Generative Dynamic Patch Attack

Xiang Li
xli62@gsu.edu
Shihao Ji
sji@gsu.edu

Department of Computer Science
Georgia State University
Atlanta, GA, USA

A Experimental Details

We first describe the three benchmark datasets and target models used in our experiments. These datasets are used to train our GDPA generator, robust models with adversarial training, and evaluate the performance of patch attacks.

A.1 VGGFace

Dataset The VGGFace dataset [7] is a benchmark for face recognition, containing 2,622 subjects and 2.6 million images in total. Same with DOA [11], we choose 10 subjects and sample face images only containing those individuals. We process the data to the size of 224×224 by standard crop-and-resize, and perform class-balanced split to generate training, validation, and test datasets with ratio 7:2:1. As a result, we obtain 3178, 922 and 470 images for training, validation and test, respectively. The training set is used to train the target model, the GDPA generator and robust models with adversarial training. Likewise, the test set is used to evaluate the target model, the performance of patch attack and adversarial defense.

Target Model We use the VGGFace CNN model [7] as the target classifier in our experiments. We use standard transfer learning on our processed dataset, keeping the convolutional layers in the VGGFace CNN model, but adjusting the number of output neurons of the last fully connected layer to 10. In order to use the pre-trained weights from the convolutional layers of VGGFace CNN model, we convert the images from RGB to BGR and subtract the mean value $[129.2, 104.8, 93.6]$. We set the batch size to 64 and use the Adam Optimizer with an initial learning rate of 10^{-4}. We drop the learning rate by 0.1 every 10 epochs. For hyperparameter tuning and model selection, we track the accuracy on validation set to avoid overfitting. We train the model on training set for 30 epochs and obtain an accuracy of 98.94% on test data.

A.2 Traffic Sign

Dataset To have a fair comparison with DOA [11], we pick the same 16 traffic signs from the dataset LISA [5] with 3,509 training and 1,148 validation images. Following the prior works [2, 11], we further sample 40 stop signs from the validation set as the test data to

evaluate performance of the stop sign classification. Similarly, all the data are processed by standard crop-and-resize to 32×32 pixels. Same with VGGFace, we use the training set to train the target model, the GDPA generator and robust models with adversarial training. We use the test set to evaluate the performance of the target model, patch attack and adversarial defense.

Target Model We use the LISA-CNN [2] as the target model, which contains three convolutional layers and one fully-connected layer. We use the Adam Optimizer with initial learning rate 0.1 and drop the learning rate by 0.1 every 10 epochs. We set the batch size to 128. After 30 epochs, we achieve an accuracy of 98.69% on the validation set, and 100% accuracy on the test data.

A.3 ImageNet

Dataset ImageNet [1] is a well-known large scale object recognition benchmark. To develop the training and validation sets to train and evaluate the GDPA generator and robust models with adversarial training, we follow Moosavi-Dezfooli et al. [6] to select a subset of $10,000$ images from ImageNet training set (randomly choose ten images for each class) as our training set, and use the whole ImageNet validation set ($50,000$ images) as our validation set.

Target Model Following Poursaeed at el. [8], we use a pre-trained VGG19 model [10] from PyTorch library as the target model. This model achieves an accuracy of 72.4% on the validation set.

B Patch Attacks

Eyeglasses Attack This is an effective physically realizable patch attack developed by Sharif et al. [9]. It first initializes the eyeglass frames with 5 different colors, and chooses the color with the highest cross-entropy loss as starting color. For each update step, it divides the gradient value by its maximum and multiplies the results with the learning rate. Then it only keeps the gradient value in the eyeglass frame area. Finally, it clips and rounds the pixel values to keep them in the valid range. We evaluate the eyeglasses attack on the test set of VGGFace.

Sticker Attack Proposed by Evtimov et al. [2], this is another physically realizable patch attack. It initializes the stickers on the stop signs with random noise at fixed locations. For each update step, it uses the Adam optimizer with the learning rate 0.1 (and default parameters) to maximize the classification loss of the target model. Just as the other patch attacks, adversarial perturbations are restricted to the mask area; in our experiments, we use the same collection of small rectangles as in [2]. We evaluate the sticker attack on the test set of Traffic Sign.

C GDPA Network Architecture and Training Details

Network Architecture For VGGFace and ImageNet, both having images of size 224×224, we adopt the encoder network structure G_E from the work of image-to-image translation [12]. For the Traffic Sign dataset, which has images of size 32×32, we adopt a CNN of 3 convolutional layers with kernel size 4 and stride 2 as the encoder network G_E. We then use a neural network of one fully-connected layer with output size $3 \times w' \times h'$ as the pattern decoder G_P, and a neural network of one fully-connected layer with output size 2 as the location decoder G_L.

GDPA Training Details Following Algorithm 1, we train the GDPA generator G by using the Adam optimizer with an initial learning rate of 0.1 for VGGFace and ImageNet, and 0.01 for Traffic Sign. We drop the learning rate by 0.2 every 10 epochs and train the generator for 30 epochs. We set the batch size to 32 and β to 3000, which we find works well across various architectures and datasets in our experiments.

GDPA-AT Training Details Following Algorithm 2, we train the GDPA generator G and target model T iteratively. We initialize the generator with a pre-trained GDPA generator and the target model with a cross-entropy trained model. We set the w' and h' to 70 for VGGFace and Imagenet and 7 for Traffic Sign during the adversarial training. We use the Adam optimizer to train the generator and the target model, with a learning rate of 0.0001 for both VGGFace and Traffic Sign and 0.001 for imagenet, and drop the learning rate by 0.2 every 50 epochs. We use batch size 32 and train for 1000 epochs for VGGFace, 100 epochs for Imagenet and 5000 epochs for Traffic Sign.

D Ablation Study

D.1 Generate *pattern* vs p

Instead of generating *pattern* from the GDPA generator, we can generate p directly by adjusting the output size of pattern decoder G_P to $3 \times w \times h$. Directly generating p can simplify the pipeline of GDPA as we do not need to translate *pattern* to generate p in two steps. Thus, it's worth investigating which design choice works better. Table 1 shows the results comparing these two design choices. As we can see, generating *pattern* achieves significantly higher ASRs than generating p directly. We conjecture that this is because p has a larger space to optimize than *pattern*, and thus is more difficult to optimize. Hence, in our GDPA pipeline we generate *pattern* first and then translate *pattern* to generate p.

	Generate *pattern*	Generate p
Traffic Sign	**87.9%**	69.7%
VGGFace	**46.3%**	24.9%
ImageNet	**96.3%**	63.8%

Table 1: ASRs of GDPA when generating *pattern* vs. p.

D.2 Visibility α vs. ASR

In Section 4.1, we investigate the impact of visibility parameter α of Eq. 6 on GDPA's ASR. Figure 1 visualizes some example perturbed images generated by GDPA with different α's and patch sizes. As we can see, by using different α's, we can control the visibility of GDPA attack.

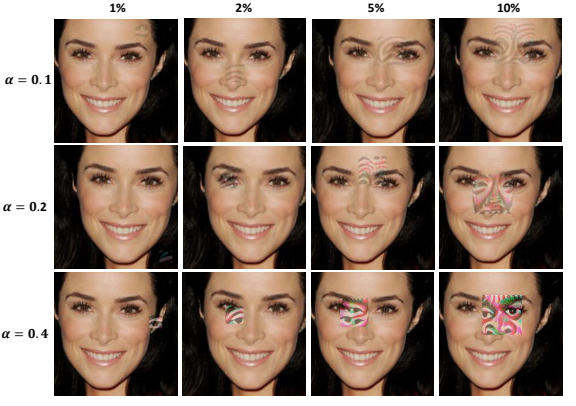

Figure 1: Perturbed images generated by GDPA with different α's and patch sizes (1%, 2%, 5% or 10% pixels).

D.3 Effect of β

The β in Eq. 1 controls the slope of tanh that constrains l_x and l_y in the range of $[-1,1]$. It is critical to find an appropriate value of β to train the GDPA generator. Intuitively, a too large or too small β value can cause different training difficulties. If β's value is too small, the tanh activation function saturates quickly and pushes l_x and l_y to the saturated value of -1 or 1, which corresponds to corners of an image. On the other hand, if β is too large, the tanh activation function has a slow transition from -1 to 1, which may not be able to push l_x and l_y away from the origin $[0,0]$ of an image, and likely causes ineffective training as well. Therefore, we treat β as a hyperparameter and tune it on the validation set. The results with different values of β on VGGFace are shown in Table 2 and Figure 2. It can be observed that we get the highest ASR with $\beta = 3000$. With small βs like 100 or 500, the patch location saturates at the corners of images; With large βs such as 5000 or 7000, the learned patch locations are close to the origin for most of the images. We find $\beta = 3000$ works well across a variety of architectures and datasets, and thus set it as the default value.

β	100	500	1000	3000	5000	7000
ASR	15.6	20.8	88.2	**88.4**	88.1	87.9

Table 2: ASRs of GDPA with different values of β. We use 5% of pixels as the patch size.

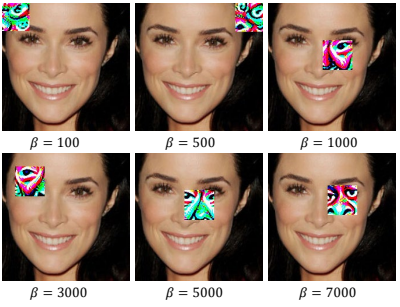

Figure 2: Perturbed images by GDPA with different values of β. The patch size is 5% of pixels.

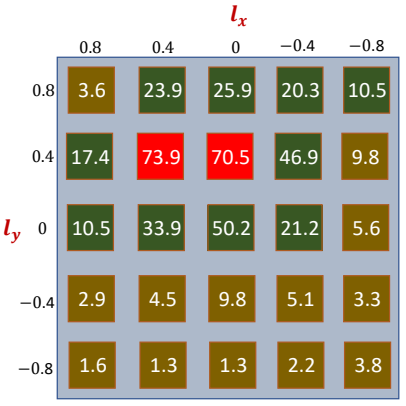

Figure 3: ASRs of static patch attack on different locations. We use different colors to denote ASRs in different ranges. Red: above 70%; Green: 10% - 70%; brown: below 10%. Dynamic GDPA achieves 76.4% ASR in this experiment.

E Eyeglass Attack Visualization

Figure 4 shows example results when using eyeglasses attack to evade a standard CE-trained model (a) and the GDPA-AT trained model (b). As we can see, the eyeglasses attack fails to attack the GDPA-AT trained model because it is not able to generate effective adversarial patterns on the eyeglass frames in 5 out of 6 cases, while being very successful on standard CE-trained model.

Figure 4: Perturbed images generated by eyeglasses attack on (a) standard CE-trained model, and (b) GDPA-AT trained model.

F Generating Static Patch Attack with GDPA

Contrary to dynamic patch attack, static patch attack uses a fixed patch location for all the images. To conduct static patch attack with GDPA, we set l_x and l_y to fix values instead of generating them from G_L. To compare the performance between dynamic and

static patch attacks, we conduct static patch attacks on VGGFace at 25 fixed locations ($l_x, l_y \in [-0.8, -0.4, 0, 0.4, 0.8]$). We use patch size 32×32 (2% of pixels) in the experiment.

Figure 3 shows the ASRs of static patch attacks at the 25 locations. As we can see, patch location is an important factor in the performance of static patch attack. Notably, patch locations around the area of eyes have the best ASRs. The highest ASR we obtain from static patch attack is 73.9%, while dynamic GDPA achieves 76.4%, demonstrating the effectiveness of dynamic GDPA.

G Generating Adversarial Attack with GDPA

Thanks to its generic formulation, we can also generate conventional adversarial attacks with GDPA by adjusting its pipeline slightly. To do this, we use a fixed mask of value 0.5 for all image pixels, and update the generator to produce p of the same size of image directly. To make sure the adversarial noise is within a small L_∞-norm bound, we multiple p by $\varepsilon/255$ such that the adversarial noise is bounded by $\varepsilon/255$. Finally, we scale the perturbed image by 2 and clip its pixel values to $[0,1]$ to create an adversarial example. We call this GDPA version of adversarial examples as GDPA-ADV.

We then compare the attack performances of GDPA-ADV with PGD [4] and PI-FGSM [3] on VGGFace and ImageNet. The PGD attack is generated with learning rate 10 for 20 iterations. The results with different ε's are provided in Table 3. We can observe that GDPA-ADV achieves slightly higher ASRs than PGD in all the cases considered. Compared with the other more competitive method PI-FGSM, GDPA-ADV has slightly worse ASR except on VGGFace when $\varepsilon = 6$. Some adversarial examples generated by GDPA-ADV on VGGFace are visualized in Figure 5. These adversarial examples look similar to the conventional adversarial examples.

		$\varepsilon = 6$	$\varepsilon = 8$	$\varepsilon = 10$
VGGFace	PGD	81.5	90.7	97.8
	PI-FGSM	79.7	**94.9**	**100**
	GDPA-ADV	**82.6**	91.9	98.2
ImageNet	PGD	88.5	90.3	91.2
	PI-FGSM	**92.9**	**94.0**	**94.8**
	GDPA-ADV	89.3	93.2	94.3

Table 3: ASRs of the adversarial attacks generated by PGD, PI-FGSM and GDPA-ADV.

H GDPA-AT against Adversarial Attack

We evaluate the robustness of models under conventional adversarial attacks, such as the PGD attack [4]. The results are reported in Table 4, where different PGD attack strengths ε have been considered. We set step size as 20 and iterations as 300. It can be observed that

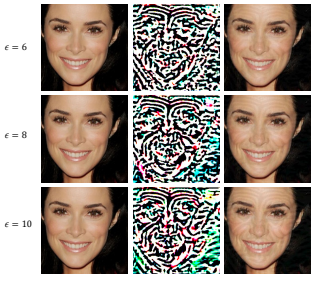

Figure 5: Adversarial examples generated by GDPA-ADV with different ε's. Left: original images; Middle: adversarial noise scaled to $[0,1]$ for visualization; Right: adversarial examples.

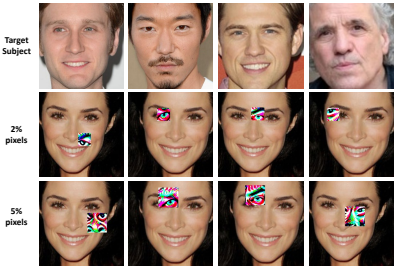

Figure 6: Perturbed images generated by GDPA with targeted attack on VGGFace. Each column corresponds to one targeted attack with a different target subject.

GDPA-AT achieves significantly higher robustness than DOA against the PGD attack. More interestingly, the accuacies that GDPA-AT achieve are almost on par with PGD-AT even though GDPA is a patch attack algorithm. We believe this is because during the adversarial training process, GPDA generates the adversarial patches to attack the classifier iteratively; even though each patch attack is localized, the combination of all patch attacks generated during the iterative process resembles a whole image attack that PGD usually produces. For this reason, the model trained by GDPA-AT can defend conventional adversarial attacks.

These results demonstrate that GDPA-AT is a generic defense algorithm that can defend both patch attacks and conventional adversarial attacks, while PGD-AT and DOA fail on one of them.

	VGG Face					Traffic Sign				
Attack Strength (ε)	0	2	4	8	16	0	2	4	8	16
CE training	98.9	44.4	1.7	0	0	98.7	89.5	61.6	24.6	5.1
PGD-AT [9]	97.3	96.9	96.6	96.1	95.8	97.5	95.8	94.6	92.9	91.0
DOA-Grad [11]	97.5	33.4	0.4	0	0	95.6	91.2	79.5	46.9	6.7
DOA-Exh [11]	98.5	35.7	0.4	0	0	92.9	89.5	77.1	42.8	5.8
GDPA-AT	**98.9**	**95.1**	**94.9**	**94.6**	**94.5**	**98.5**	**94.7**	**93.5**	**92.2**	**90.3**

Table 4: The accuracies of different robust models on (a) VGGFace, and (b) Traffic Sign when under the PGD attack.

I Cross Attacks and Defenses

In this section, we compare the defense performances of PGD-AT, DOA and GDPA-AT when they are attacked by their corresponding attack algorithms. In this experiment, the PGD attack uses $\varepsilon = 8$, and ROA and GDPA use 10% pixels as patch size. The results on

VGGFace are shown in Table 5. As we can see, PGD-AT achieves the highest robustness under the PGD attack, but is not very robust under the ROA and GDPA attacks. On the other hand, DOA achieves decent robustness under the ROA and GDPA attacks, but fails completely under the PGD attack. Notably, GDPA-AT is the only defense algorithm that achieves almost the highest robustness under all three attacks. It's expected that GPDA-AT would be robust under the ROA and GDPA attacks since both are patch attacks. An explanation of the robustness of GDPA-AT under the PGD attack is provided in Section 4.2.

AT \ Attack	PGD	ROA	GDPA
PGD-AT	**96.1**	32.8	30.5
DOA	0	88.1	86.9
GDPA-AT	94.6	**90.4**	**88.2**

Table 5: Accuracies of adversarially trained models under PGD, ROA and GDPA attacks.

J Additional Results on Targeted Attack

Figure 6 provides additional perturbed images generated by targeted GDPA attack on VGGFace. The top row shows the target subjects, while the bottom two rows show the perturbed images with different patch sizes. As we can see, the patches generated by GDPA attempt to replace the corresponding face features with the ones from the target subjects.

References

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 2009.

[2] Ivan Evtimov, Kevin Eykholt, Earlence Fernandes, Tadayoshi Kohno, Bo Li, Atul Prakash, Amir Rahmati, and Dawn Song. Robust physical-world attacks on machine learning models. In *CoRR*, 2017.

[3] Lianli Gao, Qilong Zhang, Jingkuan Song, Xianglong Liu, and Heng Tao Shen. Patch-wise attack for fooling deep neural network. In *European Conference on Computer Vision*, 2020.

[4] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Machine Learning*, 2018.

[5] Andreas Mogelmose, Mohan Manubhai Trivedi, and Thomas B Moeslund. Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey. *IEEE Transactions on Intelligent Transportation Systems*, 2012.

[6] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[7] Omkar M Parkhi, Andrea Vedaldi, and Andrew Zisserman. Deep face recognition. *BMVC*, 2015.

[8] Omid Poursaeed, Isay Katsman, Bicheng Gao, and Serge Belongie. Generative adversarial perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[9] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 acm sigsac conference on computer and communications security*, 2016.

[10] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015.

[11] Tong Wu, Liang Tong, and Yevgeniy Vorobeychik. Defending against physically realizable attacks on image classification. In *International Conference on Learning Representations*, 2020.

[12] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networkss. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.