

Efficient Object Detection in Large Images Using Deep Reinforcement Learning

Burak Uzkent, Christopher Yeh, and Stefano Ermon
Department of Computer Science, Stanford University

Detection in Large Images - Sliding Window

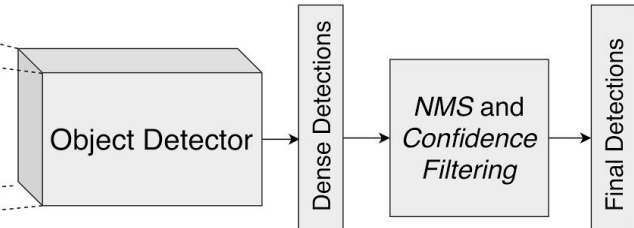
- No need to downsample
- Low memory requirement
- Large runtime to process each window



>1000 pixels

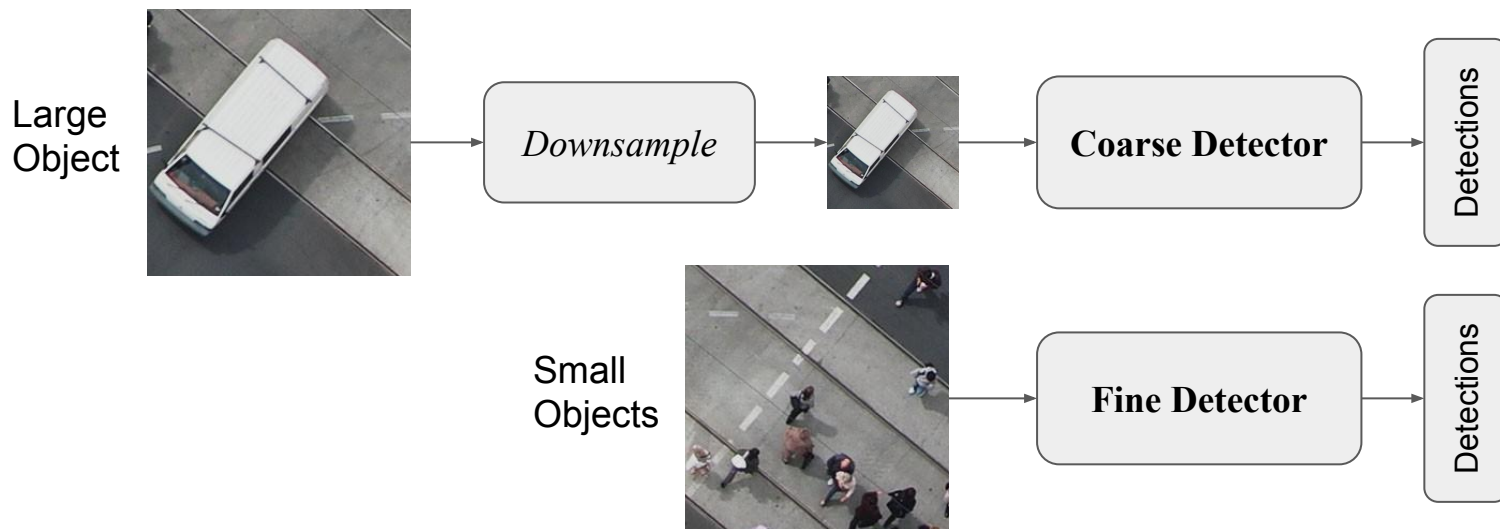


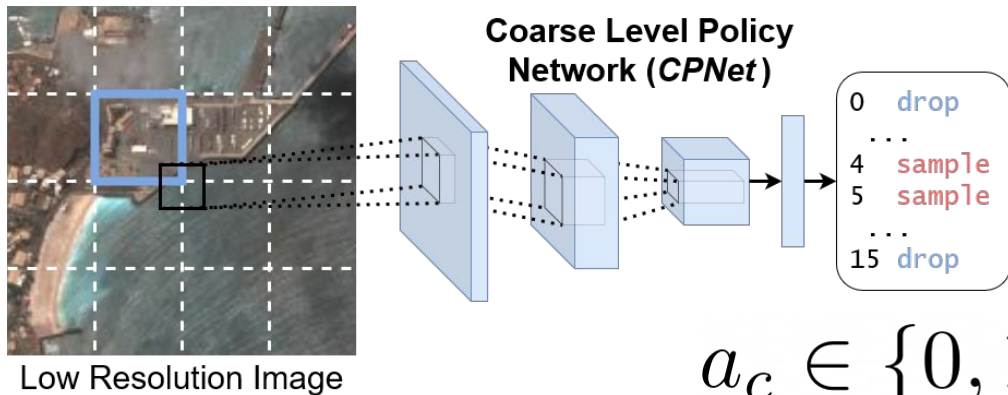
300 - 500 pixels



Proposed Method - Adaptive Sliding Window

Small objects requires fine-level information whereas large objects can be detected at coarse-level.



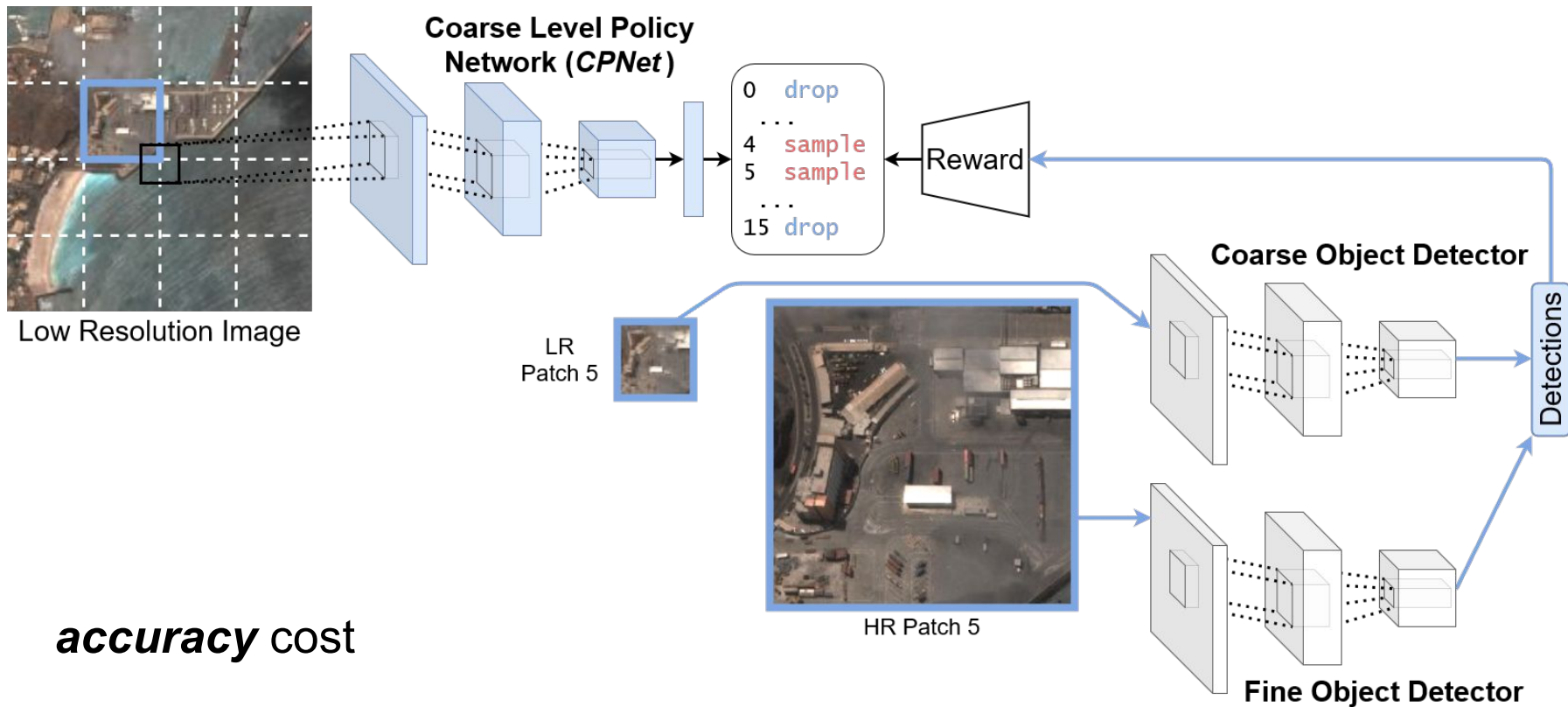


$$a_c \in \{0, 1\}^{P_c}$$

Policy network treats sampling each image patch as a Bernoulli random variable

$$\pi_c(a_c | x_L, \theta_p^c) = \prod_{i=1}^{P_c} (s_c^i)^{a_c^i} (1 - s_c^i)^{(1-a_c^i)}$$

$$s_c = f_p^c(x_L; \theta_p^c)$$



accuracy cost

$$R_{acc} = \sum_{i=1}^{P_c} \left(Recall \left(\hat{Y}_i^f, Y_i \right) - \left(Recall \left(\hat{Y}_i^c, Y_i \right) + \beta \right) \right) \cdot N_i$$

Modeling the Policy Networks

Policy network treats sampling each image patch as a Bernoulli variable

$$\pi_c(a_c | x_L, \theta_p^c) = \prod_{i=1}^{P_c} (s_c^i)^{a_c^i} (1 - s_c^i)^{(1-a_c^i)}$$
$$s_c = f_p^c(x_L; \theta_p^c)$$

Policy network is trained with policy gradient method, with advantage function

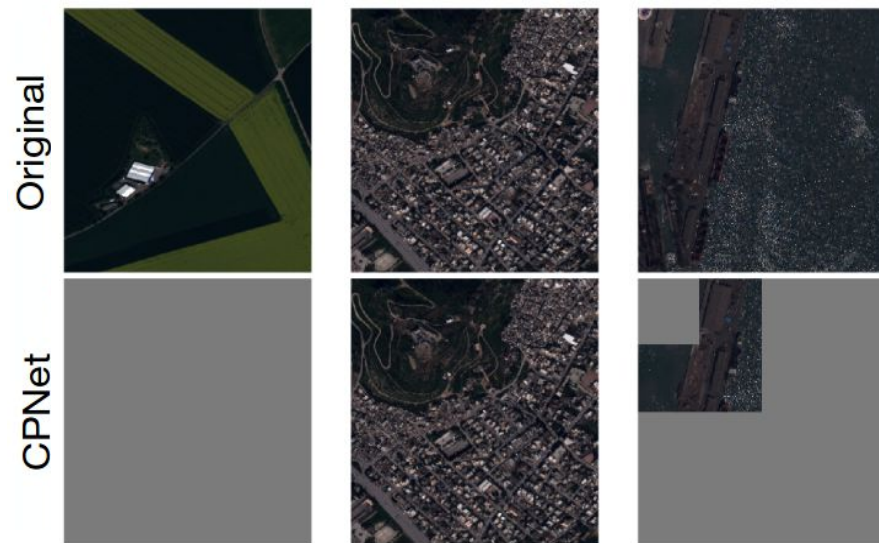
$$J_c = \mathbb{E} [R_c(a_c, a_d, Y)]$$
$$\nabla_{\theta_p^c} J_c = \mathbb{E} \left[R_c \cdot \nabla_{\theta_p^c} \log \pi_{\theta_p^c}(a_c | x_L) \right]$$

Experiments - xView

Experiments on the xView dataset, consisting of 847 very large images (>3000 x >3000 px).

Model/Metric	HR	AP	AR	Run-time	HR
Random (5×)	50	24.1	47.1	1408	31
Entropy (5×)	50	25.4	47.2	1415	31
Sliding Window-L (5×)	0	26.3	39.8	640	0
Sliding Window-H	100	39.0	60.9	3200	100
Gao et al. [7] (5×)	35.4	35.2	55.5	1551	31.6
Ours (5×)	35.5	38.1	59.7	1484	31.5

Table 1 : Results on the xView test set.

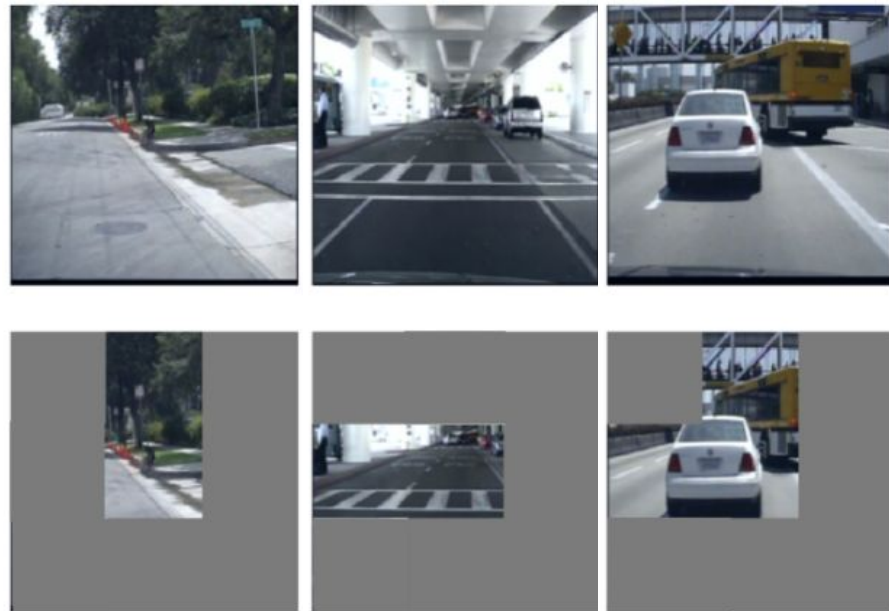


Experiments - Caltech Pedestrian

Experiments on the Caltech Pedestrian dataset ($>800 \times >800$ px).

Model/Metric	AP	AR	Run-time	HR
Random ($\times 5$)	30.9	62.1	248	44.4
Entropy ($\times 5$)	34.0	63.9	250	44.4
Sliding Window-L ($\times 5$)	21.2	46.3	90	0
Sliding Window-H	64.7	74.7	450	100
Gao et al. [7] ($\times 2$)	64.5	73.1	295	7.1
Gao et al. [7] ($\times 5$)	57.3	70.7	309	43.3
CPNet ($\times 2$)	64.4	74.5	267	6.6
CPNet ($\times 5$)	61.7	74.1	270	44.4

Table 2 : Results on the Caltech Pedestrian test set.



Code

<https://github.com/uzkent/EfficientObjectDetection>

uzkent / **EfficientObjectDetection** Watch 2 Star 5 Fork 0

<> Code Issues 0 Pull requests 0 Actions Projects 0 Wiki Security Insights Settings

PyTorch Implementation of Efficient Object Detection in Large Images with RL Edit

[Manage topics](#)

56 commits 1 branch 0 packages 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

uzkent	Readme update	Latest commit 81ae891 4 days ago
dataset	file renaming	6 days ago
figures	README update	6 days ago
utils	Policy Network Update to R18	4 days ago
README.md	Readme update	4 days ago
constants.py	Paths entered, updated readme	4 days ago
environment.yml	Requirements added	6 days ago
train.py	Variable Names Changed in Constants	6 days ago

README.md Edit

Efficient Object Detection in Large Images with Deep Reinforcement Learning

This repository contains PyTorch implementation of our IEEE WACV20 paper on Efficient Object Detection in Large Images with Deep Reinforcement Learning. The arxiv version of the paper can be found [here](#).