Adversarial perturbation - jalon 1

Samuel Landeau
Romain Pierre
Mohamed Kherraz
Christian Banide

Département Informatique
Novembre 2024

1 Deepfool

The general idea behind the Deepfool algorithm is to find the minimal perturbation to add to an image so it gets wrongly classified. This perturbation is computed using two main equations :

$$\hat{l}(x_0) = \arg\min_{k \neq \hat{k}} \frac{|f_k(x_0) - f_{\hat{k}}(x_0)|}{\|w_k - w_{\hat{k}}\|_2} \tag{1}$$

This first equation represents the closest classification frontier to approach, with k being the original class of x_0, and $\hat{k}$ another class. f_k represents the classification function output for a given class k, and w_k is the direction vector of said function for a given class. When we know what classification $\hat{k}$ is the closest, we can compute the minimal perturbation to get the image to pass the frontier and be classified as $\hat{k}$:

$$r_*(x_0) = \frac{|f_{\hat{l}(x_0)}(x_0) - f_k(x_0)|}{\|w_{\hat{l}(x_0)} - w_k\|_2^2}(w_{\hat{l}(x_0)} - w_k) \tag{2}$$

We have re-written the Deepfool algorithm and used it on five images to test it. For each image, the algorithm will take the 10 most probable classifications and try to make the model converge toward one of those other classification by creating a perturbation. The perturbation chosen is then the one with the smallest norm. That's why we can see on figure 1 that perturbed classifications tend to be similar to the original ones.

Figure 1: Results of the deepfool algorithm on 5 images classified by ResNet34

The perturbations are almost to totally imperceptible for the human eye. If we look closely, it's possible to see some alterations, like a red patch of pixel on the back of the boat in the third image. By observing in more details the perturbations, it's possible to see that different images will be perturbed in totally differents manners. For example, we can see on figure 2 that some images, like the third one, will be very locally altered but quite intensely, while other images like the fifth one will be globally altered but with little intensity.

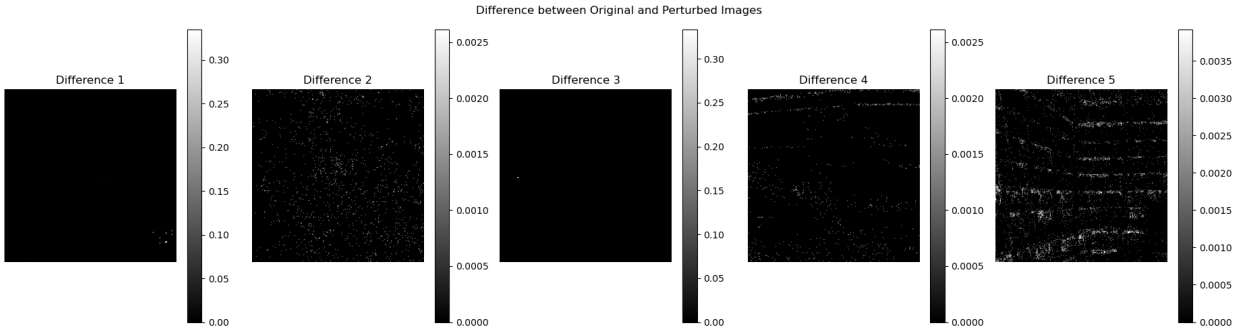

Figure 2: Perturbations applied by the algorithm to the five test images (original pixel values were scaled from 0 to 1)

2 Localised Deepfool

We have reworked DeepFool to apply it in a localized manner on the image. We started by selecting a specific region of the image and then enforcing a transformation on that particular area. The new equation used to compute minimal perturbation restricted to a region R can then be written as follow :

$$r_*(x_0) = \frac{|f_{\hat{l}(x_0)}(x_0) - f_k(x_0)|}{\|w_{\hat{l}(x_0)} - w_k\|_{2,R}^2}(w_{\hat{l}(x_0)} - w_k)^R \tag{3}$$

The algorithm still works this way, but sometimes will need to alter greatly the given region to fool the classification. On the first image of figure 3, we can for example clearly see the chosen region for perturbation.

Figure 3: Results of the deepfool algorithm restrained on a local area on 5 images classified by ResNet34

Compared to generalised DeepFool which can perturb images without it being perceptible every time, the application of the localised Deepfool to a randomly selected restricted area of the image can significantly affects the algorithm's performance. The image of the macaw becomes heavily degraded when DeepFool is applied specifically to the selected region in this experiment.

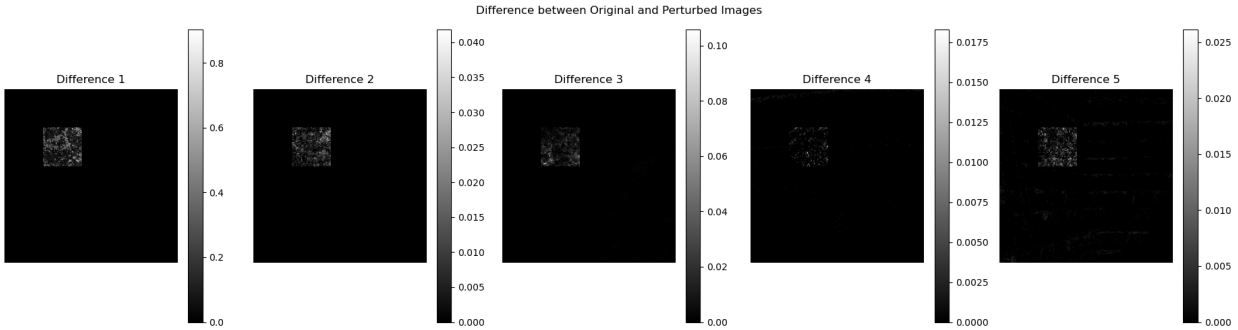

Figure 4: Perturbations applied localy by the algorithm to the five test images (original pixel values were scaled from 0 to 1)

We then asked ourselves if there are specific areas in an image that should be prioritised for optimal degradation. To explore this, we divided each image using a 4x4 grid and applied localized DeepFool on each grid cell. The results for the macaw image show that certain areas are clearly more effective for perturbation, as illustrated in Figure 5.

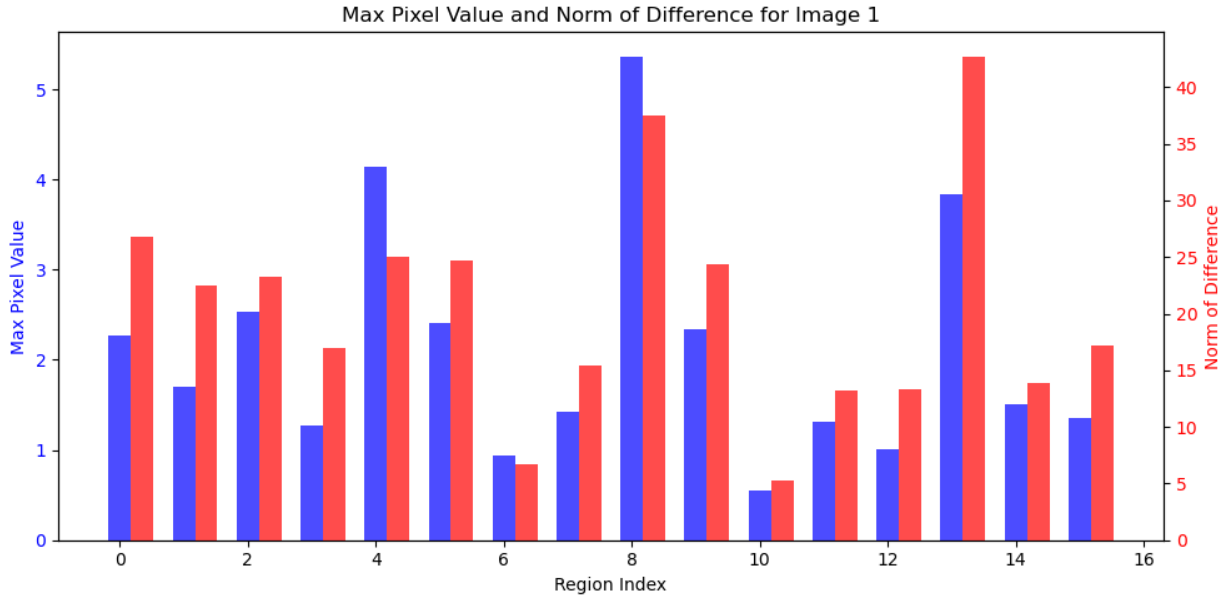

Figure 5

Those results are similar for the other images, and so it should be possible to develop an algorithm that tries to minimize not only the norm of the perturbation but also its size for any images. That could be used to find more localised perturbation that could be used in cases like the hat that makes faces of people being misclassified.

3 Universal perturbation

The general idea behind universal perturbation is trying to get a perturbation that can be applied to any images given as input to a model, and fool the model most of the time. This perturbation is considered "universal" because it can be applied regardless of the image and original classification of the image perturbed.

The algorithm will iterate on many images and apply the Deepfool algorithm on it, and each time update the perturbation with the results of Deepfool. The maximum norm authorized is set beforehand, so the perturbation can be controlled to not alter the image too much. When the perturbation can fool the model at a chosen minimum rate, the algorithm stops and returns the actual perturbation.

Figure 6: Results of a universal perturbation applied on the dataset stl10 classified by ResNet

On figure 6, we can see the results of our implementation of universal perturbation on the stl10 dataset classified by a fine-tuned version of ResNet. The perturbation fools the model 40% of the time, which is the goal we set for the algorithm to get results in a satisfying time.

4 Image Detection

We have tested YOLO for image detection, on both live video and pictures. This represents a first step towards the next phase of the project, where we will attempt to deceive classifiers on live video streams.

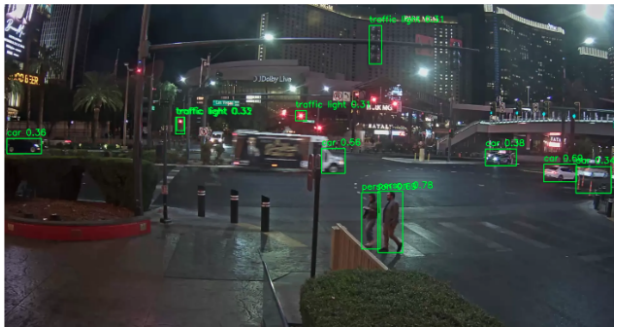

Figure 7: Use of YOLO on live video

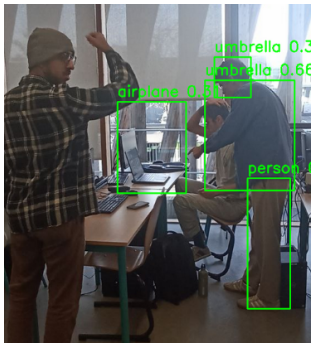

Figure 8: Use of YOLO on a picture

5 Next Steps

From now on we'll try to shift our focus to real-time detection, and see how we can alter the vision of models like Yolo. The goal is to have similar results as the ones on the presentation of the project, like the hoodie that makes people invisible.

While doing so, we'll also try to explore a little the possibilities with the algorithms we have already implemented, for example to compare the efficiency of a same perturbation on differents models, or try using universal perturbation with the localised Deepfool algorithm to see if its possible to have acceptable universal results with a localised perturbation.