Eigenpatches — Adversarial Patches from Principal Components

Jens Bayer[1,2] , Stefan Becker[1,2] , David Münch[1,2] , and Michael Arens[1,2]

[1] Fraunhofer Center for Machine Learning,
[2] Fraunhofer IOSB,
Gutleuthausstr. 1, 76275 Ettlingen, Germany
`jens.bayer@iosb.fraunhofer.de`

Abstract. Adversarial patches are still a simple yet powerful white box attack that can be used to fool object detectors by suppressing possible detections. The patches of these so-called evasion attacks are computational expensive to produce and require full access to the attacked detector. This paper addresses the problem of computational expensiveness by analyzing 375 generated patches, calculating the principal components of these and show, that linear combinations of the resulting "eigenpatches" can be used to fool object detections successfully.

Keywords: Adversarial Patch Attack · Object Detection · Principal Component Analysis.

1 Introduction

Despite the good performance in image classification, object detection and semantic segmentation, computer vision systems are still prone to adversarial attacks. A prominent white box attack scenario is an evasion attack against object detectors using adversarial patches, whereas these patches are sometimes also referred to as a form of camouflage [6] or invisibility cloak [27,28]. This is possible since the patches depict specifically calculated patterns, that alter the activations of the attacked object class for an object detector.

As "simple" and straightforward the generation process of such a patch is, as time-consuming and computation intensive it is as well. To speed up the patch calculation and gain a more profound understanding if there are common attributes that can be extracted to harden an object detector, this paper investigates:

1. If patches that are combinations of principal components generated by a set of precalculated patches can be used to create valid patches.
2. How many principal components are necessary, for recreating patches with a sufficient high impact on the detection score.
3. The influence of set size used to generate the principal components, regarding the quality of the resulting patches.
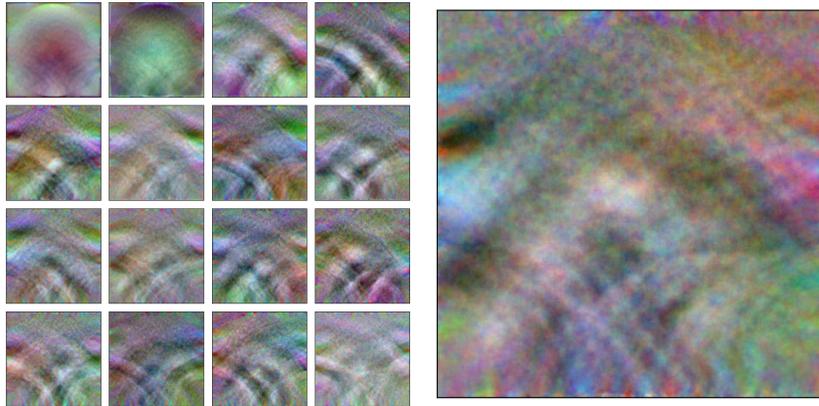
Fig. 1. Left: the first sixteen principal components extracted from the trained patches. Right: the unweighted mean image of these sixteen principal components.

Towards this end, the well known YOLOv7 object detector [23] is attacked with various adversarial patches. These patches are trained on a subset of the INRIA Person dataset [4] and are analyzed with a principal component analysis (PCA). The extracted principal components are then used to recreate the patches and even create new patches by combining them linearly (e.g., Figure 1).

To the best of our knowledge, this is the first approach in investigating and creating adversarial patches that attack object detectors by using principal components.

The organization of the paper is as follows: In section 2 the related work regarding the investigation and analysis of adversarial attacks is addressed. A brief introduction of what eigenpatches are and how they can be crafted is given in section 3. Section 4 covers all necessary information to recreate the experimental results. The used object detector, the dataset, and the generation process of the patches that are the base for eigenpatch generation are further explained. The results of the experiments and the used metrics are presented in section 5. Finally, section 6 summarizes the paper, gives a conclusion and a brief outlook.

2   Related Work

In the following section, selected works regarding the analysis of adversarial attack patterns and sampling from lower dimensional embeddings are presented. For a broader overview, the interested reader is encouraged to read the survey papers for adversarial attacks [1,3,13,24] and the YOLO object detector family [19].

The approach of using dimensionality reduction algorithms like principal component analysis to investigate the patterns generated by adversarial attacks for image classifiers is an active research area. Nonetheless, most investigations

are performed solely on image classifiers with attacks, that induce high frequent noise on the whole image [21,25,15,5,14,26,8]. Only a few works [18] investigate adversarial patches.

Tramer et al. explore the space of transferable adversarial examples by proposing methods for estimating the dimensionality of the space of adversarial inputs [21]. By investigating untargeted misclassification attacks, they show that perturbing a data point in such a way, that it crosses a model's decision boundary, is likely to result in similar performance degradation when applied to other models.

Wang et al. present a fast black-box adversarial attack that finds key differences between different classes based on PCA that later can be used to drive a sample to a target class or the nearest other class [25].

Energy Attack [15] is a transfer-based black-box L_∞ adversarial attack that uses PCA to obtain the energy distribution of perturbations, generated by whitebox attacks on a surrogate model. For the attack, patches are sampled from the energy distribution, tiled and applied on the target image.

Dohmatob et al. investigate, whether neural networks are vulnerable to blackbox attacks inherently [5]. After analyzing low-dimensional adversarial perturbations, they hypothesize that adversarial perturbations exist with high probability in low dimensional subspaces, that are much smaller than the dimension of the image space.

Theoretical bounds on the vulnerability of classifiers against adversarial attacks are shown by Shafahi et al. using the unit sphere and unit cube [14]. They claim that by using extremely large values for the class density functions, the bounds can be potentially escaped.

Weng et al. perform a Singular Value Decomposition to improve adversarial attacks on convolutional neural networks [26]. They combine the top-1 decomposed singular value-associated features for computing the output logits with the original logits, used to optimize adversarial examples and thus boost the transferability of the attacks.

Recently, Tarchoun et al. examined adversarial patches from an information theory perspective and measured the entropy of random crops of the patches [18]. The results indicate that the entropy of adversarial patches have a higher mean entropy than natural images. Based on these findings, they create a defense mechanism against adversarial patches.

Further research on the relationship between adversarial vulnerability and the number of perturbed dimensions is performed by Godfrey et al. [9]. Their results strengthen the hypothesis, that adversarial examples are a result of the locally linear behavior of neural networks with high dimensional input spaces.

3   Eigenpatches

This section covers the generation process of the eigenpatches. The term eigenpatches is derived from eigenfaces (the most prominent example of eigenpictures [16]), which are calculated similarly. Eigenpictures is the name of the eigenvectors, that can be derived from a set of training images. They can be used as

a low-dimensional representation of the original training images and recreating these through a linear combination.

Given a set of adversarial patches

$$\mathcal{P} = \{\boldsymbol{P}_i | i = 1, \ldots, n\}, \quad \boldsymbol{P}_i \in [0,1]^{C \times H \times W} \tag{1}$$

where H is the height in pixel, W is the width in pixel and C is the number of channels. A principal component analysis is now performed on $\mathcal{P}$. With the top k principal components $\boldsymbol{E}_j, j \in \{1, \ldots, k\}$ and the weights $\lambda_{i,j}$ the set

$$\hat{\mathcal{P}} = \{\hat{\boldsymbol{P}}_i | i = 1, \ldots, n\} \quad \hat{\boldsymbol{P}}_i = \sum_{j=1}^{k} \lambda_{i,j} \boldsymbol{E}_j \tag{2}$$

can be generated, that consists of linear combinations of the principal components and is a recreation of $\mathcal{P}$. Figure 1 shows the first sixteen principal components as well as the patch resulting from calculating the mean of the components.

4   Experimental Setup

In the following, all the necessary information to recreate the experimental results is presented. In addition, the full source code is also provided[3]. The attacked object detector is YOLOv7 [22] with the official pretrained weights. The used Dataset is the INRIA Person dataset [4].

4.1   Object Detector

Due to its prominence and good performance, YOLOv7 [23] is selected as the object detector to be attacked. The smallest model of YOLOv7 with an input size of 640×640 pixels and the official pretrained weights are used. Despite a small fix of an error, that causes problems during the training of the patches, the source code is not modified. The generated patches can therefore be used without specific finetuned weights, and the results can easily be verified.

4.2   Dataset

For the generation of the patches, the positive images of the INRIA Person dataset [4] are used. Instead of using the provided ground truth bounding boxes of the dataset, the object detector first generates new ground truth data for each train image. After this, the newly created bounding boxes are manually reviewed. False positives and bounding boxes of highly obscured target classes (persons) are removed. The resulting ground truth data for both, the train and test split, are available in the source code repository. Since the input size of the used YOLOv7 model is 640×640 pixels, the images are resized and padded to match this size. A resized and patched example image as it would be used in the evaluation is given in Figure 2.

---

[3] https://github.com/JensBayer/PCAPatches

Fig. 2. Patched input image as it would be used in the evaluation. The image is resized and padded to match the required input size of 640×640 pixels. After this, the tested patch is embedded in the center of each bounding box and resized according to a scaling factor.

4.3   Patch Generation

The training procedure of the patches follows the procedure, described in [20]: During the generation process of a patch, it is placed inside the bounding boxes of selected target classes in images of a training dataset. To improve the robustness of the patch, different randomized transformations, such as translation, rotation, perspective and also color jitter are applied. The altered image is then propagated through the object detector and the objectness score of the target classes is minimized. In addition, a smoothness loss that punishes high frequent noise in the patches is also calculated and applied.

All patches are initialized with random values in the range $[0, 1]$ and optimized with the AdamW [12] optimizer with an initial learning rate of 0.01. The random color jitter changes the brightness, contrast, and saturation of the patch randomly by a values in the range of $[0.9, 1.1]$, $[0.95, 1.05]$, and $[0.97, 1.03]$ respectively. For the perspective, the distortion scale parameter is set to 0.5. The resize range of the patch, the learning rate scheduler and the number of training epochs are defined by the values in Table 1.

The random translation of the patch is performed in such a way, that the resized and transformed patch is placed inside the bounding box.

5   Evaluation

This section covers the used metrics, the results, and a final limitations section that discusses, under which circumstances these results are generalizable and

Run ID	n	Epochs	LR-Scheduler	Resize range	Rotation
191	175	100	StepLR	[0.75, 1.0]	30
885	100	100	CosineAnnealingLR	[0.75, 1.0]	30
905	50	100	StepLR	[0.75, 1.0]	45
371	25	125	StepLR	[0.5, 0.75]	30
243	25	125	StepLR	[0.5, 0.75]	45

Table 1. Different parameterization for the patch generation. The resize range is the range of the scaling factor, relatively to the bounding box.

valid they are. The ↑ and ↓ in the following tables indicate, whether a higher or lower value in the column is desired.

5.1   Metrics

To quantify the results, the well-known mean average precision object detector metrics mAP@.5 [7] and mAP@.5:.95 [11] are used. While mAP@.5 is the mean average precision for bounding boxes with an intersection over union (IoU) threshold of at least fifty percent, the mAP@.5:.95 is the average across ten IoU thresholds and therefore more strict. Both metrics are calculated for the unpatched and the patched images. The higher the difference between the unpatched and patched metric scores, the stronger is the impact of the patches on the detector. For Table 2, the differences (Δ) between the unpatched and patched inputs images is also given.

5.2   Experimental Setup

Similar to the training data, the test split and positive images of the INRIA Person dataset are used for the evaluation. New ground truth bounding boxes are generated the same way as described in subsection 4.2. To generate reproducible results, the evaluation procedure is as follows: A patch is placed in the center of each bounding box as given in the ground truth information and resized to a relative size of 0.75 times the longer side of the bounding box. Other transformations such as the ones mentioned in subsection 4.3 are not applied. An example of a test image is given in Figure 2. After this, the mAP@.5 and mAP@.5:.95 is calculated, according to the provided detector evaluation script.

5.3   Results

As shown in Table 2, the trained patches have a significant impact on the object detector. While the mAP of the detector on the test data is about 0.73 (0.71), the mAP drops to 0.46 (0.35) when attacked with the adversarial patches. To verify, that this is not a result of covering the persons in the image with the patches, we also checked eleven single-colored gray valued patches. Each of these patches has one particular shade of gray, out of the uniformly distributed range from

Applied Patches	Count	$mAP@.5 \downarrow$	$mAP@.5:.95 \downarrow$	$\Delta mAP@.5 \uparrow$	$\Delta mAP@.5:.95 \uparrow$
No	1	0.73	0.71		
Gray value	11	0.74 ± 0.03	0.71 ± 0.03	-0.01 ± 0.03	0.00 ± 0.02
Trained	375	$\mathbf{0.46 \pm 0.03}$	$\mathbf{0.35 \pm 0.03}$	$\mathbf{0.27 \pm 0.03}$	$\mathbf{0.36 \pm 0.03}$
PCA(2) recovered	375	0.63 ± 0.03	0.58 ± 0.03	0.10 ± 0.03	0.13 ± 0.03
PCA(4) recovered	375	0.60 ± 0.04	0.54 ± 0.04	0.13 ± 0.04	0.17 ± 0.04
PCA(8) recovered	375	0.58 ± 0.03	0.51 ± 0.03	0.15 ± 0.03	0.20 ± 0.03
PCA(16) recovered	375	0.57 ± 0.03	0.51 ± 0.03	0.16 ± 0.03	0.20 ± 0.03
PCA(32) recovered	375	0.57 ± 0.03	0.51 ± 0.04	0.16 ± 0.03	0.20 ± 0.04
PCA(64) recovered	375	0.57 ± 0.04	0.49 ± 0.04	0.16 ± 0.04	0.22 ± 0.04
PCA(128) recovered	375	0.55 ± 0.05	0.46 ± 0.06	0.18 ± 0.05	0.25 ± 0.06
PCA(256) recovered	375	0.51 ± 0.07	0.41 ± 0.07	0.22 ± 0.07	0.30 ± 0.07

Table 2. Mean average precision for the unpatched and patched test data. "Gray value" refers to gray values in the range of zero to one with a linear step size. "Count" is the numbers of patches on which the result mean and standard deviation values are calculated. The last two columns are the differences of the mAPs of the patched and unpatched images.

zero to one. As the second row of Table 2 shows, the gray valued patches do not cover important parts and even improve the mAP@.5 slightly. In comparison, the 375 crafted patches result in heavy mAP drops.

As expected, the PCA recovered patches are not as good as the trained ones and the more principal components are used to recreate the patch, the higher the mAP drop is (see Figure 3). The rising but still small standard deviation suggests that some patches can be recreated better than others, which seems plausible, since the number of patches with the same parameterization as given in Table 1 is unevenly distributed. In addition, the mAP@.5 difference between the patches that use 128 principal components for recreation and the patches that only use eight principal components with about 0.03 is relatively small. Yet, the impact on the mAP@.5:.95 is almost twice as high (0.05).

In comparison to Figure 3, the curves in Figure 4 are not that intuitive. Here, the number of patches used to compute the PCA and the resulting mAPs are plotted. Since a PCA with n components requires at least n elements in the input set, the number of principal components is given by $min(n, 64)$. As a result, the number of principal components used for this plot equals either the input set size or 64. As expected, the recreation of all 375 patches with only two components results in a small standard deviation of 0.01 while achieving a mean mAP:.5 of 0.52, which differs from the PCA(256) by only a small percentage. A possible reason is, that with such a small input set size, the resulting patch recreations do not vary a lot and reassemble, at best, one of the two patches that are given in the input set. A look of the reassembled patches (see Figure 5) as well as the minimum achieved mAP values (0.41, 0.31) support this statement.
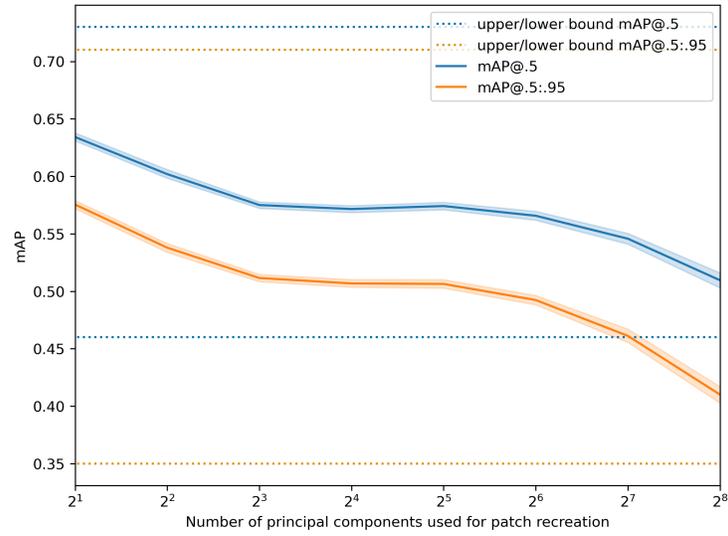
Fig. 3. Drop of the mAP@.5 and mAP@.5:.95 with different numbers of principal components to the recovered patches.
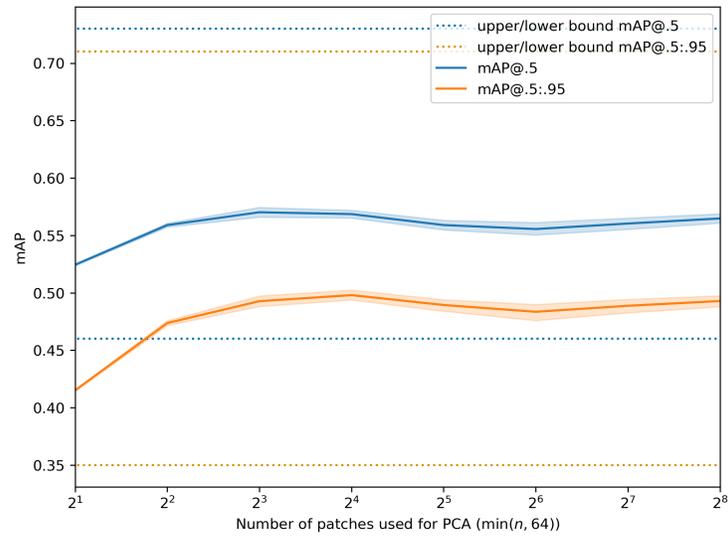

Fig. 4. mAP@.5 and mAP@.5:.95 curve with different numbers of input elements for the PCA to recreate the patches with $\min(n, 64)$ principal components.
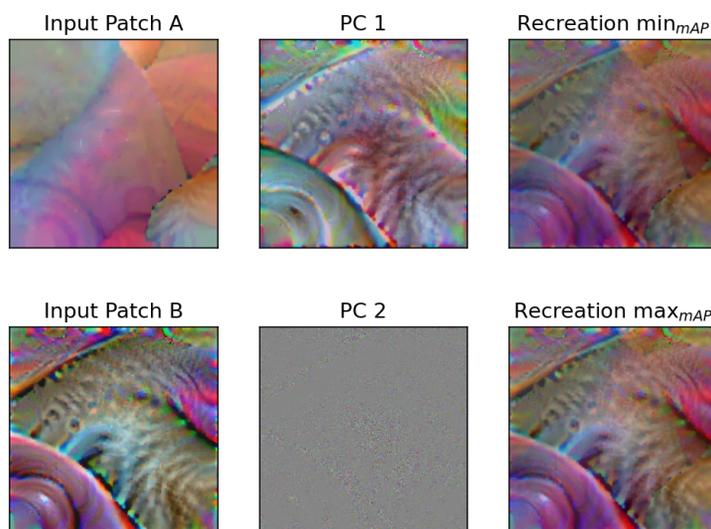
Fig. 5. Left: The input set for the PCA. Middle: The computed principal components. Right: Example recreations of the patches by the two principal components.

6   Conclusion

This paper investigates the applicability of the principal component analysis for adversarial patches to fool object detectors. The evaluation shows, that the recreation and sampling of patches based on the principal components is possible. As expected, those patches are generally not as good as carefully trained ones, yet they can be used as an initialization to finetune new patches. The evaluation also shows that the more principal components are used, the higher the mAP drop is. Nonetheless, the usage of the first eight principal components results already in a noticeable mAP drop. The influence of the set size used for the PCA is after eight patches also quite stable.

Since these experiments are performed on a comparable small dataset and only a single object detector, future work should check the behavior with larger datasets (e.g., OpenImages [10]) and other object detectors (e.g. EfficientDet [17] or DETR [2]).

Acknowledgements

References

1. Akhtar, N., Mian, A.: Threat of Adversarial Attacks on Deep Learning in Computer Vision: A Survey. IEEE Access **6**, 14410–14430 (2018). https://doi.org/10.1109/ACCESS.2018.2807385
2. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-End Object Detection with Transformers. In: ECCV. vol. 12346 LNCS, pp. 213–229. Springer International Publishing (2020). https://doi.org/10.1007/978-3-030-58452-8_13
3. Chakraborty, A., Alam, M., Dey, V., Chattopadhyay, A., Mukhopadhyay, D.: A survey on adversarial attacks and defences. CAAI Transactions on Intelligence Technology **6**(1), 25–45 (2021). https://doi.org/10.1049/cit2.12028
4. Dalal, N., Triggs, B.: Histograms of Oriented Gradients for Human Detection. In: CVPR. vol. 1, pp. 886–893. IEEE (2005). https://doi.org/10.1109/CVPR.2005.177
5. Dohmatob, E., Guo, C., Goibert, M.: Origins of Low-dimensional Adversarial Perturbations. In: Proceedings of The 26th International Conference on Artificial Intelligence and Statistics. vol. 206, pp. 9221–9237. PMLR (2023)
6. Duan, Y., Chen, J., Zhou, X., Zou, J., He, Z., Zhang, J., Zhang, W., Pan, Z.: Learning Coated Adversarial Camouflages for Object Detectors. In: Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence. pp. 891–897. International Joint Conferences on Artificial Intelligence Organization, California (7 2022). https://doi.org/10.24963/ijcai.2022/125
7. Everinghm, M., Zisserman, A., Williams, C.K., Van Gool, L., Allan, M., Bishop, C.M., Chapelle, O., Dalal, N., Deselaers, T., Dorkó, G., Duffner, S., Eichhorn, J., Farquhar, J.D., Fritz, M., Garcia, C., Griffiths, T., Jurie, F., Keysers, D., Koskela, M., Laaksouen, J., Larlus, D., Leibe, B., Meng, H., Ney, H., Schiele, B., Schmid, C., Seemann, E., Shawe-Taylor, J., Storkey, A., Szedmak, S., Triggs, B., Ulusoy, I., Viitaniemi, V., Zhang, J.: The 2005 PASCAL visual object classes challenge. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **3944 LNAI**, 117–176 (2006). https://doi.org/10.1007/11736790_8
8. Garcia, W., Chen, P.y., Clouse, H.S., Jha, S., Butler, K.R.B.: Less is More : Dimension Reduction Finds On-Manifold Adversarial Examples in Hard-Label Attacks. In: First IEEE Conference on Secure and Trustworthy Machine Learning (2023)
9. Godfrey, C., Kvinge, H., Bishoff, E., Mckay, M., Brown, D., Doster, T., Byler, E.: How many dimensions are required to find an adversarial example? In: CVPR (2023)
10. Krasin, I., Duerig, T., Alldrin, N., Ferrari, V., Abu-El-Haija, S., Kuznetsova, A., Rom, H., Uijlings, J., Popov, S., Kamali, S., Malloci, M., Pont-Tuset, J., Veit, A., Belongie, S., Gomes, V., Gupta, A., Sun, C., Chechik, G., Cai, D., Feng, Z., Narayanan, D., Murphy, K.: OpenImages: A public dataset for large-scale multi-label and multi-class image classification. Dataset available from https://storage.googleapis.com/openimages/web/index.html (2017)
11. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common objects in context. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **8693 LNCS**(PART 5), 740–755 (2014). https://doi.org/10.1007/978-3-319-10602-1_48
12. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. 7th International Conference on Learning Representations, ICLR 2019 (2019)

13. Pauling, C., Gimson, M., Qaid, M., Kida, A., Halak, B.: A Tutorial on Adversarial Learning Attacks and Countermeasures. In: arXiv preprint (2022), `http://arxiv.org/abs/2202.10377`

14. Shafahi, A., Huang, R., Studer, C., Feizi, S., Goldstein, T.: Are adversarial examples inevitable? 7th International Conference on Learning Representations, ICLR 2019 (2019)

15. Shi, R., Yang, B., Jiang, Y., Zhao, C., Ni, B.: Energy Attack: On Transferring Adversarial Examples. In: arXiv preprint (2021), `http://arxiv.org/abs/2109.04300`

16. Sirovich, L., Kirby, M.: Low-dimensional procedure for the characterization of human faces. Journal of the Optical Society of America A **4**(3), 519 (1987). https://doi.org/10.1364/josaa.4.000519

17. Tan, M., Pang, R., Le, Q.V.: EfficientDet: Scalable and Efficient Object Detection. In: CVPR. pp. 10778–10787. IEEE (6 2020). https://doi.org/10.1109/CVPR42600.2020.01079

18. Tarchoun, B., Khalifa, A.B., Mahjoub, M.A., Abu-ghazaleh, N.: Jedi : Entropy-based Localization and Removal of Adversarial Patches. In: CVPR. pp. 4087–4095 (2023)

19. Terven, J., Cordova-Esparza, D.: A Comprehensive Review of YOLO: From YOLOv1 to YOLOv8 and Beyond. In: arXiv preprint. pp. 1–27 (2023), `http://arxiv.org/abs/2304.00501`

20. Thys, S., Ranst, W.V., Goedeme, T.: Fooling automated surveillance cameras: Adversarial patches to attack person detection. CVPR Workshops **2019-June**, 49–55 (2019). https://doi.org/10.1109/CVPRW.2019.00012

21. Tramèr, F., Papernot, N., Goodfellow, I., Boneh, D., McDaniel, P.: The Space of Transferable Adversarial Examples. In: arXiv preprint. pp. 1–15 (2017), `http://arxiv.org/abs/1704.03453`

22. Wang, C.Y., Bochkovskiy, A., Liao, H.Y.M.: Scaled-YOLOv4: Scaling Cross Stage Partial Network. In: CVPR. pp. 13024–13033. IEEE (6 2021). https://doi.org/10.1109/CVPR46437.2021.01283

23. Wang, C.Y., Bochkovskiy, A., Liao, H.Y.M.: YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In: CVPR. pp. 7464–7475 (2023)

24. Wang, S., Veldhuis, R., Strisciuglio, N.: The Robustness of Computer Vision Models against Common Corruptions: a Survey. arXiv preprint pp. 1–23 (5 2023), `http://arxiv.org/abs/2305.06024`

25. Wang, Z.M., Gu, M.T., Hou, J.H.: Sample Based Fast Adversarial Attack Method. Neural Processing Letters **50**(3), 2731–2744 (2019). https://doi.org/10.1007/s11063-019-10058-0

26. Weng, J., Luo, Z., Lin, D., Li, S., Zhong, Z.: Boosting Adversarial Transferability via Fusing Logits of Top-1 Decomposed Feature. Association for Computing Machinery (2023)

27. Wu, Z., Lim, S.N., Davis, L.S., Goldstein, T.: Making an Invisibility Cloak: Real World Adversarial Attacks on Object Detectors. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **12349 LNCS**, 1–17 (2020). https://doi.org/10.1007/978-3-030-58548-8_1

28. Zhu, X., Hu, Z., Huang, S., Li, J., Hu, X.: Infrared Invisible Clothing: Hiding from Infrared Detectors at Multiple Angles in Real World. In: CVPR. pp. 13307–13316. IEEE (6 2022). https://doi.org/10.1109/CVPR52688.2022.01296