Luring Transferable Adversarial Perturbations for Deep Neural Networks

Remi Bernhard[*], Pierre-Alain Moellic[*]
Jean-Max Dutertre[†]

[*] CEA-Leti, France
[†] Mines Saint-Etienne, France

IJCNN 2021

Context: Large-scale deployment of ML models.
→ Embedded / Cloud-based systems.

Adversarial examples: Attacks against the integrity of a machine learning model

Threat: Black-box transfer attacks
→ Defenses in the black-box context are weakly covered in the literature as compared to the numerous approaches focused on white-box attacks.

The luring effect

Main idea: Use a deception based approach
$\rightarrow$ Rather than try to prevent an attack, let's fool the attacker.

Implementation:

- A network $P : \mathcal{X} \rightarrow \mathcal{X}$ is pasted to M before the input layer. Augmented model: $T(x) = M \circ P(x)$ $(x \in \mathcal{X})$.
- P is designed such that adversarial examples do not transfer from $M \circ P$ to M.

P is designed and trained with a twofold objective:

- ***Prediction neutrality:*** $T(x) = M \circ P(x) = M(x)$;
- ***Adversarial luring:*** $M \circ P(x') \neq M(x')$ Best case: x' is inefficient (i.e. $M(x') = y$)

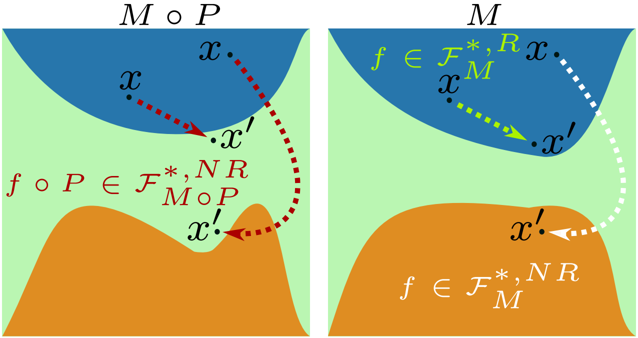
Specificities:

- Training P does not require a labeled data set, and fits any already trained model
- Compatible with existing white-box and purifier-based defense methods

Feature-based formalism from Ilyas et al., 2019:
A model learns useful features as functions $f : \mathcal{X} \rightarrow \mathbb{R}$.
For a given adversarial perturbation, a useful feature can be robust or non-robust.

Luring effect:
The adversary targets a non-robust feature of $M \circ P$, in the form of $f \circ P$, with f a useful feature for M.

- $\mathcal{F}^{*,R}_{M}$:
  robust useful features of M
- $\mathcal{F}^{*,NR}_{M}$:
  non-robust useful features of M
- $\mathcal{F}^{*,NR}_{M\circ P}$:
  non-robust useful features of $M \circ P$



Goal: Force M and $M \circ P$ to rely on different concepts to perform prediction.
$\Rightarrow$ The same adversarial perturbation does not fool M and $M \circ P$ the same way...
$\Rightarrow$ or fools $M \circ P$ but not M

How ?
Act on the logits sequence order of $M \circ P$ relatively to M:

- M: "class α is predicted, class β is the second possible class"
- $M \circ P$: "class α is predicted, the higher confidence given to class α, the smaller confidence given to class β"

Notations:

$h_i^M(x)$: logits of M for input x and class i
$h_i^{M\circ P}(x)$: logits of $M \circ P$ for input x and class i
α: predicted class by M for input x
β: second maximum value of h^M for input x
c: second maximum value of $h^{M\circ P}$ for input x

Luring Loss:

$$\mathcal{L}(x, M) = \underbrace{-\lambda\left(h_{\alpha}^{M\circ P}(x) - h_{\beta}^{M\circ P}(x)\right)}_{\text{widen the logit gap}} + \underbrace{max\left(0, h_{c}^{M\circ P}(x) - h_{\alpha}^{M\circ P}(x)\right)}_{\text{compulsory for prediction neutrality}}$$

Characterization of the luring effect

Baselines for comparison. Isolate the *luring effect* from other factors:

- ***Stack* model**: $M \circ P$ is retrained as a whole with the cross-entropy loss
- ***Auto* model**: P is an auto-encoder trained separately with binary cross-entropy loss
- ***C_E* model**: P is trained with the cross-entropy loss between the confidence score vectors $M \circ P(x)$ and $M(x)$ in order to mimic the decision of the target model M

Results

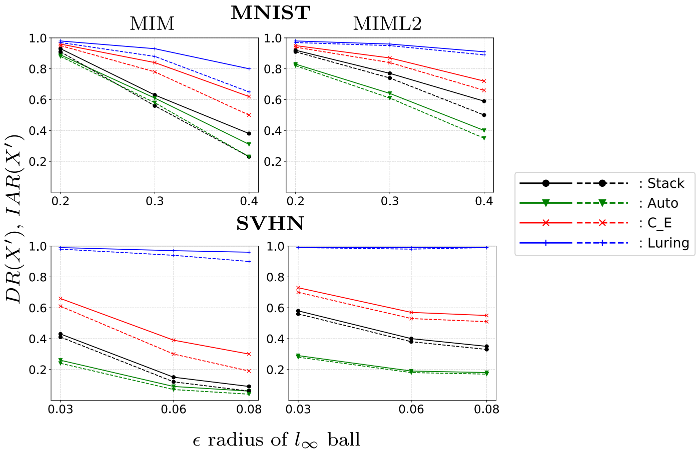


Figure: Disagreement Rate (solid line) and Inefficient Adversarial examples Rate (dashed line) for different attacks.

Complementary analysis

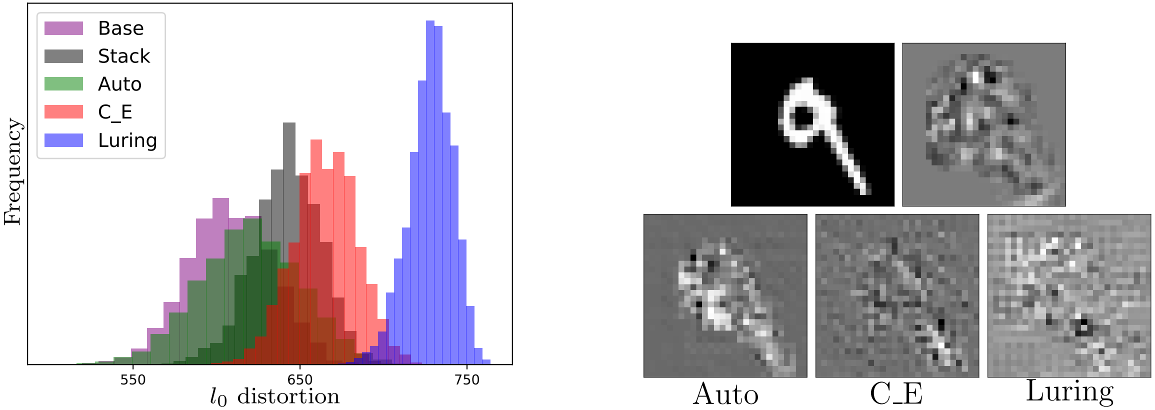


Figure: MNIST: l_0 adversarial distortion and saliency maps: (top) clean image and gradient of the cross-entropy loss with respect to input; (bottom) mapping gradients $\nabla_x P(x)$ for 3 augmented models.

Evaluation

Attacks

Gradient-free attacks:

- SPSA: the adversary has access to the logits of $M \circ P$
- ECO: score-based attack

Gradient-based attacks:

To perform an even more strict evaluation, and to anticipate future gradient-free attacks, we report the best results obtained with state-of-the-art transferability tuned attacks (noted MIM-W).

Evaluation

Results: Adversarial accuracy for $M \circ P$ (AC_{MoP}), M (AC_M), and Detection Adversarial Accuracy (DAC).

SVHN		STACK			AUTO			C_E			LURING		
	ϵ	AC_{MoP}	AC_M	DAC	AC_{MoP}	AC_M	DAC	AC_{MoP}	AC_M	DAC	AC_{MoP}	AC_M	DAC
SPSA	0.03	0.10	0.54	0.56	0.06	0.37	0.38	0.06	0.67	0.68	0.0	**0.96**	**0.97**
	0.06	0.01	0.21	0.24	0.0	0.10	0.11	0.0	0.37	0.42	0.0	**0.96**	**0.96**
	0.08	0.0	0.13	0.15	0.0	0.06	0.06	0.0	0.23	0.28	0.0	**0.94**	**0.96**
ECO	0.03	0.06	0.42	0.44	0.14	0.48	0.49	0.18	0.66	0.68	0.20	**0.97**	**0.98**
	0.06	0.0	0.11	0.12	0.06	0.09	0.11	0.1	0.35	0.39	0.1	**0.86**	**0.88**
	0.08	0.0	0.03	0.07	0.06	0.09	0.09	0.08	0.29	0.32	0.09	**0.84**	**0.86**
MIM-W	0.03	0.04	0.32	0.35	0.01	0.20	0.21	0.03	0.41	0.45	0.11	**0.81**	**0.87**
	0.06	0.0	0.06	0.09	0.0	0.03	0.05	0.0	0.10	0.18	0.0	**0.58**	**0.71**
	0.08	0.0	0.03	0.06	0.0	0.01	0.02	0.0	0.06	0.13	0.0	**0.48**	**0.67**

Results

Setup: ImageNet (ILSVRC2012), MobileNetV2

Results:

Table: ImageNet. AC_{MoP}, AC_M and DAC for different source model architectures.

	ϵ	C_E			LURING		
		AC_{MoP}	AC_M	DAC	AC_{MoP}	AC_M	DAC
MIM-W	4/255	0.0	0.23	0.35	0.00	**0.4**	**0.55**
	5/255	0.0	0.15	0.25	0.00	**0.28**	**0.43**
	6/255	0.0	0.08	0.18	0.00	**0.18**	**0.33**

Conclusion

Conclusion

Contributions:

- A conceptually innovative approach to improve the robustness of a model against transfer black-box adversarial perturbations: the *luring effect*
- Simple implementation: fits any pre-trained model, and does not require a labeled data set
- Characterization of the *luring effect* on MNIST, SVHN, CIFAR10, and extension to a black-box defense strategy
- Scalability to ImageNet

Perspectives:

Extend the *luring effect* to design a gray-box or white-box defense scheme