Fool Object Detectors with L_0-Norm Patch Attack

Honglin Li[*]
Westlake University
Hangzhou, Zhejiang, China
lihonglin@westlake.edu.cn

Yunqing Zhao
Singapore University of Technology and Design
Singapore
yunqing_zhao@mymail.sutd.edu.sg

ABSTRACT

Deep Neural Networks based Object Detection algorithms have shown their remarkable performance and been widely applied in various aspects in recent years. However, for those areas which draw much attention to the robustness and security of the model, like Autonomous Driving and Biomedical Image Analysis, there are still challenges to make domain users look at these detectors as reliable methods. In this paper, we mainly focus on object detection and propose our method to fool object detectors with with L_0-Norm patch attack.

CCS CONCEPTS

• **Computing methodologies** → **Object detection**; • **Security and privacy**;

KEYWORDS

object detection, adversarial attack, projected gradient descent

1 INTRODUCTION

Deep Neural Networks based computer vision algorithms have shown their remarkable performance but also faced with vulnerability and security problems. Adversarial attack on Image Classification has made significant progress [4], also spurred the advancement of the robustness of the classification models [12], and some attack research on object detection has also been made [16]. To make it deeper, CIKM-2020 and Aliyun-Tianchi host the AnalytiCup workshop competition [3] about generating adversarial sample from COCO [8] to attack 4 mainstream object detection models. In this paper, we propose our solution based on Projected Gradient Descent [10] with l_0-norm towards this competition. To meet the rule of the competition that the modified areas should be as smaller as better and must be constructed as connected domain, we transfer such rule into solvable problem, and make our efforts by k-means and Prim for higher score. Eventually, we get a fine result and rank 10th out of 1701 teams. Code has been made publicly available at our github code repo.

2 BACKGROUND

In this section, we first make description and understanding of the competition, then provide some background knowledge and review the related works about adversarial attack on Object Detection.

2.1 Competition Understanding

The competition adopts two object detectors known by the competitors (known as white-box attacks) and another two black-box object detectors for evaluation. The competitors are asked to generate adversarial examples by adding a small number of patches (less than 10) to each image (also known as L0 attacks) offline.

2.1.1 White-box Models: The 2 white-box model are the famous Faster R-CNN [11] and YOLOv4 [1] respectively. Faster R-CNN is known as a 2-stage detectors. The first stage, or RPN [11], will classify a coarse fore/back-ground binary result for each anchor, then most of the background anchors will be neglected by threshold, and the second stage will make final prediction based on remaining anchors. YOLOv4 is an 1-stage detector, and both of this 2 white-boxes will predict on two aspects: the object's bounding box size and location, which is a regressor, and the object's category, which is a classifier.

2.1.2 Data, Constrains and Evaluation Metric: The competition provides 1000 COCO test samples without annotation, where object detectors will predict the 81 (80 types of foreground object, and 1 for background) categories' confidence and the bounding-box' size and location.
First limitation is the maximum limit of changed pixels rate. Second is the maximum limit of patches' number. The goal of the adversaries is to make all bounding boxes failed to be returned, by adding the patches to images. As for evaluation score, on one hand the less bounding boxes given by the adversarial example, the higher the score, on the other hand the less changed pixel rate, the higher the score. More specific definition can be found in [3].

2.2 Related Work

A number of attacks for object detectors have been developed recently [16]. [15] extends the attack method from classification to detection and demonstrates that it is possible to attack objectors using a designed classification loss. [9] generate adversarial examples that fool detectors for stop sign and face detections. [7] proposes to attack the RPN with a specially designed hybrid loss incorporating both classification and localization terms. Apart from the full images, [6] attack detectors by restricting the attacks to be within a local region.

2.3 Projected Gradient Descent(PGD)

The PGD [10] attack aims at maximizing the loss: $max\{Jy_{target}, y_{pred}\}$ from the viewpoint of robustness optimization. In each iteration, the PGD first modifies x by $\nabla_x L$, then it will take projection to norm ball.

For some attack problems where the ground-truth label is not given, we can propose a loss function towards bad prediction by

[*] Corresponding author

In: Dimitar Dimitrov, Xiaofei Zhu (eds.): Proceedings of the CIKM AnalytiCup 2020, 22 October, 2020, Gawlay (Virtual Event), Ireland, 2020, published at http://ceur-ws.org.

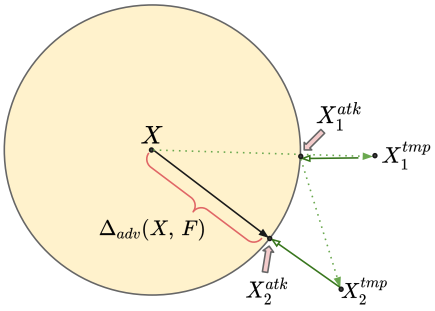


Figure 1: PGD iteration: X is the original input, also initial state of adversarial sample X_0^{atk}, in each gradient descent iter step $i = 1,2,...,n$, X_{i-1}^{atk} will be modified into X_i^{tmp}. Then, if X_i^{tmp} lies outside of the norm box, it will be projected onto the norm box' edge as X_i^{atk}, else $X_i^{atk} = X_i^{tmp}$.

setting synthetic bad label, e.g. in classification problem we may set all labels into a specific category, like 0 in MNIST. Then the original PGD can be used by $min\{Jy_{synthetic}, y_{pred}\}$.

2.4 Sparse l_0-attack

In an l_0-attack one is interested in finding the smallest number of pixels which need to be changed so that the decision changes. We show an adversarial image with l0-attack in Fig.1. From a practical point of view the l0-attack tests basically how vulnerable the model is to failure of pixels or large localized changes on an object e.g. a sticker on a refrigerator or dirt/dust on a windshield [2], or stained points on cell microscopy image.

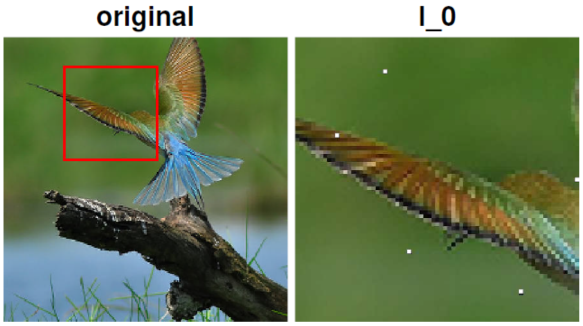


Figure 2: Left: original image with box for zoom. Right: l_0-attack, only 0.04% are changed, but the final classification result is quite different[2].

2.5 Better Black-box Attack, or More Transferable

There are some techniques proposed for better transferable attack on black-box. And in this paper, we only discuss 2 gradient-based methods.

Ensemble of Models has been used to improve traditional prediction. When attacking ensemble of models, the adversarial sample can trade-off between the attack ability and the transferability[4]. While there are K models, we should fuse the output of these models, e.g. for binary classification logits output, we can use weighted averaging $Lx = w_k l_k x$.

The momentum iterative-FGSM[4] also trade-off between the white-box attacks and the transferability. Intuitively, the adversarial example can easily drop into poor local optima in searching landscape and 'overfit' the specific model. So, using the momentum of gradients can smooth the above problem and makes it more transferable.

2.6 Minimum Spanning Tree

An edge-weighted graph is a graph where we associate weights or costs with each edge. A minimum spanning tree (MST) of an edge-weighted graph is a spanning tree whose weight (the sum of the weights of its edges) is no larger than the weight of any other spanning tree.[13]

There are two mainly algorithms to solve the prblem: Prim's algorithm, and Kruskal's algorithm.[13]

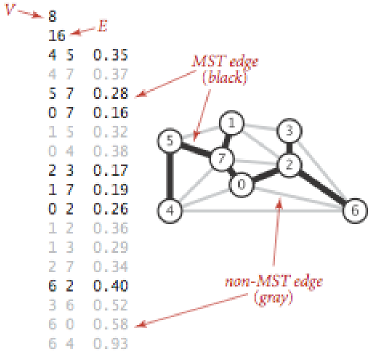


Figure 3: A MST sample, V: vertex, E: edge

3 METHODOLOGY

In this section, we elaborate the details of the proposed method. We first illustrate how to employ the PGD to attack the detectors with l_0-norm. Then we describe our approach to meet the constrains, and tell how to trade off the dilemma that fewer modified pixels results in higher score coefficient but lower attack performance.

3.1 Loss Function

Considering the fact that only if the classifier output as foreground and the class confidence is bigger than NMS threshold, the final predict result will be given, our method can be quite straight forward: only attack the classifier of detectors. Even though the competition does not provide any annotation, thanks for the NMS threshold we can make attack. The attack loss function is defined as follow:

$$Jx = \frac{1}{N}\sum_{i=1}^{N} smlx \cdot \mathbb{1}_{smlx>\tau}, \tag{1}$$

Where N is the number of predicted objects, and sml is the softmax output, and τ is a threshold more strict than NMS threshold. $\mathbb{1}$ is a indicator function that return 1 if $smlx > \tau$ else 0. The target of this function is trying to make all foreground prediction can not pass NMS.

3.2 Attacking Procedure

Our attacking framework is summarized in Algorithm 1. Input sample x to the given detectors, then the softmax probability of a specific category can be obtained. Since background will be neglected by NMS, we only store the foreground's probability. Above input to store process can be denoted as $sml_k x$. By calculate the loss function, we can make back propagation the gradient on x is got.(We just give a simple description here about how to iterate the momentum, more details like its cold start problem and various variants can be seen in [5].) At last, if the attack result is outside of the box, we should projected it onto the norm bound like Fig.1.

Algorithm 1 PGD l_0-attack on ensemble of models

Input: A sample x, the foreground's softmax function of K models: $sml_1, sml_2, ..., sml_K$ and weights: $w_1, w_2, ..., w_K$.
Hyper-params: Training epochs T, gradient momentum update factor μ, learning rate lr.
Output: An adversarial example x^*.

```
$x_0^* = x$, momentum $g_0 = 0$;
for $t = 0$ to $T - 1$ do
    Input $x_t^*$, get $sml_0 x_t^*, sml_1 x_t^*, ..., sml_K x_t^*$;
    Fuse the K logits as $lx_t^* = {}^{K}_{k=1} w_k sml_k$;
    Get loss $Jx_t^*$ based on $lx_t^*$ and Eq.(1);
    Obtain current gradient $\nabla_x Jx_t^*$
    $g_{t1} = g_t * 1 - \mu \ \nabla_x Jx_t^* * \mu$
    $x_{t1}^* = x_t^* - lr * g_{t1}$
    update $x_{t1}^*$ by projection onto $l_0$-norm box
end for
return $x^* = x_T^*$
```

3.3 Construct Connected Domains

We solve the connected domains rule of this competition by K-means and then convert sub-problem to a MST. We first set a specific pixels' number as β for l_0-attack, and train the adversarial sample with Algorithm.1 for certain epochs. Then, we group these pixels into K areas by K-means, which can be visualized in Fig.4. The distance metric of two pixels is given by Chessboard Distance, or known as Chebyshev Distance[14]. After grouping, we connect the pixels within their group by method in 2.1.4. During connecting process, we use these pixels' value before l_0-norm projection, and in fact this is also a projection. If the value is same to original input, we make slight modification on that value to keep connectivity.

4 EXPERIMENTS

In this section, we show our experiments based on the methodology in section 3. In our experiments, we only attack the 2 provided white-box models due to the computational resources limitation.
YOLOv4 make binary classification on whether the object is foreground, and its NMS threshold (0.4) is different from Faster R-CNN(0.3). We set the threshold τ in Eq.1 as 0.15 for YOLO and 0.25 for Faster R-CNN. The latter output 81 class while the former is binary, so the corresponding τ is relatively bigger. The value also take the output's unbalanced distribution into consideration.
We set our learning rate as 60000, because the attack on image's value are $0 - 255$ while the gradients are quite small. The momentum update factor $\mu = 0.5$ is soft, and weights relatively high on the gradients of current iteration compared with the usage other task, like traditional image classification. We set the first 30 epochs to train sample with 8000 pixels l_0 limitation, then use 200 epochs to train sample with less pixels with connecting operation, finally we train 30 epochs to make sure the the sample quantized into uint8 type which helps maintain most adversarial ability. The first 30 'pre-train' epochs' improvement can be visualized in Fig.5.

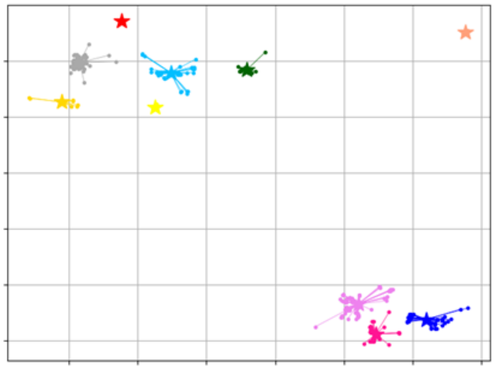


Figure 4: Group pixels to 10 areas with k-means clustering, the dots are the pixels, the stars are centers of each group.

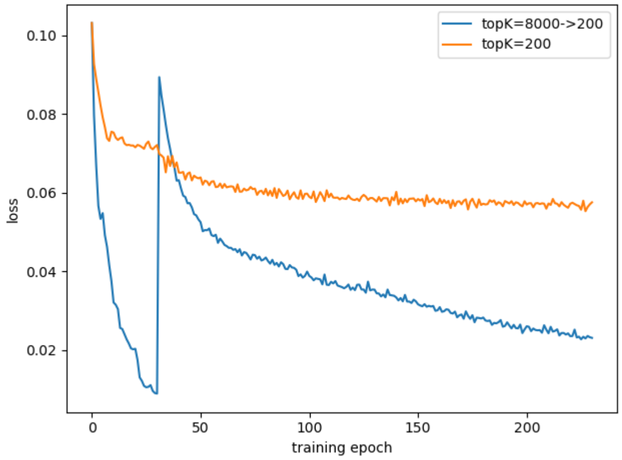


Figure 5: The training process for a sample. Due to the *fixed form* of mask after epoch 30, the pre-train with more top K pixels is useful.

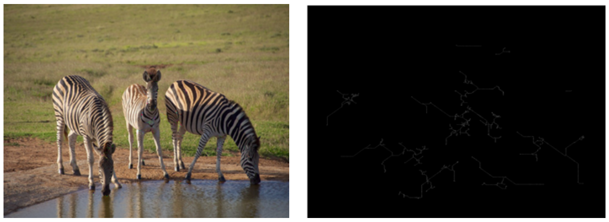
We show our result in Table.1 by adding the methods during the competition, the overall score list are recorded at the our subscription result on the competition entrance. And we visualized a perturbed image and its patch in Fig.6.

5 DISCUSSION AND CONCLUSION

In this paper, we show our PGD l_0-attack adversarial solution towards the competition's target. We find that there is no need to make hand-crafted patches into the image, because by PGD searching

Score Module	A	YOLO	Overall	Black
handcraft patch, A	-	-	61	-
k-means	250	31	304	23
YOLO only	49	683	783	51
A + YOLO	231	651	1036	154
momentum	281	997	1635	357
Prims, 1-stage A	820	1255	2547	472
quantized training	955	1335	2836	546

Table 1: The 'A' means Faster R-CNN. Firstly we use handcrafted patches and only attack A, the pixels side patches will be modified. Then we use K-means to get 10 proposal areas, and the score gets better. We also attack YOLO only, and find that YOLO is much easier to be destroyed. By training on ensemble of the 2 models, we get higher black box score, even though the white-box score shows a little decrease. By adding momentum, the increase is promising. To lower the number of pixels, we optimized the connecting method by Prims together with hybrid stage-1 of A into loss function, and we get a notable improvement. Finally, we quantize the adversarial sample after every 3 epochs, which lowers the train/test error.



(a) perturbed image

(b) modified areas

Figure 6: (a) is a perturbed image result; (b) is the difference of original image and perturbed image. Most of the perturbed pixels lie near to the 3 zebra objects.

this can be done automatically. We also use K-means and Prims to handle the game's constrain. For l_0 attack, the top-K selection is intuitive because the amplitude of gradients on the input interpret how important they are for the prediction target. For the reason that the connecting process is too slow, we just connect once and keep patches fixed, which may hinder the searching result and would be optimized in future work.

REFERENCES

[1] Alexey Bochkovskiy, Chien Yao Wang, and Hong Yuan Mark Liao. 2020. YOLOv4: Optimal Speed and Accuracy of Object Detection. (2020). https://arxiv.org/abs/2004.10934

[2] Francesco Croce and Matthias Hein. 2019. Sparse and Imperceivable Adversarial Attacks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

[3] Dimitar Dimitrov and Xiaofei Zhu. 2020. *CIKM2020 Analyticup: Alibaba-Tsinghua Adversarial Challenge on Object Detection*. Retrieved January 12, 2021 from https://www.cikm2020.org/adversarial-challenge-on-object-detection/

[4] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. 2018. Boosting Adversarial Attacks With Momentum. *CVPR* (2018), 9185–9193.

[5] Diederik Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations* (12 2014).

[6] Yuezun Li, Xian Bian, and Siwei Lyu. 2018. Attacking Object Detectors via Imperceptible Patches on Background. *ArXiv* abs/1809.05966 (2018). http://arxiv.org/abs/1809.05966

[7] Yuezun Li, Daniel Tian, Ming-Ching Chang, Xiao Bian, and Siwei Lyu. 2018. Robust Adversarial Perturbation on Deep Proposal-based Models. *CoRR* abs/1809.05962 (2018). http://arxiv.org/abs/1809.05962

[8] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollar, and Larry Zitnick. 2014. Microsoft COCO: Common Objects in Context. In *ECCV* (eccv ed.). European Conference on Computer Vision. https://www.microsoft.com/en-us/research/publication/microsoft-coco-common-objects-in-context/

[9] Jiajun Lu, Hussein Sibai, and Evan Fabry. 2017. Adversarial Examples that Fool Detectors. *CoRR* abs/1712.02494 (2017). http://arxiv.org/abs/1712.02494

[10] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards Deep Learning Models Resistant to Adversarial Attacks. (06 2017).

[11] Shaoqing Ren, Kaiming He, Ross. Girshick, and Jian. Sun. 2017. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39, 6 (2017), 1137–1149. https://doi.org/10.1109/TPAMI.2016.2577031

[12] Andrew Ross and Finale Doshi velez. 2017. Improving the Adversarial Robustness and Interpretability of Deep Neural Networks by Regularizing their Input Gradients. (11 2017).

[13] Robert Sedgewick and Kevin Wayne. 2000. *Algorithms, 4th Edition*. Retrieved January 12, 2021 from https://algs4.cs.princeton.edu/home/

[14] F. Van der Heijden, Robert Duin, D. Ridder, and David Tax. 2004. Classification, Parameter Estimation and State Estimation: An Engineering Approach Using MATLAB. (01 2004). https://doi.org/10.1002/0470090154

[15] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Yuille. 2017. Adversarial Examples for Semantic Segmentation and Object Detection. In *2017 IEEE International Conference on Computer Vision (ICCV)*. 1378–1387. https://doi.org/10.1109/ICCV.2017.153

[16] H. Zhang and J. Wang. 2019. Towards Adversarially Robust Object Detection. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 421–430. https://doi.org/10.1109/ICCV.2019.00051