國立高雄科技大學
National Kaohsiung University of Science and Technology

Faster R-CNN
Towards Real-Time Object Detection with Region Proposal Networks

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun

IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI), 2017

Speaker: Shih-Shinh Huang

September 24, 2020

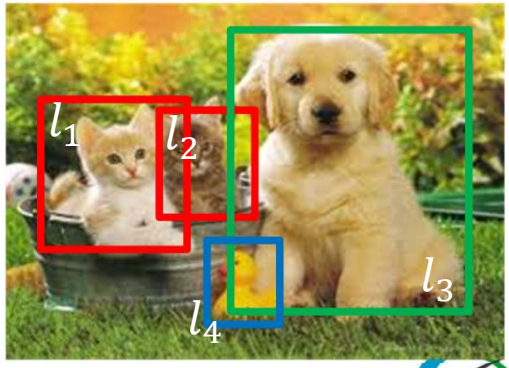
Outline

- Introduction
  - About Object Detection
  - Background
  - Architecture Overview
- Network Ingredient
  - Region Proposal Network (RPN)
  - RoI Pooling
  - Detection Network
- Faster R-CNN Training
  - Training Overview
  - RPN Loss
  - Detection Loss

Introduction

- About Object Detection
  - Input:
    - I: input image
    - $\{c_1, c_2, \ldots, c_n\}$: object classes to be detected
  - Output:
    - $\{r_1, r_2, \ldots r_m\}$: bounding boxes of m detected objects
    - $\{l_1, l_2, \ldots l_m\}$: class labels of all detected objects

$C = \{\mathbf{cat}, \mathbf{dog}, \mathbf{duck}\}$



$l_1 = \mathbf{cat}$

$l_2 = \mathbf{cat}$

$l_3 = \mathbf{dog}$

$l_4 = \mathbf{duck}$

Introduction

- Background
  - Region-based CNN (R-CNN) has made remarkable advance in object detection.

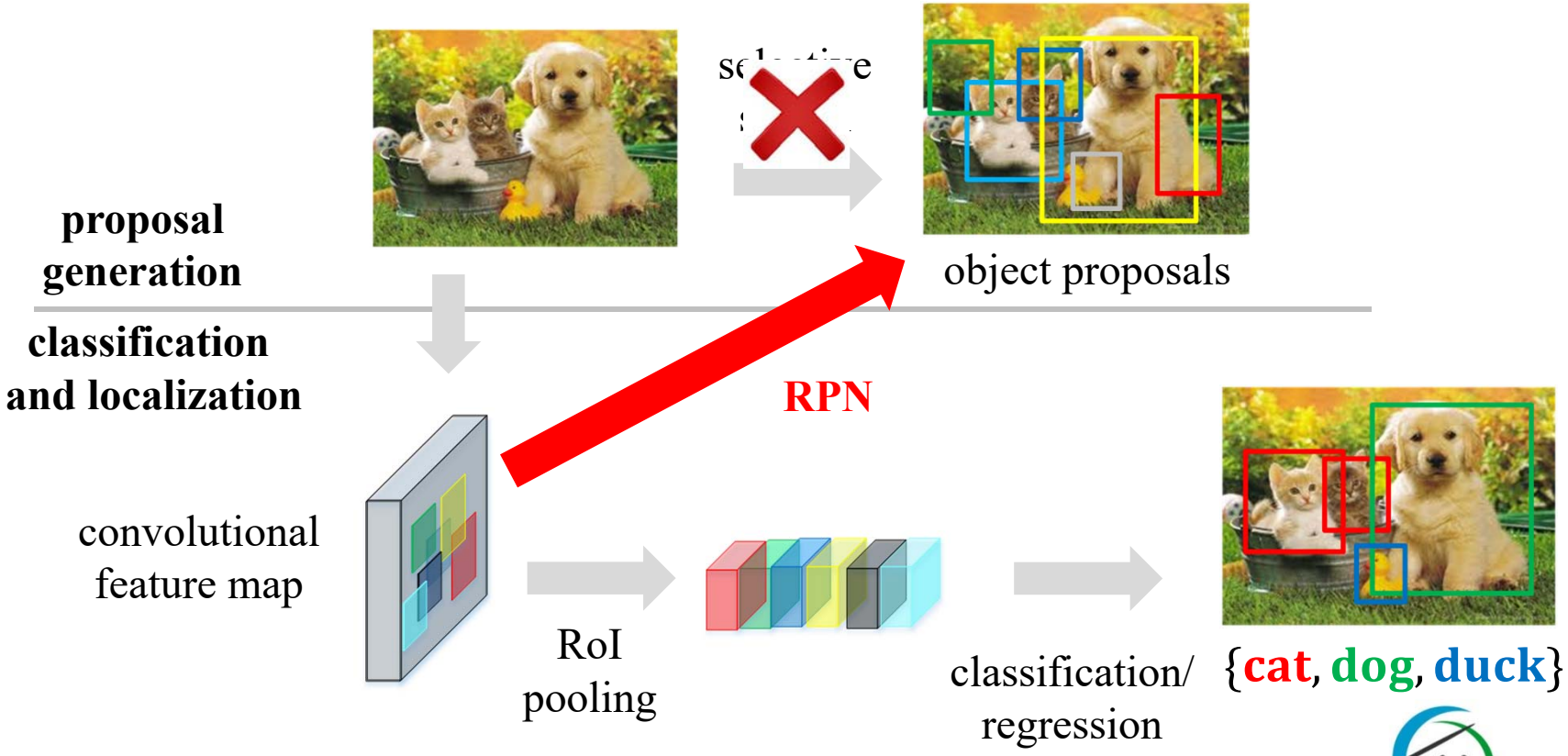
    R. Girshick, et. al. “Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation,” *CVPR*, 2014

  - Fast R-CNN improves detection speed by sharing convolution feature map across proposals.

    R. Girshick, “Fast R-CNN,” *CVPR*, 2015

Introduction

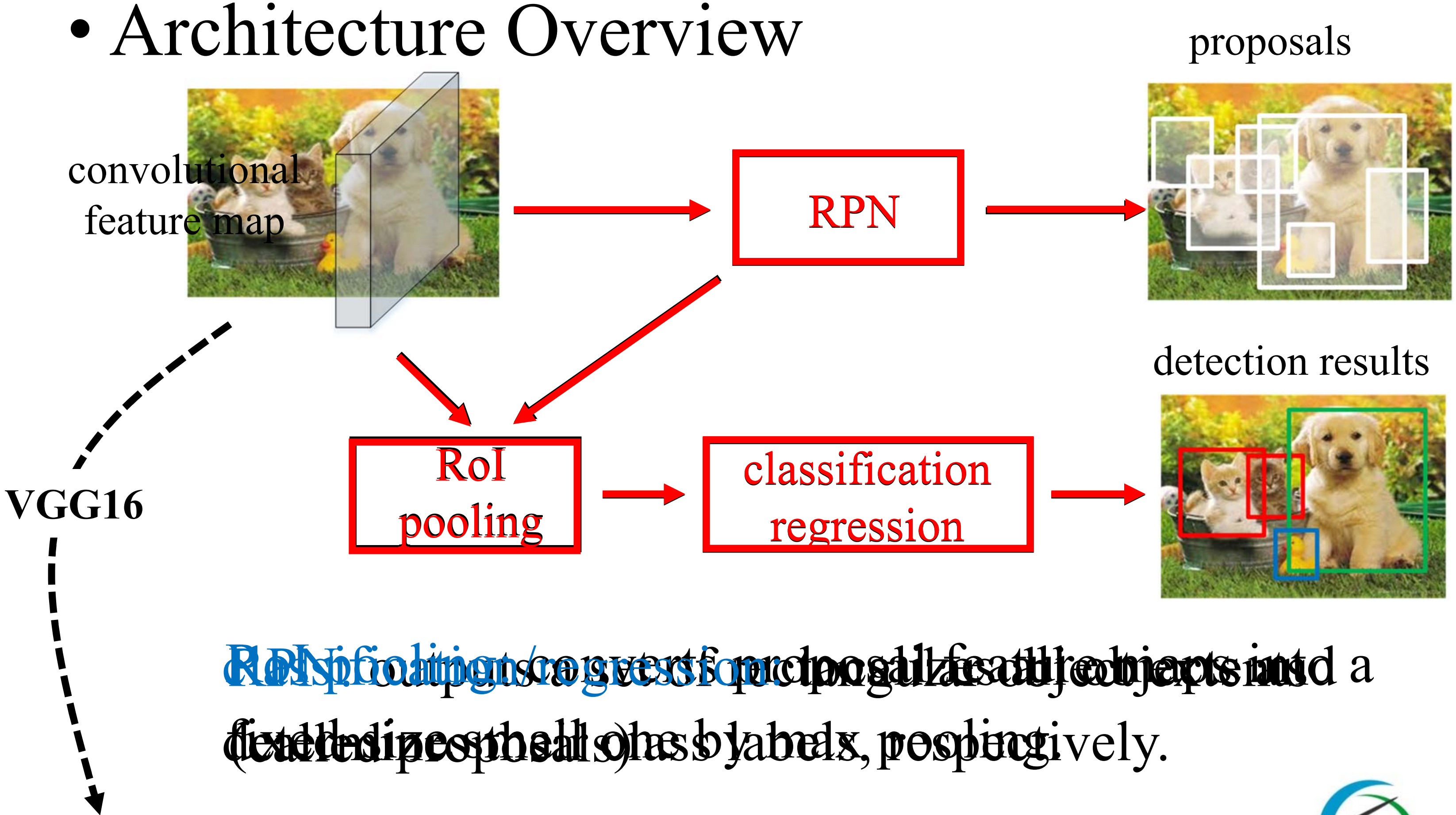
- Background: Faster R-CNN



J. R. Uijlings et. al. "Selective Search for Object Detection" *IJCV*, 2013.

Introduction

- Architecture Overview



RPN: outputs a set of rectangular object proposals (called proposals).

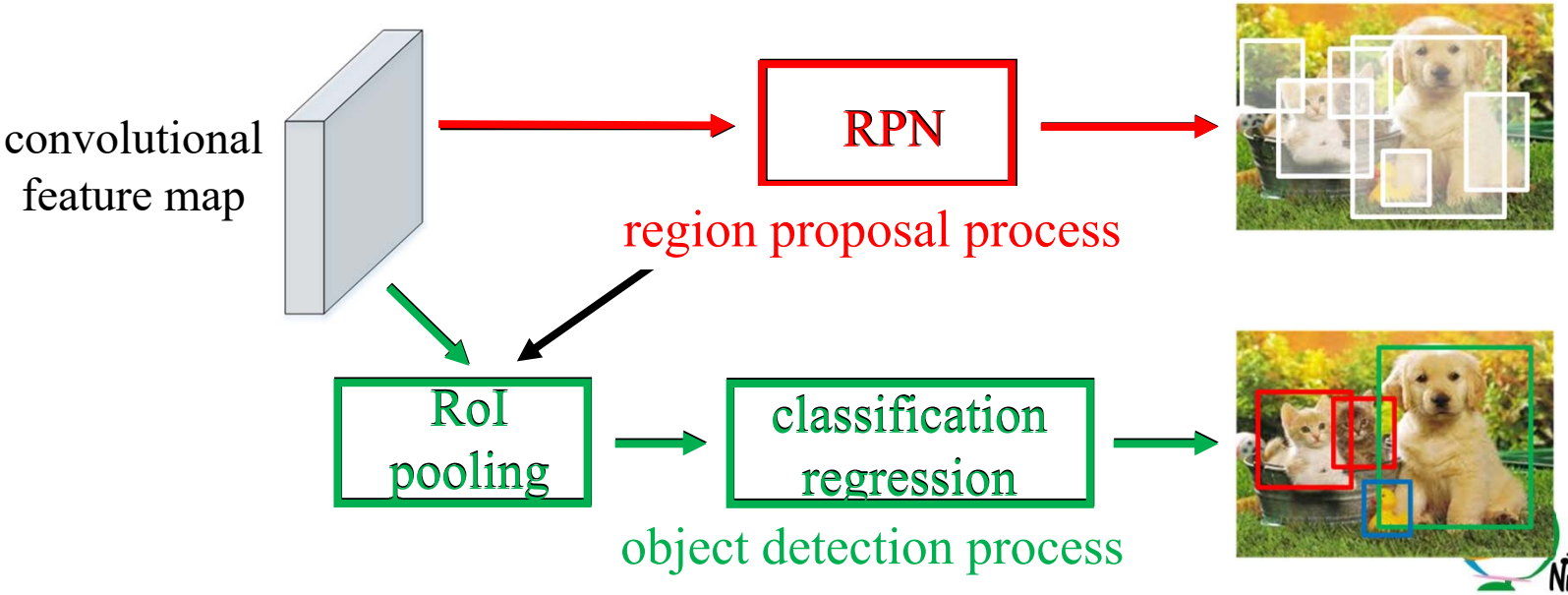
RoI pooling: converts proposal features into a fixed-size small one by max pooling.

Classification/regression: localizes the objects and detects their class labels, respectively.

K. Simonyan et. al. "Very Deep Convolutional Networks for Large-Scale Image Recognition," *ICLR*, 2015.

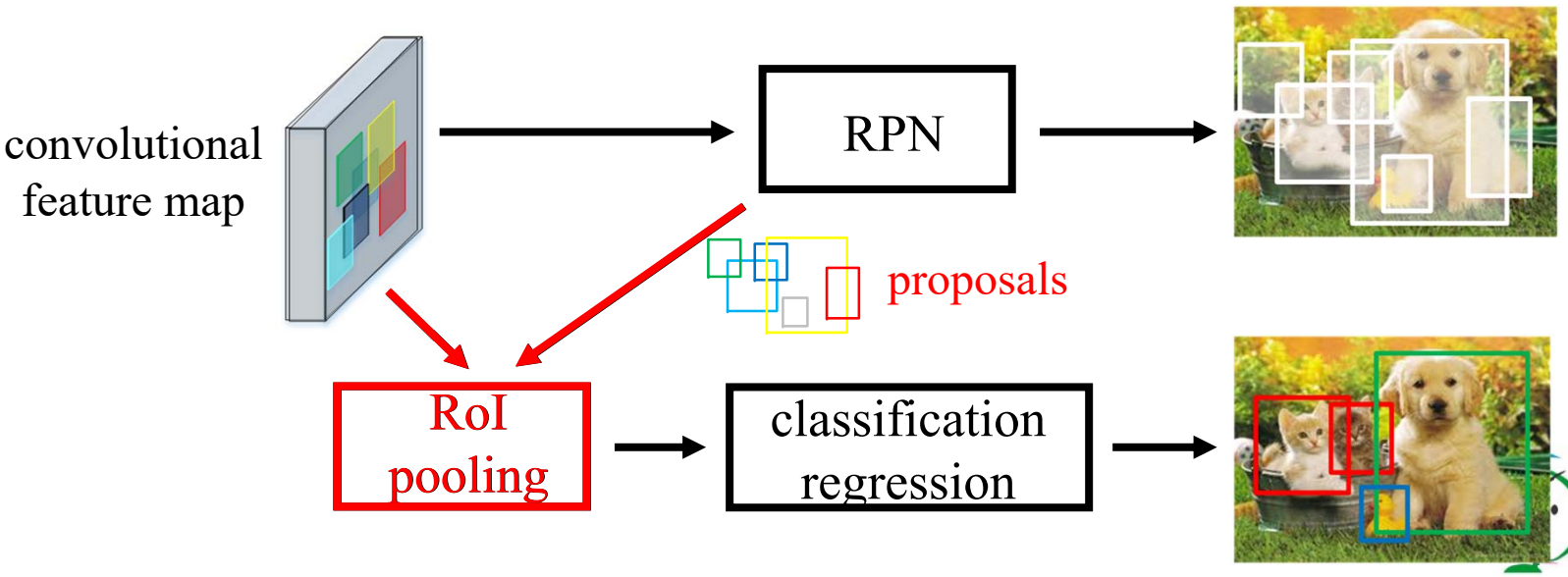
Introduction

- Architecture Overview
  - Efficiency: two-level feature sharing
    - first-level: region proposal and detection
    - second-level: among generated proposals



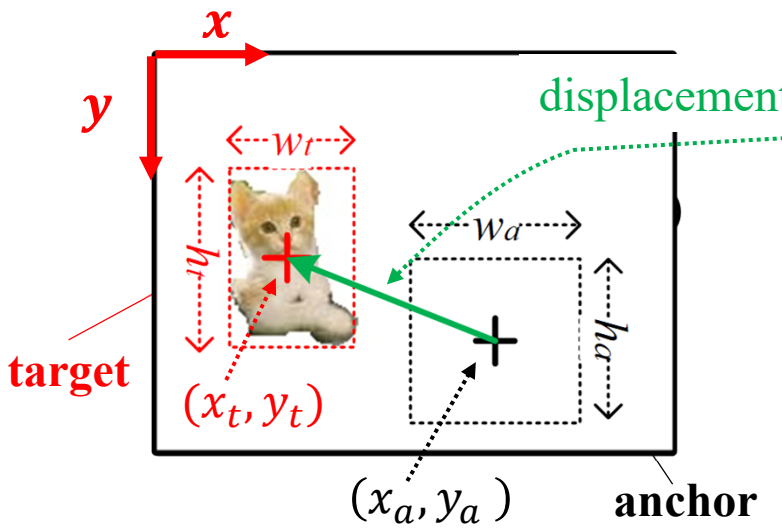
Introduction

- Architecture Overview
  - Efficiency: two-level feature sharing
    - first-level: region proposal and detection
    - second-level: among generated proposals



Introduction

- Architecture Overview
  - Effectiveness: two-level anchor mechanism.
    - Anchor is a reference box (x_a, y_a, w_a, h_a)
    - Anchor parameterizes a target box (x_t, y_t, w_t, h_t) by regression offset (d_x, d_y, d_w, d_h)



displacement

$$d_x = (x_t - x_a)/w_a$$

$$d_y = (y_t - y_a)/h_a$$

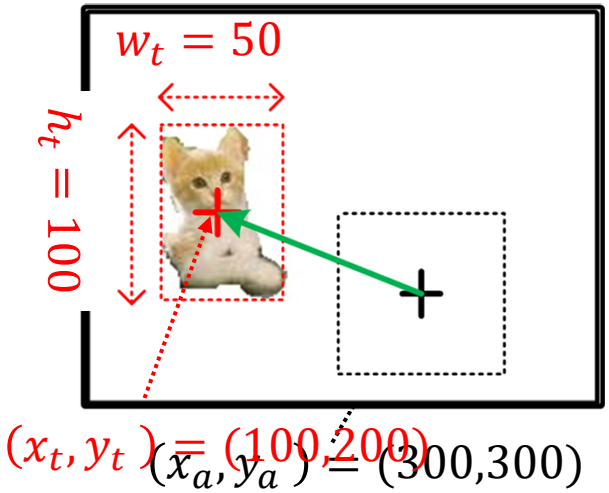
scaling

$$d_w = w_t/w_a$$

$$d_h = h_t/h_a$$

Introduction

- Architecture Overview
  - Effectiveness: two-level anchor mechanism.
    - Anchor is a reference box (x_a, y_a, w_a, h_a)
    - Anchor parameterizes a target box (x_t, y_t, w_t, h_t) by regression offset (d_x, d_y, d_w, d_h)



$(x_a, y_a, w_a, h_a) = (100,200,50,100)$

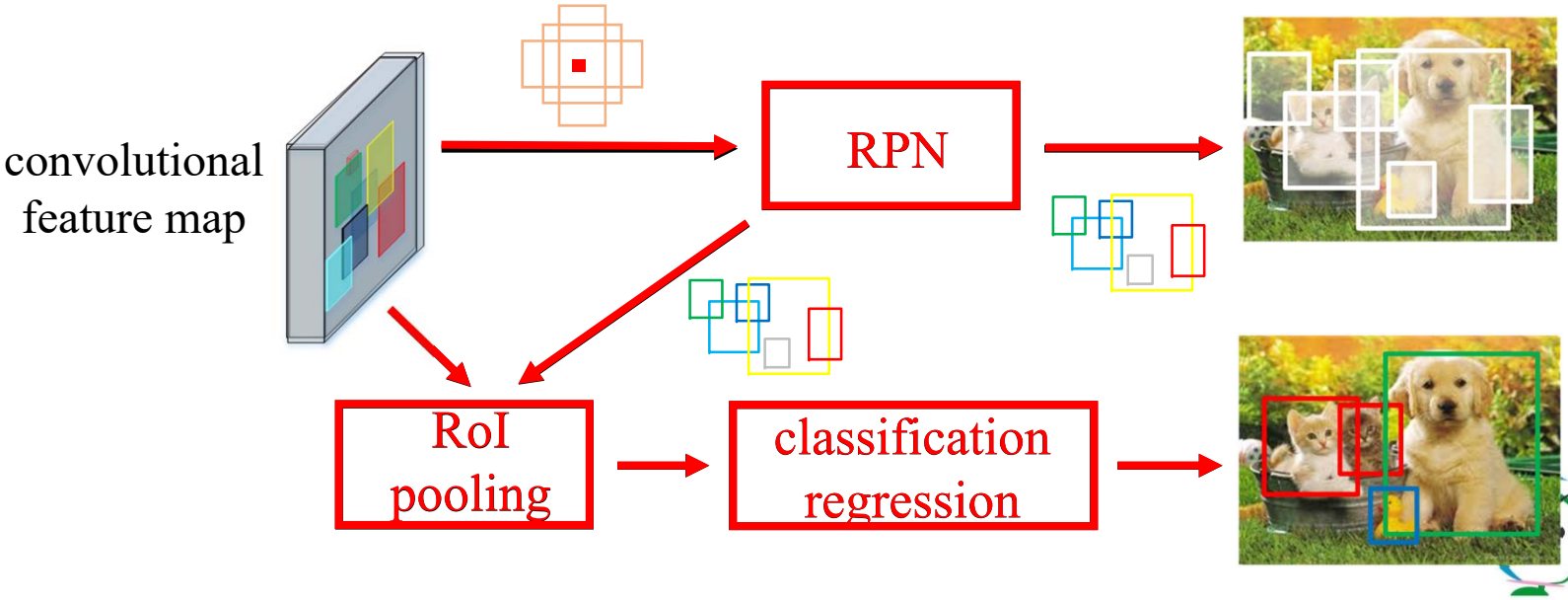
$(d_x, d_y, d_w, d_h) = (-2.0, -1.0, 0.5, 1.0)$

$x_t = 100$ $-3($ $w_t = 50$ l_w $0.5 = w_t/100$

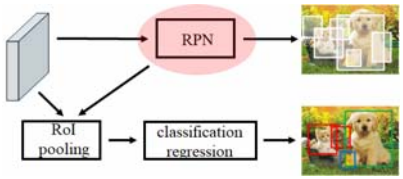
$y_t = 200$ $-3($ $h_t = 100$ h $1.0 = h_t/100$

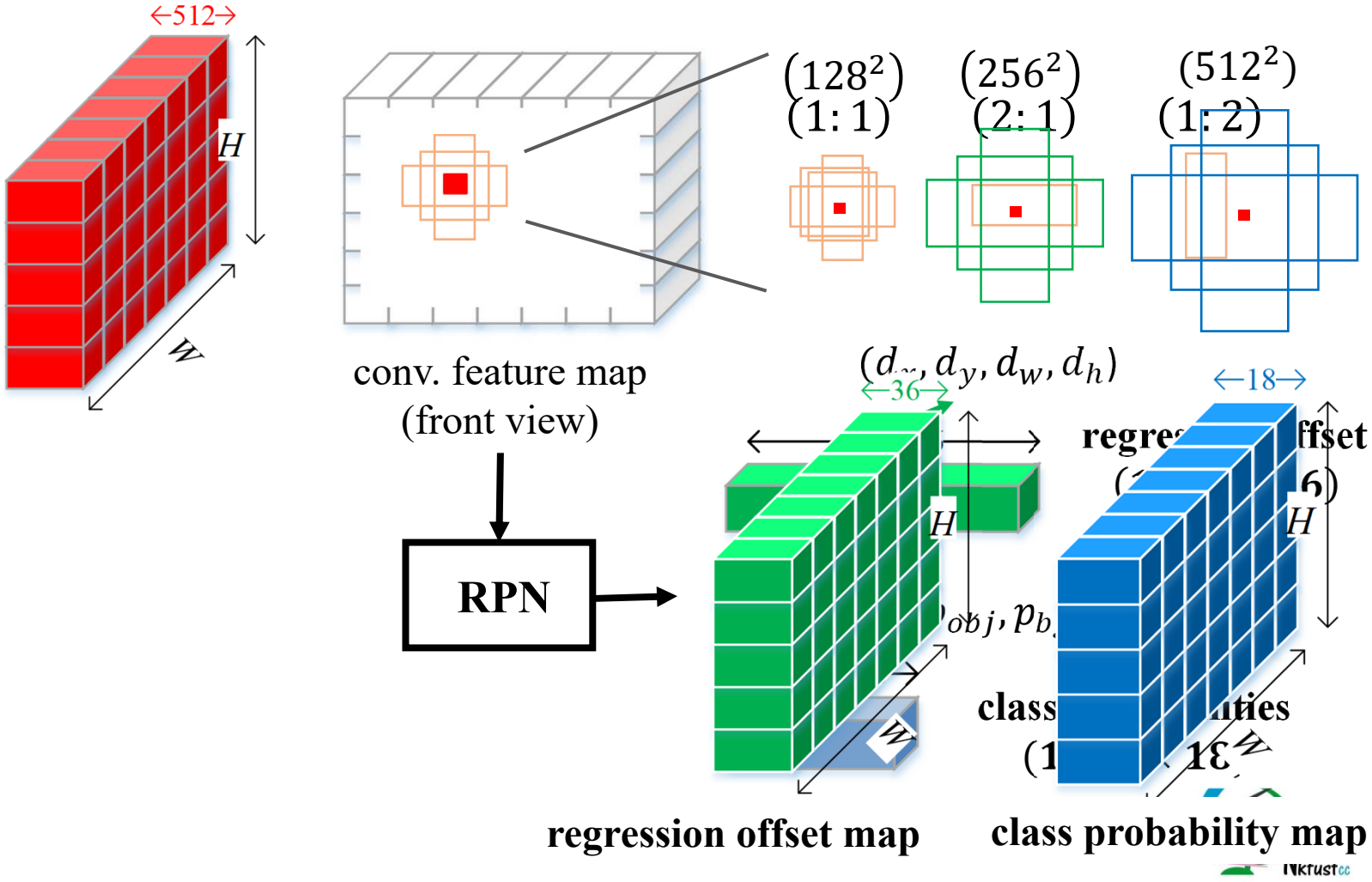
Introduction

- Architecture Overview
  - Effectiveness: two-level anchor mechanism
    - first-level: put several anchors at every point (dense)
    - second-level: take proposals as anchors (sparse)

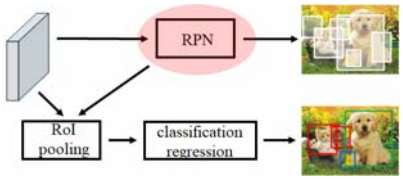


Network Ingredient

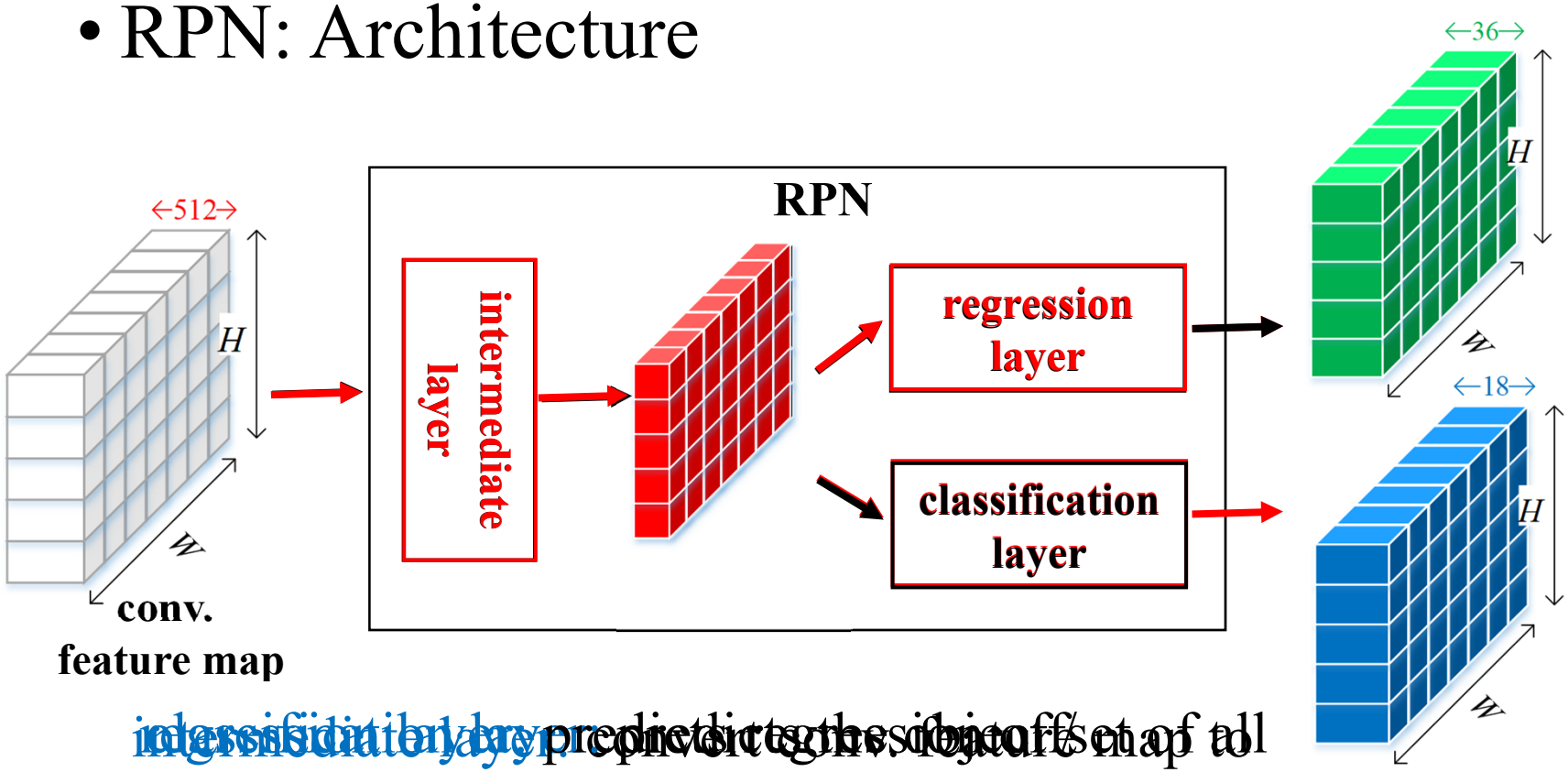




Network Ingredient



- RPN: Architecture

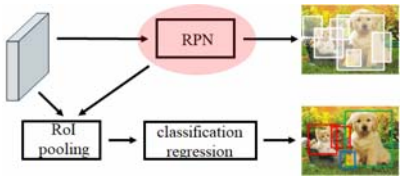


intermediate layer: convert the conv. feature map to the proposal.

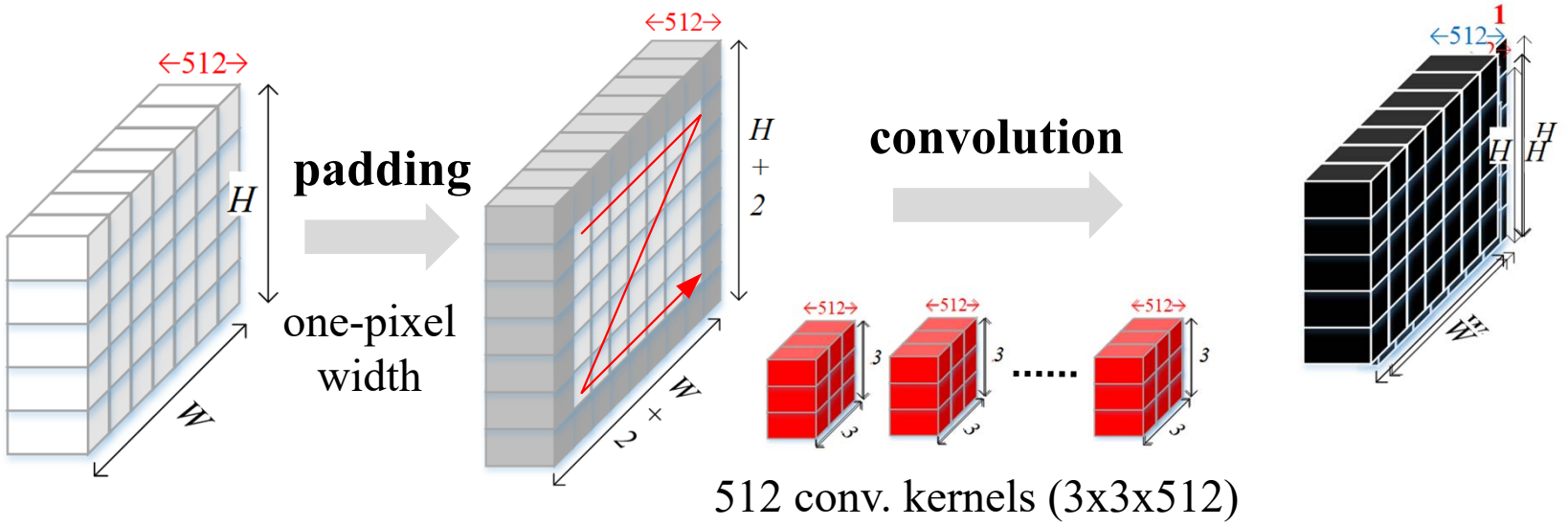
regression layer: predicts the offset of all proposals specifically for proposal generation.

classification layer: predicts the probabilities of all proposals.

Network Ingredient

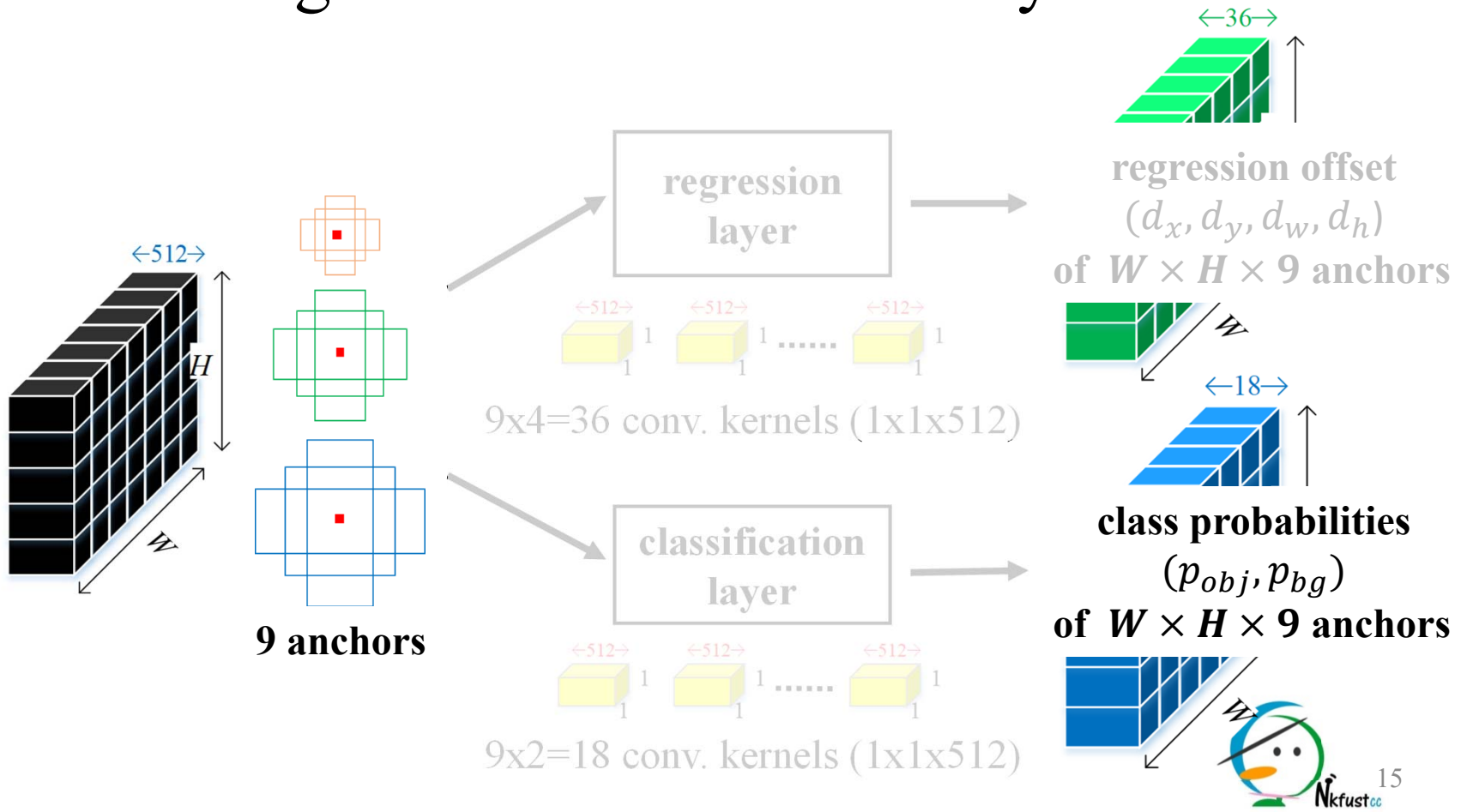


- RPN: intermediate layer
  - extract feature map for proposal generation.



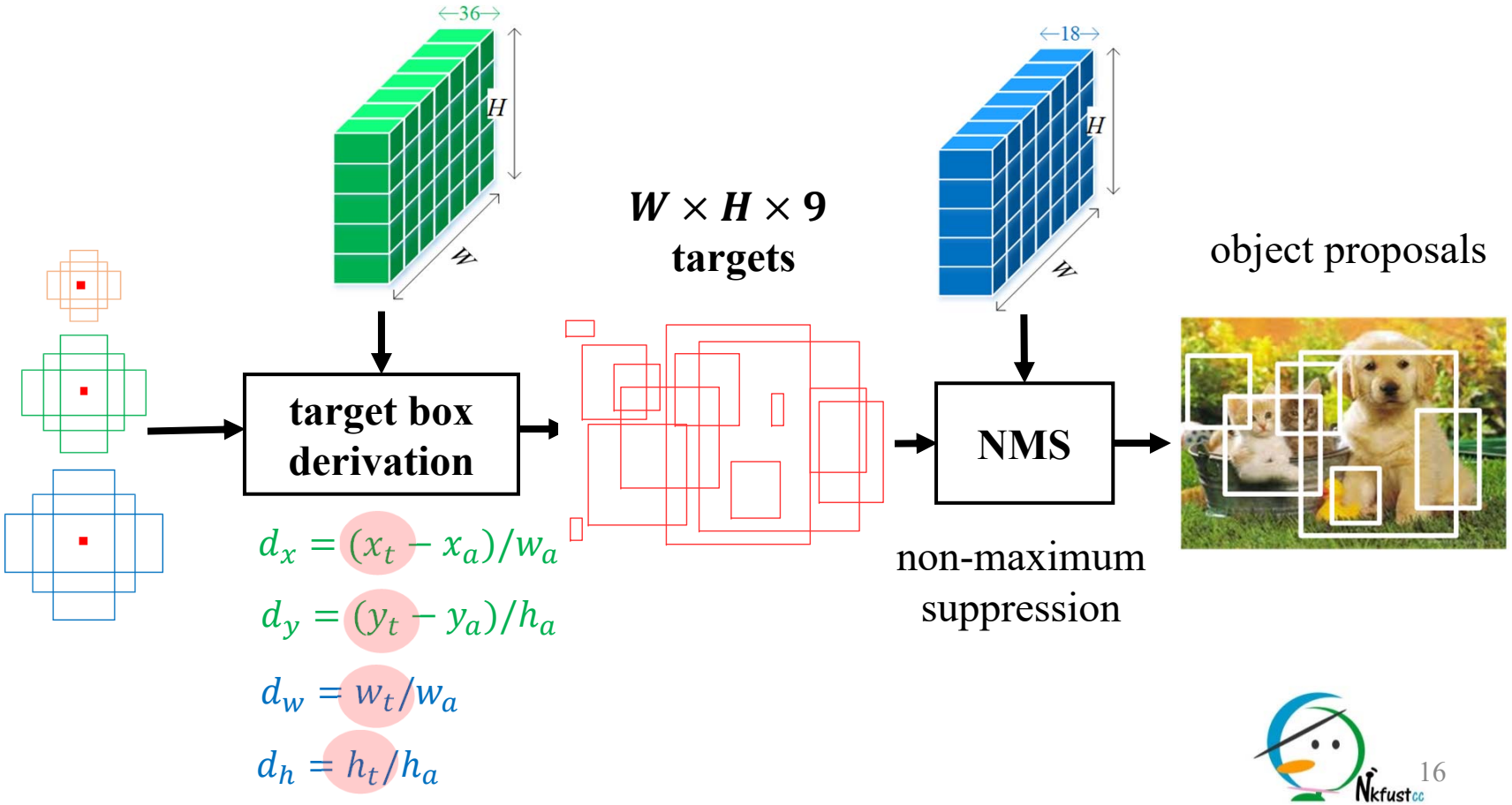
Network Ingredient

- RPN: regression/classification layers

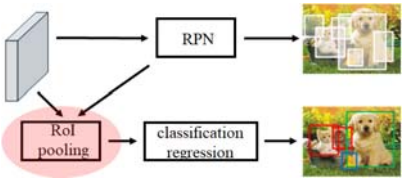


Network Ingredient

- RPN: regression/classification layers



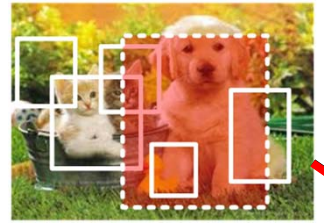
Network Ingredient

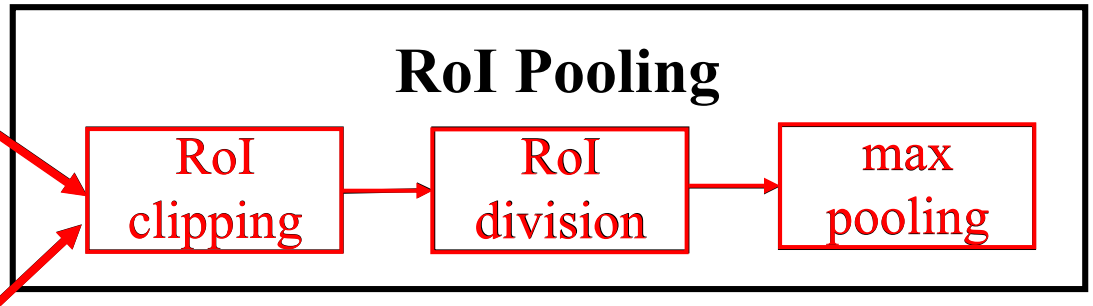


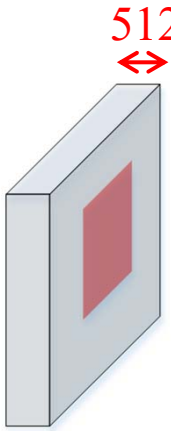
- RoI Pooling

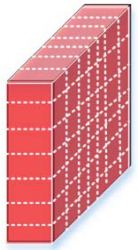
RoI clipping: crop the conv. feature map of the object proposal box

max pooling: apply max pooling to every grid of the RoI feature map

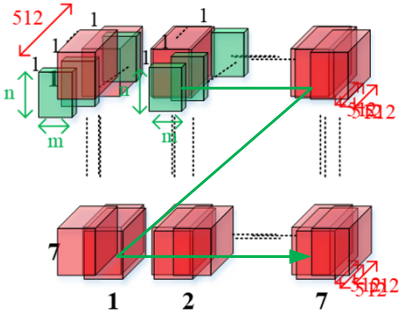




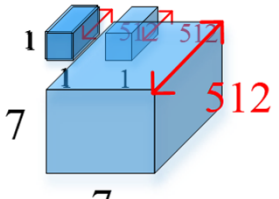




RoI feature clip

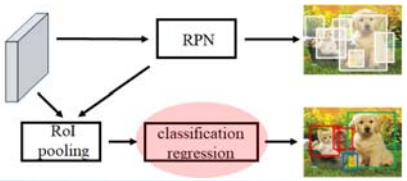


7 × 7 grids (front view)

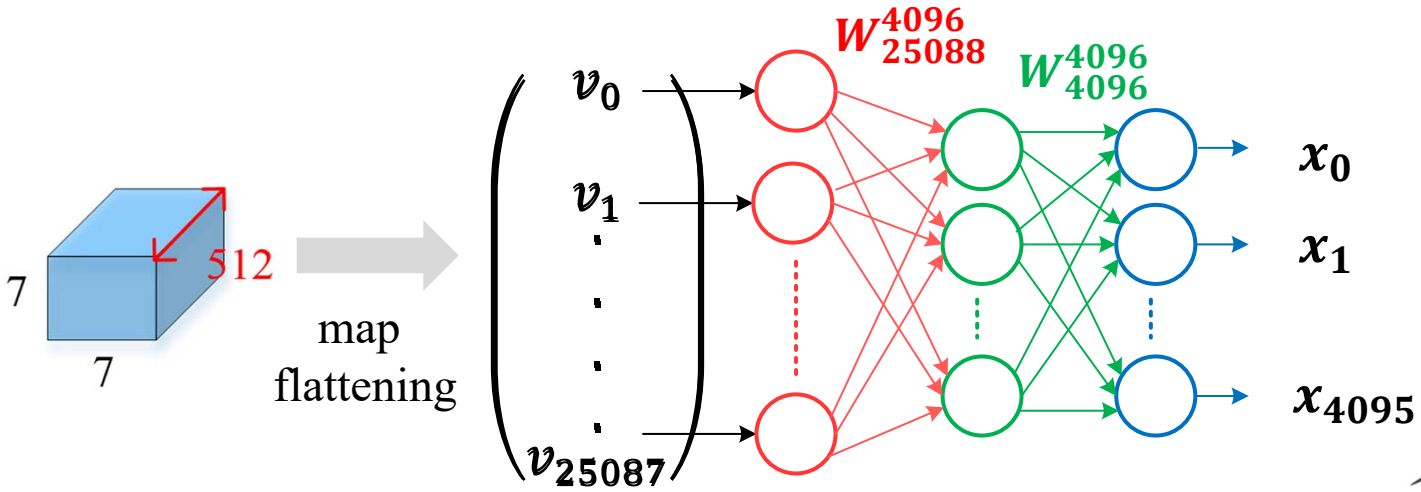


proposal feature map

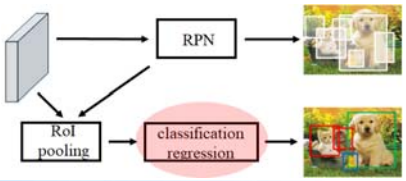
Network Ingredient



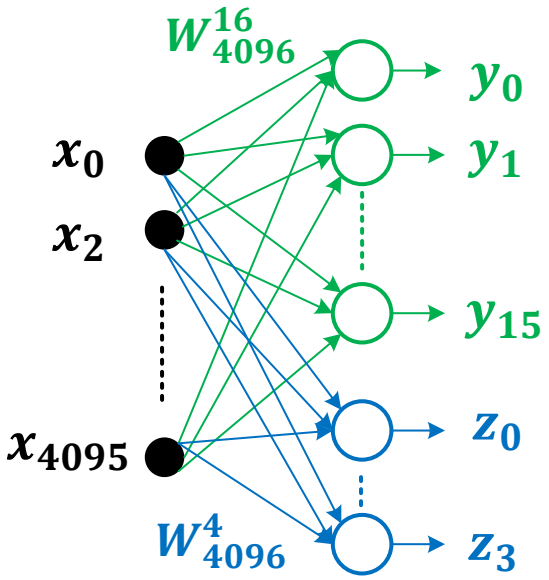
- Detection Network
  - flatten proposal feature map to a $7 \times 7 \times 512 (= 25088)$ feature vector
  - use two fully connected layers for reducing feature dimension to 4096



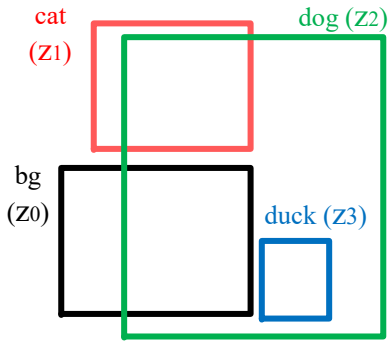
Network Ingredient

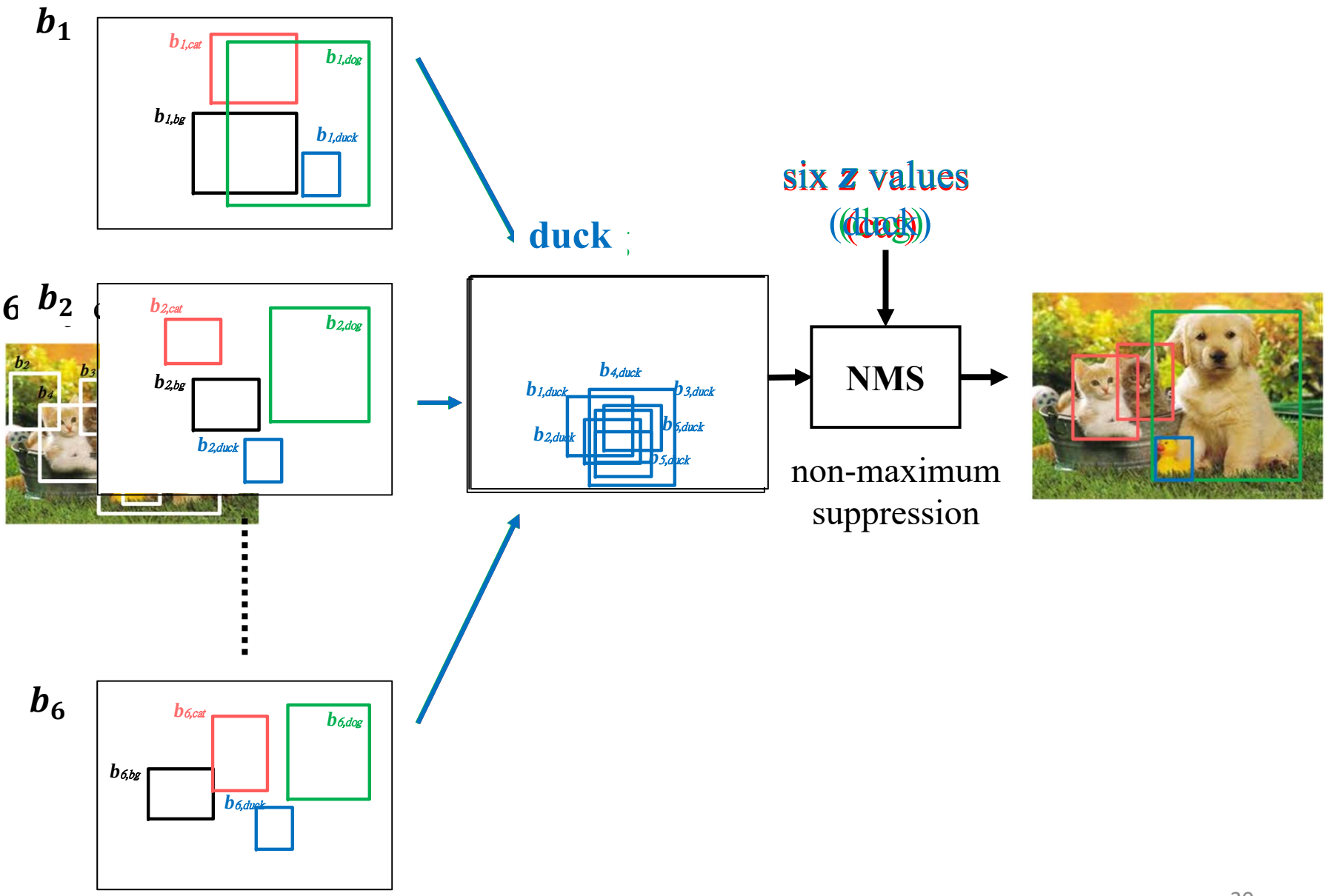


- Detection Network
  - apply two fully connected branches for
    - regression: $4096 \rightarrow (N_{class} + 1) \times 4$ background (bg)
    - classification: $4096 \rightarrow (N_{class} + 1)$ offset: (d_x, d_y, d_w, d_h)



$\hat{C} = \{\mathbf{bg}, \mathbf{cat}, \mathbf{dog}, \mathbf{duck}\}$ $N_{class} = 3$




b_1
$b_{1,cat}$
$b_{1,dog}$
$b_{1,bg}$
$b_{1,duck}$
b_2
$b_{2,cat}$
$b_{2,dog}$
$b_{2,bg}$
$b_{2,duck}$
b_6
$b_{6,cat}$
$b_{6,dog}$
$b_{6,bg}$
$b_{6,duck}$
duck
$b_{4,duck}$
$b_{1,duck}$
$b_{3,duck}$
$b_{2,duck}$
$b_{5,duck}$
six z values
NMS
non-maximum
suppression

Faster R-CNN Training

- Training Overview
  - select an image with its labels from dataset
  - flow through Faster R-CNN network to obtain
    - RPN: regression offset map and classification map
    - Detection Network: object regression offset $\{y_0, y_1, \dots\}$ and class probabilities $\{z_0, z_1, \dots.\}$
  - compute prediction loss for updating parameters
    - RPN: convolutional kernels
    - Detection Network: weight matrices

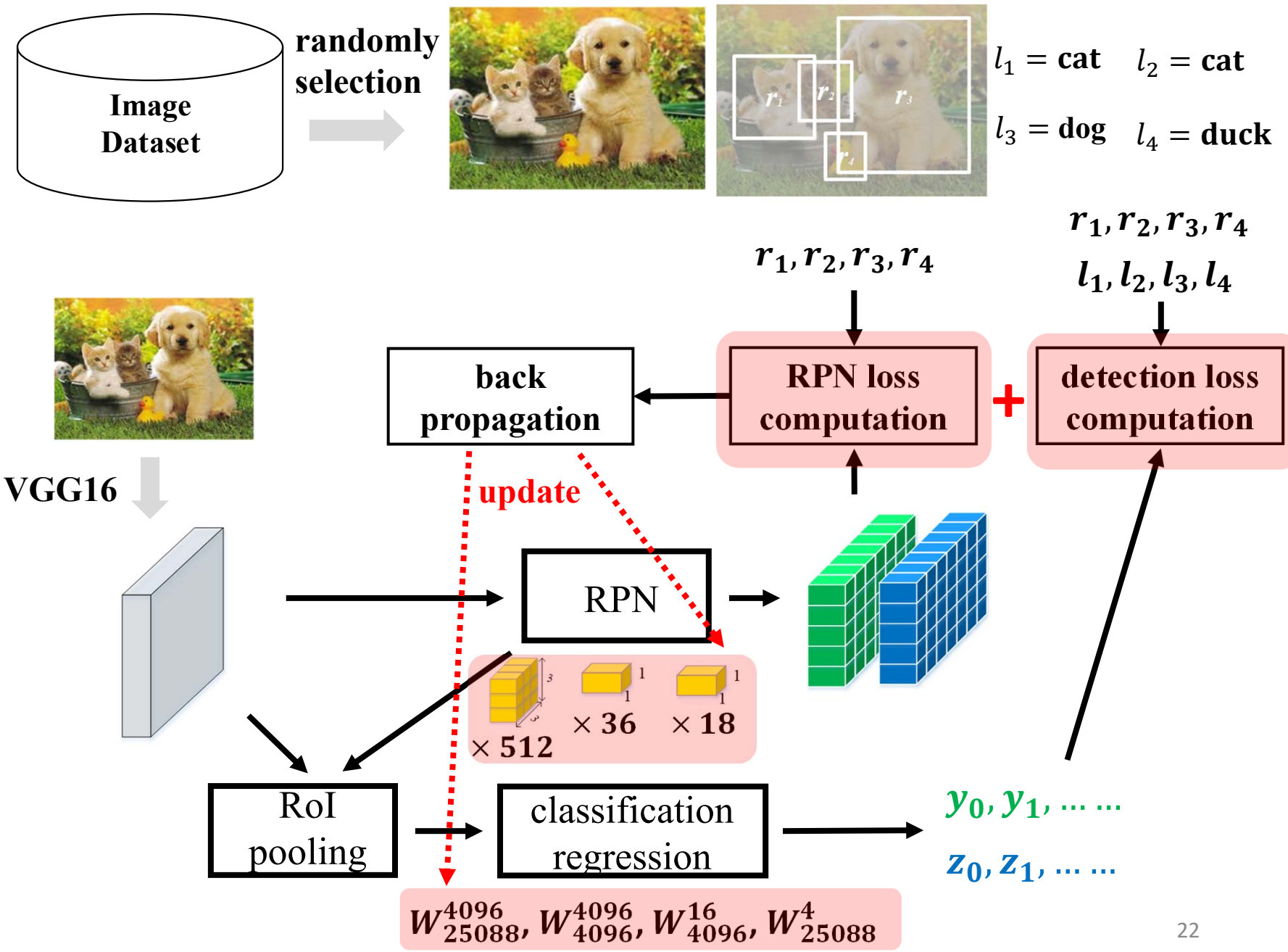


Image Dataset
randomly selection
l_1 = cat
l_2 = cat
l_3 = dog
l_4 = duck
r_1, r_2, r_3, r_4
r_1, r_2, r_3, r_4
l_1, l_2, l_3, l_4
back propagation
RPN loss computation
+
detection loss computation
VGG16
update
RPN
× 512
× 36
× 18
RoI pooling
classification regression
$y_0, y_1, \ldots\ldots$
$z_0, z_1, \ldots\ldots$
$W^{4096}_{25088}, W^{4096}_{4096}, W^{16}_{4096}, W^{4}_{25088}$

Faster R-CNN Training

- RPN Loss $L^{(r)}(.)$
  - The computation of RPN loss is on a mini-batch
    - 128 positive (object) anchors
    - 128 negative (background) anchors

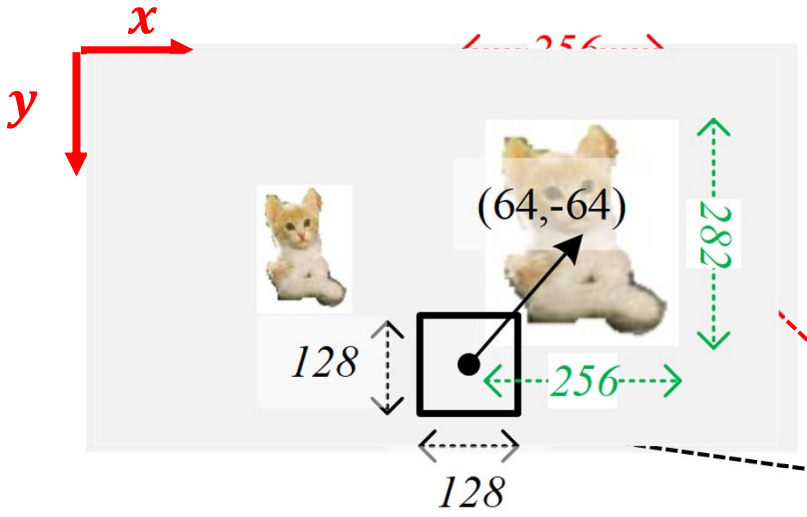
Quarter Unit: Region Proposal Network (PRN)

(00:13:40)

https://youtu.be/0tBhRfEzUWs

Faster R-CNN Training

- RPN Loss $L^{(r)}(.)$
  - $D = \{\hat{d}^{(i)}, \hat{p}^{(i)}\}_{i=1}^{256}$: selected anchors
    - $\hat{p}^{(i)} = \left(\hat{p}_{obj}^{(i)}, \hat{p}_{bg}^{(i)}\right)$: class probabilities of ith anchor
    - $\hat{d}^{(i)} = \left(\hat{t}_x^{(i)}, \hat{t}_y^{(i)}, \hat{t}_w^{(i)}, \hat{t}_h^{(i)}\right)$: regression offset of ground-truth box with respect to ith anchor



$$\hat{d}^{(1)} = \begin{pmatrix} 32 & 32 & 256 & 282 \\ 0.125 & 0.125 & 1.0 & 1.1 \\ 256 & 256 & 256 & 256 \end{pmatrix}$$

$$\hat{d}^{(2)} = \begin{pmatrix} 64 & -64 & 256 & 282 \\ 0.25 & -0.25 & 2.0 & 2.2 \\ 256 & 256 & 128 & 128 \end{pmatrix}$$

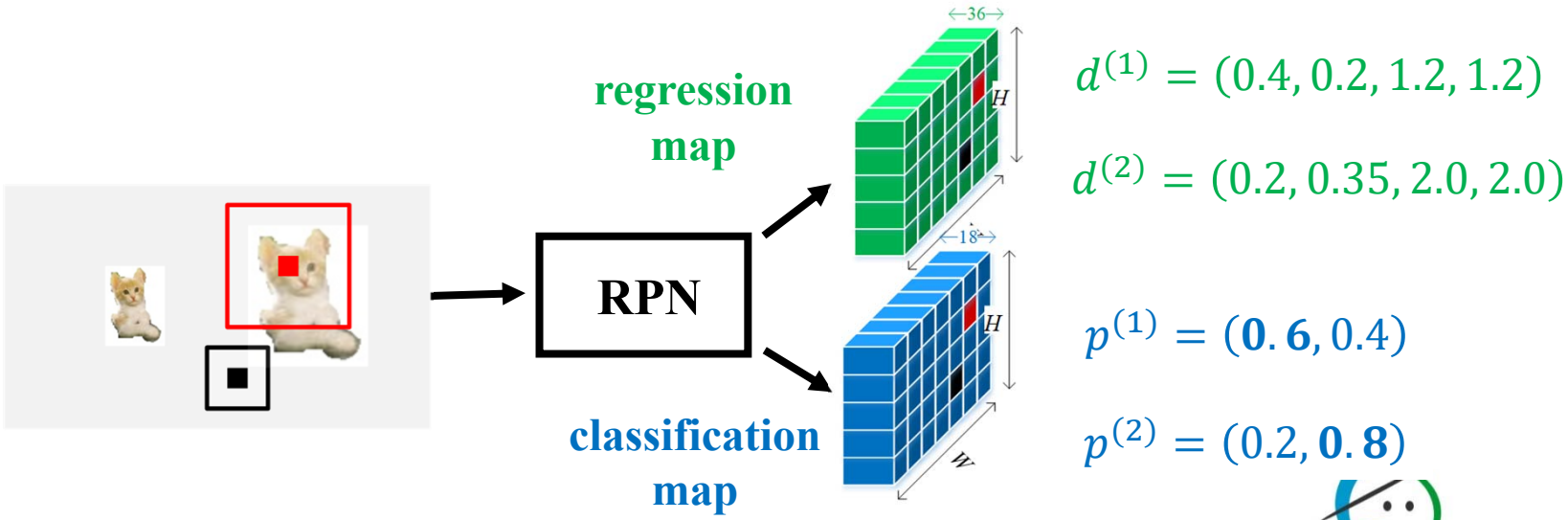
$$\hat{p}_{obj}, \hat{p}_{bg}$$

$$\hat{p}^{(1)} = (\mathbf{1.0}, 0.0)$$

$$\hat{p}^{(2)} = (0.0, \mathbf{1.0})$$

Faster R-CNN Training

- RPN Loss $L^{(r)}(.)$
  - $\{d^{(i)}, p^{(i)}\}$: predicted parameters from RPN
  - $L^{(r)}(.)$ evaluates fitness of $\{d^{(i)}, p^{(i)}\}$ to $\{\hat{d}^{(i)}, \hat{p}^{(i)}\}$



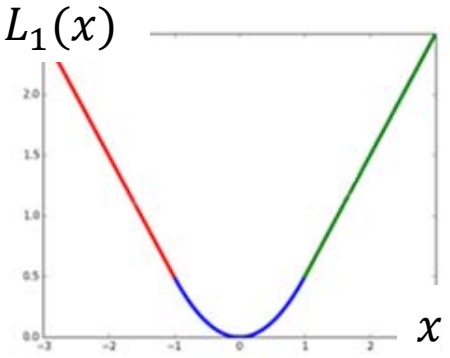
Faster R-CNN Training

- RPN Loss $L^{(r)}(.)$

regression term　　classification term

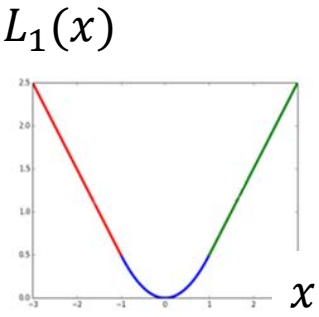
$$L^{(r)}\left(\{\boldsymbol{d}^{(i)}, \boldsymbol{p}^{(i)}\}\right) = \sum_i \hat{\boldsymbol{p}}_{\boldsymbol{obj}}^{(i)} \times L_{reg}^{(r)}\left(\boldsymbol{d}^{(i)}, \hat{\boldsymbol{d}}^{(i)}\right) + \sum_i L_{cls}^{(r)}\left(\boldsymbol{p}^{(i)}, \hat{\boldsymbol{p}}^{(i)}\right)$$

  - regression term
    - depend only on positive anchors
    - $L_{reg}^{(r)}$: smooth L_1 function
  - classification term
    - $L_{cls}^{(r)}$: log loss function



$$L_1(x) = \begin{cases} 0.5x^2, & if\,|x| < 1 \\ |x| - 0.5, & otherwise \end{cases}$$

Faster R-CNN Training



- RPN Loss $L^{(r)}(.)$: $\sum_i \hat{p}_{obj}^{(i)} \times L_{reg}^{(r)}(d^{(i)}, \hat{d}^{(i)})$

$$L_{reg}^{(r)}(d^{(i)}, \hat{d}^{(i)}) = L_1\left(d_x^{(i)} - \hat{d}_x^{(i)}\right) + L_1\left(d_y^{(i)} - \hat{d}_y^{(i)}\right)$$
$$+ L_1\left(\log\left(d_w^{(i)}\right) - \log\left(\hat{d}_w^{(i)}\right)\right) + L_1\left(\log\left(d_h^{(i)}\right) - \log\left(\hat{d}_h^{(i)}\right)\right)$$

	d_x	d_y	d_w	d_h
$d^{(1)}$ =	(0.4	0.2	1.2	1.2))
$\hat{d}^{(1)}$ =	(0.125	0.125	1.0	1.1))

$$= L_1(0.4 - 0.125) + L_1(0.2 - 0.125) + L_1(\log(1.2) - \log(1.0)) + L_1(\log(1.2) - \log(1.1))$$
$$= (0.5 \times (0.275)^2) + (0.5 \times (0.075)^2) + (0.5 \times (0.18)^2) + (0.5 \times (0.09)^2)$$

Faster R-CNN Training

- RPN Loss $L^{(r)}(.)$: $\sum_i L_{cls}^{(r)}\left(p^{(i)}, \hat{p}^{(i)}\right)$

$$L_{cls}^{(r)}\left(p^{(i)}, \hat{p}^{(i)}\right) = -\left(\hat{p}_{obj}^{(i)} \times \log\left(p_{obj}^{(i)}\right) + \hat{p}_{bg}^{(i)} \times \log\left(p_{bg}^{(i)}\right)\right)$$

	p_{obj}	p_{bg}
$p^{(1)}$ =	0.6	0.4
$\hat{p}^{(1)}$ =	1.0	0.0
$p^{(2)}$ =	0.2	0.8
$\hat{p}^{(2)}$ =	0.0	1.0

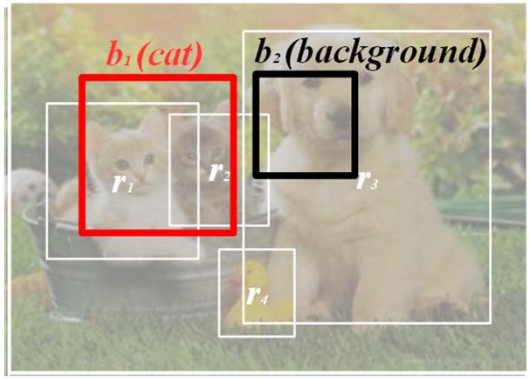
$$\sum_i L_{cls}^{(r)}\left(p^{(i)}, \hat{p}^{(i)}\right)$$

$$= -\left(\hat{p}_{obj}^{(1)} \times \log\left(p_{obj}^{(1)}\right) + \hat{p}_{bg}^{(1)} \times \log\left(p_{bg}^{(1)}\right)\right)$$

$$-\left(\hat{p}_{obj}^{(2)} \times \log\left(p_{obj}^{(2)}\right) + \hat{p}_{bg}^{(2)} \times \log\left(p_{bg}^{(2)}\right)\right)$$

Faster R-CNN Training

- Detection Loss $L^{(d)}(.)$
  - A mini-batch for computing detection loss consists of 128 proposals (anchors)
    - 32 object proposals (max. IoU $\geq \mathbf{0.5}$)
    - 96 background proposals ($\mathbf{0.1} \leq$ max. IoU $< \mathbf{0.5}$)

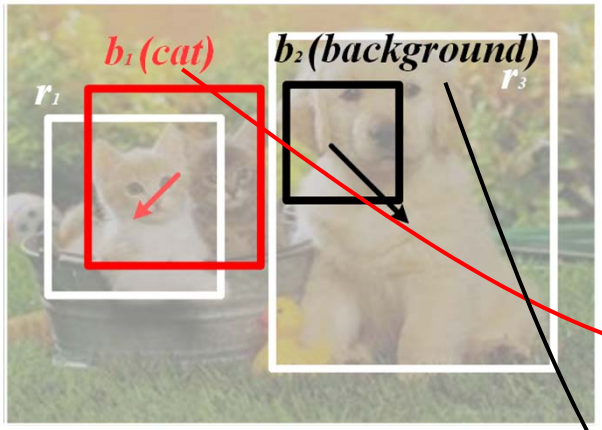


	cat	*cat*	*dog*	*duck*		
	r_1	r_2	r_3	r_4		
b_1	0.8	0.6	0.0	0.0	≥ 0.5	*cat* proposal
b_2	0.0	0.1	0.2	0.0	< 0.5	*bg* proposal

IoU: Intersection over Union

Faster R-CNN Training

- Detection Loss $L^{(d)}(.)$
  - $D = \{\hat{d}^{(i)}, \hat{p}^{(i)}\}_{i=1}^{128}$: selected proposals

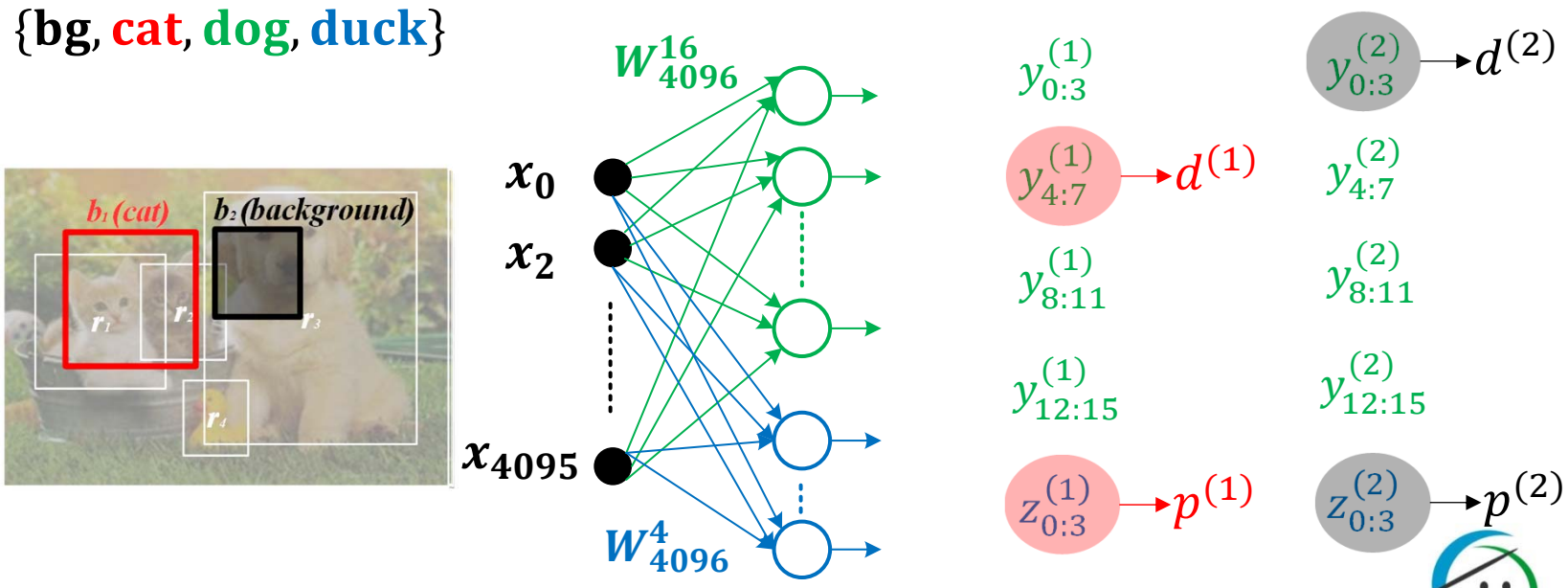


	d_x	d_y	d_w	d_h
$\hat{d}^{(1)}$ =	(-0.2	0.2	1.0	1.0)
$\hat{d}^{(2)}$ =	(1.0	1.0	2.0	2.0)

	p_{bg}	p_{cat}	p_{dog}	p_{duck}
$\hat{p}^{(1)}$ =	(0.0	**1.0**	0.0	0.0)
$\hat{p}^{(2)}$ =	(**1.0**	0.0	0.0	0.0)

Faster R-CNN Training

- Detection Loss $L^{(d)}(.)$
  - $\{\boldsymbol{d}^{(i)}, \boldsymbol{p}^{(i)}\}$: predicted parameters from detection network.



Faster R-CNN Training

- Detection Loss $L^{(d)}\left(\{\boldsymbol{d}^{(i)}, \boldsymbol{p}^{(i)}\}\right)$

$$= \sum_i \left(1.0 - \hat{p}_{bg}^{(i)}\right) \times L_{reg}^{(d)}\left(\boldsymbol{d}^{(i)}, \hat{\boldsymbol{d}}^{(i)}\right) + \sum_i L_{cls}^{(d)}\left(\boldsymbol{p}^{(i)}, \hat{\boldsymbol{p}}^{(i)}\right)$$

regression term　　classification term

$\boldsymbol{y}_{4:7}^{(1)}$ $\boldsymbol{p}_{bg}$ $\boldsymbol{d}_x$ $\boldsymbol{p}_{cat}$ $\boldsymbol{d}_y$ $\boldsymbol{p}_{dog}$ $\boldsymbol{d}_w$ $\boldsymbol{p}_{duck}$ $\boldsymbol{d}_h$

$\hat{\boldsymbol{p}}^{(1)}$ = (0.0 **1.0** 0.0 0.0)　　$\boldsymbol{d}^{(1)}$ = (-0.1 0.3 0.8 1.2)

$\hat{\boldsymbol{p}}^{(2)}$ = (**1.0** 0.0 0.0 0.0)　　$L_{reg}^{(d)}\left(\boldsymbol{d}^{(i)}, \hat{\boldsymbol{d}}^{(i)}\right) = L_{reg}^{(r)}\left(\boldsymbol{d}^{(i)}, \hat{\boldsymbol{d}}^{(i)}\right)$

RPN regression

$\hat{\boldsymbol{d}}^{(1)}$ = (-0.2 0.2 1.0 1.0)

Faster R-CNN Training

- Detection Loss $L^{(d)}\left(\left\{\boldsymbol{d}^{(i)}, \boldsymbol{p}^{(i)}\right\}\right)$

$$= \sum_i \left(\mathbf{1.0} - \hat{\boldsymbol{p}}_{\boldsymbol{bg}}^{(i)}\right) \times L_{reg}^{(d)}\left(\boldsymbol{d}^{(i)}, \hat{\boldsymbol{d}}^{(i)}\right) + \sum_i L_{cls}^{(d)}\left(\boldsymbol{p}^{(i)}, \hat{\boldsymbol{p}}^{(i)}\right)$$

classification term

$$L_{cls}^{(d)}\left(\boldsymbol{p}^{(i)}, \hat{\boldsymbol{p}}^{(i)}\right) = -\left(\hat{\boldsymbol{p}}_{\boldsymbol{bg}}^{(i)} \times \log\left(\boldsymbol{p}_{\boldsymbol{bg}}^{(i)}\right) + \hat{\boldsymbol{p}}_{\boldsymbol{cat}}^{(i)} \times \log\left(\boldsymbol{p}_{\boldsymbol{cat}}^{(i)}\right)\right)$$

p_{bg} p_{cat} p_{dog} p_{duck}

$z_{0:3}^{(0)} = p^{(1)} = (\ 0.1 \quad 0.7 \quad 0.1 \quad 0.1\ )$

$z_{0:3}^{(1)} = p^{(2)} = (\ 0.6 \quad 0.0 \quad 0.2 \quad 0.2\ )$

$\hat{p}^{(1)} = (\ 0.0 \quad \mathbf{1.0} \quad 0.0 \quad 0.0\ )$

$\hat{p}^{(2)} = (\ \mathbf{1.0} \quad 0.0 \quad 0.0 \quad 0.0\ )$

$L_{cls}^{(d)}\left(\boldsymbol{p}^{(1)}, \hat{\boldsymbol{p}}^{(1)}\right)$

$$-\left(\hat{\boldsymbol{p}}_{\boldsymbol{dog}}^{(i)} \times \log\left(\boldsymbol{p}_{\boldsymbol{dog}}^{(i)}\right) + \hat{\boldsymbol{p}}_{\boldsymbol{duck}}^{(i)} \times \log\left(\boldsymbol{p}_{\boldsymbol{duck}}^{(i)}\right)\right)$$

$L_{cls}^{(d)}\left(\boldsymbol{p}^{(2)}, \hat{\boldsymbol{p}}^{(2)}\right)$


thank you
© 1996, 2002 SANRIO CO., LTD.