DPATCH: An Adversarial Patch Attack on Object Detectors

Xin Liu¹, Huanrui Yang¹, Ziwei Liu², Linghao Song¹, Hai Li¹, Yiran Chen¹

¹Duke University ²The Chinese University of Hong Kong
Outline

• Object Detectors
• Background
  • Adversarial Attack
  • Adversarial Patch
• DPatch Formulation
• DPatch Evaluation
Object Detectors

- **Faster R-CNN**
  - 2-stage object detector
  - First, propose regions through a deep fully convolutional network
  - Second, use the proposed regions for object classification

  ![Faster R-CNN Diagram]

  S.Ren et al, ICCV 2015

- **YOLO**
  - 1-stage object detector
  - Simultaneously predict bounding box and class confidence

  ![YOLO Diagram]

  Joseph. R et al, CVPR 2015

- Both **bbox regression and object classification** should be attacked — hundreds of attack targets.
Adversarial Attacks

- Traditional adversarial attacks
  - Small additive noise perturbing the whole image
  - Need to change every pixel value in the whole image
  - Not practical to be applied in a real-world scene, e.g., web cameras, autonomous vehicles.

Physical-world Adversarial Attacks
Adversarial Patch

- **Original adversarial patch**
  - A patch that can mislead a CNN classifier when added to the scene
  - The classifier outputs the object label with the highest confidence score while ignoring others works well against CNN
  - Object detector will both locate and classify multiple objects, not just the object with the highest confidence.
  - Not effective on object detectors

Brown et al. 2017
**DPatch Formulation**

- **DPatch Training System**
  - Add a randomly initialized DPatch to the input image
  - Utilize the detector network to do object classification and bounding box regression based on the DPatch ground truth
  - Back propagation, update pixel values in DPatch
  - Do not modify anything in detector networks → black-box setup

![Diagram showing DPatch training system](image)
DPatch Formulation

- **Untargeted** Attack
  - Find a patch pattern that maximizes the loss of the object detector to the real class label and bbox label when the DPatch is applied to the input scene.

  \[ \hat{P}_u = \arg \max_{P} \mathbb{E}_{x,s} [L(A(x,s,P); \hat{y}, \hat{B})] \]

- **Targeted** Attack
  - Find a patch pattern that minimizes the loss of the object detector to the targeted DPatch label when the DPatch is applied to the input scene.

  \[ \hat{P}_t = \arg \min_{P} \mathbb{E}_{x,s} [L(A(x,s,P); y_t, B_t)] \]

- **Transferability**
  - Detector architecture and datasets are unknown
  - Transfer among detector networks: YOLO-trained DPatch attacks Faster R-CNN, and vice versa
  - Transfer among datasets: COCO(*more feature*)-trained DPatch attacks Pascal VOC-trained detectors
DPatch Evaluation

- Untargeted Attack on YOLO
  - No DPatch on the left upper corner
  - DPatch (targeted label: tv) on the left upper corner
  - DPatch on the left upper corner
  - The key and U-Disk are detected as person.
  - No extra matrix that simulates the function of the camera during training.
  - DPatch can work well for most cameras.

- Targeted Attack on Faster R-CNN
  - No DPatch on the left upper corner
  - DPatch on the left upper corner
  - DPatch on the left upper corner

- Transferability:
  - YOLO-trained DPatch disables Faster R-CNN

- Physical DPatch Demo
  - The key and U-Disk are detected as person.
## DPatch Evaluation

### Compare with others

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Attack targets</strong></td>
<td>1 (only stop sign)</td>
<td>1 (only stop sign)</td>
<td>All objects in the scene</td>
</tr>
<tr>
<td><strong>Performance</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>untargeted</td>
<td>93%</td>
<td>85.6%</td>
<td>100%</td>
</tr>
<tr>
<td>targeted</td>
<td>87%</td>
<td>N/A</td>
<td>98.15%</td>
</tr>
<tr>
<td><strong>Setup</strong></td>
<td>white-box</td>
<td>white-box</td>
<td>black-box</td>
</tr>
<tr>
<td><strong>Transferability</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>detectors*</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>datasets*</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

* [1] is aimed to attack just Faster RCNN. [2] can be transferred among YOLO and Faster RCNN, but not different datasets. Our DPATCH can not only be transferred among different detectors, but also datasets. The DPatch trained by COCO can successfully attack the detector trained via PASCAL VOC.
