Adversarial Perturbations

Become invisible to AI

Romain PIERRE
Samuel LANDEAU
Christian BANIDE
Mohamed KHERRAZ

Examples of adversarial perturbations

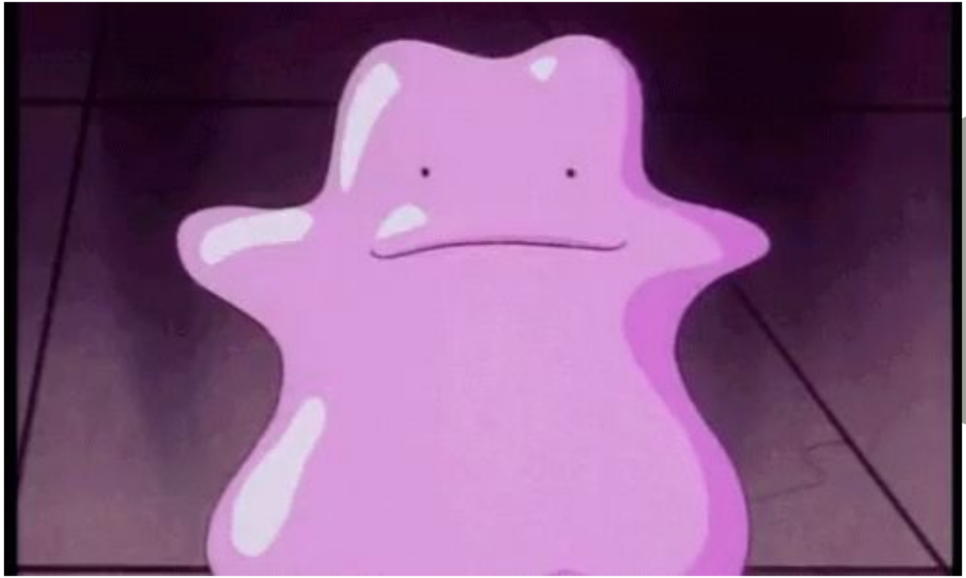


Table of contents

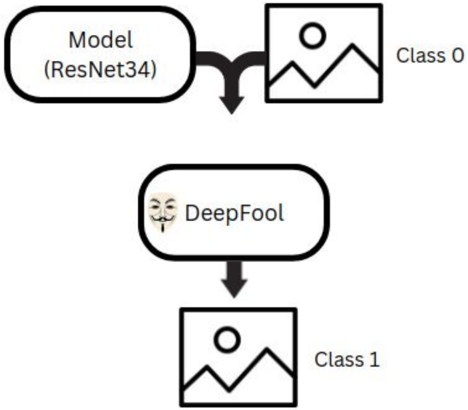
Class perturbation

Perturbing the class of an image using deepfool variations.

Class perturbation - Deepfool

Deepfool is a method that uses gradient and prediction logits of an image when passed in a machine learning model and adds the needed transformation to change its class.

- General Deepfool
  - Closest class
  - Target class
- Region Deepfool
  - Closest class
  - Target class

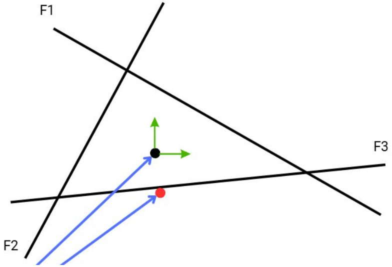


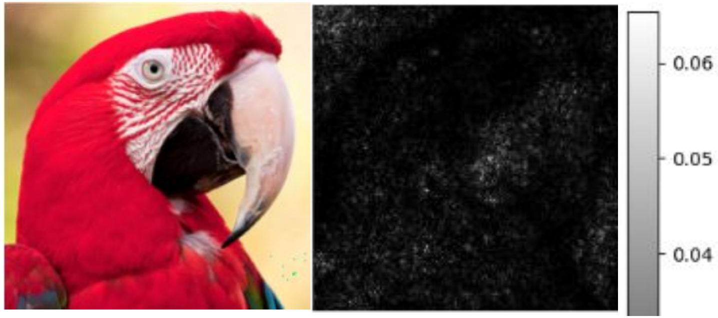
Class perturbation - General Deepfool

Algorithm 1 DeepFool: multi-class case

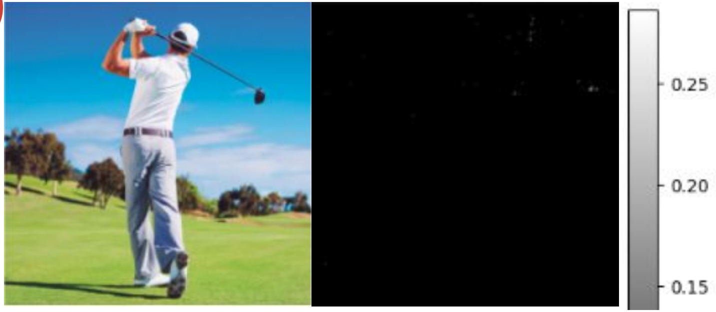
Require: Image x, classifier f.
Ensure: Perturbation $\hat{r}$.

Initialize $x_0 \leftarrow x$, $i \leftarrow 0$.
while $\hat{k}(x_i) = \hat{k}(x_0)$ **do**
  for $k \neq \hat{k}(x_0)$ **do**
    $w'_k \leftarrow \nabla f_k(x_i) - \nabla f_{\hat{k}(x_0)}(x_i)$
    $f'_k \leftarrow f_k(x_i) - f_{\hat{k}(x_0)}(x_i)$
  end for
  $\hat{l} \leftarrow \arg\min_{k \neq \hat{k}(x_0)} \frac{|f'_k|}{\|w'_k\|_2}$
  $r_i \leftarrow \frac{|f'_{\hat{l}}|}{\|w'_{\hat{l}}\|_2^2} w'_{\hat{l}}$
  $x_{i+1} \leftarrow x_i + r_i$
  $i \leftarrow i + 1$
end while
return $\hat{r} \leftarrow \sum_i r_i$





Flamingo



Assault rifle

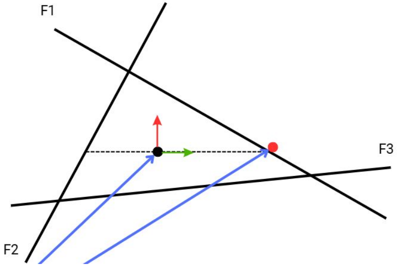
Class perturbation - Region Deepfool

Algorithm 2 DeepFool: multi-class case

Require: Image x, classifier f, mask M.

Ensure: Perturbation $\hat{r}$.

```
Initialize x_0 ← x, i ← 0.
while k̂(x_i) = k̂(x_0) do
    for k ≠ k̂(x_0) do
        w'_k ← M(i) · (∇f_k(x_i) − ∇f_{k̂(x_0)}(x_i))
        f'_k ← f_k(x_i) − f_{k̂(x_0)}(x_i)
    end for
    l̂ ← argmin_{k≠k̂(x_0)} |f'_k| / ||w'_k||_2
    r_i ← (|f'_l̂| / ||w'_l̂||_2^2) w'_l̂
    x_{i+1} ← x_i + r_i
    i ← i + 1
end while
return r̂ ← Σ_i r_i
```



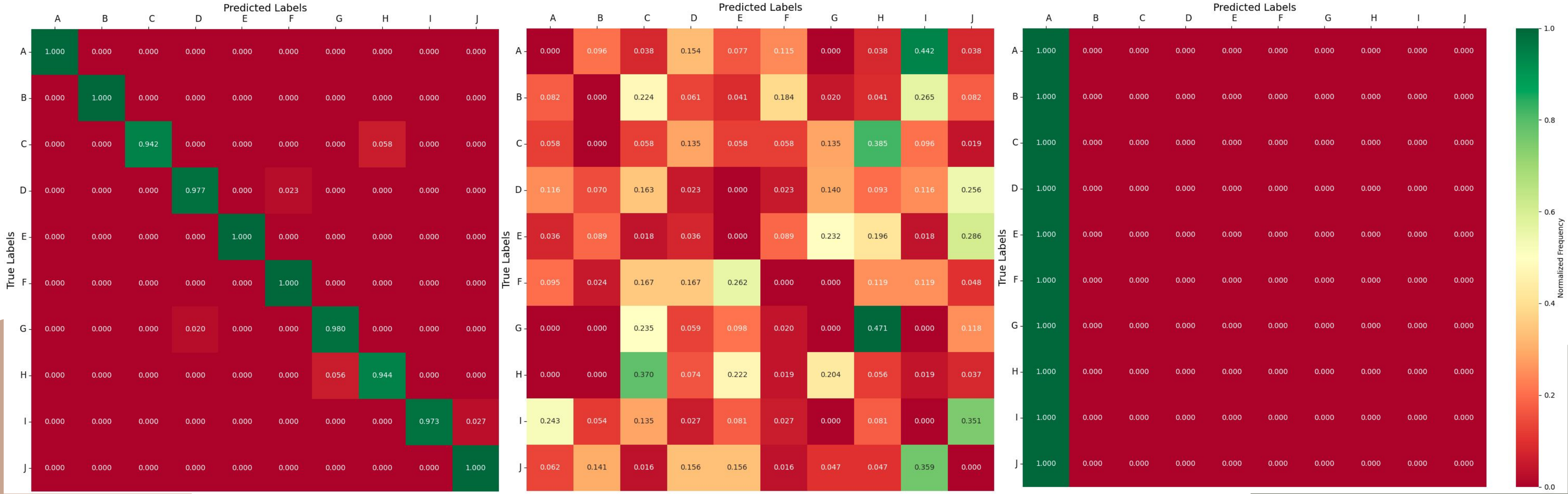
Oxcart (plane)

Assault rifle

Class perturbation - Evaluation

Note : The variations of methods do not change the confusion matrices but it changes the mean norm of the perturbations.

Dataset : CIFAR-10

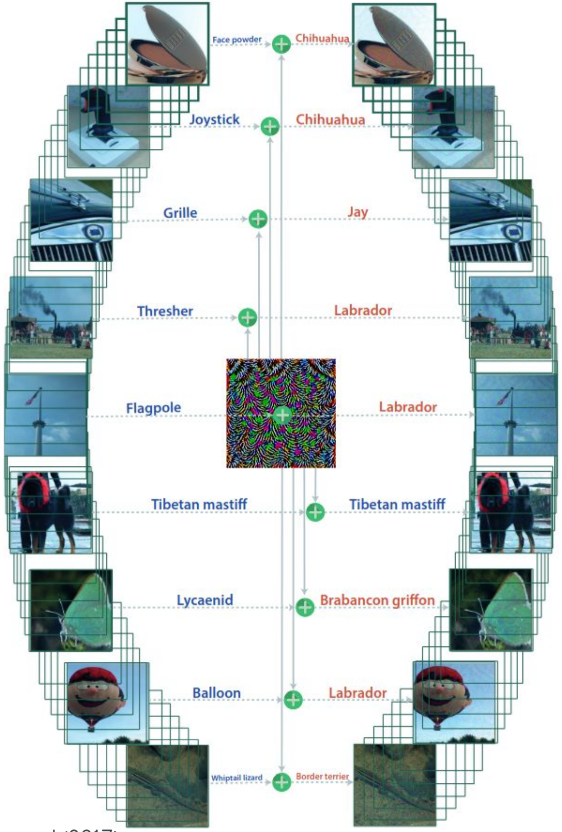


Universal perturbation

Perturbing the class of multiple images with the same noise.

Universal

- Instead of perturbing one image in particular, we are looking to create a universal perturbation for all the images.
- As this perturbation is based on a (model, dataset) pair, we will then compare it on different architectures in order to assess its generalization and effectiveness.



Source: Universal adversarial perturbations, Moosavi-Dezfooli, O. Fawzi, A. Fawzi, Frossard (2017)

Algorithm

Algorithm 1 Computation of universal perturbations.

```
input: Data points X, classifier k̂, desired ℓ_p norm of the perturbation ξ, desired accuracy on perturbed samples δ.
output: Universal perturbation vector v.
Initialize v ← 0.
while Err(X_v) ≤ 1 − δ do
    for each datapoint x_i ∈ X do
        if k̂(x_i + v) = k̂(x_i) then
            Compute the minimal perturbation that sends x_i + v to the decision boundary:
                Δv_i ← arg min_r ||r||_2 s.t. k̂(x_i + v + r) ≠ k̂(x_i).
            Update the perturbation:
                v ← P_{p,ξ}(v + Δv_i).
        end if
    end for
end while
```

1. **input:** Data points X, classifier $\hat{k}$, desired ℓ_p norm of the perturbation ξ, desired accuracy on perturbed samples δ.
2. **output:** Universal perturbation vector v.
3. Initialize $v \leftarrow 0$.
4. **while** $\mathrm{Err}(X_v) \leq 1 - \delta$ **do**
5. **for** each datapoint $x_i \in X$ **do**
6. **if** $\hat{k}(x_i + v) = \hat{k}(x_i)$ **then**
7. Compute the *minimal* perturbation that sends $x_i + v$ to the decision boundary:

$$\Delta v_i \leftarrow \arg\min_r \|r\|_2 \text{ s.t. } \hat{k}(x_i + v + r) \neq \hat{k}(x_i).$$

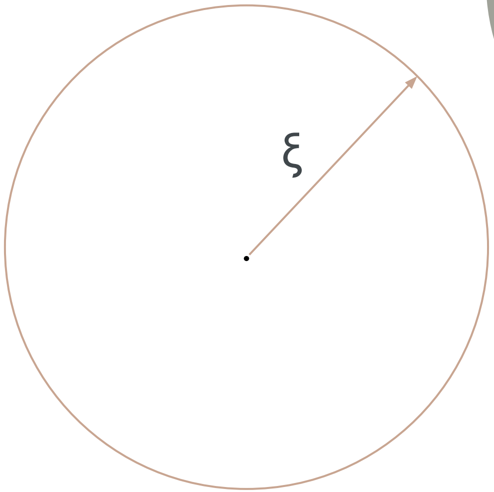
<- Deepfool

8. Update the perturbation:

$$v \leftarrow \mathcal{P}_{p,\xi}(v + \Delta v_i).$$

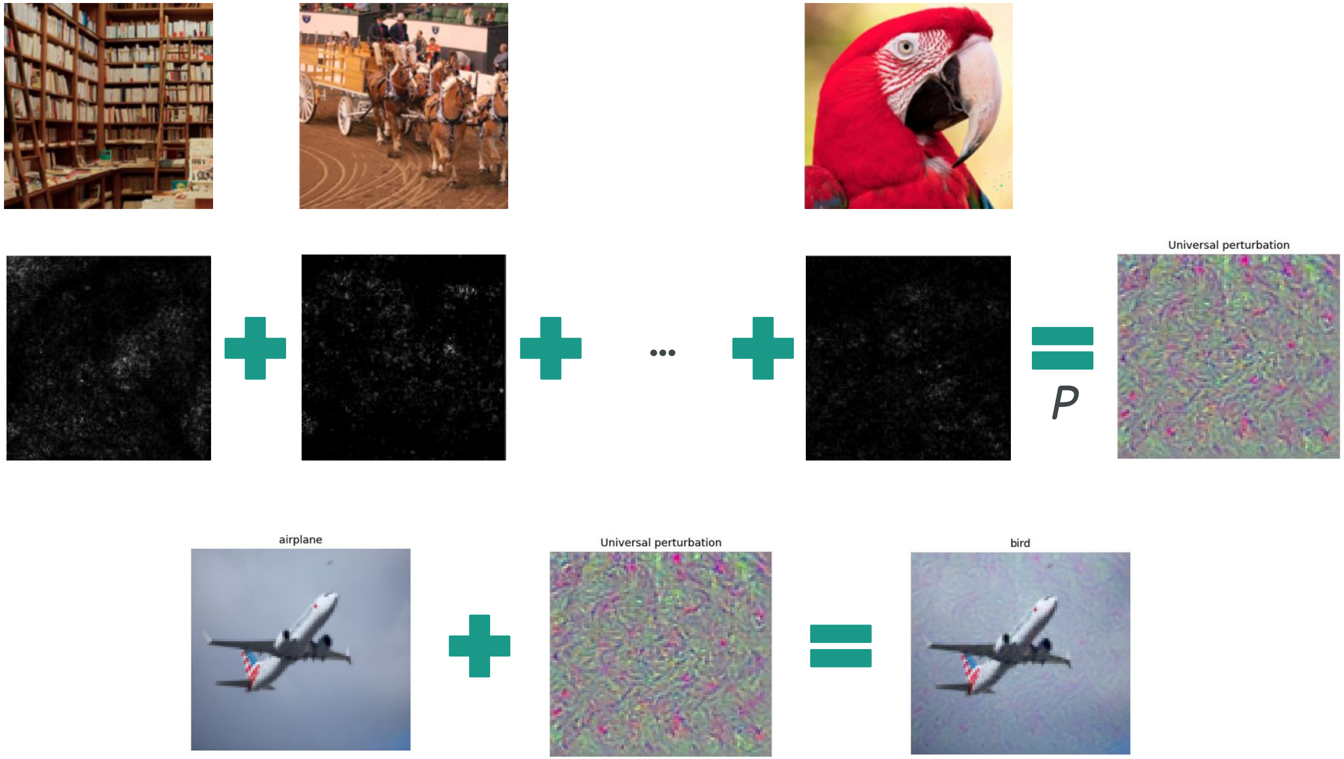
<- Projection

9. **end if**
10. **end for**
11. **end while**



Source: Universal adversarial perturbations, Moosavi-Dezfooli, O. Fawzi, A. Fawzi, Frossard (2017)

Example


Universal perturbation
P
airplane
Universal perturbation
bird

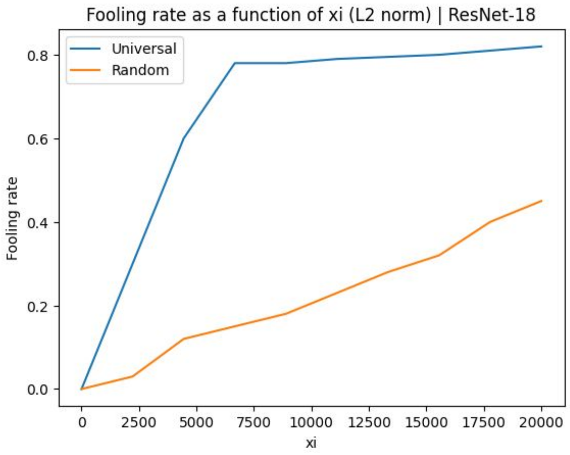
Generalization

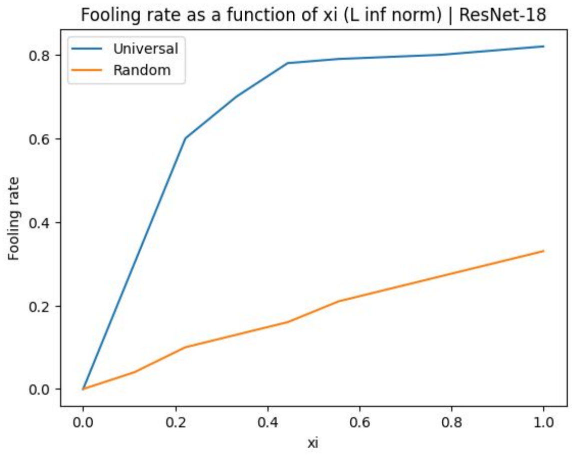
Dataset: STL-10: (3, 96, 96) x 5 000 [10 classes]

Network \ Perturbation	ResNet-18	VGG-11	MobileNet-V2	Random
ResNet-18	**76.4%**	26.7%	27.1%	19.0%
VGG-11	43.4%	**73.6%**	15.3%	12.5%
MobileNet-V2	53.3%	56.6%	**57.4%**	22.9%

- Universal perturbations are most effective on the model they are crafted for
- ResNet-18 has the best results and seems to be the most generalizable
- The efficiency of the algorithm compared with random noise is clearly visible

Fooling rate as a function of xi





- Quickly reach a plateau, whatever the norm used or the model considered

- It is impossible to fool 100% of the dataset without completely destroying the images

Real time perturbation

Perturbing the class and detection of objects in a sequence of images (video)

Real time perturbation

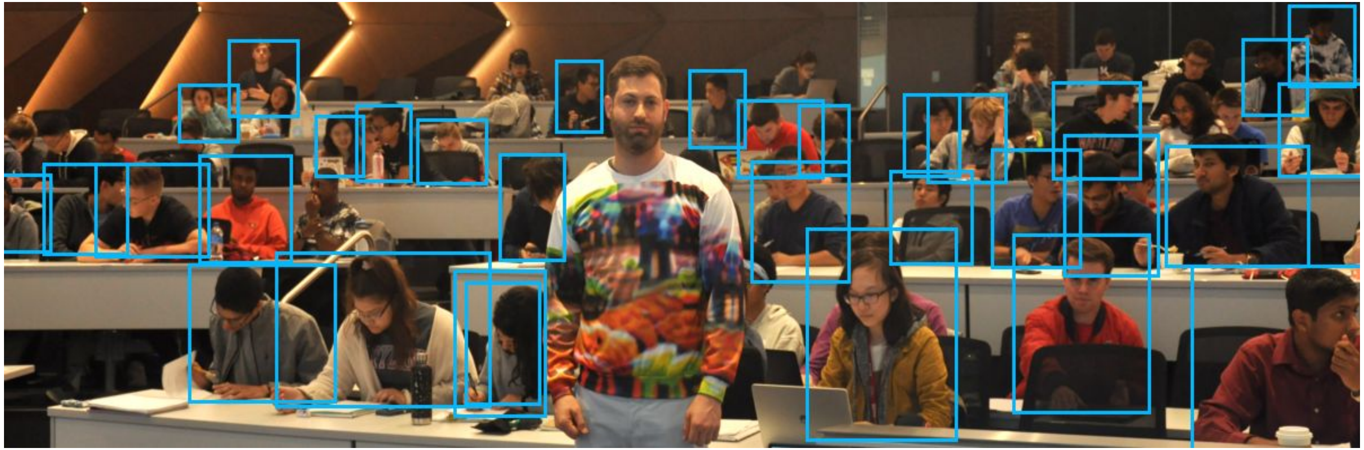
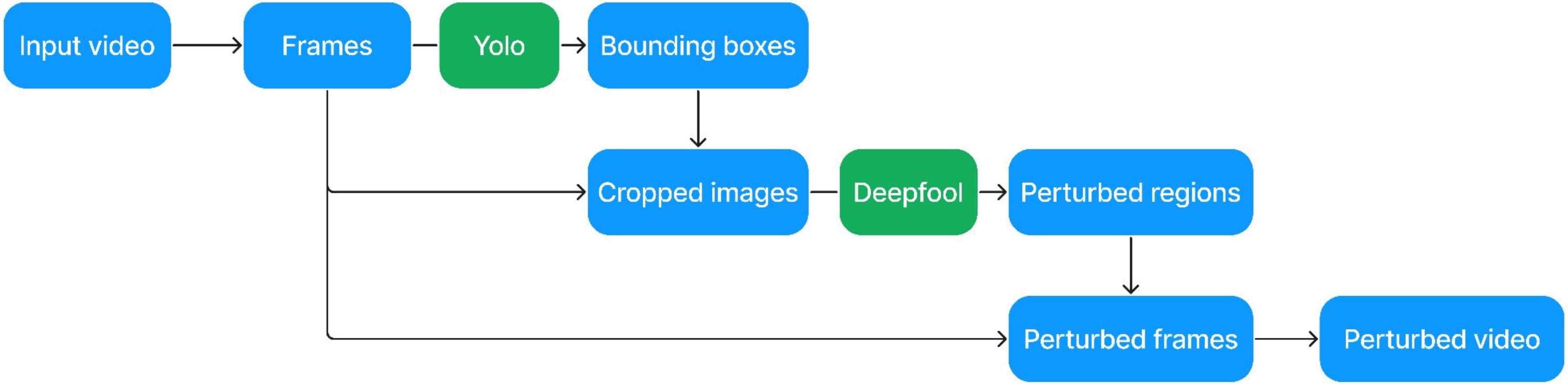
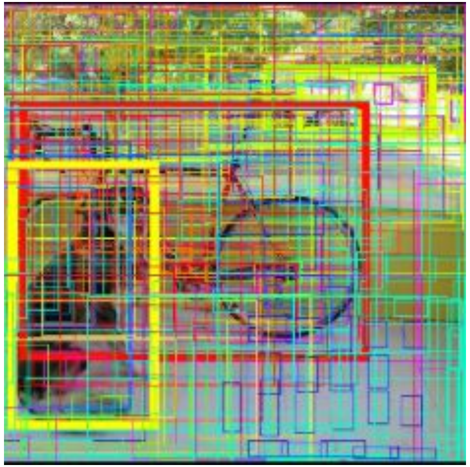


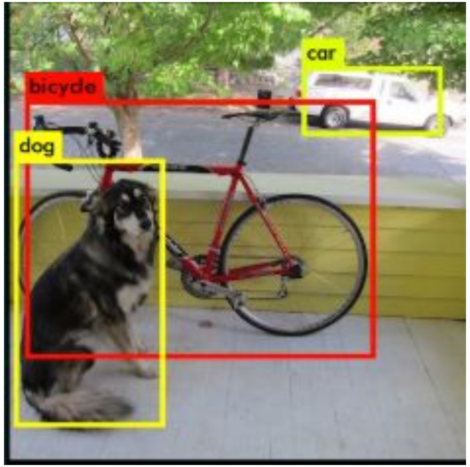
Image from the paper “Making an Invisibility Cloak: Real World Adversarial Attacks on Object Detectors”

Real time perturbation

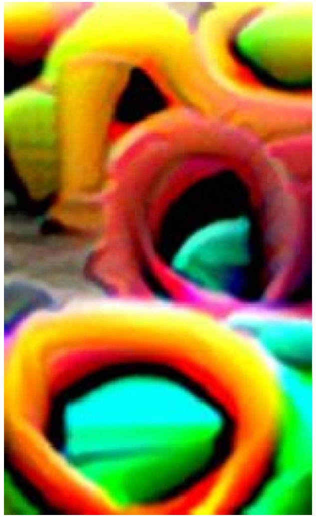


YOLO detection

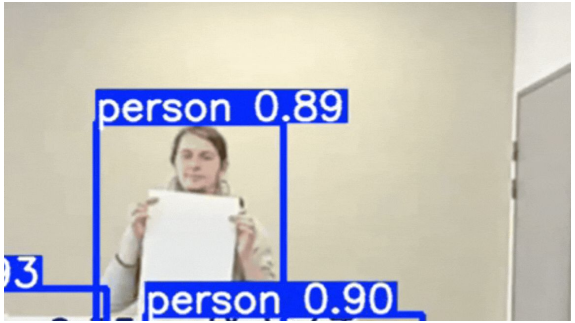




Real time perturbation

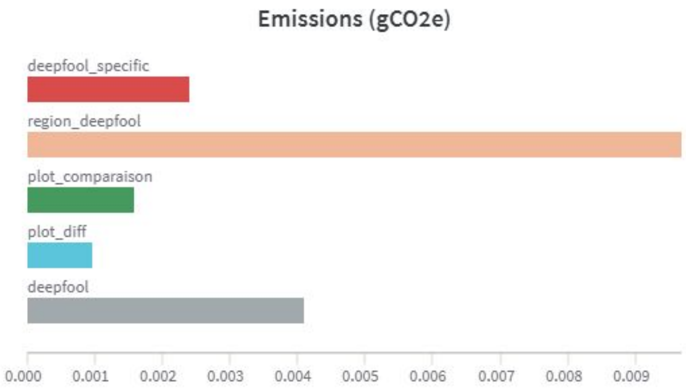


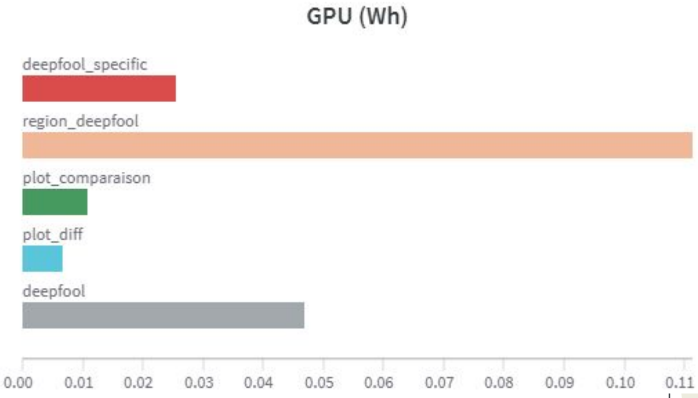
YOLOV3



Environmental impact

Name (118 visualized)		User	Emissions (gCO2e)	GPU (Wh)	CPU (Wh)	Energy (Wh)	RAM (Wh)
Name: plot_comparaison	3	notthestallion	0.0015864	0.010805	0.013555	0.02831	0.0039491
Name: plot_diff	3	notthestallion	0.00095841	0.0066414	0.0081013	0.017103	0.0023599
Name: deepfool_specific	25	notthestallion	0.0023868	0.025453	0.013272	0.042591	0.0038661
Name: local_deepfool	25	notthestallion	0.0096839	0.11135	0.047592	0.17281	0.013868
Name: deepfool	62	notthestallion	0.0055044	0.063125	0.02718	0.098224	0.0079193





Environmental impact

- France, Nouvelle-Aquitaine
- OS: Linux 6.12
- CPU: Intel Xeon 1270
- GPU: RTX 3060

Standard use

- Duration: 3s
- Energy consumed: 2.8e-5 kWh
- Emission: 1.5e-8 kg.CO2eq

x 2000

Deepfool

- Duration: 10s
- Energy consumed: 1.7e-4 kWh
- Emission: 9e-8 kg.CO2eq

x 400

Universal (ResNet-18/STL-10):

- Duration: 4m 33s
- Energy consumed: 1e-2 kWh
- Emission: 5.8 x 10^{-3} kg.CO2eq

x 50

= 2.9e-1 kg.CO2eq = 5.71e-2 km in car

Conclusion