A Systematic Study of Masked Projected Gradient Descent Attack on Object Detection Models

Haiying Huang 804757410 [1] **Xiaojian Chen 905118702** [1] **Zongnan Bao 405711837** [1] **Cheng Lu 405726283** [1]

1. Abstract

We study adversarial attacks on object detection models under constraints on the size and connectivity of the adversarial patches using Masked Projected Gradient Descent (MGDP). Particularly, we study how to construct masks that targets highly sensitive regions of the image while satisfying the constraints. In this project, we implement and evaluate four mask heuristics including Random, Greedy, Beam-Search, and Half-Neighbor masks for attacking two mainstream object detectors (i.e. Fast-RCNN and YOLO), and show that our Greedy and Beam-Search mask have better successful rate than the other two. Finally, we present a new method that, in theory, is able to find the "optimal" mask by reduction to Weighted Model Counting (**WMC**) but may be computationally intractable. Our code can be found at: https://github.com/lc4324/260-PGD.

2. Introduction

Neural Networks are shown to be sensitive, that is, a small, deliberate perturbation on the input data may cause the model to misclassify(Szegedy et al., 2013). However, attacking object detectors can be more difficult due to its two-folded goal: 1) a bounding box around an interested object and 2) a label of the box.

We investigate adversarial attacks on object detectors under limitations on the size and shape of the adversarial patch. The goal is to perturb only a small number of pixels in the image and limit their connectivity, so that the generated adversarial examples are less likely to be defended against or human perceptible. Mask Projected Gradient Descent (MPGD) (Smilkov et al., 2017; Zhang et al., 2021; Madry et al., 2017; Goodfellow et al., 2015) is commonly used for attack in this settings. This method first identifies a mask of the image that meets the patch constraints, and then perform projected gradient descent only on pixels within the mask. However, it remains a challenge how to find a good PGD mask, that is, how to select pixels from the salience map in order to achieve the strongest perturbation effect while satisfying predefined constraints. Previous study has proposed some heuristics for mask generation such as Half-Neighbor Mask (Zhang et al., 2021). However, there is a lack of systematic study and analysis on the mask generation process, which is the focus of this project.

Our main contributions are:

1. Proposed a metric for comparing the performance of different mask heuristics;
2. Theoretically analyze the problem of optimal PGD mask by framing it as a Weighted Model Counting (**WMC**) problem.
3. Develop some other heuristics for PGD masks, including greedy and beam-search masks, and evaluate them on the state-of-the-art object detection models such as R-CNN(S. Ren & Sun, 2017) and Yolo(Redmon et al., 2016).

3. Literature Review

PGD (Madry et al., 2017) has been widely used in the white-box setting for generating attack samples to attack deep learning based classifiers. (Zhang et al., 2021) proposed a variation, Half-Neighbor Masked Projected Gradient Descent(HNM-PGD), to attack object detection models under the white-box setting. Their idea can be divided into two parts as shown in Figure 1

The first part is Finding the mask. They first input the target image to the detection model and use the SmoothGrad (Smilkov et al., 2017) to generate the saliency map, which reflects how sensitive the model is to the specific pixel in terms of making the prediction. Given an input example $\mathbf{X} \in \mathcal{R}^{m \times n}$ and a target model f, the saliency map $S \in \mathcal{R}^{m \times n}$ can be computed as,

$$S(x) = \frac{1}{n} \sum_{i=1}^{n} \nabla_x L(f(x + \eta_i)) \tag{1}$$

They use the saliency map as a basepoint and propose a heuristic function (Half-neighbor algorithm) to analyze the saliency map to gradually reduce the number of perturbation

[1]Department of Computer Science, University of California, Los Angeles.

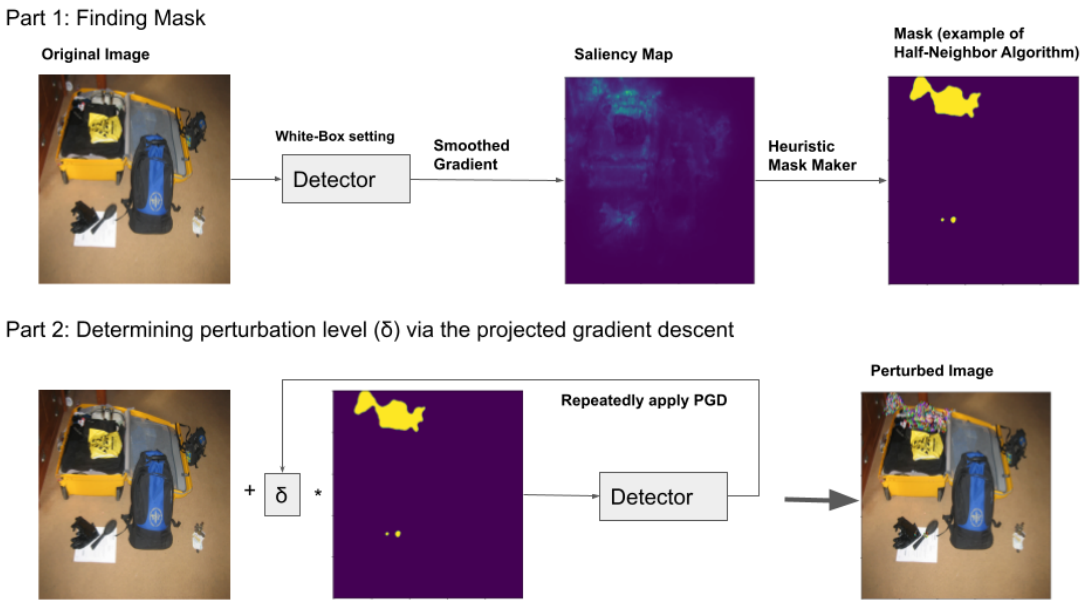


Figure 1. Mask-PGD Pipeline

pixels to meet certain perturbation constraints. The result is a mask such that The bright regions are the masked pixel regions that will be perturbed.

The second part is determining the perturbation level. Once the mask is constructed, by using an iterative approach, they use the Projected gradient descent (PGD) (Madry et al., 2017) to determine the minimum magnitude of the perturbation we need to add to the masked regions. Thus the perturbed image will force the detector to misclassify most of the regions into the background class with high confidence.

4. Method

4.1. Masked Projected Gradient Descent

We consider two types of constraints on the perturbed region following (Zhang et al., 2021):

1. The maximum number of changed pixels (e.g. 2%);
2. The maximum number of 8-connected pixels in the perturbed region (e.g. 10), which is the occurrences that all neighbors of a changed pixel are also changed.

As discussed in section 3, Masked-PGD first identify a mask on the image that satisfy the constraints and then perform PGD on pixels within the mask. In this project, we focus on the mask generation process. Given an image $\mathbf{X}$ and a model f, we can capture the sensitivity of each pixel $x \in \mathbf{X}$ to the final prediction f of the model by computing the salience map S_x in eqn 1. Intuitively, we want to choose a subset of pixels $m \subseteq \mathbf{X}$ such that the total sensitivity of pixels in m is maximized.

$$\widehat{m} = \underset{m \subseteq \mathbf{X}}{\operatorname{argmax}} \sum_{x \in M} S(x) \quad \text{s.t. } |m| = n \text{ and } \mathcal{C}(m) = \texttt{true} \tag{2}$$

where n is the maximum number of changed pixels and $\mathcal{C}$ are connectivity constraints on the perturbed region. Due to the combinatorial natural of connectivity constraints, finding the optimal mask is probably intractable. Therefore, (Zhang et al., 2021) proposes a heuristic Half-Neighbor (HN) mask, which iteratively choose pixels based on whether half of its neighbors are also chosen in the mask. In this project, we propose three simple-and-intuitive heuristics, including Random, Greedy and Beam-Search masks, and apply them for attacking Fast-RCNN(S. Ren & Sun, 2017) and Yolo-v4(Redmon et al., 2016).

4.2. Heuristic Masks

We formally present Greedy, Beam-Search, Random, and Half-Neighbor masks.

Greedy Mask: the greedy method adds pixels to the mask one at a time. During each iteration, it chooses the pixel x with the top salience value among the remaining pixels and check if adding x to the current mask m meets the connectivity constraint. If so, we add x to m; otherwise, pixel x is discarded. We repeat this process until we have a mask with n pixels. The pseudocode is provided in alg 1.

Algorithm 1 GREEDY_MASK($X, S, \mathcal{C}, N$)

Input: image X, salience map S, mask size N, and constraints $\mathcal{C}$
Output: a mask of size N
  $X \leftarrow$ sort(X, key=S)
  $mask \leftarrow [\,]$
  while $mask$.length $< N$ **do**
    $x =$ next(X)
    if $mask \cup \{x\}$ satisfy $\mathcal{C}$ **then**
      $mask$.append(x)
    end if
  end while
  return mask

Algorithm 2 BEAM_SEARCH_MASK($X, S, \mathcal{C}, N, k$)

Input: image X, salience map S, mask size N, constraints $\mathcal{C}$, and beam width k
Output: a mask of size N
  $X \leftarrow$ sort(x, key $= S$)
  $beam \leftarrow [[\,]]$ ▷ empty mask initially
  for i in range(N) **do**
    $NC \leftarrow [\,]$ ▷ new candidates
    for $mask \in beam$ **do**
      $X_2 \leftarrow X \setminus mask$
      NC += $generate\_k\_candidates(mask, X_2, \mathcal{C}, k)$
    end for
    sort NC by total salience value
    $beam \leftarrow NC[1:k]$
  end for
  $mask \leftarrow beam[0]$
  return mask

Beam Search Mask: one limitation of greedy mask is that once a pixel is added to a mask, it can never be removed so it easily leads to a sub-optimal solution. Beam search improves the searching strategy by expanding a set of most promising candidates instead of a single one. At each iteration, beam search maintains the best k candidates, generate k new candidates from each of the existing one, and then keep the best k among those k^2 new candidates for the next iteration.
We can use beam search to approximate the optimal mask under the constraints C. We will use the sum of salience value as score to compare the masks. The pseudocode is provided in alg 2.

Random Mask: Random mask is relatively simple compared to other mask heuristics. We just randomly chooses N pixels into random mask until we have a mask that satisfies the connectivity constraint. It is used as a baseline.

Half-neighbor mask The Half-Neighbor Mask is proposed by (Zhang et al., 2021) based on the half-neighbor heuristics

Algorithm 3 Generate_K_Candidates($mask, X, \mathcal{C}, k$)

Input: current mask $mask$, pixels X, constraints $\mathcal{C}$, and beam width k
Output: k candidate masks
  $NC \leftarrow [\,]$
  while NC.length $< k$ **do**
    $x =$ next(X)
    $candidate \leftarrow mask \cup \{x\}$
    if $candidate$ satisfy $\mathcal{C}$ **then**
      NC.append($candidate$)
    end if
  end while
  Return NC

Algorithm 4 Half Neighbor Mask

Input: A given a saliency map S_x, control parameter ϕ, control step γ
  while M doesn't meets our constraints **do**
    $z_{resp} =$ Mean(S_x) $+ \phi$Std(S_x)
    $M = S_x$
    Define $M' = 0$, where M has a same shape as M
    for $M'_{i,j} \in M'$ **do**
      if $\#(\{M_{i-1,j}, M_{i+1,j}, M_{i,j-1}, M_{i,j+1}\} \geq z_{resp}) \geq 2$ **then**
        $M'_{i,j} = M_{i,j}$
      end if
    end for
    $M = M'$
    $\phi = \phi + \gamma$
  end while
  Set all non-zero $M_{i,j}$ of M to 1.
  Return M

that is, if half of a pixel's neighbors are chosen in the mask, then the pixel should be chosen. The saliency map gives us information about how the detector model responds to each pixel of the input image. So we use it as a starting point of a mask M^0, then repeatedly apply the half neighbor heuristics to reduce the pixels of the M until it meets our constraints.

Note that, in beginning each iteration, we only consider the pixel that the model's responds is larger than the certain threshold $z_{resp} =$ Mean(S_x) $+ \phi$Std(S_x) as a valid chosen pixel of previous mask M^{i-1}. In here, we use ϕ as the control parameter, it will be increased by a control step γ to tight up the threshold. Then is the core idea of the Half-Neighbor algorithm: if half of a pixel's neighbors have been chosen by the previous mask M^{i-1}, we keep this pixel in the current mask M^i, otherwise, we will discard it.

5. Optimal mask: Weighted Model Counting

We present one novel approach which, theoretically, is guaranteed to find the optimal mask as defined in eqn 2. This method attempts to reduce the problem of optimal masks to Weighted Model Counting(Chavira & Darwiche, 2008). Given an image $\mathbf{X}$, a salience map S_x, and a set of constraints $\mathcal{C}$ on the perturbed region, the reduction procedure is as follows:

- Create a boolean variable $v_x \in \mathbf{V}$ to represent whether a pixel $x \in \mathbf{X}$ is chosen to be perturbed (true for selected, otherwise false).
- Assign the weight of each bool variable v_x as the salience value $\log S(x)$.
- Specify constraints $\mathcal{C}$ as a boolean formula f over variables $\mathbf{V}$.
- Find an instantiation $\mathbf{v}$ of that satisfies f and the product of weights for true variables in $\mathbf{v}$ is maximized.

It should be clear now the solution to WMC corresponds to an optimal mask. The real challenge is to *efficiently* encode the constraints $\mathcal{C}$ on the perturbed region as boolean formula. Unfortunately, we discover that this is not always possible. We are able to encode the constraints on maximum number of perturbed pixels as Ordered Binary Decision Diagram (OBDD) of polynomial size. However, to encode the connectivity constraint, we can do no better than enumerating and hardcoding every possible mask.

5.1. Encode Cardinality Constraint

To use the WMC solver, one must first write the constraints as a boolean formula. It is hard to write the cardinality constraint in Conjunctive Normal Form (CNF) or Disjunctive Normal Form (DNF) as we would have to enumerate all possible instantiations. Instead, we propose to encode the cardinatlity constraint in the form of Ordered Binary Decision Diagrams (OBDDs) (RANDAL E, 1992).
Let n be the total number of pixels. Let k be the max number of pixels allowed. To encode the cardinality constraint, construct the following OBDD as shown in Figure 2.

Being at a node in row i column j means that, for all the pixels we have seen so far, $i-1$ of them is 1, $j-1$ of them is 0. If node 1 is reached, the mask passes the cardinality constraint. Otherwise the mask fails the cardinality constraint.

5.2. Encode Connected Region Constraint

We find it very difficult to encode the constraints on 8-connected pixels as boolean formula. One naive method is to enumerate every possible mask $m \subseteq \mathbf{X}$ of size N, and manually count the number of 8-connected pixels. Then we hardcode the masks with less than specified threshold of 8-connected pixels into a Disjunctive Normal Form (DNF). However, this would lead to a DNF of exponential size and totally defeats our original goal for problem reduction.

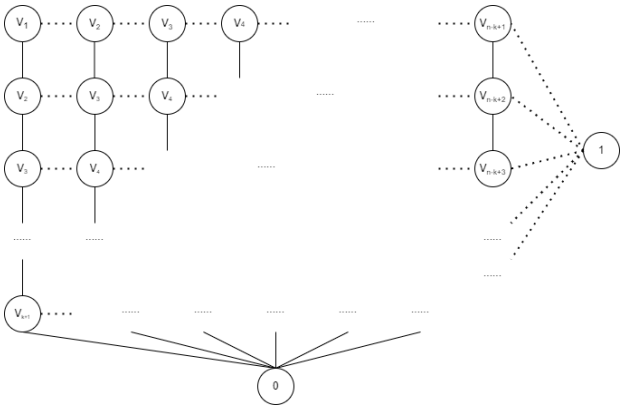


Figure 2. Cardinality Constraint

6. Experiments

6.1. Metric for Mask Performance

We formally define the metric for mask performance as follow:

- By fixing number of perturbed pixels N, we can find the minimum perturbation magnitude δ needed to fool the model using mask returned by a heuristic.
- Plot minimum δ over the number of perturbed pixels N.
- The smaller the AUC (area under the curve), the better the heuristics is.

6.2. Masked-PGD Attack Results

We applied different kinds of mask heuristics on PGD, and attacked two object detection models, YOLO and Faster R-CNN, respectively. As shown in Table 1, given fixed pixel contraint, (e.g. 1% means the mask covers 1% of total number of pixels), we found the minimum pixel perturbation needed to successfully fool the model (i.e. the model outputs 0 detected object) by grid searching. We plotted percentage of pixels against magnitude needed to fool both YOLO and Faster-RCNN in Figure 3. Visualizations of different heuristic masks are shown in Figure 5, along with the original image Figure 4.

As shown in Figure 3 and Table 1, greedy and beam search mask have similar performance and they outperform the

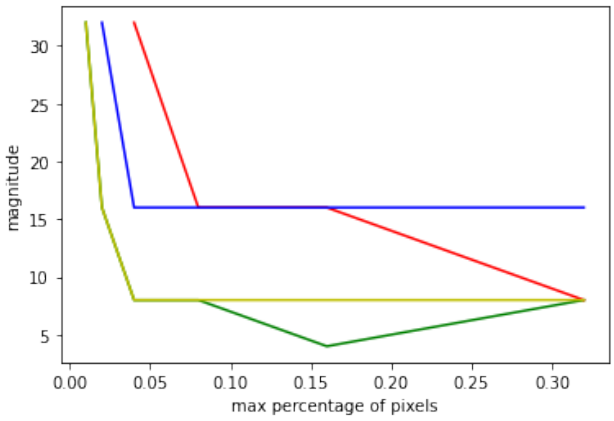
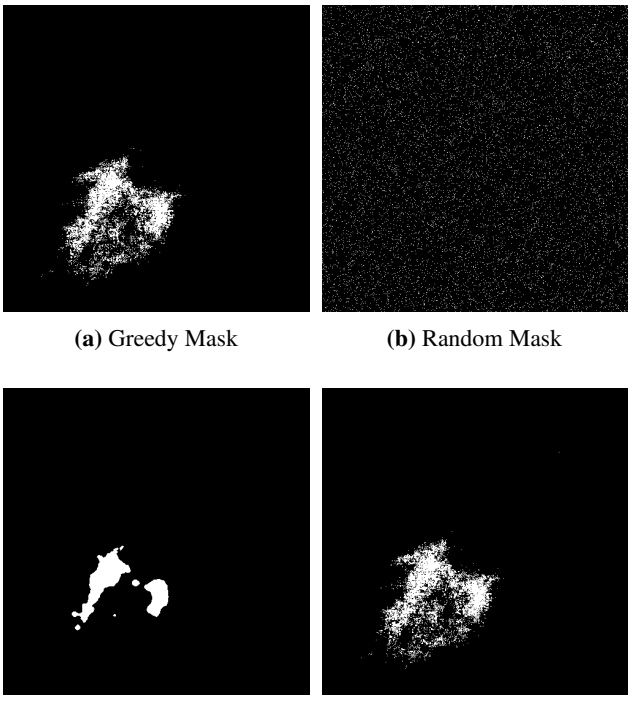


Figure 3. percentage of pixels vs magnitude. green: greedy; red: random; blue: half neighbor; yellow: beam search

Figure 4. Example image



(a) Greedy Mask **(b)** Random Mask

(c) Half-Neighbor Mask **(d)** Beam-Search Mask

Figure 5. Visualizing Masks

other two heuristic. Upon inspecting the visualized masks (Figure 5), we can see that both greedy and beam search mask concentrates on the objects of interest, forcing the PGD attack to focus on objects that YOLO and Faster R-CNN are trying to detect. This could potentially be the reasons why greedy and beam search mask have better performance. An interesting phenomenon worth noting is that, though beam search is the refined version of greedy search, in our experiments greedy mask is slightly better than beam search mask. To be more specific, they have the exact same result under every setting except one, where greedy mask only need a magnitude of $4/255$ to fool Faster R-CNN with 16% pixels while beam search mask needs $8/255$. This is because of the randomness in our projected gradient descent and smooth grad. Therefore this result does not indicate that beam search mask is worse than greedy. The more important problem is, why can't beam search mask outperforms greedy mask even though the former mask is strictly better than the latter. This is mostly because under our patch constraints, the mask generated by beam search is not significantly better than that of greedy search, as shown in Table 2.

The half neighbor mask, though outperformed by greedy and beam search mask, has better performance than random mask when the number of pixels allowed to be perturbed is small. Similar to greedy and beam search mask, half neighbor mask concentrates on the object of interest. But

Mask \ Percent / Victim	Number of perturbed pixels N											
	1%		**2%**		**4%**		**8%**		**16%**		**32%**	
	Yolo	FRCNN	Yolo	FRCNN	Yolo	FRCNN	Yolo	FRCNN	Yolo	FRCNN	Yolo	FRCNN
Greedy	**16/255**	**32/255**	**8/255**	**16/255**	**8/255**	**8/255**	**8/255**	**8/255**	**4/255**	**4/255**	**8/255**	**4/255**
Random	N/A	N/A	64/255	N/A	32/255	32/255	16/255	16/255	8/255	16/255	8/255	8/255
Half-Neighbor	**16/255**	N/A	16/255	32/255	16/255	16/255	16/255	16/255	8/255	16/255	16/255	16/255
Beam-Search	**16/255**	**32/255**	**8/255**	**16/255**	**8/255**	**8/255**	**8/255**	**8/255**	**4/255**	8/255	**8/255**	**4/255**

Table 1. Given pixel constraints, each cell represents the minimum pixel perturbation magnitude (lower is better) needed to successfully fool the victim model. For instance, masks generated using Greedy heuristics can successfully attack YOLO using 8/255 perturbation magnitude under 32% percent pixel constraint. N/A means the attack is failed even using 32/255 pixel perturbation magnitude. (the highest pixel perturbation we used in the grid search process)

Pixels	Greedy	Beam
1%	5679.57	5680.98
2%	8532.56	8533.51
4%	12280.25	12280.83
8%	17377.64	17377.93
16%	19360.34	19360.42
32%	21414.00	21414.02

Table 2. saliency score of masks

instead of picking pixels from the region of interest, half neighbor mask finds a set of connected regions around the object of interest. Due to the way the mask is generated, it is not guaranteed that these connected regions completely overlaps with the object, which could potentially explain why it is worse than greedy/beam search mask. For the same reason, half neighbor mask fails to utilize all the pixels when the number of pixels constraint is relaxed, which is why it is better than random mask at first but worse than random mask when the constraint is relaxed.

7. Conclusion & Future Work

In conclusion, among the four heuristics we evaluated, greedy mask and beam search mask provide the best performance on attacking fast-RCNN and Yolo models. While beam search should be able to produce slightly better mask in terms of salience score, greedy mask is more computationally cheaper and at the same time has similar performance.

As for finding the optimal mask, unfortunately, our proposed method turned out to be computationally intractable. To be more specific, our method failed to encode the connected region constraints as boolean formula, as doing so would cause the size of resulting boolean circuit to blow up. However, our idea of reducing the *Opt-Constrained-PGD-Mask* problem to Weighted Model Counting is still sound. The problem is that we could not implement this specific shape constraint. For future work, we can investigate the possibility of encoding other shape constraint for optimal PGD mask problem.

8. Github

Our code is available at: https://github.com/lc4324/260-PGD.

References

Chavira, M. and Darwiche, A. On probabilistic inference by weighted model counting. *Artificial Intelligence*, 172(6):772–799, 2008. ISSN 0004-3702. doi: https://doi.org/10.1016/j.artint.2007.11.002. URL https://www.sciencedirect.com/science/article/pii/S0004370207001889.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples, 2015.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

RANDAL E, B. Symbolic boolean manipulation with ordered binary-decision diagrams. *ACM Computing Surveys (CSUR)*, 24(3):293–318, 1992. ISSN 0004-3702. doi: https://dl.acm.org/doi/abs/10.1145/136035.136043.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. You only look once: Unified, real-time object detection, 2016.

S. Ren, K. He, R. G. and Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137-1149*, 2017.

Smilkov, D., Thorat, N., Kim, B., Viégas, F., and Wattenberg, M. Smoothgrad: removing noise by adding noise, 2017.

Zhang, Y., Wang, F., and Ruan, W. Fooling object detectors: Adversarial attacks by half-neighbor masks. *arXiv preprint arXiv:2101.00989*, 2021.