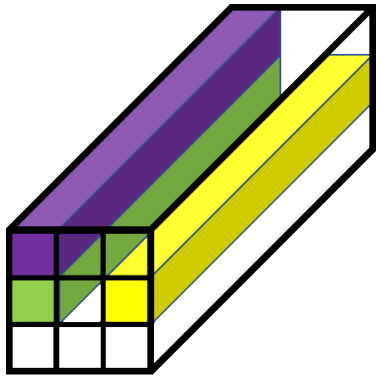
YOLO

Labels for training for each grid cell:



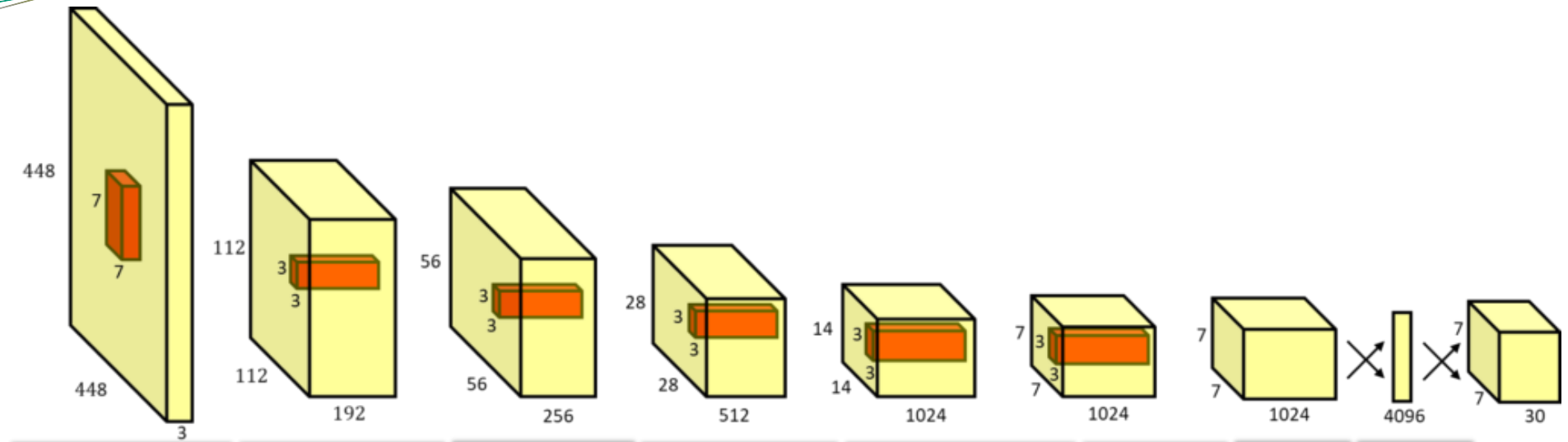
$$y = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} \quad \begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix} \quad \begin{bmatrix} 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 0 \\ 1 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

→ bounding box

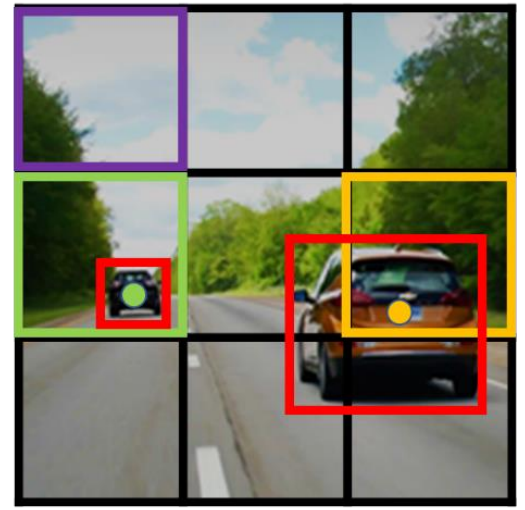


$3 \times 3 \times 8$

YOLO



Conv. Layer $7 \times 7 \times 64 - s - 2$ Maxpool Layer $2 \times 2 - s - 2$	Conv. Layer $3 \times 3 \times 192$ Maxpool Layer $2 \times 2 - s - 2$	Conv. Layers $1 \times 1 \times 128$ $3 \times 3 \times 256$ $1 \times 1 \times 256$ $3 \times 3 \times 512$ Maxpool Layer $2 \times 2 - s - 2$	Conv. Layers $\left.\begin{matrix}1 \times 1 \times 256\\3 \times 3 \times 512\end{matrix}\right\} \times 4$ $1 \times 1 \times 512$ $3 \times 3 \times 1024$ Maxpool Layer $2 \times 2 - s - 2$	Conv. Layers $\left.\begin{matrix}1 \times 1 \times 512\\3 \times 3 \times 1024\end{matrix}\right\} \times 4$ $3 \times 3 \times 1024$ $3 \times 3 \times 1024 - s - 2$	Conv. Layers $3 \times 3 \times 1024$ $3 \times 3 \times 1024$	Conn Layer	Conn Layer



100

100

YOLO

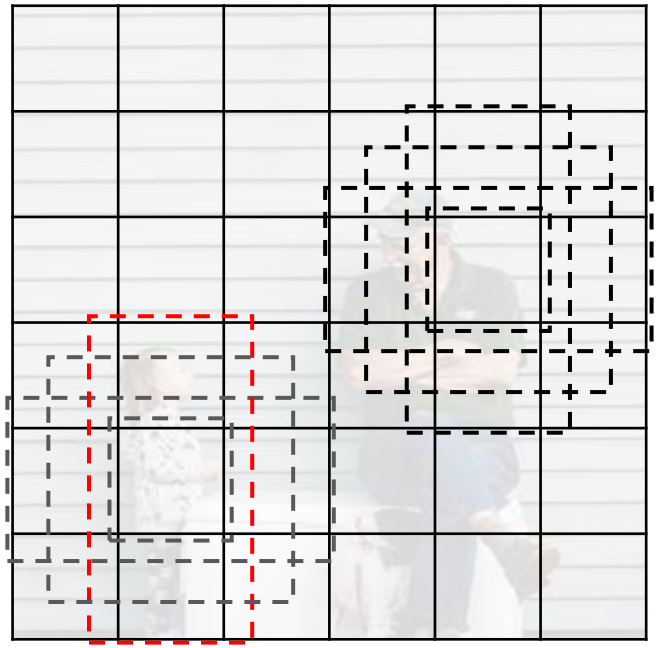
Real-Time Detectors	Train	mAP	FPS
100Hz DPM [31]	2007	16.0	100
30Hz DPM [31]	2007	26.1	30
Fast YOLO	2007+2012	52.7	**155**
YOLO	2007+2012	**63.4**	45
Less Than Real-Time			
Fastest DPM [38]	2007	30.4	15
R-CNN Minus R [20]	2007	53.5	6
Fast R-CNN [14]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[28]	2007+2012	73.2	7
Faster R-CNN ZF [28]	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21

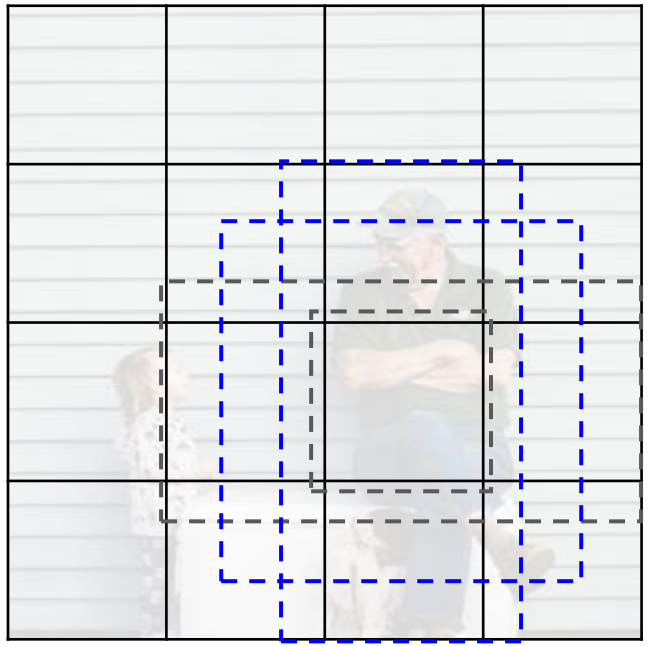
SSD: Single Shot MultiBox Detector

- روش YOLO یک روش آموزش end-to-end شبکه است که نیازی به تولید ناحیه پیشنهادی و تغییر ابعاد آنها ندارد
- سرعت روش YOLO از روش‌هایی که از بخش تولید ناحیه‌های پیشنهادی استفاده می‌کنند بهتر است اما دقت پائین‌تری دارد
- در مقاله SSD بهبودهایی داده شده است که در ضمن افزایش سرعت، دقت نیز افزایش یافته است (به خصوص برای اشیاء کوچک)
- مهمترین نوآوری SSD آن است که برای تشخیص اشیاء و محل آنها از چند لایه استفاده کرده است تا اشیاء با ابعاد مختلف قابل تشخیص باشند

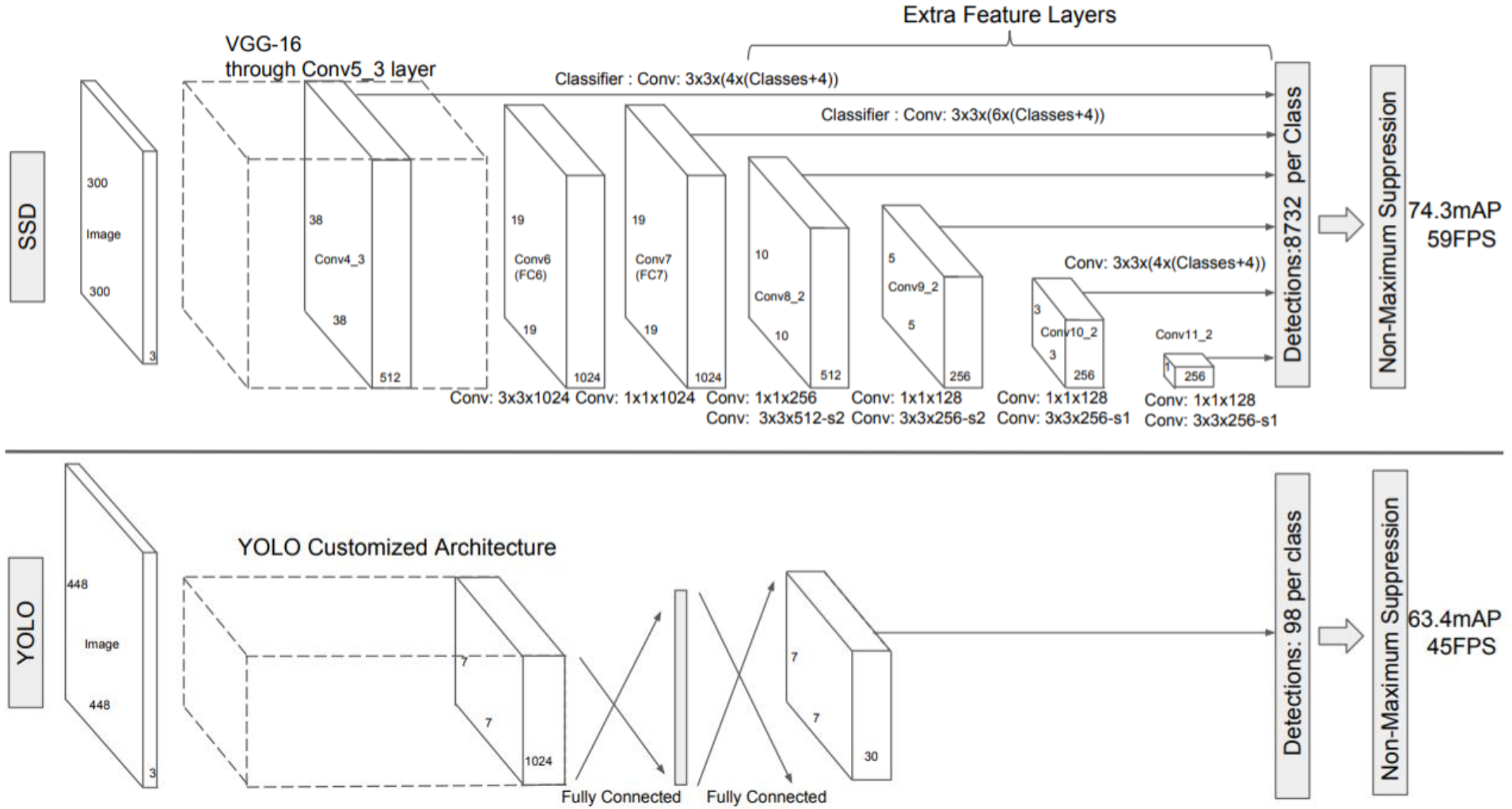
Liu, W., et al. "Ssd: Single shot multibox detector“, *European conference on computer vision*, (2016).

SSD





SSD vs YOLO



http://cvmldlcnns.blogspot.com/2017/05/ssd-single-shot-multibox-detector.html

RetinaNet

- یکی از مشکلات آموزش در شبکه‌های طراحی شده برای تشخیص اشیاء عدم توازن شدید میان نمونه‌های کلاس‌ها است
- به طور خاص، تعداد ناحیه‌هایی که هیچکدام از اشیاء مورد نظر در آن قرار ندارند به مراتب بیش از ناحیه‌های مربوط به کلاس‌های دیگر است
- ایده اصلی در مقاله RetinaNet پیشنهاد تابع هزینه‌ای است که بهینه‌سازی آن با استفاده از داده‌های نامتوازن منجر به عملکرد مناسب‌تری شود

Lin, T., et al., "Focal loss for dense object detection“, *Proceedings of the IEEE international conference on computer vision*, (2017).

Focal Loss

- تابع هزینه متداول در شبکه‌های عصبی cross entropy است

$$CE(p,y) = -\sum y_i \log(p_i)$$

- حالت دو کلاسه:

$$CE(p,y) = \begin{cases} -\log(p) & if \;\; y = 1 \\ -\log(1-p) & otherwise \end{cases}$$

$$p_t = \begin{cases} p & if \;\; y = 1 \\ 1-p & otherwise \end{cases}$$

$$CE(p,y) = CE(p_t) = -\log(p_t)$$

Focal Loss

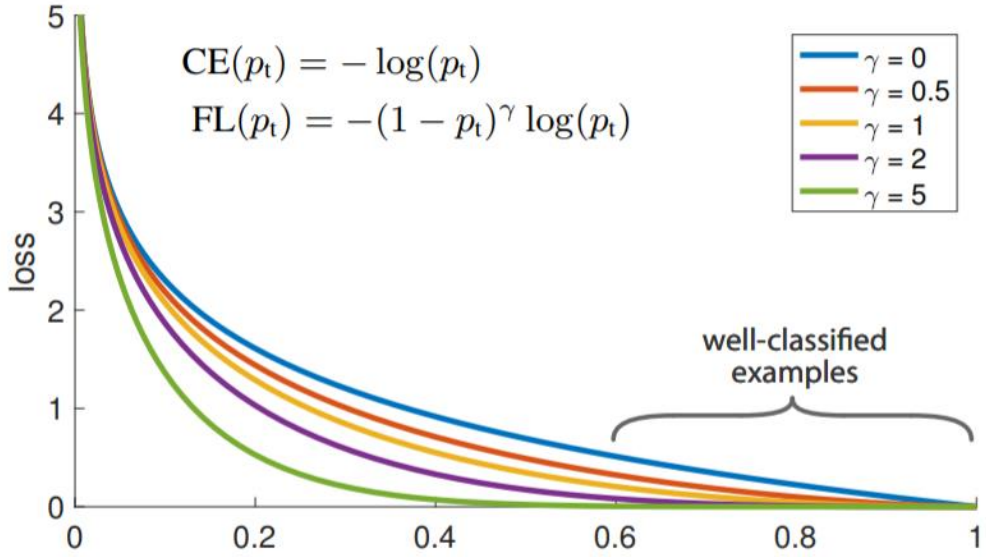
- تابع هزینه cross entropy تلاش می‌کند تا تمام نمونه‌ها را با احتمال کامل درست بگوید
- به عبارت دیگر، دسته‌بندی با احتمال بالا کفایت نمی‌کند و این مقادیر هزینه کوچک زمانیکه برای تعداد بسیار زیادی نمونه با یکدیگر جمع می‌شوند عدد قابل توجهی می‌شود
- در تابع هزینه focal، مقدار ضرر برای داده‌هایی که به خوبی شناسایی شده‌اند کاهش می‌یابد

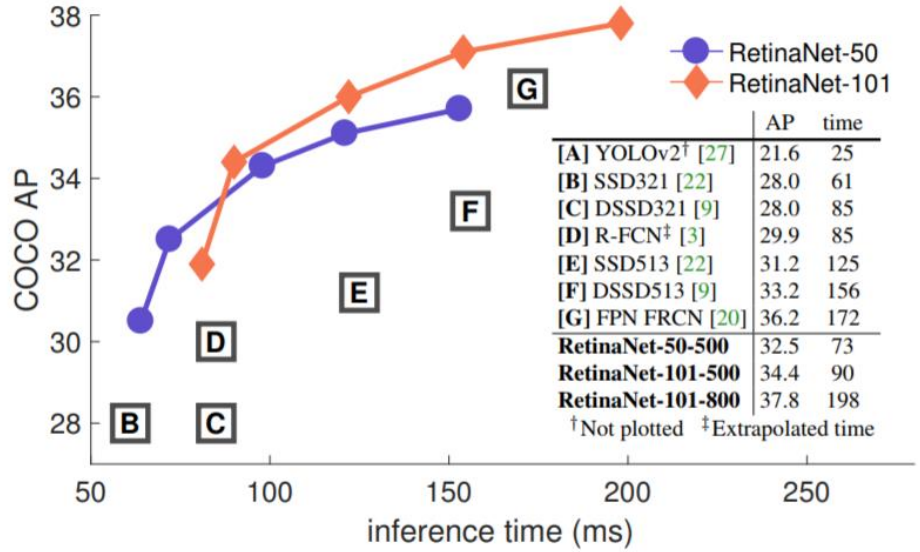
$$FL(p_t) = -(1-p_t)^{\gamma}\log(p_t)$$

$$p_t = \begin{cases} p & if \;\; y = 1 \\ 1-p & otherwise \end{cases}$$

$$CE(p,y) = CE(p_t) = -\log(p_t)$$

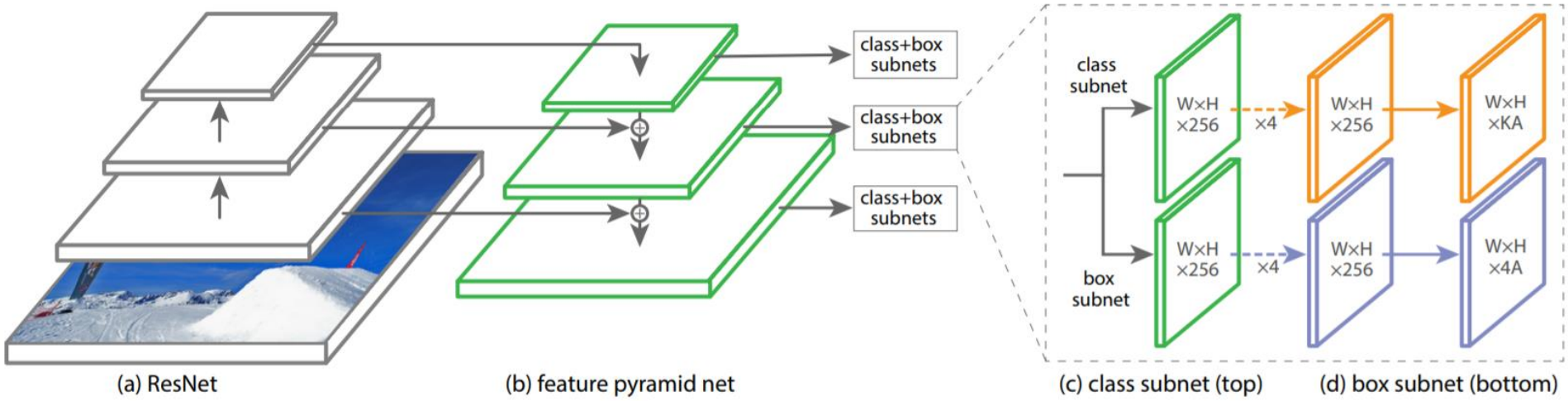
RetinaNet





	AP	time
[A] YOLOv2† [27]	21.6	25
[B] SSD321 [22]	28.0	61
[C] DSSD321 [9]	28.0	85
[D] R-FCN‡ [3]	29.9	85
[E] SSD513 [22]	31.2	125
[F] DSSD513 [9]	33.2	156
[G] FPN FRCN [20]	36.2	172
RetinaNet-50-500	32.5	73
RetinaNet-101-500	34.4	90
RetinaNet-101-800	37.8	198

†Not plotted ‡Extrapolated time



مطالب تکمیلی (جزء مباحث آزمون پایان‌ترم)

- https://www.aparat.com/v/GBTgu?playlist=1001309
- https://www.aparat.com/v/FbT8U?playlist=1001309

به پایان آمد این دفتر

حکایت همچنان باقیست ...