Detection Attack

By Wei Gao

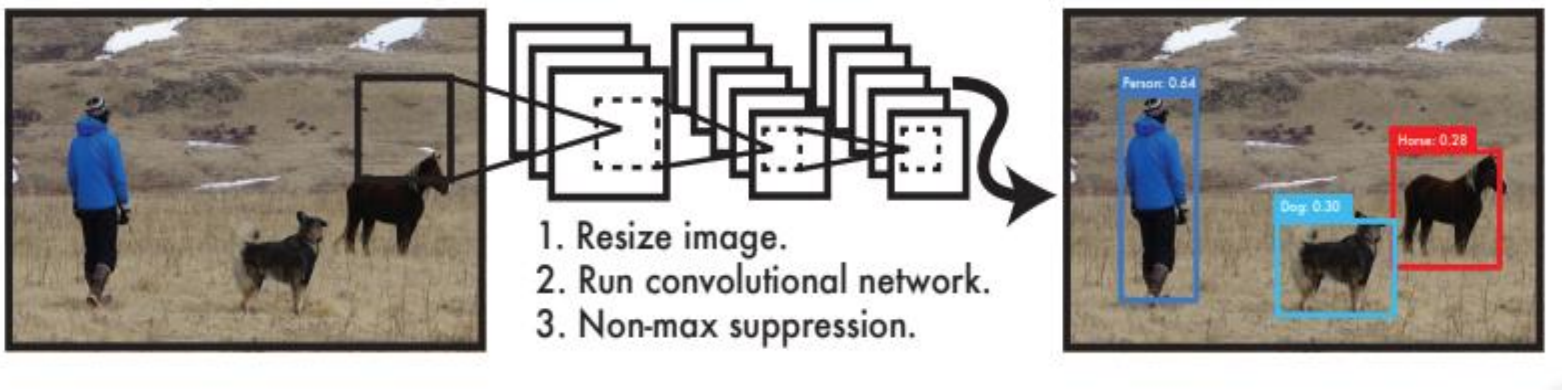
Detection

- Valuable Concepts
- One-Stage
- Two-Stage
- Comparison

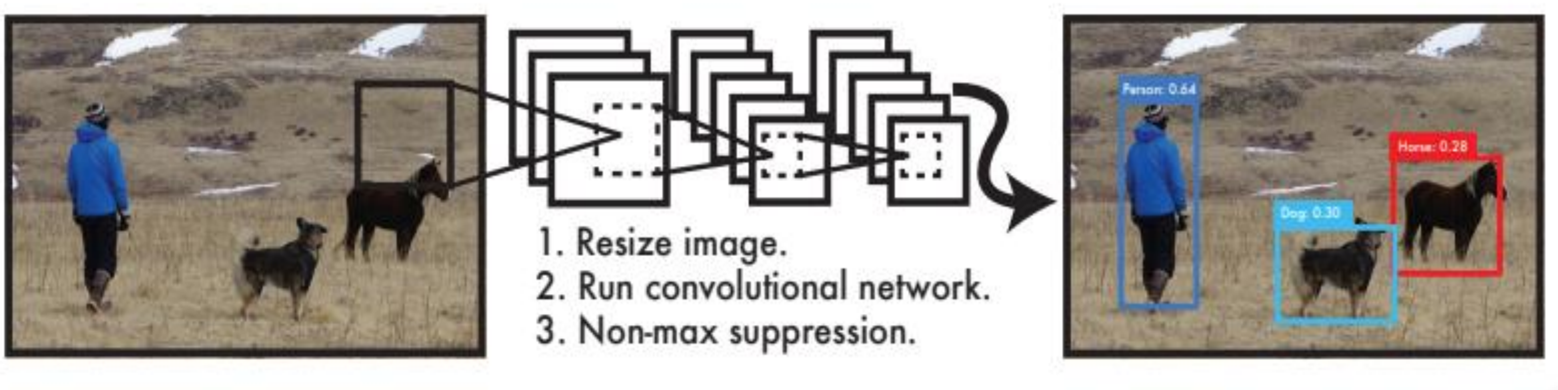
Anchor

- Backbone -- Feature Extractor
- Anchor -- Predefine Bounding Box
- Proposal -- Possible Bounding Box
- ROI-Pooling -- Feature Aligner
- MAP -- Evaluation Metric
- IOU

One-Stage(YOLO)



One-Stage(YOLO)



One-Stage(YOLO)

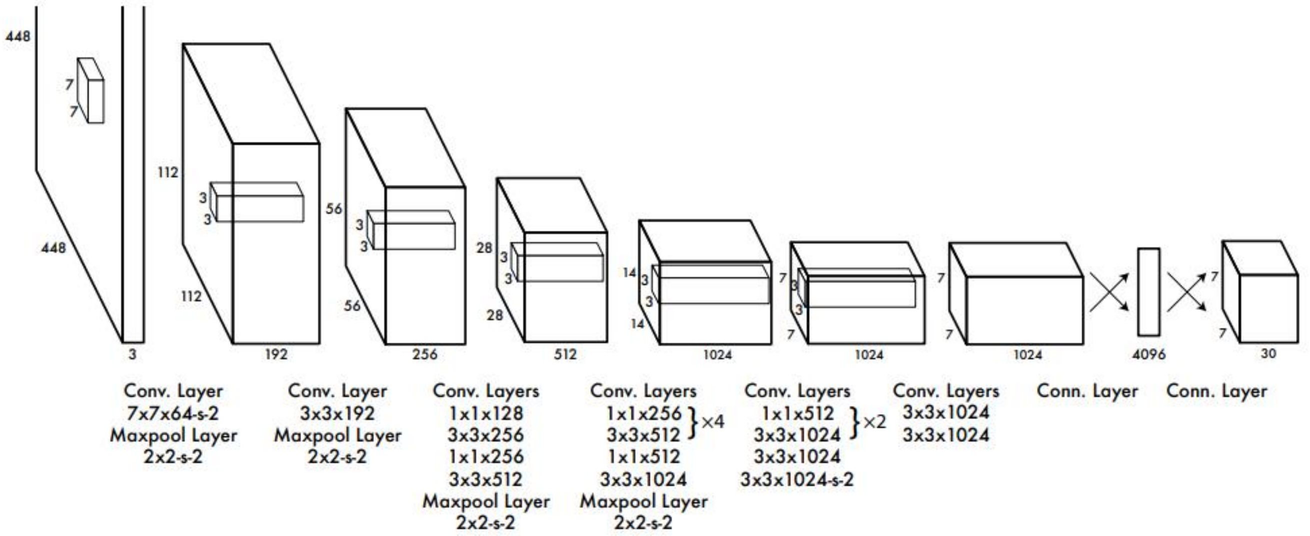
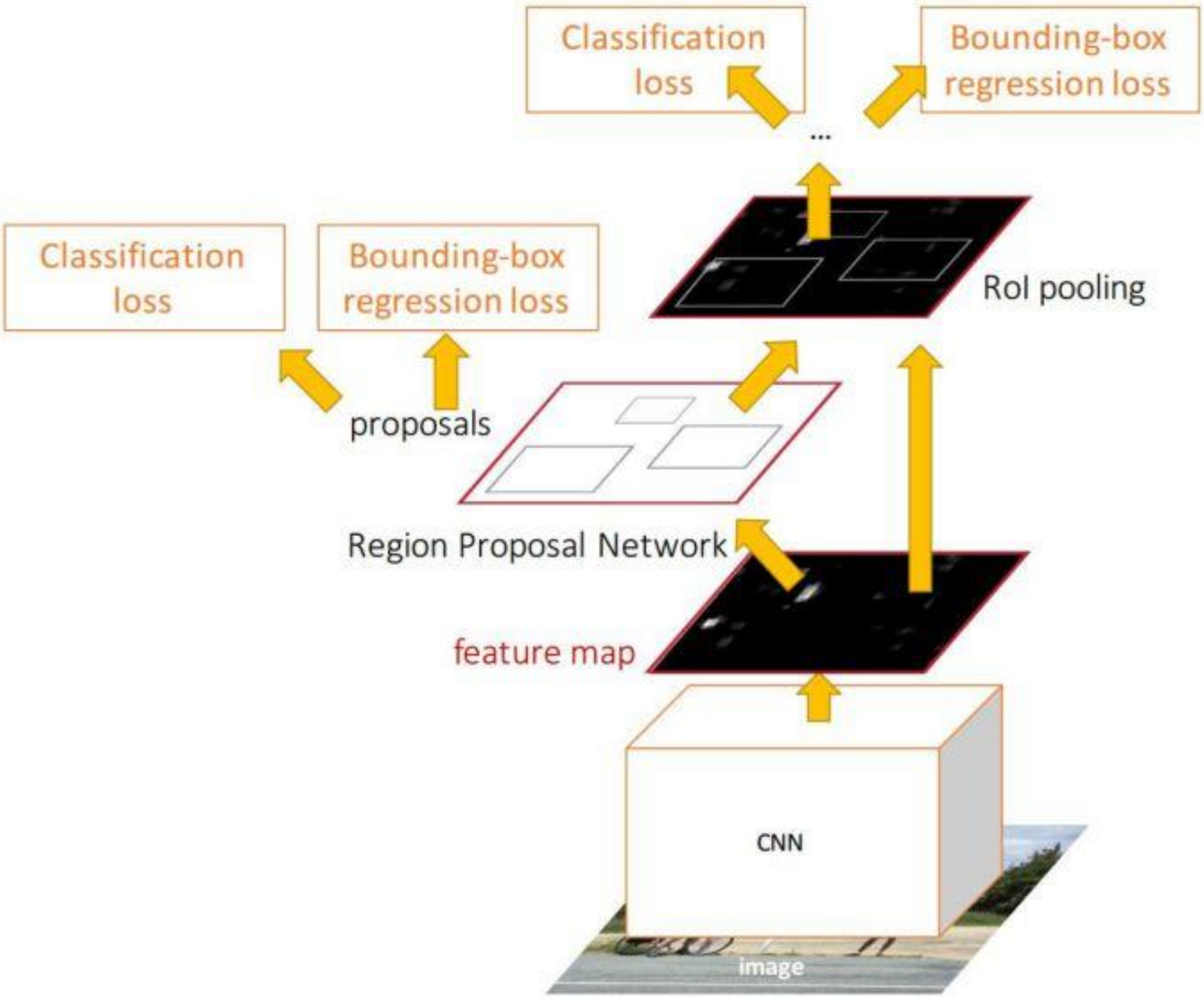
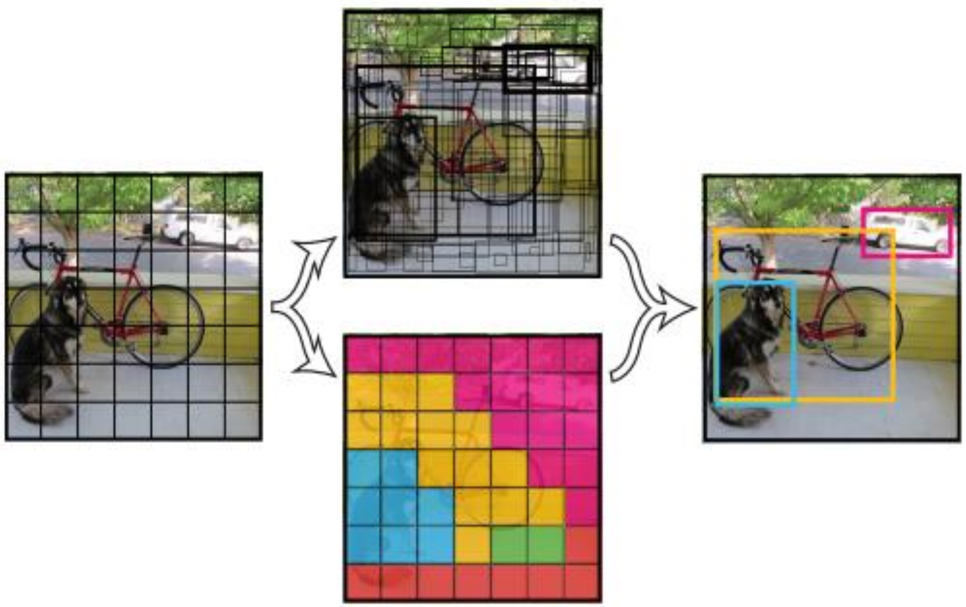


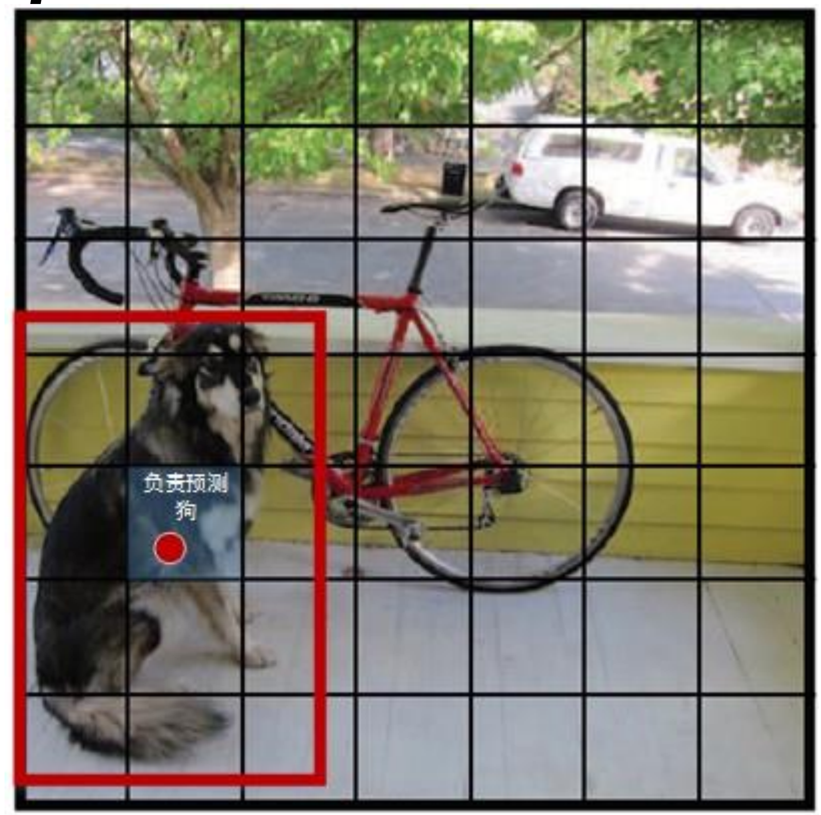
Figure 3: The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1×1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224×224 input image) and then double the resolution for detection.

Two-Stage(FRCNN)

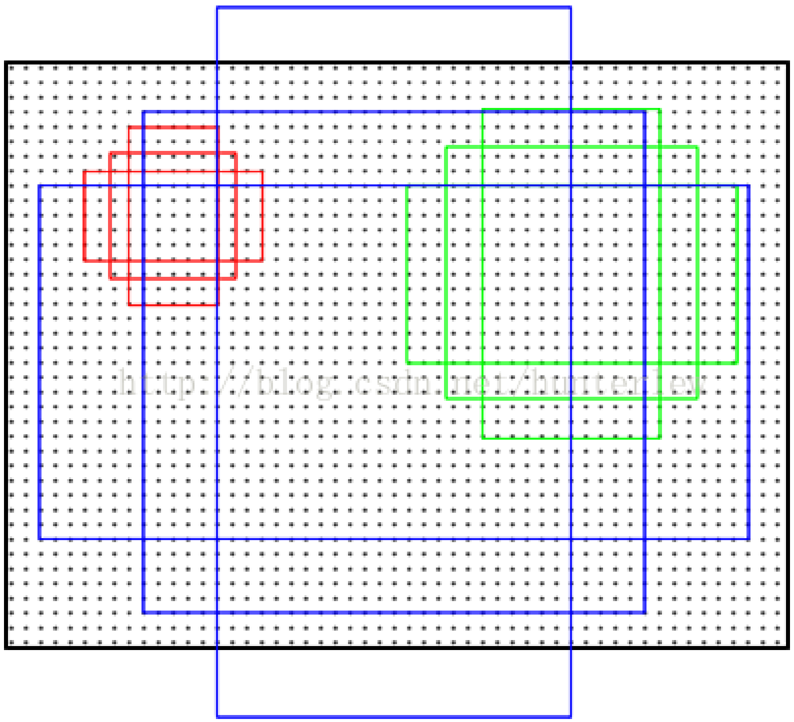


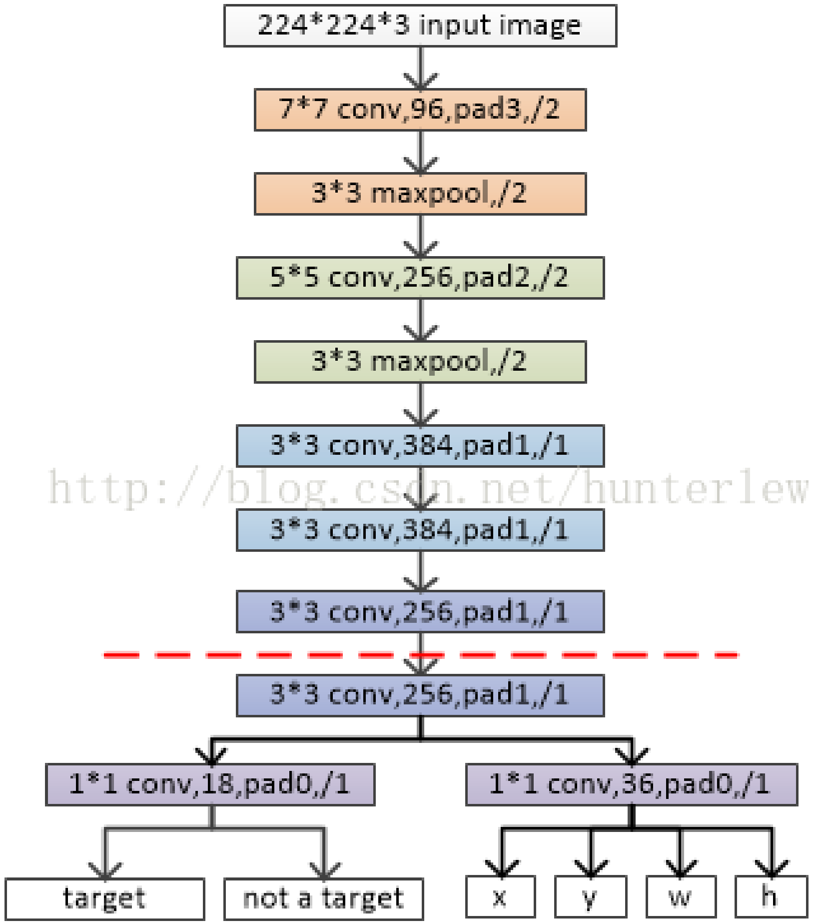
Anchor/Grid(YOLO)



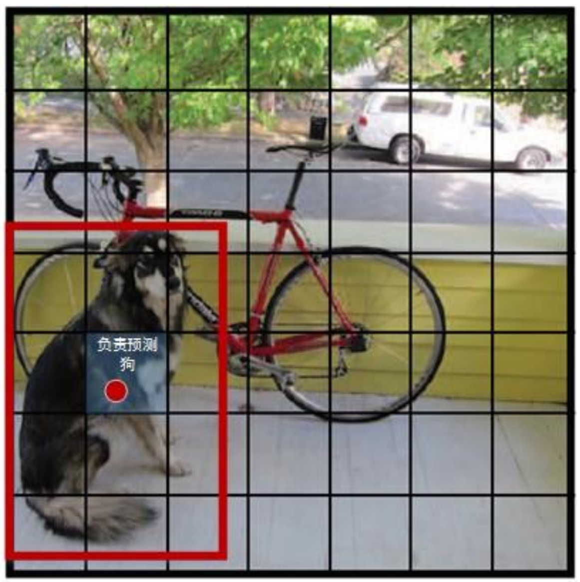


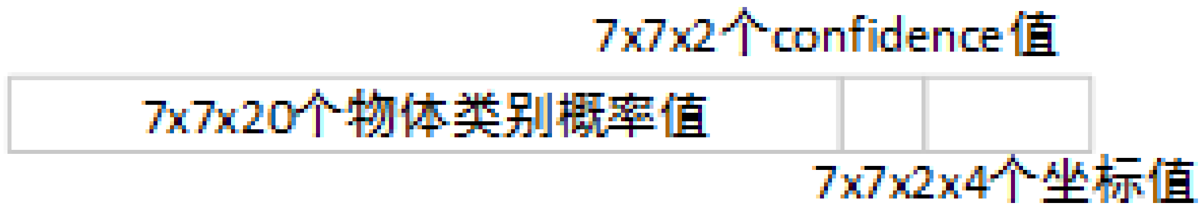
Anchor(FRCNN)



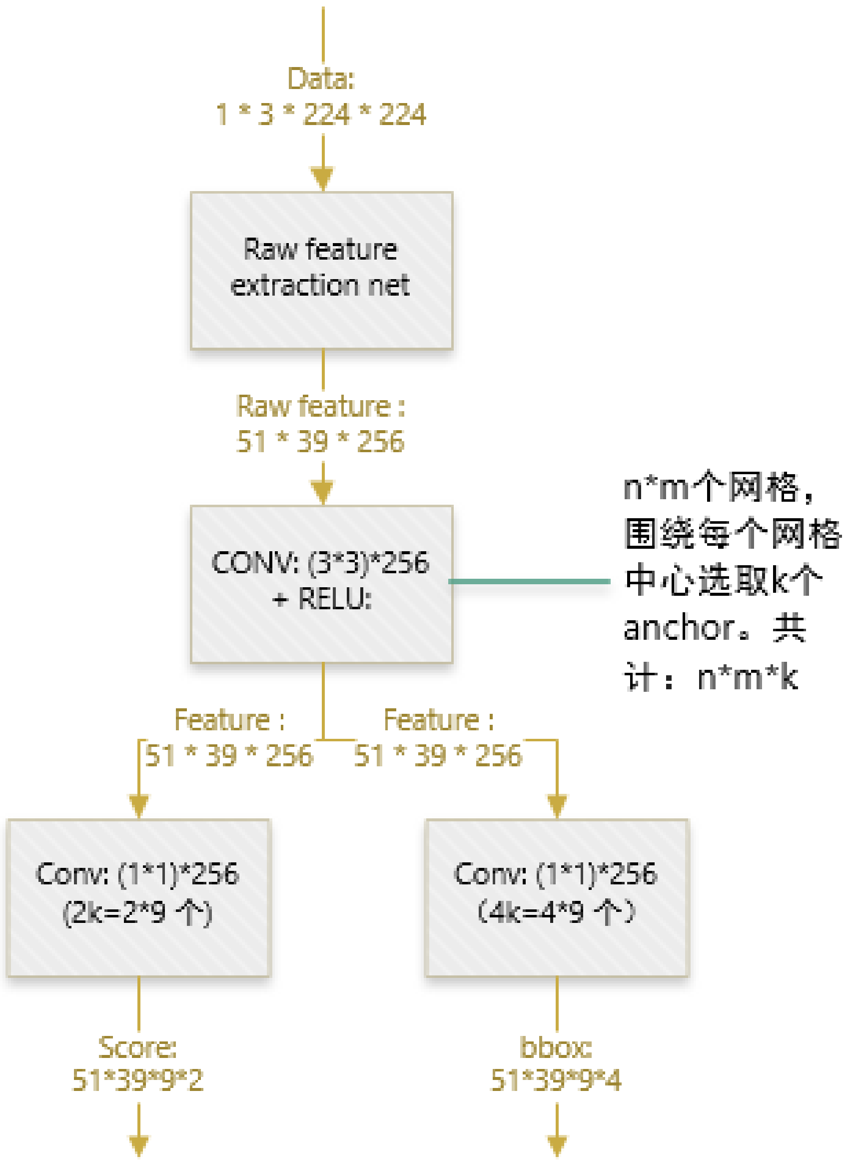


Output(YOLO)

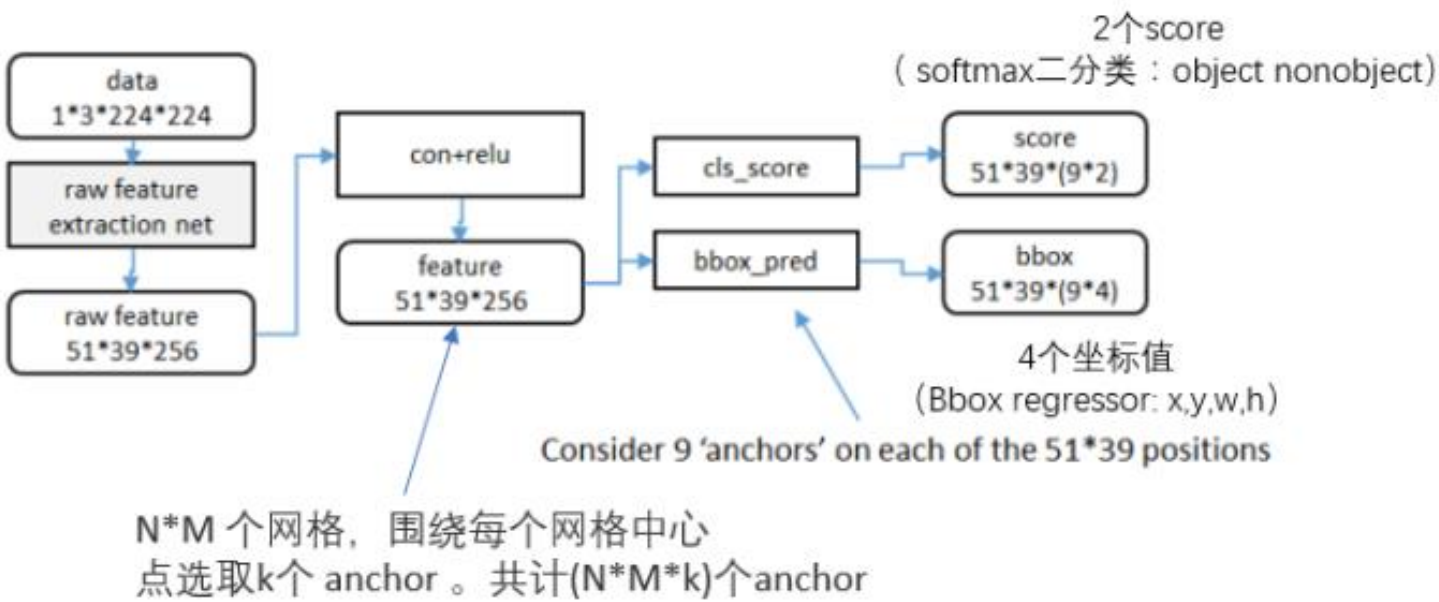




Output(FRCNN)



Output(FRCNN)



LOSS(YOLO)

$$loss = \sum_{i=0}^{S^2} coordError + iouError + classError$$

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

坐标误差

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

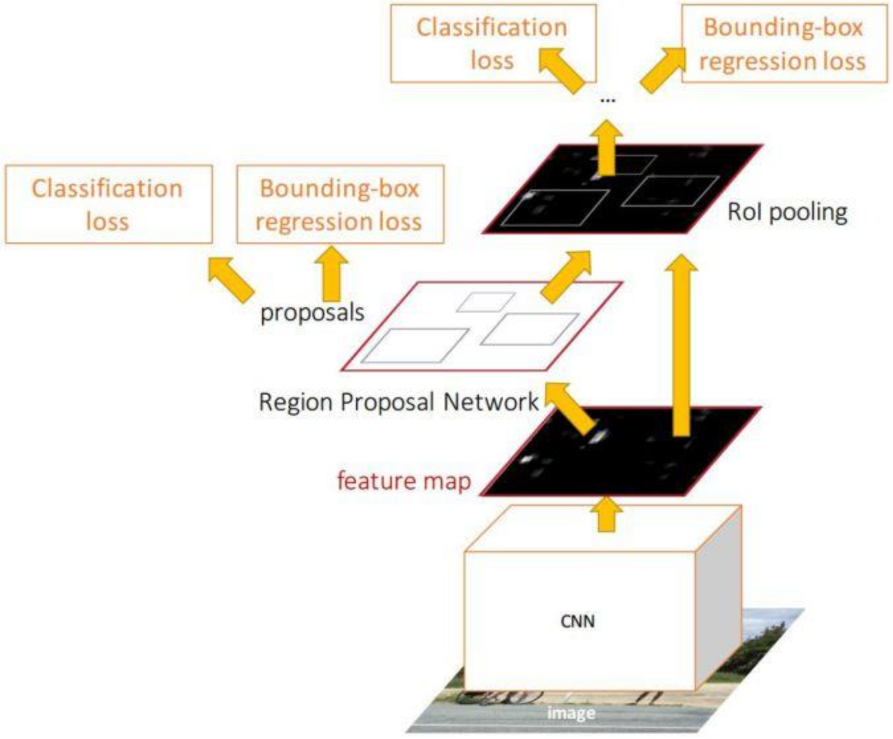
IOU误差

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2$$

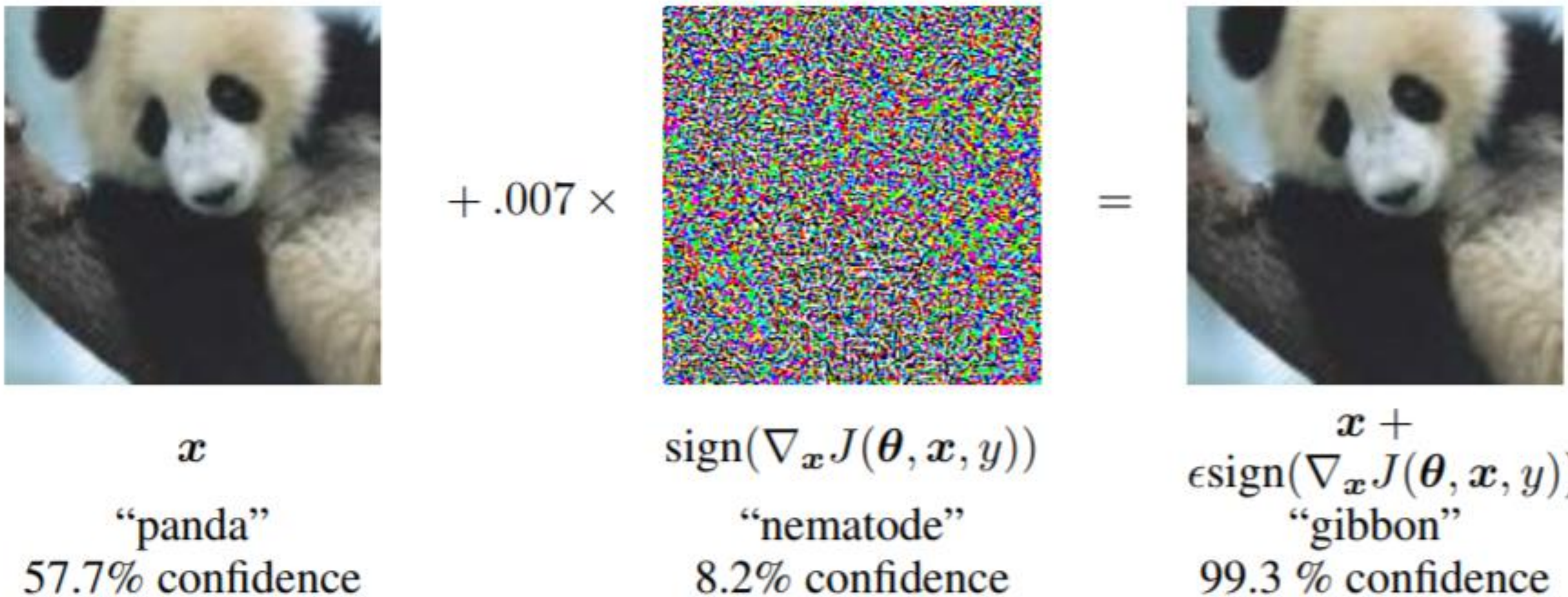
分类误差

LOSS(FRCNN)

$$L(\{p_i\}\{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*)$$



Attack



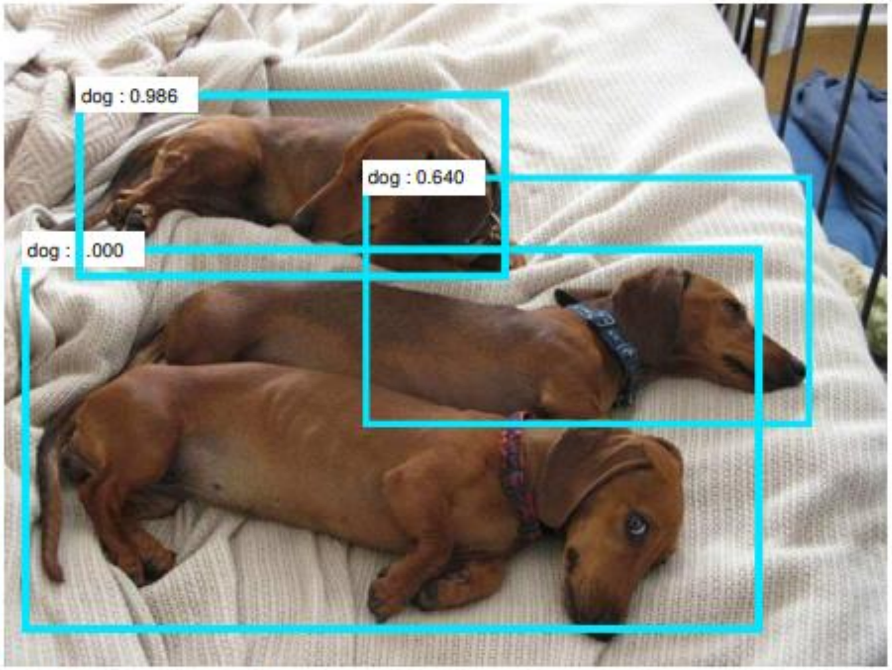
$$\boldsymbol{x}_{adv} = \boldsymbol{x}_{benign} + \varepsilon * sign(\nabla_{\boldsymbol{x}_{benign}} \boldsymbol{J}(\boldsymbol{\theta}, \boldsymbol{x}_{benign}, y))$$

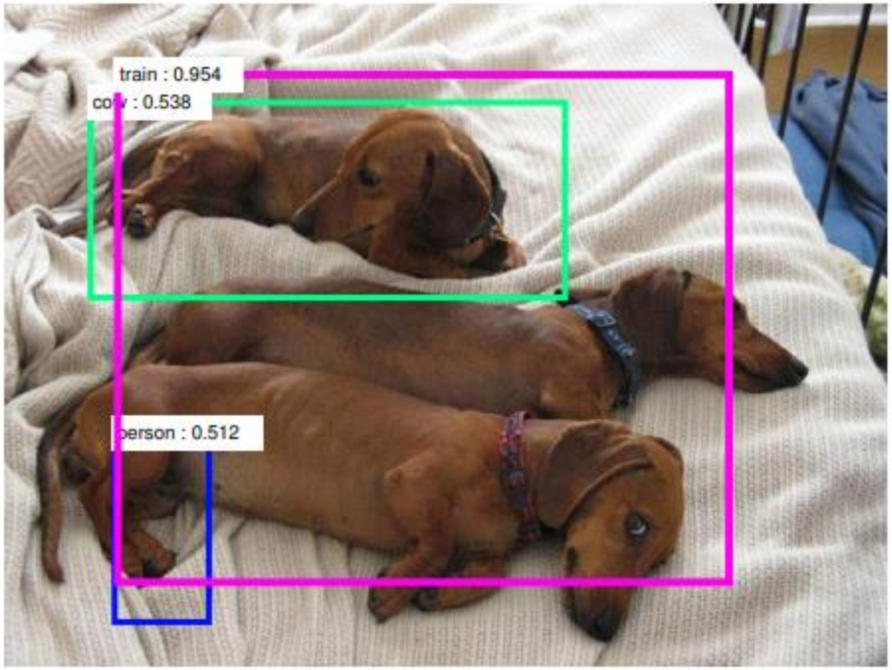
Detection Attack

- Dense Attack
- GAN Attack
- Proposal Attack(Global)
- Proposal Attack(Local)

Dense Attack

Adversarial Examples for Semantic Segmentation and Object Detection





Dense Attack

Adversarial Examples for Semantic Segmentation and Object Detection

- Proposal out of RPN as Target Set
- Change Threshold(IOU) to Increase Number of Proposal
- Preserve All Positive Proposals and Discard Left

Dense Attack

Adversarial Examples for Semantic Segmentation and Object Detection

Adversarial Perturbations from	**FR-ZF-07**	**FR-ZF-0712**	**FR-VGG-07**	**FR-VGG-0712**	**R-FCN-RN50**	**R-FCN-RN101**
None	58.70	61.07	69.14	72.07	76.40	78.06
FR-ZF-07 ($\mathbf{r}_1$)	**3.61**	22.15	66.01	69.47	74.01	75.87
FR-ZF-0712 ($\mathbf{r}_2$)	13.14	**1.95**	64.61	68.17	72.29	74.68
FR-VGG-07 ($\mathbf{r}_3$)	56.41	59.31	**5.92**	48.05	72.84	74.79
FR-VGG-0712 ($\mathbf{r}_4$)	56.09	58.58	31.84	**3.36**	70.55	72.78
$\mathbf{r}_1 + \mathbf{r}_3$	**3.98**	21.63	**7.00**	44.14	68.89	71.56
$\mathbf{r}_1 + \mathbf{r}_3$ (permute)	58.30	61.08	68.63	71.82	76.34	77.71
$\mathbf{r}_2 + \mathbf{r}_4$	13.15	**2.13**	28.92	**4.28**	63.93	67.25
$\mathbf{r}_2 + \mathbf{r}_4$ (permute)	58.51	61.09	68.68	71.78	76.23	77.71

Table 2: Transfer results for detection networks. **FR-ZF-07**, **FR-ZF-0712**, **FR-VGG-07** and **FR-VGG-0712** are used as four basic models to generate adversarial perturbations, and **R-FCN-RN50** and **R-FCN-RN101** are used as black-box models. All models are evaluated on the **PascalVOC-2007** test set and its adversarial version, which both has 4952 images.

GAN Attack

Transferable Adversarial Attacks for Image and Video Object Detection

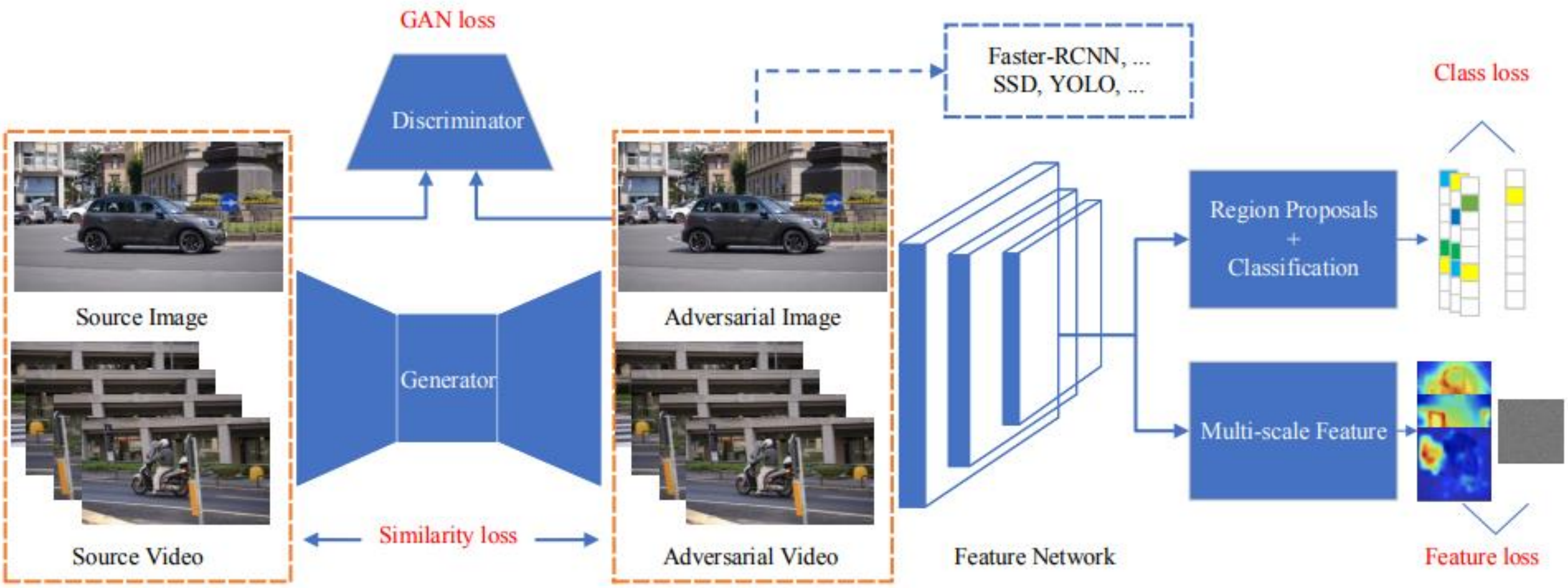


Figure 2. The overall framework of our Unified and Efficient Adversary (UEA). We formulate DAG's high-level class loss with the proposed low-level multi-scale feature loss into GAN framework to jointly train a better generator. For the coming images or video frames, the generator is to output the corresponding adversarial images or frames to simultaneously fool the different kinds of object detectors.

GAN Attack

Transferable Adversarial Attacks for Image and Video Object Detection

$$\mathcal{L} = \mathcal{L}_{cGAN} + \alpha\mathcal{L}_{L_2} + \beta\mathcal{L}_{DAG} + \epsilon\mathcal{L}_{Fea}, \qquad (5)$$

$$\mathcal{L}_{cGAN}(\mathcal{G}, \mathcal{D}) = \mathbb{E}_I[log\mathcal{D}(I)] + \mathbb{E}_I[log(1 - \mathcal{D}(\mathcal{G}(I)))], \qquad (1)$$

$$\mathcal{L}_{L_2}(\mathcal{G}) = \mathbb{E}_I[||I - \mathcal{G}(I)||_2]. \qquad (2)$$

GAN Attack

Transferable Adversarial Attacks for Image and Video Object Detection

$$\mathcal{L}_{DAG}(\mathcal{G}) = \mathbb{E}_I\left[\sum_{n=1}^{N}[f_{l_n}(\mathbf{X}, t_n) - f_{\hat{l}_n}(\mathbf{X}, t_n)]\right], \quad (3)$$

where $\mathbf{X}$ is the extracted feature map from the feature network of Faster-RCNN on I, and $\tau = \{t_1, t_2, ..., t_N\}$ is the set of all proposal regions on $\mathbf{X}$. The symbol t_n is the n-th proposal region from the Region Proposal Network (RPN). l_n is the ground-truth label of t_n, and $\hat{l}_n$ is the wrong label randomly sampled from other incorrect classes.

GAN Attack

Transferable Adversarial Attacks for Image and Video Object Detection

DAG loss function is specially designed for attacking Faster-RCNN, therefore its transferability to other kinds of models is weak. To address this issue, we propose the following multi-scale feature loss:

$$\mathcal{L}_{Fea}(\mathcal{G}) = \mathbb{E}_I[\sum_{m=1}^{M} ||\mathbf{X}_m - \mathbf{R}_m||_2], \qquad (4)$$

where $\mathbf{X}_m$ is the extracted feature map in the m-th layer of the feature network. $\mathbf{R}_m$ is a randomly generated feature map, and its size is the same with $\mathbf{X}_m$. Eq.(4) enforces the random permutation of feature maps. In the experiments, we choose the Relu layer after conv3-3 and the Relu layer after conv4-2 in VGG16 to destroy their feature maps.

GAN Attack

Transferable Adversarial Attacks for Image and Video Object Detection

Table 2. The comparison results between DAG and UEA versus three aspects.

Methods	FR	SSD	SSIM	Time(s)
Clean Images	0.70	0.68	1.00	\
DAG	0.05	0.64	**0.98**	9.3
UEA	**0.05**	**0.28**	0.81	**0.01**

Proposal Attack(Global)

Robust Adversarial Perturbation on Deep Proposal-based Models

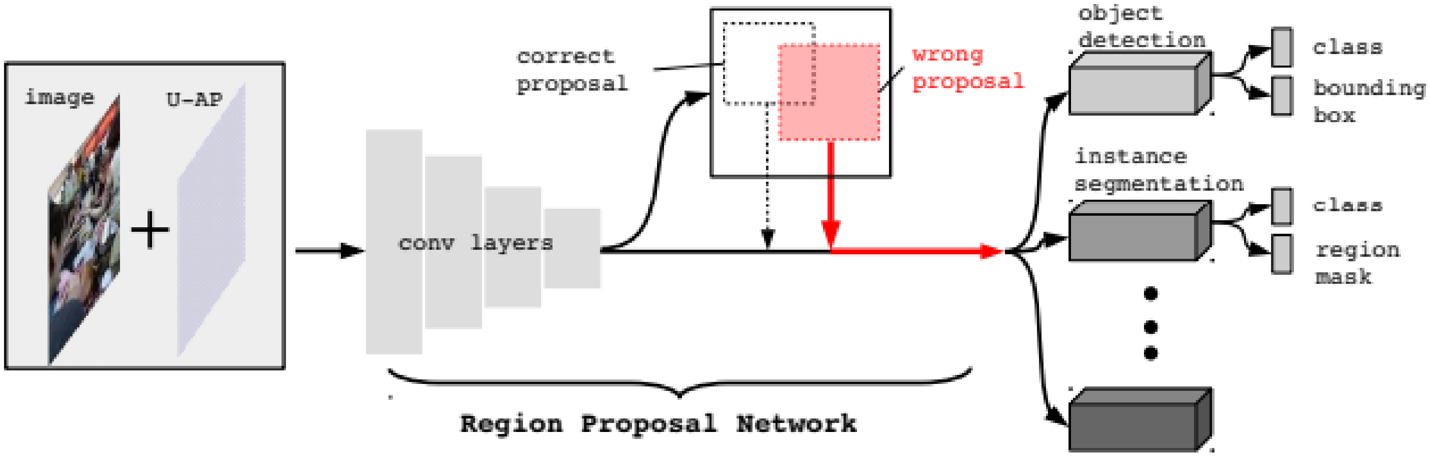
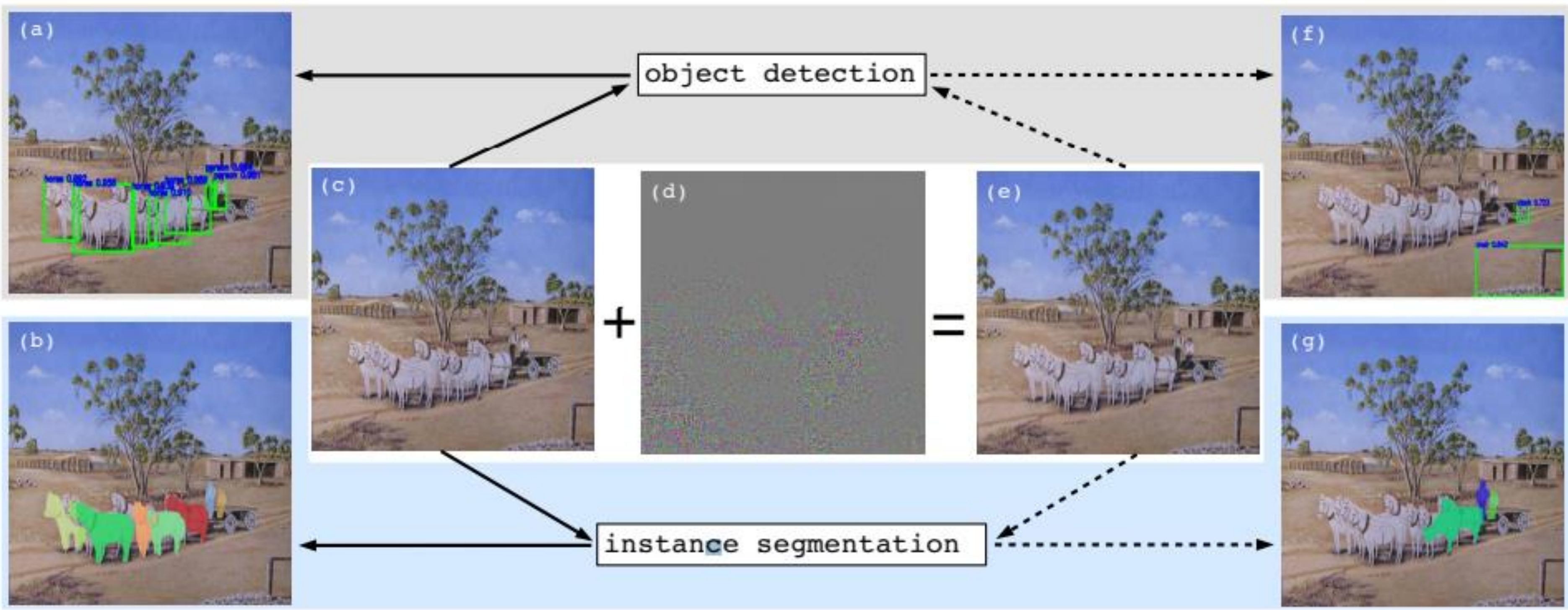


Fig. 1. *Overview of the Robust Adversarial Perturbation (R-AP) method. Our method attacks Region Proposal Network (RPN) [11] in deep proposal-based object detectors and instance segmentation algorithms.*

Proposal Attack(Global)

Robust Adversarial Perturbation on Deep Proposal-based Models



Proposal Attack(Global)

Robust Adversarial Perturbation on Deep Proposal-based Models

$$\min_{\mathcal{I}} \quad L_{label}(\mathcal{I};\mathcal{F}_\theta) + L_{shape}(\mathcal{I};\mathcal{F}_\theta), \;\; \text{s.t.} \;\; \text{PSNR}(\mathcal{I}) \geq \epsilon, \tag{1}$$

where $\text{PSNR}(\mathcal{I})$ denotes the PSNR of luminance channel in image $\mathcal{I}$, ϵ is the lower bound of PSNR. We describe the label loss L_{label} and shape loss L_{shape} in sequel.

$$L_{label}(\mathcal{I};\mathcal{F}_\theta) = \sum_{j=1}^{m} z_j \log(s_j). \tag{2}$$

In other words, minimizing this loss is equivalent to decreasing confidence score of positive proposals.

we define a new loss function L_{shape} as

$$L_{shape}(\mathcal{I};\mathcal{F}_\theta) = \sum_{j=1}^{m} z_j((\Delta x_j - \tau_x)^2 + (\Delta y_j - \tau_y)^2 + (\Delta w_j - \tau_w)^2 + (\Delta h_j - \tau_h)^2), \tag{3}$$

where $\tau_x, \tau_y, \tau_w, \tau_h$ are large offsets defined to substitute the real offset between anchor boxes and matched ground truth bounding boxes. We are only concerned

Proposal Attack(Global)

Robust Adversarial Perturbation on Deep Proposal-based Models

---

Algorithm 1 *Adversarial Perturbation Generation*

Require: RPN model $\mathcal{F}_\theta$; input image $\mathcal{I}$; maximal iteration number T.

1: $\mathcal{I}_0 = \mathcal{I}, t = 0$;
2: **while** $t < T$ and $\sum_{j=1}^{m} z_j \neq 0$ **do**
3: $\quad \hat{p}_t = \nabla_{\mathcal{I}_t}(L_{label} + L_{shape})$;
4: $\quad p_t = \frac{\lambda}{||\hat{p}_t||_2} \cdot \hat{p}_t$; $\triangleright$ *λ is a fixed scale parameter*
5: $\quad \mathcal{I}_{t+1} = \text{clip}(\mathcal{I}_t - p_t)$;
6: $\quad$ **if** $\text{PSNR}(\mathcal{I}_t) < \varepsilon$ **then**
7: $\quad\quad$ break
8: $\quad t = t + 1$;
9: $p = \mathcal{I}_t - \mathcal{I}_0$;

Ensure: adversarial perturbation p

Proposal Attack(Global)

Robust Adversarial Perturbation on Deep Proposal-based Models

Table 1. *Performance of R-AP on 6 state-of-the-art object detectors at mAP 0.5 and 0.7. Lower value denotes better attacking performance.*

	FR-v16	**FR-mn**	**FR-rn50**	**FR-rn101**	**FR-rn152**	**RFCN** [12]
origin	59.2/47.3	47.1/32.6	59.5/49.4	63.5/53.6	64.8/54.5	60.1/50.0
random	58.7/46.5	46.5/32.6	59.6/48.9	63.2/53.2	64.6/54.4	59.9/49.6
v16 (p_1)	**5.1/3.1**	34.8/22.2	47.9/36.8	52.7/42.4	55.5/45.0	54.5/43.8
mn (p_2)	56.8/44.4	**11.0/6.1**	56.7/45.2	60.6/50.2	62.3/51.4	57.5/46.6
rn50 (p_3)	53.8/41.2	39.5/25.7	**10.5/6.6**	52.8/42.2	55.9/44.7	53.7/42.6
rn101 (p_4)	54.8/42.6	41.0/27.4	50.0/39.2	**16.8/11.0**	56.0/45.3	52.0/40.4
rn152 (p_5)	55.0/41.9	41.8/27.4	49.8/38.3	53.6/42.2	**17.3/10.6**	54.4/42.9
$\mathbf{P} = \alpha \cdot \sum_{i=1}^{5} p_i$	37.5/25.6	26.4/16.5	31.3/21.3	37.9/27.2	41.4/30.1	**47.0/35.9**

Proposal Attack(Local)

DPATCH: An Adversarial Patch Attack on Object Detectors

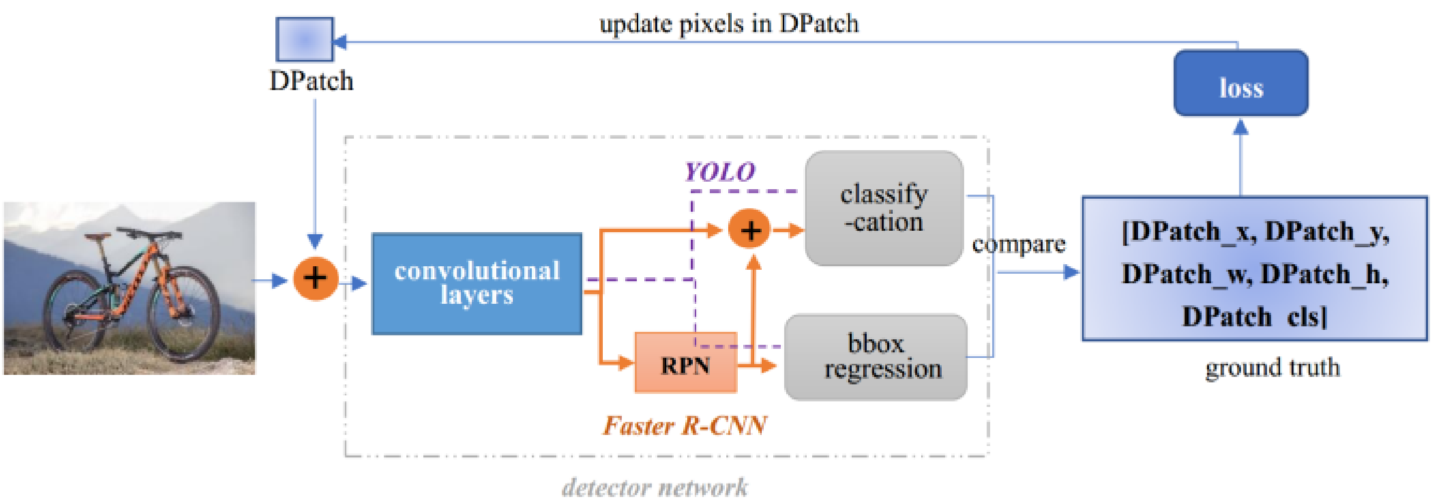


Figure 2: DPATCH training system: we add a randomly-initialized DPATCH to the image, utilize the detector network to do classification and bounding box regression based on the ground truth **[DPATCH_x, DPATCH_y, DPATCH_w, DPATCH_h, target_label]**. During back-propagation, we update the pixels of DPATCH.

Proposal Attack(Local)

DPATCH: An Adversarial Patch Attack on Object Detectors

F-RCNN: make the region where the DPATCH exists as the only valid RoI, while other potential proposal should be considered not to own an object and thus, ignored.

YOLO: the grid containing a DPATCH has higher confidence score than others with normal objects.

Proposal Attack(Local)

DPATCH: An Adversarial Patch Attack on Object Detectors

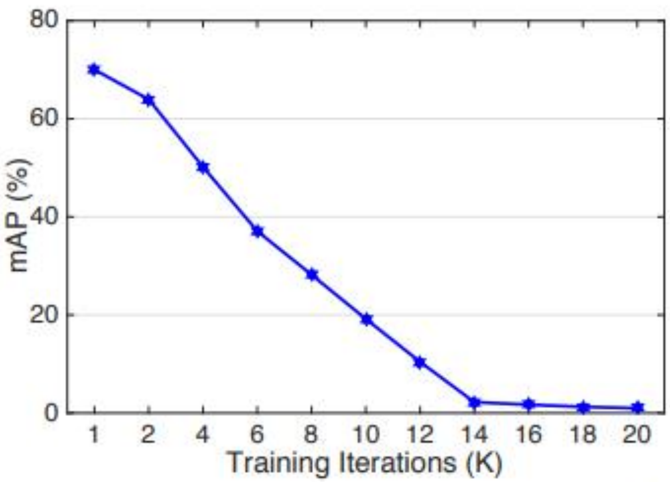


Figure 6: As training iterations accumulate, the falling speec of mAP gradually slow down, meaning the attack effects o DPATCH will saturate at a point. For tv, the saturate point i approximately 200k training iterations.

(a) targeted DPATCH attacking Faster R-CNN

(b) targeted DPATCH attacking YOLO

Proposal Attack(Local)

DPATCH: An Adversarial Patch Attack on Object Detectors

Table 1: Results on Pascal VOC 2007 test set with Fast R-CNN and ResNet101 when applying DPATCH of different types

Faster R-CNN	plane	bike	bird	boat	bottle	bus	car	cat	chair	cow	table
no DPATCH	74.80	80.20	77.60	64.50	61.50	81.10	86.70	86.40	55.70	89.30	69.60
untargeted DPATCH	0.10	3.20	4.30	0.00	5.40	0.00	9.80	0.00	11.20	10.60	5.20
targeted DPATCH	0.02	0.00	0.00	0.00	0.00	0.53	0.08	0.61	0.00	0.02	0.00
YOLO trained DPATCH	2.27	**0.51**	**0.87**	2.27	**0.78**	1.52	4.55	0.62	1.17	3.03	2.10
	dog	horse	motor	person	plant	sheep	sofa	train	tv	mAP	
	87.40	84.50	80.00	78.60	47.70	76.00	74.60	76.60	73.70	75.10	
	0.30	0.59	0.00	1.69	0.00	4.68	0.00	0.00	1.00	**2.90**	
	9.09	0.16	0.00	9.09	0.16	0.00	9.09	0.00	0.00	**0.98**	
	2.02	3.37	1.30	0.94	0.53	0.43	3.03	1.52	1.52	**1.72**	

Table 2: Results on Pascal VOC 2007 test set with YOLO when applying DPATCH of different types

YOLO	plane	bike	bird	boat	bottle	bus	car	cat	chair	cow	table
no DPATCH	69.50	75.60	64.00	52.30	35.60	73.40	74.00	79.60	42.10	66.10	66.90
untargeted DPATCH	0.00	1.50	9.10	1.30	9.10	0.00	9.10	0.00	9.10	9.10	0.40
targeted DPATCH	0.00	4.55	9.09	0.00	0.09	0.00	9.09	1.82	0.01	0.00	0.36
Faster R-CNN trained DPATCH	0.01	**0.00**	**0.23**	0.02	**0.00**	0.00	0.00	0.00	0.01	0.00	0.00
	dog	horse	motor	person	plant	sheep	sofa	train	tv	mAP	
	78.10	80.10	78.20	65.90	41.70	62.00	67.60	77.60	63.10	65.70	
	0.00	0.00	0.00	0.00	9.10	9.10	0.00	0.00	1.00	**0.00**	
	0.01	0.00	0.00	1.73	0.00	0.00	1.07	0.00	9.09	**1.85**	
	0.00	0.03	0.00	0.07	0.00	0.00	0.00	0.00	0.01	**0.02**	

Proposal Attack(Local)

Fooling automated surveillance cameras: adversarial patches to attack person detection

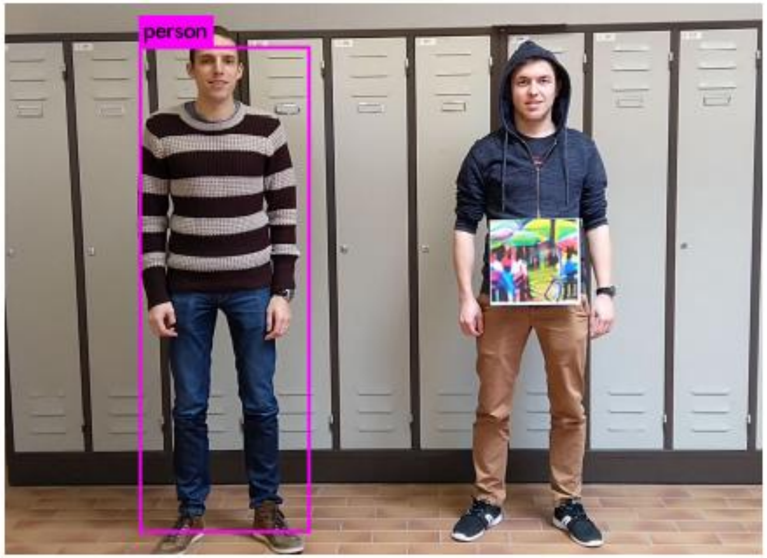


Figure 1: We create an adversarial patch that is successfully able to hide persons from a person detector. Left: The person without a patch is successfully detected. Right: The person holding the patch is ignored.

Proposal Attack(Local)

Fooling automated surveillance cameras: adversarial patches to attack person detection

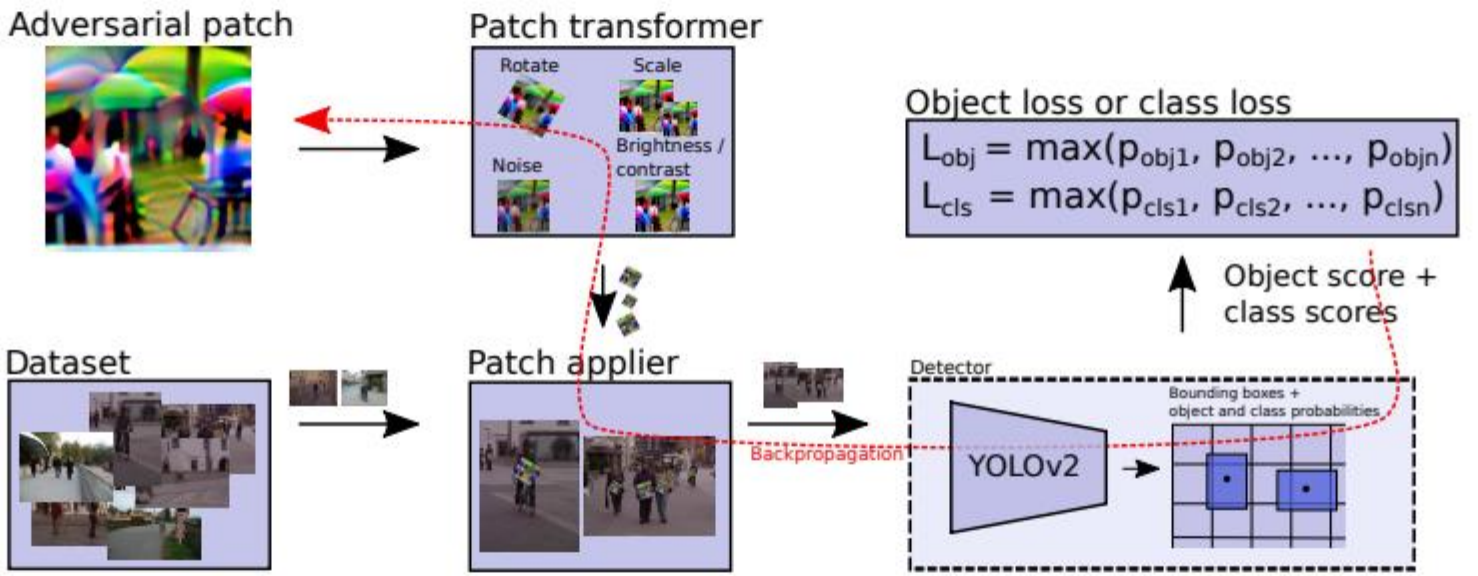


Figure 3: Overview of the pipeline to get the object loss.

Proposal Attack(Local)

Fooling automated surveillance cameras: adversarial patches to attack person detection

- L_{nps} The non-printability score [17], a factor that represents how well the colours in our patch can be represented by a common printer. Given by:

$$L_{nps} = \sum_{p_{\text{patch}} \in p} \min_{c_{\text{print}} \in C} |p_{\text{patch}} - c_{\text{print}}|$$

  Where p_{patch} is a pixel in of our patch P and c_{print} is a colour in a set of printable colours C. This loss favours colors in our image that lie closely to colours in our set of printable colours.

- L_{tv} The total variation in the image as described in [17]. This loss makes sure that our optimiser favours an image with smooth colour transitions and prevents noisy images. We can calculate L_{tv} from a patch P as follows:

$$L_{tv} = \sum_{i,j} \sqrt{((p_{i,j} - p_{i+1,j})^2 + (p_{i,j} - p_{i,j+1})^2}$$

  The score is low if neighbouring pixels are similar, and high if neighbouring pixel are different.

Proposal Attack(Local)

Fooling automated surveillance cameras: adversarial patches to attack person detection

- L_{obj} The maximum objectness score in the image. The goal of our patch is to hide persons in the image. To do this, the goal of our training is to minimize the object or class score outputted by the detector. This score will be discussed in depth later in this section.

Out of these three parts follows our total loss function:

$$L = \alpha L_{nps} + \beta L_{tv} + L_{obj}$$

We take the sum of the three losses scaled by factors α and β which are determined empirically, and optimise using the Adam [10] algorithm.

Proposal Attack(Local)

Fooling automated surveillance cameras: adversarial patches to attack person detection

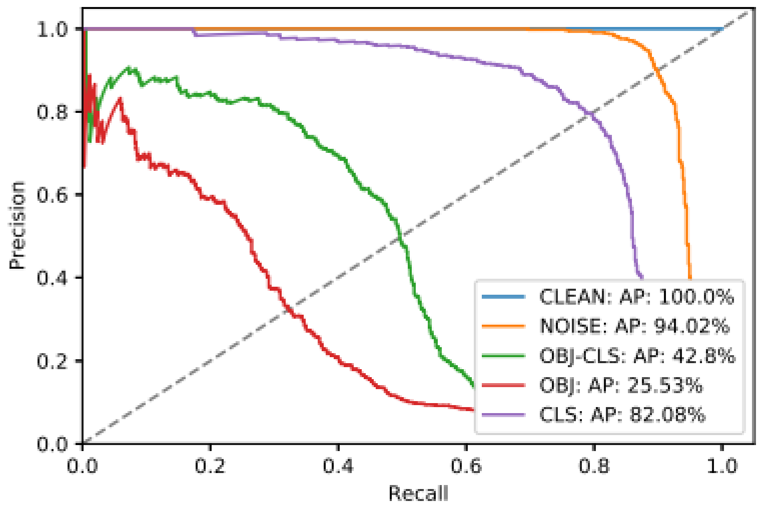


Figure 5: PR-curve of our different approaches (OBJ−CLS, OBJ and CLS), compared to a random patch (NOISE) and the original images (CLEAN).