Interpreting License Plate Detection Model: A SHAP-based Analysis and Adversarial Attack Exploration

Robert Laskowski RL394414@STUDENTS.MIMUW.EDU.PL
Szymon Sadkowski SS406325@STUDENTS.UW.EDU.PL
University of Warsaw, Poland

Abstract

This project investigates the robustness of a YOLOv5-based model(Ultralytics (2021)), fine-tuned for license plate detection, against adversarial attacks using a variation of DPatch Liu et al. (2019). Additionally, we explore the interpretability of the model's predictions through SHAP Lundberg and Lee (2017).

1. Introduction

In recent years, computer vision models have achieved widespread success. They find application in various domains, highlighting their practical significance. However, ensuring the strength and transparency of these models is crucial, particularly given their deployment in real-world scenarios. To make sure the model is reliable, it's crucial to actively look for weaknesses using methods like adversarial attacks. Also, understanding how the model behaves becomes really important, so we need to make efforts to explain and interpret its predictions. This two-pronged approach not only makes the model stronger but also helps us better grasp how it performs in real-world situations.

2. Methodology

SHAP Lundberg and Lee (2017) or SHapley Additive exPlanations, is a methodology grounded in cooperative game theory. By assigning Shapley values to features based on their impact, SHAP enhances the interpretability of complex neural network models, offering insights into feature contributions.
In our work, we used a custom approach for calculating SHAP values for object detection problem.

- For a single image we only explain single detection
- We use slic image segmentation algorithm to divide images into patches. For SHAP calculation we use those patches as features (if the feature is not present, according patch is filled with mean image color)
- SHAP classification v function (applied to predicted classification probabilites) is multiplied by $IOU(target\_bbox, predicted\_bbox)$ to take into account changes in bbox regression part of model

DPatch Liu et al. (2019) is an adversarial-patch-based attack method designed for mainstream object detectors like Faster R-CNN Girshick et al. (2014) and YOLO. It disrupts both bounding box regression and object classification simultaneously, significantly reducing the mean Average Precision (mAP) of targeted detectors. Notably Lee and Kolter (2019) introduce a specific variation of DPatch attack which aims to solve several problems found in the original algorithm.
YOLO Redmon et al. (2015) is a "one-shot" object detector known for its high efficiency, outperforms other models, running up to 3× faster. YOLO divides the input image into an S × S grid, where each cell predicts B bounding boxes and their confidence scores.

The model and dataset employed in this paper originate from Hugging Face, `https://huggingface.co/keremberke/yolov5m-license-plate` with the dataset consisting of 8,832 images. The model is built upon YOLOv5 and fine-tuned for the task of license plate detection. According to the model's author, it achieves an mAP@0.5 score of 0.988 on the validation set.

Note that the term *patch* is used with two distinct meanings in this work: as an adversarial patch strategically placed on an image to deceive the detector, and as a segment of an image identified during our SHAP analysis.

3. Experimental results

3.1 DPatch adversarial attack

The initial phase of our work focused on attacking the model through the generation of adversarial examples with DPatch method. The patch training utilized a set of 100 random images, with the final training duration lasting 12 hours on a single P5000 GPU. In contrast to the original paper, which used 300,000 iterations, the training involved 50,000 iterations. The process employed implementation based on Lee and Kolter (2019) from the Adversarial Robustness Toolbox (ART) library: `https://github.com/Trusted-AI/adversarial-robustness-toolbox`. The resulting DPatch was sized at 250x250 and positioned at (50x50). All images underwent rescaling to 640x640 to meet model's requirements. We initially set the learning rate to 0.5 and decayed it by 0.95 every 5000 iterations. Notably, smaller DPatches were found to be ineffective during the training phase, and hyperparameter tuning did not yield discernible improvements. The training process stabilized around 30,000 iterations, with further steps showing no significant improvement. Performance evaluation during training involved monitoring model losses (total loss, objectness loss, box loss), which were expected to increase, and tracking the number of images on which the model failed to detect any object after applying the patch—an indicator that was anticipated to rise as well.

After applying the trained DPatch, 33% of the original detections were missing, contrasting with 2% when using a random patch of the same size and location. Successful cases for the DPatch were predominantly observed when the patch was close to the license plate, including instances where it intersected with it. It is noteworthy that even when applying a random patch that intersected with the license plate, the model still successfully detected it. Upon examining resulting detections on patched images without explanatory methods like SHAP, it appeared that the DPatch primarily exerted local effects. Overall, the results indicated that the DPatch needs to be relatively large and positioned near the license plate

to disrupt object detection effectively. Additionally, the attack success rate did not align well with the reported results in the original paper. Graphs illustrating the training process and DPatch visualizations can be found in Appendix A.

Several factors could explain the observed limitations in the success of the DPatch attack. Firstly, the DPatch papers targeted the YOLOv3 model, which is an older version of YOLO. It is plausible that more recent iterations of YOLO exhibit increased resilience to adversarial attacks. Secondly, the task of detecting license plates may inherently be simpler than detecting a diverse range of classes from the COCO dataset Lin et al. (2015), as employed in the original paper. Additionally, the model in focus underwent fine-tuning for the specific task of license plate detection. Unlike the original paper, where attacks could target multiple classes, our DPatch training had a narrower scope, focusing solely on one class. Consequently, it couldn't effectively exploit vulnerabilities related to class loss. Lastly, considering the varied success rates observed across classes in the original paper, with some achieving nearly 0 mAP after applying the patch and others exhibiting lower success rates, license plates may fall into the latter category, proving to be challenging targets for effective adversarial attacks.

3.2 SHAP explonations

The second phase of our work focused on understanding the model predictions. We used a variation of the SHAP method and conducted a comparative analysis of prediction explanations on original images, images with the applied trained DPatch, and images with a random patch. The results of these explanations revealed the contributions of image segments to license plate detection. Upon initial visual examination, we observed that our trained DPatch did not alter the positive contribution of the segment containing the license plate to detection. However, we noted that patch containing DPatch was causing a negative contribution to the detection. Additionally, SHAP explanations revealed that the DPatch had a contribution to the detection only when it was in proximity to the license plate. We also aimed to use metrics that would help to quantify and compare explanations across 3 sets of images: **orig** (original images), **dpatch** (images with DPatch applied), **rand** (images with random patch applied). The employed metrics were:

- **most-pos-inf-invariant(A, B)** - counts what percent of corresponding images in two sets A, B (e.g. orig and dpatch) have the same patch with highest positive contribution.
- **most-neg-inf-invariant(A, B)** - counts what percent of corresponding images in two sets have the same patch with lowest negative contribution.
- **mean-abs-contr-diff(A, B)** - An average of absolute differences between contribution of the same patches in corresponding images from set A and B.

	orig	DPatch	rand
orig	x	0.84	0.92
DPatch	0.84	x	0.86
rand	0.92	0.86	x

Table 1: Most positive patch invariant

	orig	DPatch	rand
orig	x	0.29	0.38
DPatch	0.29	x	0.32
rand	0.38	0.32	x

Table 2: Most negative patch invariant

	orig	DPatch	rand
orig	x	0.024936	0.014256
DPatch	0.024936	x	0.0198
rand	0.014256	0.0198	x

Table 3: Mean abs patch contribution difference

- $most-neg-patch-inv(orig, dpatch) < most-neg-patch-inv(orig, random)$ indicates that DPatch has a stronger impact on changing the most negatively contributing segments compared to random patch.
- $most-pos-patch-inv(orig, dpatch) < most-neg-patch-inv(orig, random)$ indicates that DPatch has a stronger impact on changing the most positively contributing segments compared to random patch.
- $most - pos - patch - inv(rand, dpatch) \sim most - pos - patch - inv(orig, dpatch)$, $most - neg - patch - inv(rand, dpatch) \sim most - neg - patch - inv(orig, dpatch)$ which implies that explonations of rand and orig are similar.
- $mean - abs - contr - diff(orig, rand) < mean - abs - contr - diff(dpatch, rand) < mean-abs-contr-diff(orig, dpatch)$ indicates that difference between explonations of orig and rand is smaller than between orig and dpatch.

All of the above seems to be coherent with intuition and initial visual examination of the explonations.

4. Conclusion

This study examines the robustness and interpretability of a fine-tuned YOLOv5 model for license plate detection using DPatch adversarial attacks and SHAP analysis.

DPatch experiments revealed limitations in perturbing the model, with reduced detection rates. Discrepancies from the original DPatch paper were attributed to specific fine-tuning for license plate detection and potential model advancements.

SHAP analysis provided insights into the model's decision-making, highlighting the local effects of DPatch and the model's focus on specific regions, such as the license plate. Numbers from Tables 1-3 align with our observations and further solidify our understanding on DPatch attacks.

References

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation, 2014.

Mark Lee and Zico Kolter. On physical adversarial patches for object detection, 2019.

Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2015.

Xin Liu, Huanrui Yang, Ziwei Liu, Linghao Song, Hai Li, and Yiran Chen. Dpatch: An adversarial patch attack on object detectors, 2019.

Scott Lundberg and Su-In Lee. A unified approach to interpreting model predictions, 2017.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection, 2015.

Ultralytics. YOLOv5: A state-of-the-art real-time object detection system. `https://docs.ultralytics.com`, 2021.

Appendix A. DPatch - training and visualization

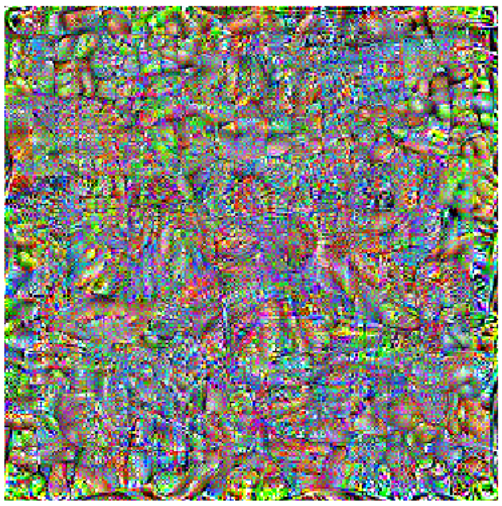

Figure 1: Our patch

Figure 2: Lee and Kolter (2019) patch

Figure 1 and 2 present a DPatch trained by us and compare it with the one presented in the original paper. It is evident that our patch lacks visible structures in comparison to the latter.

Figure 3: Detections on image without applied patch

Figure 4: Detections on image with random patch

Figure 5: Detections on image with trained patch

Figure 6: Original image

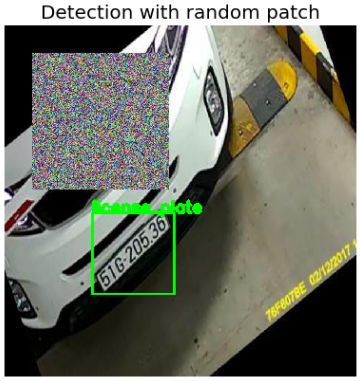

Figure 7: Random patch

Figure 8: DPatch

Figures 3-8 present one of the successful patch attacks when DPatch was in near proximity of license plate.

Figure 9: Original image

Figure 10: Random patch

Figure 11: DPatch

Figures 9-11 present a case when applying DPatch had no effect on detection.

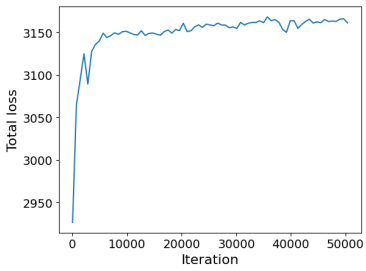

Figure 12: Total loss

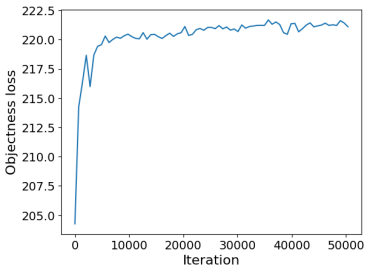

Figure 13: Objectness loss

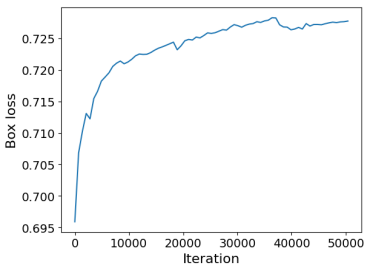

Figure 14: Box loss

Figures 12 - 14 present how model losses increased during training.

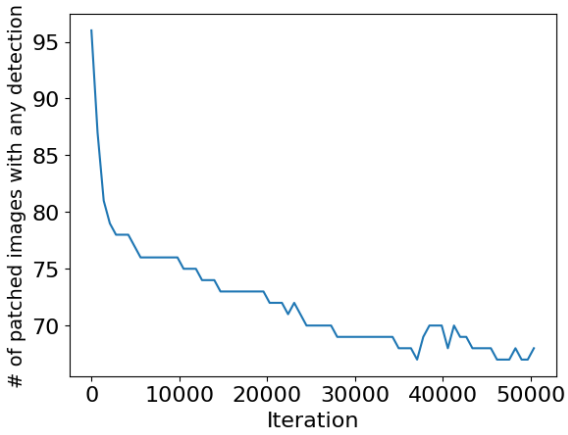

Figure 15: Number of images with any detection after applying DPatch

Figure 15 present number of detections on images with DPatch applied decreased during training and stabilized at around 67 at the end.

Appendix B. SHAP explonations

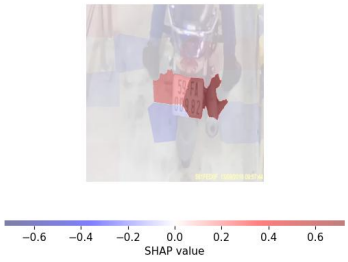

Figure 16: SHAP on original image

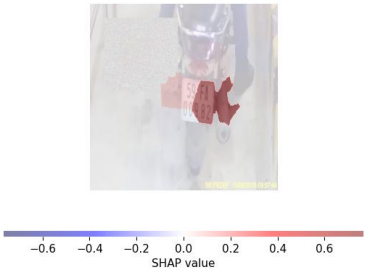

Figure 17: SHAP on image with random patch

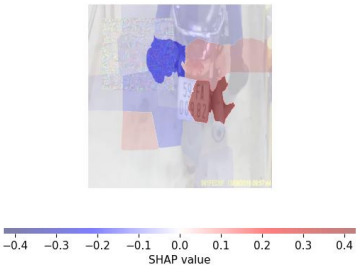

Figure 18: SHAP on image with DPatch

Figures 16, 17, 18 present contributions of different segments of images on the license plate prediction. Blue segments have a negative impact, while red segments have a positive impact. Segment with a DPatch on figure 18 has a large negative contribution. a

Appendix C. Possible Improvements

We suggest a few improvements that can improve the quality of this approach.

- In future work, we could investigate training smaller DPatches and dynamically applying them to the license plate, such as in the corners. This strategy aims to address practical scenarios where individuals attempt to deceive the detection system with small stick-on modifications to their plates. The rationale is that this approach may be effective due to the DPatch's substantial impact near the detectable objects.
- One can try normalizing SHAP values to bring more meaning into numeric metrics.
- due to our way of calculating SHAP values, all of shap values are 0 when model returns no detections - providing us with no explanation. Instead, we could disable non max suppression and calculate regular shap based on detection score attached to bbox closest (ex. in IOU terms) to target bbox.
- new metrics could be introduced. 1. $GT - most - pos - influencial$ which returns 1 if the patch with the greatest positive contribution has non-zero intersection with target bbox, otherwise 0. This could allow us to test our observations that target bbox (licence plate) should have greatest positive contribution and find anomalies. 2. $DPatch - most - neg - influencial$ which returns 1 if the patch with the smallest negative contribution has a non-zero intersection with applied DPatch, otherwise 0. This one could allow us to test our observations that patches introduce local negative influence and find anomalies.