The Space of Adversarial Strategies

USENIX'23

Ryan Sheatsley*, **Blaine Hoak*,** Eric Pauley, Patrick McDaniel

*Equal Contribution. Thursday, August 10, 2023









Basic Iterative Method LaserAttack LowProFool Adversarial Texture Wasserstein Attack **Brendel and Bethge Attack** Adversarial Patch NewtonFool Targeted Universal Perturbation Attack **Projected Gradient Descent** Auto Conjugate Attack Frame Saliency Attack Square Attack Carlini and Wagner I₀ Geometric Decision Based Attack Carlini and Wagner I_2 Malware Gradient Descent Elastic Net Attack DPatch DeepFool Spatial Transforms Attack Fast Gradient Method RobustDPatch Sign-OPT ThresholdAttack Over The Air Flickering Attack **PixelAttack** Auto Attack Virtual Adversarial Method GRAPHITE Carlini and Wagner I. High Confidence Low Uncertainty Attack Universal Perturbation Attack **Boundary Attack** Carlini and Wagner ASR Decision Tree Attack Jacobian Saliency Map Approach HopSkipJump Feature Adversaries Zeroth-Order Optimization Imperceptible ASR Shadow Attack Simple Black-box Adversarial Attack





Surfaces compute the perturbation

$$BIM \begin{array}{l} \boldsymbol{X}_{0}^{adv} = \boldsymbol{X} \\ \boldsymbol{X}_{N+1}^{adv} = Clip_{X,\epsilon} \left\{ \boldsymbol{X}_{N}^{adv} + \alpha \underline{\operatorname{sign}} \left(\nabla_{X} J(\boldsymbol{X}_{N}^{adv}, y_{true}) \right) \right\} \end{array}$$

Travelers apply perturbations and other techniques that update the input

EAR	$s \leftarrow \underset{l \neq c}{\operatorname{argmin}} \frac{ f_{l}(x^{(i)}) - f_{c}(x^{(i)}) }{\ \nabla f_{l}(x^{(i)}) - \nabla f_{c}(x^{(i)})\ _{q}}$ $\delta^{(i)} \leftarrow \operatorname{proj}_{p}(x^{(i)}, \pi_{s}, C)$ $\delta^{(i)}_{\operatorname{orig}} \leftarrow \operatorname{proj}_{p}(x_{\operatorname{orig}}, \pi_{s}, C)$	$\begin{bmatrix} z^{(k+1)} \leftarrow P_{\mathcal{S}} \left(x^{(k)} + \eta \nabla f(x^{(k)}) \right) \\ x^{(k+1)} \leftarrow P_{\mathcal{S}} \left(x^{(k)} + \alpha (z^{(k+1)} - x^{(k)}) \\ + (1 - \alpha) (x^{(k)} - x^{(k-1)}) \right) \end{bmatrix} APGD$
FAD	compute α as in Equation (9) $x^{(i+1)} \leftarrow \operatorname{proj}_C \left((1-\alpha) \left(x^{(i)} + \eta \delta^{(i)} \right) \right)$	$\hat{l} \leftarrow \arg\min_{k \neq \hat{k}(\boldsymbol{x}_0)} \frac{ f'_k }{\ \boldsymbol{w}'_k\ _2}$ $\boldsymbol{r}_i \leftarrow \frac{ f'_i }{\ \boldsymbol{w}'_i\ _2} \boldsymbol{w}'_i$ DeepFool
	$+ \alpha (x_{\text{orig}} + \eta \delta_{\text{orig}}))$ Compute forward derivative $\nabla \mathbf{F}(\mathbf{X}^*)$	$egin{aligned} egin{aligned} egi$
JSMA	$p_1, p_2 = \text{saliency}_map(\nabla \mathbf{F}(\mathbf{X}^*), \Gamma, \mathbf{Y}^*)$ Modify p_1 and p_2 in \mathbf{X}^* by θ	$x^{t+1} = \Pi_{x+S} \left(x^t + \alpha \operatorname{sgn}(\nabla_x L(\theta, x, y)) \right) PGD$





Components of the traveler and surface. Arrows represent the progression of an an attack through components at each iteration.





Surface Components										Traveler Components							
Losses: Cross-Entropy Carlini-Wagner Loss Identity Loss Difference of Logits Ratio Loss							Cl	Rand	dom-R	estart:	Ena	bled, D	Disable	d			
Saliency Maps: ℓ_p -norm		$\mathbb{SM}_{J}, \mathbb{SM}_{D}, \mathbb{SM}_{I}$ $\ell_{0}, \ell_{2}, \ell_{\infty}$						Change of variables: Optimizer:			SGD, Adam, MBS, BWSGD						
BIM	•	0	0	0	0	0	•	0	0	•	0	0	•	0	0	0	
PGD	•	0	0	0	0	Ō	•	0	0	•	•	0	•	0	0	0	
JSMA	0	0	•	0	•	0	0	•	0	0	0	0	•	0	0	0	
DF	0	0	•	0	0	•	0	0	•	0	0	0	•	0	0	0	
CW	0	•	0	0	0	0	•	0	•	0	0	•	0	•	0	0	
APGD-CE	•	0	0	0	0	0	•	0	0	٠	•	0	0	0	٠	0	
APGD-DLR	0	0	0	•	0	0	•	0	0	•	•	0	0	0	•	0	
FAB	0	0	•	0	0	•	0	0	•	0	0	0	0	0	0	•	
	C.	Chr	\sim	DIP	SM	SNS	SM	6	5	lo	Rp	Sr	SCD	Adap	MBS	Physic	



Attack Algorithms **Surface Components Traveler Components** Random-Restart: Enabled, Disabled Losses: Cross-Entropy Carlini-Wagner Loss Identity Loss Difference of Logits Ratio Loss Saliency Maps: SM_J, SM_D, SM_I Change of Variables: Enabled, Disabled $\ell_0, \ell_2, \ell_\infty$ ℓ_p -norm Optimizer: SGD, Adam, MBS, BWSGD BIM PGD JSMA DF CW APGD-CE APGD-DLR FAB PANSCO Sr Rp MBS Co Chi 0ZS SA SA SCS Ż SA 6 5 Co

PGD:



DeepFool:

 $\nabla f_y(x)$



						At	tack A	lgori	thms							
Surface Components									Traveler Components							
Losses: Saliency Maps: ℓ _p -norm		Cros Carl Iden Dif: SM_J , ℓ_0 , ℓ	ss-Er lini- ntity ferer SM _D , $2, \ell_{\infty}$	Ntropy Wagne Loss Nce of SM _I	/ er Lo: 3 E Log:	ss its R	atio	Loss	Ch	Rand ange	lom-R of Vari Opti	estart: iables: mizer:	Ena Ena SGD	bled, E bled, E , Adam,	Disable Disable	d d BWSGD
BIM	•	0	0	0	0	0	•	0	0	•	0	0	•	0	0	0
PGD		0	0	0	0	0		0	0		•	0		0	0	0
JSMA	0	0		0				•	0	0	0	0		0	0	0
DF	0	0		0	0	•	0	0		0	0	0		0	0	0
CW	0		0	0	0		•	0		0	0	•	0	•	0	0
APGD-CE		0	0	0	0	0	•	0	\bigcirc			\circ	\bigcirc	0	•	0
APGD-DLR	\bigcirc	\bigcirc	\bigcirc	•	\bigcirc	0		\bigcirc	\bigcirc		•	\circ	\bigcirc	\circ	•	0
FAB	0	0	•	0	0	•	0	0	٠	\bigcirc	0	0	0	0	0	•
	C.	Chil	Ż	DIP	SA	SNO	SM	C	\$	lo	Rp	SV	SCD	Adam	MBS	Physic

PGD:

DeepFool:

 $\nabla \text{CE}(x, y)$ $\frac{|f_y(x) - f_k(x)|}{\nabla f_y(x) - \nabla f_k(x)}$

$$(\nabla f_y(x) - \nabla f_k(x)))$$

9



						At	tack A	lgori	thms									
Surface Components										Traveler Components								
Los Saliency Me ℓ _p -n	ses: aps: orm	Cross-Entropy Carlini-Wagner Loss Identity Loss Difference of Logits Ratio Loss SM_J, SM_D, SM_I $\ell_0, \ell_2, \ell_{\infty}$								Random-Restart: Change of Variables: Optimizer:			Enabled, Disabled Enabled, Disabled SGD, Adam, MBS, BWS(d d BWSGD		
BIM	•	0	0	0	0	0	•	0	0	•	0	0	•	0	0	0		
PGD		0	0	0	0	0	•	0	0		•	0	•	0	0	0		
JSMA	0	0		0		0	0	•			0	0		0	0	0		
DF	0	0		0	0		0	0	•	0	0	0		0	0	0		
CW	\bigcirc		0	0	0	0		0	•	0	0	•	\bigcirc	•	0	0		
APGD-CE		0	\bigcirc	0	0	0	•	0	0	•	•	0	0	\bigcirc	•	0		
APGD-DLR	\bigcirc	0	\circ		0	0		0	\bigcirc			0	\bigcirc	\circ		0		
FAB	0	0	•	0	0	•	0	0	•	0	0	0	0	0	0	•		
	C.	Chil	Ż	DIP	SM	SAN	SM	6	\$	Co	Rp	Ser	SCD	Adam	MBS	BASG		

PGD:
$$\delta = \alpha \cdot \text{sgn}(\nabla \text{CE}(x, y))$$

DeepFool: $\delta = \frac{|f_y(x) - f_k(x)|}{\|\nabla f_y(x) - \nabla f_k(x)\|_2^2} \cdot (\nabla f_y(x) - \nabla f_k(x))$



PGD:
$$\delta = \alpha \cdot \text{sgn}(\nabla \text{CE}(x, y)), x_0 = x + \mathcal{U}(-\epsilon, \epsilon)$$

DeepFool: $\delta = \frac{|f_y(x) - f_k(x)|}{\|\nabla f_y(x) - \nabla f_k(x)\|_2^2} \cdot (\nabla f_y(x) - \nabla f_k(x)), x_0 = x$



PGD:
$$\delta = \alpha \cdot \text{sgn}(\nabla \text{CE}(x, y)), x_0 = x + \mathcal{U}(-\epsilon, \epsilon), x_{i+1} = x_i + \delta$$

DeepFool: $\delta = \frac{|f_y(x) - f_k(x)|}{\|\nabla f_y(x) - \nabla f_k(x)\|_2^2} \cdot (\nabla f_y(x) - \nabla f_k(x)), x_0 = x, x_{i+1} = x_i + \delta$



Our **extensible** decomposition of **mutually compatible and independent** components allows us to build a vast attack space containing 576 attacks.



The Pareto Ensemble Attack





Evaluation

- Questions
 - When and why are attacks performant?
- Setup
 - Adversary has access to model parameters
 - CIC-MalMem2022, Malware Detection, 58k total (k-Fold), 4 classes
 - CIFAR-10, Object Classification, 50k train, 10k test, 10 classes
 - Fashion-MNIST, Clothing Classification, 60k train, 10k test, 10 classes
 - MNIST, Digit Recognition, 60k train, 10k test, 10 classes
 - NSL-KDD, Network Intrusion Detection, 125k train, 22k test, 5 classes
 - Phishing Websites, Phishing Detection, 10k total (k-Fold), 2 classes
 - UNSW-NB15, Network Intrusion Detection, 101k train, 53k test, 10 classes



	Component H ₁		Component H ₂		Condition	p-value	Effect Size
1.	SGD	is better than	BWSGD	when	Dataset = MNIST	$< 2.2 \times 10^{-308}$	99%
2.	Adam	is better than	BWSGD	when	Dataset = MNIST	$< 2.2 \times 10^{-308}$	99%
		:			:		
84.	Identity Loss	is better than	Difference of Logits Ratio Loss	when	Dataset = NSL-KDD	$< 2.2 \times 10^{-308}$	93 %
85.	SGD	is better than	BWSGD	when	<pre>SaliencyMap = Jacobian Saliency Map</pre>	$< 2.2 \times 10^{-308}$	92 %
		:			1		
393.	DeepFool Saliency Map	is better than	Jacobian Saliency Map	when	Dataset = FMNIST	$< 5 \times 10^{-6}$	66 %
394.	Cross-Entropy	is better than	Carlini-Wagner Loss	when	Change of Variables = Disabled	$< 5 \times 10^{-6}$	61 %
					1		
1689.	ℓ_0	is better than	ℓ_2	when	$\texttt{Threat Model} = \ell_2 + 1.0$	$9.8 imes 10^{-1}$	50 %
1690.	Identity Saliency Map	is better than	DeepFool Saliency Map	when	<code>Threat Model = $\ell_\infty + 0.4$</code>	1.0	49 %



- Change of Variables \rightarrow 100% disabled, 0% enabled
- Optimizers \rightarrow 50% Adam, 33% SGD, 16% MBS, 1% BWSGD
- Random Restart \rightarrow 61% enabled, 39% disabled
- Saliency Maps → 70% no Saliency Map, 30% either DeepFool or JSMA Saliency Map
- Loss → 47% Identity Loss, 34% Cross Entropy, 18% Carlini Wagner Loss, 1% DLR Loss



- Change of Variables \rightarrow 100% disabled, 0% enabled
- Optimizers \rightarrow 50% Adam, 33% SGD, 16% MBS, 1% BWSGD
- Random Restart \rightarrow 61% enabled, 39% disabled
- Saliency Maps → 70% no Saliency Map, 30% either DeepFool or JSMA Saliency Map
- Loss → 47% Identity Loss, 34% Cross Entropy, 18% Carlini Wagner Loss, 1% DLR Loss



- Change of Variables \rightarrow 100% disabled, 0% enabled
- Optimizers \rightarrow 50% Adam, 33% SGD, 16% MBS, 1% BWSGD
- Random Restart \rightarrow 61% enabled, 39% disabled
- Saliency Maps → 70% no Saliency Map, 30% either DeepFool or JSMA Saliency Map
- Loss → 47% Identity Loss, 34% Cross Entropy, 18% Carlini Wagner Loss, 1% DLR Loss



GitHub Repo: https://github.com/sheatsley/attacks



	main 🚽 🥲 1 branch 💿 0 tags		Go to file Add file -	<> Code -
•	sheatsley add perturbation visualizati	on figure for mnist	c23e797 19 hours ago	3 401 commits
	aml	adapt version computation to load static ver	sion file for non-dev ins	2 days ago
	examples	add perturbation visualization figure for mni	st	19 hours ago
۵	.gitmodules	add overleaf ref		7 months ago
۵	Dockerfile	refactor dockerfile to build off of models		20 hours ago
۵	LICENSE	add license		6 months ago
۵	README.md	typofixes		2 days ago
۵	setup.py	always generate version file		2 days ago
۵	tests.py	reset seeds after fit to ensure results consis	tency when loading pret	yesterday

Adversarial Machine Learning

Adversarial Machine Learning (am1) is a repo for measuring the robustness of deep learning models against white-box evasion attacks. Designed for academics, it is principally designed for use in fundamental research to understand adversarial examples, inputs designed to cause models to make a mistake. At its core, am1 is based on a series of techniques used in eight popular attacks:

- 1. APGD-CE (Auto-PGD with CE loss)
- 2. APGD-DLR (Auto-PGD with DLR loss)
- 3. BIM (Basic Iterative Method)
- 4. CW-L2 (Carlini-Wagner with I2 norm)
- 5. DF (DeepFool)

README.md

- 6. FAB (Fast Adaptive Boundary)
- 7. JSMA (Jacobian Saliency Map Approach)
- 8. PGD (Projected Gradient Descent)





Thank you

https://hoak.me



S@blaine_hoak





