

Adversarial Dictionary Learning

Jordan Frecon, Lucas Anquetil, Gilles Gasso, Stéphane Canu

Chaire IA Raimo, ANR
Normandie Univ, INSA Rouen, LITIS, Normandie

CAp 2021

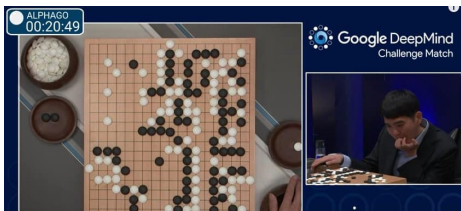
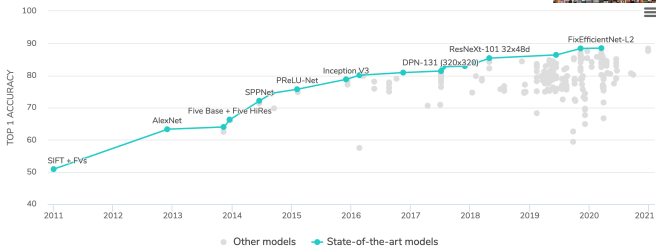


- 1 Adversarial learning
- 2 Adversarial dictionary learning framework
 - Principle
 - Algorithmic solutions
 - Generation of adversary examples
 - Defense mechanism
- 3 Numerical experiments
- 4 Conclusion

SCADA -> ADiL (Adversarial Dictionary Learning)

The amazing achievements of deep learning

Image Classification on ImageNet



Attacks against autonomous vehicles



Eykholt et al, Robust Physical-World Attacks on Deep Learning Visual Classification, CVPR 2018

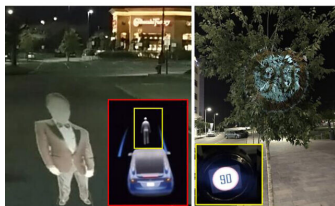


Zhang et al., CAMOU: Learning Physical Vehicle Camouflages to Adversarially Attack Detectors in the Wild, ICLR 2019



<https://www.mcafee.com/blogs/other-blogs/mcafee-labs/model-hacking-ad-as-to-pave-safer-roads-for-autonomous-vehicles/>

Nassi et al., Phantom of the ADAS: Securing Advanced Driver-Assistance Systems from Split-Second Phantom Attacks, 2020
 Qayyum, et al., Securing Connected & Autonomous Vehicles: Challenges Posed by Adversarial ML, IEEE Communications, 2019

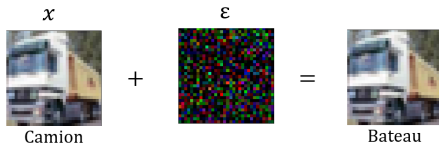


Definition (Adversarial example x')

Given an observed (also called natural or clean) example x , x' is

- a slight modification of x (e.g. such that $\|x - x'\| \leq \epsilon$)
- but having a different label prediction by f (i.e. fooling f)

$$\operatorname{argmax}_{k \in \{1, \dots, c\}} f(x'; \theta) \neq \operatorname{argmax}_{k \in \{1, \dots, c\}} f(x; \theta),$$



Explaining and Harnessing Adversarial Examples, I. Goodfellow et al, ICLR 2015

How to craft adversarial examples?

- Specific: for a given x_i

$$x'_i = x_i + \varepsilon(x_i)$$

- ▶ FGSM [GSS15, KGB17]

$$\varepsilon(x_i) = \delta \operatorname{sign}(\nabla_{x_i} H(f(x_i; \theta), y_i)),$$

- ▶ DeepFool [MFF16]

$$\varepsilon(x_i) = \operatorname{argmin}_{\varepsilon} \|\varepsilon\|, \text{ s.t. } \operatorname{argmax}_k f(x_i + \varepsilon; \theta) \neq \operatorname{argmax}_k f(x_i; \theta)$$

- Universal [MDFFF17]: for any example

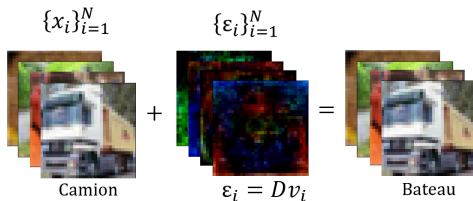
$$\varepsilon(x_i) = \operatorname{argmax}_{\varepsilon} \sum_{j=1}^N H(f(x_j + \varepsilon; \theta), y_j) \quad \text{s.t.} \quad \|\varepsilon\|_p \leq \epsilon,$$

- Use a dictionary D :

$$\varepsilon(x_i) = Dv_i$$

- 1 Adversarial learning
- 2 Adversarial dictionary learning framework
 - Principle
 - Algorithmic solutions
 - Generation of adversary examples
 - Defense mechanism
- 3 Numerical experiments
- 4 Conclusion

Adversarial dictionary learning: $\varepsilon(x_i) = Dv_i$



$$\underset{[D, v]}{\text{minimize}} \sum_{i=1}^N \underbrace{\ell_i(x_i + Dv_i)}_{\text{adversary}} + \underbrace{\lambda_1 \|v_i\|_1}_{\text{sparse}} + \underbrace{\lambda_2 \|Dv_i\|_2^2}_{\varepsilon_i \text{ small}}$$

$$D = \left(\begin{array}{c} \text{[7 adversarial perturbation images]} \end{array} \right)$$

D universal, $v_i \in \mathbb{R}^M$ specific ($M \ll N$)

Algorithmic solutions

- Full-batch version: ADiL
 - ▶ with proofs
- Stochastic version: SADiL
 - ▶ that scales...

Full-batch version: ADiL

$$\underset{\substack{D \in \mathbb{R}^{P \times M} \\ V \in \mathbb{R}^{M \times N}}}{\text{minimize}} \mathcal{L}(D, V) \triangleq F(D, V) + \Omega(D, V)$$

- Smooth supervised fitting term

$$F(D, V) = \sum_{i=1}^N \lambda_2 \|Dv_i\|^2 + H(f(x_i + Dv_i; \theta), t_i)$$

- Non-smooth regularization

$$\Omega(D, V) = \iota_{\mathcal{C}}(D) + \sum_{i=1}^N \lambda_1 \|v_i\|_1, \quad \mathcal{C} = \{D \mid \forall m, \|d_m\|_2 \leq 1\}$$

A sparse representation for a better dictionary

The proximal step

$$(D^{(k+1/2)}, V^{(k+1/2)}) = \underset{\substack{D \in \mathbb{R}^{P \times M} \\ V \in \mathbb{R}^{M \times N}}}{\operatorname{argmin}} F(D, V) + \Omega(D, V),$$

The proximal step

$$\begin{pmatrix} D^{(k+1/2)} \\ V^{(k+1/2)} \end{pmatrix} = \operatorname{prox}_{\gamma_k \Omega} \left(\begin{pmatrix} D^{(k)} \\ V^{(k)} \end{pmatrix} - \gamma_k \nabla F(D^{(k)}, V^{(k)}) \right),$$

Ω being separable, it yields that

$$\begin{pmatrix} D^{(k+1/2)} \\ V^{(k+1/2)} \end{pmatrix} = \begin{pmatrix} \operatorname{Proj}_{\mathcal{C}} & (D^{(k)} - \gamma_k \nabla_D F(D^{(k)}, V^{(k)})) \\ \operatorname{Soft}_{\gamma_k \lambda_1} & (V^{(k)} - \gamma_k \nabla_V F(D^{(k)}, V^{(k)})) \end{pmatrix},$$

Algorithm 1 ADiL

Require: Parameter $\delta \in]0, 1[$, $D^{(0)} \sim \mathcal{N}(0_{P \times M}, 1_{P \times M})$, $V^{(0)} = 0_{M \times N}$

for $k = 0$ to $K - 1$ **do**

Proximal-gradient step

$$D^{(k+1/2)} = \text{Proj}_{\mathcal{C}}(D^{(k)} - \gamma_k \nabla_D F(D^{(k)}, V^{(k)}))$$

$$V^{(k+1/2)} = \text{Soft}_{\gamma_k \lambda_1}(V^{(k)} - \gamma_k \nabla_V F(D^{(k)}, V^{(k)}))$$

Armijo-like backtracking

$$d_D^{(k)} = D^{(k+1/2)} - D^{(k)}$$

$$d_V^{(k)} = V^{(k+1/2)} - V^{(k)}$$

$$i_k = 0$$

repeat

$$\tilde{D}^{(k)} = D^{(k)} + \delta^{i_k} d_D^{(k)}$$

$$\tilde{V}^{(k)} = V^{(k)} + \delta^{i_k} d_V^{(k)}$$

$$i_k = i_k + 1$$

until decreasing criterion satisfied

$$D^{(k+1)} = \tilde{D}^{(k)}$$

$$V^{(k+1)} = \tilde{V}^{(k)}$$

end for

return $\{D^{(K)}, V^{(K)}\}$

Convergence

Theorem (Convergence [BLP⁺17])

Let $\{D^{(k)}, V^{(k)}\}_{k \in \mathbb{N}}$ be the sequence of ADiL Algorithm 1. Then,

- each limit point of $\{D^{(k)}, V^{(k)}\}_{k \in \mathbb{N}}$ is a stationary point of ADiL
- $\{\mathcal{L}(D^{(k)}, V^{(k)})\}_{k \in \mathbb{N}}$ converges to the limit point objective value

In addition, if \mathcal{L} satisfies the Kurdyka-Łojasiewicz property at any point, then the sequence converges to a stationary point of ADiL

Stochastic version: SADiL

Two ingredients: an **alternating scheme**

$$\begin{cases} D^{(k+1)} &= \text{Proj}_{\mathcal{C}} \left(D^{(k)} - \gamma_k \tilde{\nabla} F(D^{(k)}, V^{(k)}) \right), \\ V^{(k+1)} &= \text{Soft}_{\gamma_k \lambda_1} \left(V^{(k)} - \gamma_k \tilde{\nabla} F(D^{(k+1)}, V^{(k)}) \right), \end{cases}$$

$\tilde{\nabla} F$: **random estimate** of the gradient on a mini-batch $\mathcal{B}_k \sim \{1, \dots, N\}$

$$\tilde{\nabla} F(D, V) = \frac{N}{|\mathcal{B}_k|} \sum_{i \in \mathcal{B}_k} \nabla F_i(D, V).$$

For $|\mathcal{B}_k| = N$, we recover SADiL

Generation of adversary examples

Design of adversarial perturbations to unseen examples.

1 Use ADiL with fixed D to find $v^{(K)}$

2 Project onto the input manifold $\mathcal{X} \subseteq \mathbb{R}^P$

$$x' = \text{Proj}_{\mathcal{X}} \left(x + Dv^{(K)} \right)$$

Defense mechanism

Problem (Defense mechanism)

$$\underset{\theta \in \Theta}{\text{minimize}} \quad \mathbb{E}_{\{x,y\} \sim \mathcal{D} \cup \mathcal{A}} H(f(x; \theta), y), \quad (1)$$

where $\mathcal{D} \cup \mathcal{A}$ is the augmented training set

Two manners of constructing the adversarial set with correct labeling.

(Adversarial training) $\mathcal{A} = \{x_i + \hat{D}\hat{v}_i, y_i\}_{i=1}^N,$

(Noise injection) $\mathcal{A} = \{x_i + \hat{D}z_i, y_i\}_{i=1}^N$ with $z_i \sim \text{Laplace}(0, b),$

where b is estimated by fitting a Laplacian distribution to the \hat{v}_i 's.

1 Adversarial learning

2 Adversarial dictionary learning framework

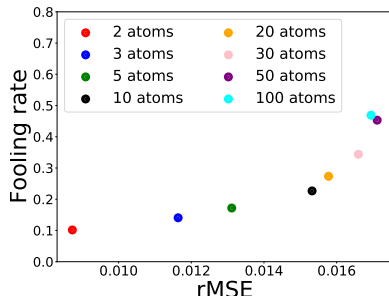
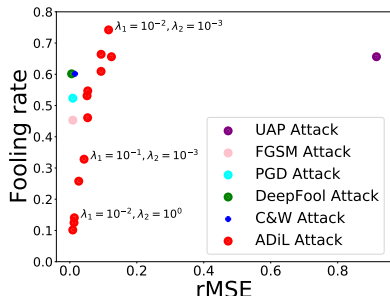
- Principle
- Algorithmic solutions
- Generation of adversary examples
- Defense mechanism

3 Numerical experiments

4 Conclusion

Experimental results: LeNet classifier on CIFAR-10

$$\text{rMSE}: (1/|\mathcal{T}_2|) \sum_{i=1}^{|\mathcal{T}_2|} \|Dv_i\|^2 / \|x_i\|^2$$



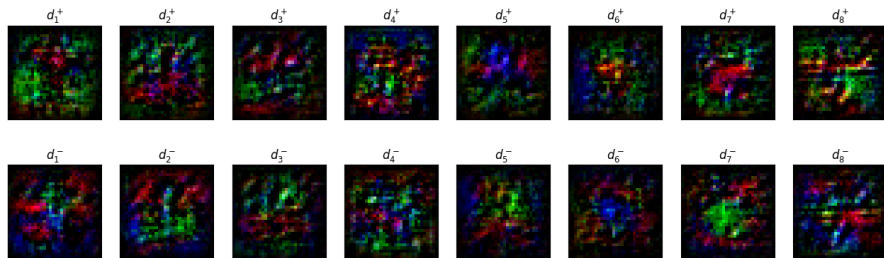
Experimental results on ResNet18 classifier

		PGD	DeepFool	C&W	ADiL	UAP
CIFAR-10	Fool. Rate	54.69%	74.22%	74.22%	90.63%	77.34%
	rMSE	0.0091	0.0056	0.032	0.071	0.747
ImageNet	Fool. Rate	22.66%	17.19%	3.91%	38.28%	100%
	rMSE	0.00054	0.00022	0.00025	0.0458	1.52

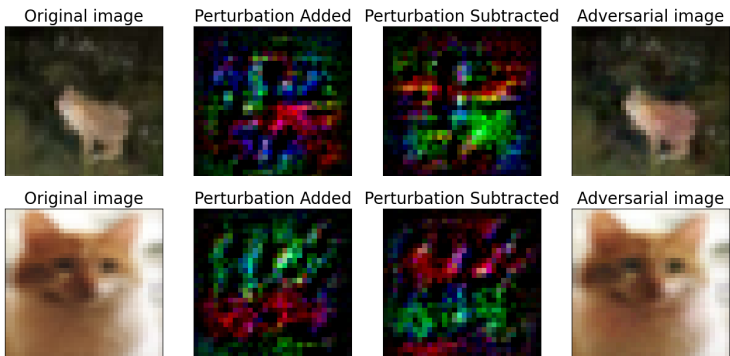
Defense mechanism for LeNet on CIFAR-10

M_{attacker}	2 atoms	5 atoms	10 atoms	15 atoms	20 atoms
No Defense	25.78%	56.25%	60.15%	46.09%	57.81%
With Defense	15.62%	30.46%	53.90%	44.53%	56.25%

Dictionary of ADiL attacks for LeNet on CIFAR-10



Two examples of ADiL attacks for LeNet on CIFAR-10



Conclusion

- A new way to generate adversarial examples
- with a universal component D
 - ▶ interpretable?
 - ▶ transferable?
- efficient way to compute specific components v_i
- improve the defence mechanism to train robust NN

References I

- [BLP⁺17] Silvia Bonettini, Ignace Loris, Federica Porta, Marco Prato, and Simone Rebegoldi, *On the convergence of a linesearch based proximal-gradient method for nonconvex optimization*, *Inverse Problems* **33** (2017), no. 5, 055005.
- [GSS15] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy, *Explaining and harnessing adversarial examples*, 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (Yoshua Bengio and Yann LeCun, eds.), 2015.
- [KGB17] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio, *Adversarial examples in the physical world*, 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings, OpenReview.net, 2017.
- [MDFFF17] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard, *Universal adversarial perturbations*, Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1765–1773.
- [MFF16] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard, *Deepfool: A simple and accurate method to fool deep neural networks*, 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016, IEEE Computer Society, 2016, pp. 2574–2582.