Transferable adversarial poisoning of deep neural network classifiers using surrogate backbones

Ambre ETIENNE
ENSAE
ambre.etienne@ensae.fr

Ilyes HAMMOUDA
ENSAE
ilyes.hammouda@ensae.fr

Valentin TORDJMAN–LEVAVASSEUR
ENSAE
valentin.tordjman-levavasseur@ensae.fr

Abstract

In this article, we propose an adversarial method to degrade the performance of a deep neural network classifier using surrogate backbones as proxies to the unknown one to be poisoned. We assess transferability to other backbones, and performance of our methodology in the white box and in the black box setting. We also assess the performance degradation when varying the number of poisoned images in the training set, and when varying the number of surrogate backbones. We propose some ablation experiments to justify our choices, and show some examples of perturbations on the German Traffic Sign Recognition dataset [7]. We observe a successful drop in performance when 100% of the training set is poisoned, yet we also observe a significant increase in classification accuracy when poisoning a fraction of the training set. We showcase transferability, and create unnoticeable perturbations.

1 Introduction and litterature review

In computer security, machine learning, while a powerful asset, becomes vulnerable when poisoned, leading to performance degradation [1]. Our entry point to this subject was the paper [8], which showed successful poisoning of diffusion networks, so that the poisoned model would be confused between targets: it would create a picture of a dog when asked for a cat. The paper [11] manages, for its parts, to poison a deep neural network classifier by slightly modifying the training images, so that the embedding of the target images to be labelled wrongly by the classifier would be in the convex polytope formed by the poisoned images. They manage to perform in the white box setting where the targetted backbone is unknown by using surrogate networks, and show some transferability. They compare their method with the so called Feature collision [6]. Such attacks could be refered as backdoor attack, as the goal is to make some images in the test set to be labelled as a label of our choice. On the other hand, we decided to explore methods to damage classifier performance by subtly poisoning the training set, aiming for perturbations imperceptible to human operators. The paper [4] highlights the existence of various adversarial techniques, including attacks at different stages of the machine learning lifecycle. For instance, at the pre-training stage, attackers may manipulate the training dataset to generate poisoned samples, leading to a data-poisoning based backdoor attack. At the post-training stage, attackers can modify model parameters to inject a trojan via weight attack injection. Here, we adopt an adversarial approach using surrogate networks as discriminators. We leverage transfer attacks, as highlighted by Szegedy et al. [3], exploiting the generalization power of deep neural networks to generate adversarial examples that can be misclassified by diverse classifiers. The discriminator task is to classify correctly both the clean and the poisoned images, and also to classify if the images are fake or not. The generator task is to create a perturbation to the original

image so that the discriminator performs poorly, with in addition an L2 penalisation, which we justify with our ablation experiments. While [11] assess their performance on the CIFAR-10 [5] dataset, we decided to try to use our method on the German Traffic Sign Recognition dataset [7]. We believe that as trafic signs were specifically designed to be recognizable to one another and not be confused with another one, it would be a rather challenging task. Besides, a successful poisoning with a significative degradation of performance could raise awareness on how crucial it is that the data used to train autonomous vehicles algorithms is protected to cyberattacks.

2 Model architecures

2.1 Generator

For our generator, we employed the default UNet++ [10] from MONAI [2], which is originally a library designed for medical semantic segmentation and offers a wide range of available models. This library facilitates the straightforward construction of a 2D segmentation model with the number of channels of our choosing. We opted for UNet++ over UNet due to peculiar aliasing issues observed in the perturbation during our initial attempts with UNet. These artifacts would have been perceptible to the human eye. The loss of our generator can be written as follows :

$$Loss = \alpha * MSE(Perturbation, 0) - LossDiscriminator$$

With α being a user-defined hyperparameter that influences the intensity of the L2 penalization, the choice of its value is crucial. A lower α, as we observe for the end user, results in more perceptible perturbations. Further exploration on this aspect is detailed in the experiments section.

During the backward pass on the generator, the discriminator's loss is exclusively computed for poisoned images, and the discriminator's weights remain frozen throughout this sub-step.

2.2 Discriminator

Architecture The discriminator architecture consists of :

- $N_{encoders}$ frozen surrogate backbones
- $N_{decoders} = N_{encoders}$ decoders, which are linear layers of size $feature\_size \times 128$
- A classification head, which is a linear layer of size $128 \times N_{classes}$, whose task is to predict the class of the input image
- A fake detector, which is a linear layer classification head of size 128×2, whose task is to say wether the image is perturbed or not

The discriminator architecture can be summurized with this schematics :

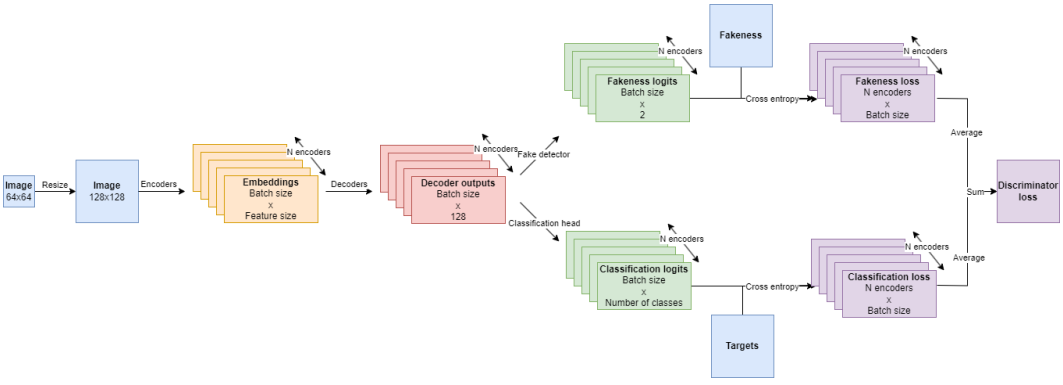


Figure 1: Discriminator architecture diagram

Loss The loss of the discriminator can be expressed as follows :

$$Loss = ClassificationLoss_{PoisonedImages} + ClassificationLoss_{RealImages}$$

$$ClassificationLoss_{PoisonedImages} = \frac{1}{N}\sum_{i=1}^{N}(\text{CrossEntropy}(\text{pred}_i, y) + \text{CrossEntropy}(\text{pred\_fakeness}_i, 1))$$

$$ClassificationLoss_{RealImages} = \frac{1}{N}\sum_{i=1}^{N}(\text{CrossEntropy}(\text{pred}_i, y) + \text{CrossEntropy}(\text{pred\_fakeness}_i, 0))$$

With each poisoned and real images loss term being the mean of the Cross entropy label classification loss for all the backbones, plus the mean of the Cross entropy fakeness classification loss for all the backbones, on the real or fake images. This way, the discriminator has to train its decoder to be able to classify correctly both the label and the fakeness of the image for all its backbones. For the backward pass on the discriminator, the pertubation is computed within a torch.no_grad() flag, so that the gradients of the generator are not computed.

3 Experiments on the German Traffic Sign Recognition dataset

3.1 Presentation of the dataset

The German Traffic Recognition dataset tackles the complex task of recognizing traffic signs, lacking systematic comparisons and comprehensive benchmarks. With over 40 classes and 50,000+ images, it reflects real-world challenges like varied appearances due to illumination changes and occlusions. Despite human accuracy near 100%, the dataset presents difficulties, especially with subsets of similar signs. Ground-truth reliability is ensured through semi-automatic annotation, and each traffic sign instance in the dataset is unique, preventing repetitions.

Here are some examples of images that can be found in the dataset:

Figure 2: Dangerous left turn sign

Figure 3: One-way sign

Figure 4: Stop sign

3.2 Examples of produced perturbations

3.2.1 Varying α

The provided examples were generated using five surrogate backbones—EfficientNetB0, ResNet34, Resnest26d, RegNetX_006, and DenseNet121. The objective is to qualitatively evaluate the perceptibility of perturbations under varying α, representing the strength of L2 penalization.

Figure 5: α = 1

Figure 6: α = 100

Figure 7: α = 1000

As α increases, the perturbations become progressively less noticeable. In the subsequent experiments, we set α to 1000. Additional examples of perturbations with this specific α and the five surrogate encoders are provided in the appendix.

3.2.2 Varying the number of encoders

This section explores the impact of varying numbers of encoders on the resulting poisoned image. We begin with a single encoder, then progress to three encoders, and finally, incorporate all five encoders mentioned earlier. The selection of these encoders is based on their distinct specificities, contributing to the overall effectiveness of the model.

Figure 8: 1 Backbone *(ResNest26D)*

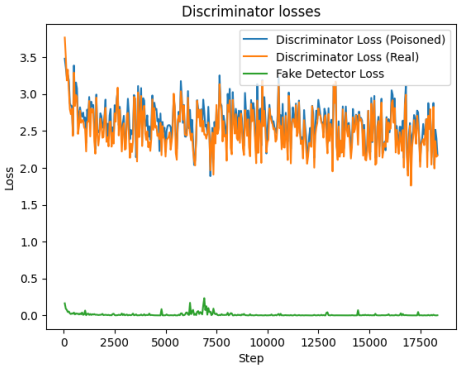
Figure 9: 3 Backbones *(ResNet34, EfficientNetB0, and Resnest26d)*

Figure 10: 5 Backbones *(ResNet34, EfficientNetB0, Resnest26d, RegNetX_006, and DenseNet121é)*

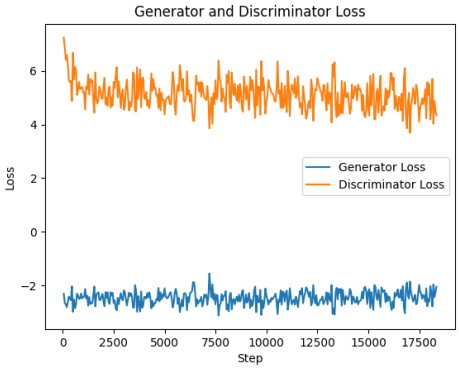
With an increase in the number of encoders, the perturbation also becomes less perceptible. However, this effect is less obvious compared to variations in the L2 penalization term.

3.3 Training plots

The presented plots were generated using the 5 previously mentioned backbones. They illustrate the progression of both the generator and discriminator losses. Additionally, the evolution of individual components of each loss is showcased, including the discriminator loss on real images, the discriminator loss on poisoned images, and the fake detector loss.



(a) Components of the discriminator losses during training



(b) Generator and discriminator losses during training

3.4 Performance degradation assessment

In our experiments, the Generative Adversarial Network (GAN) underwent training for 15 epochs, utilizing Adam as the optimizer for both the discriminator and the generator. The learning rate was set to 5.10^{-4}, and a weight decay of 1.10^{-4} was applied. The classifiers were trained for 5 epochs, employing the same optimizer parameters.

3.4.1 Black box setting

In the context of a black box setting, we refer to a scenario where the attacked network differs from the surrogate ones. Performance degradation is evaluated across various backbones, considering both 3 and 5 surrogate networks. For the 5-surrogate setting, all five previously mentioned networks are considered, while the 3-surrogate setting excludes RegNetX_006 and DenseNet121.

Below 1 are the accuracies obtained for ResNest101e, for its similarity with ResNest26D:

	Clean	10% poisoned	50% poisoned	100% poisoned
3 backbones	0.5283	0.6518	0.6214	0.1988
5 backbones	0.5283	0.6515	0.6329	0.2013

Table 1: Accuracies obtained with ResNest101e, with either 3 or 5 backbones for the discriminator, with varying quantity of poisoned images.

With 100% of the training set being poisoned, the accuracy drops from 53% to 20%. Notably, an intriguing observation is that with a fraction of the dataset poisoned, the accuracy surpasses that of the clean dataset, a phenomenon we will explore further in the Discussion section.

Additionally, we aimed to evaluate transferability to transformer-based backbones. In this regard, we utilized Tiny ViT 21M, and the results are as follows:

	Clean	10% poisoned	50% poisoned	100% poisoned
3 backbones	0.6458	0.6052	0.6345	0.261
5 backbones	0.6458	0.6051	0.6172	0.2301

Table 2: Accuracies obtained with Tiny ViT, with either 3 or 5 backbones for the discriminator, with varying quantity of poisoned images.

One notable observation is that the peak accuracy is achieved when 50% of the dataset is poisoned, in contrast to the previous scenario where the the best accuracy was obtained with a 10% poisoning. This suggests the existence of an optimal poisoned ratio that influences the overall performance, which depends on the targeted classifier.

Additionally, poisoning the entire dataset significantly impacts the accuracy.

3.4.2 White box setting

In a white box setting, we refer to a scenario where the attacked network is known, which is somewhat unrealistic. In this case, only one surrogate network is required, specifically the one intended to be poisoned. Performance assessment is conducted in the white box setting using ResNet34 and ResNest26D.

Below are the results:

	Clean	10% poisoned	50% poisoned	100% poisoned
ResNet34	0.3977	0.4717	0.439	0.1477
ResNest26D	0.5615	0.7265	0.642	0.2754

Table 3: Accuracies obtained in the white box setting, with either varying quantity of poisoned images

Surprisingly, when injecting a 10% poison into the dataset for ResNet26D, we observe an unexpected increase in accuracy from 56% to 72%. Despite the intended goal of degrading model performance, this peculiar finding presents an intriguing opportunity for improvement with a small percentage of poisoning. However, as the percentage of the dataset poisoned increases further, the model consistently exhibits a decline in accuracy, aligning with the intended deterioration of model performance.

3.5 Ablation experiments

The following experiments were conducted utilizing the 5 previously mentioned surrogate backbones, with the objective of substantiating the use of the ablated part.

Figure 12: No ablation

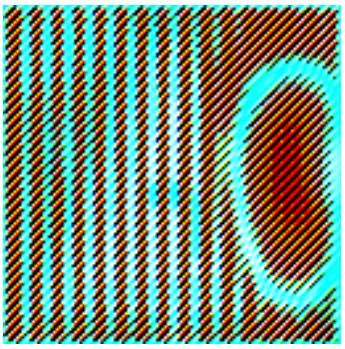


Figure 13: Without L2 regularization

Figure 14: Without fake detector

In both scenarios, we observe a more pronounced perturbation, validating our choice to include both components. It's worth noting that even without L2 regularization, the fake detector loss tends to decrease to 0, resembling the non-ablated case, despite the obvious perturbation.

4 Discussion

Some extension to this work could be done by exploring our method in the grey box setting, in which the attacked backbone is in the surrogate backbones but with different weights. This is a more realistic setting compared to the white box setting, as backbones are more often than not unfrozen during training to get better performances. During our journey, we sometimes noticed that the perturbation concentrated itself in the corner of the image, which makes it easily dodgeable using a center crop. While this is not the case for our final perturbations, it would be rather interesting to explore some other post processing strategies that an operator could use to mitigate the effect of a known poisoning. We also would have liked to use more surrogate backbones, but were stuck because of our GPU memory. We believe that using more backbones could lead to better results. Finally, we think that our generator could benefit from a pre training on ImageNet, and a fine tuning on the dataset to be poisoned. Some work could be done to see if it actually increases the performance degradations.

In our research, an intriguing phenomenon emerged during experiments involving dataset poisoning. Contrary to the anticipated decrease in accuracy with the introduction of adversarial elements, we observed a surprising outcome when poisoning a fraction of the dataset. Rather than witnessing a decline in accuracy, we noted a noticeable increase (56% to 72% in the white box setting for ResNest26D). While this unexpected behavior challenges conventional assumptions, it aligns with a complex interplay of factors such as overfitting, data augmentation effects, and stochasticity in the training process. This anomaly we observed raises questions about the intricate dynamics between adversarial elements and model performance, warranting further exploration and investigation.

It would be worthwhile to explore the impact of injecting a small amount of poison after unfreezing the backbones and assess whether it enhances model performance. This avenue of investigation could provide valuable insights into the dynamics of the model's response to poisoning when the backbone layers are open for adjustment. Additionally, extending this experimentation to other image-related tasks, such as semantic segmentation, could offer a broader understanding of how these observations generalize across various computer vision tasks. For example, the paper [9] investigates the vulnerability of convolutional neural networks in aerial image semantic segmentation to adversarial attacks.

5 Conclusion

In this study, an adversarial method was proposed to degrade the performance of a deep neural network classifier by subtly poisoning the training set. Surrogate backbones were used as discriminators to assess transferability to other backbones and evaluate performance in both white and black box settings. The experiments on the German Traffic Sign Recognition dataset demonstrated successful poisoning, resulting in diminished classifier accuracy in both settings. Surprisingly, injecting a fraction of poison led to increased accuracy, challenging conventional assumptions and prompting further exploration. Ablation experiments validated the inclusion of both perturbation and fake detector components in the methodology. The unexpected findings suggest a complex interplay between adversarial elements and model performance, warranting further investigation and potential applications in other computer vision tasks.

References

[1] M. Barreno, B. Nelson, A. Joseph, and J. Tygar. The security of machine learning. *Machine Learning*, 81:121–148, 11 2010.

[2] M. J. Cardoso, W. Li, R. Brown, and N. Ma. Monai: An open-source framework for deep learning in healthcare, 2022.

[3] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples, 2015.

[4] S. Kotyan. A reading survey on adversarial machine learning: Adversarial attacks and their understanding, 2023.

[5] A. Krizhevsky, V. Nair, and G. Hinton. Cifar-10 (canadian institute for advanced research).

[6] A. Shafahi, W. R. Huang, M. Najibi, O. Suciu, C. Studer, T. Dumitras, and T. Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. *CoRR*, abs/1804.00792, 2018.

[7] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, (0):–, 2012.

[8] L. Struppek, M. Hentschel, C. Poth, D. Hintersdorf, and K. Kersting. Leveraging diffusion-based image variations for robust training on poisoned data. In *NeurIPS 2023 Workshop on Backdoors in Deep Learning - The Good, the Bad, and the Ugly*, 2023.

[9] Z. Wang, B. Wang, Y. Liu, and J. Guo. Global feature attention network: Addressing the threat of adversarial attack for aerial image semantic segmentation. *Remote Sensing*, 15(5), 2023.

[10] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang. Unet++: A nested u-net architecture for medical image segmentation, 2018.

[11] C. Zhu, W. R. Huang, A. Shafahi, H. Li, G. Taylor, C. Studer, and T. Goldstein. Transferable clean-label poisoning attacks on deep neural nets, 2019.

Appendix

You may find below some more poisoning examples that were computed using our 5 surrogate backbones setting, with $\alpha = 1000$.

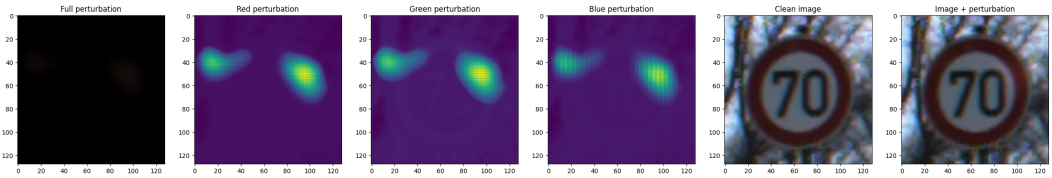


Figure 15: Example 1 of perturbation with 5 surrogate backbones, $\alpha = 1000$

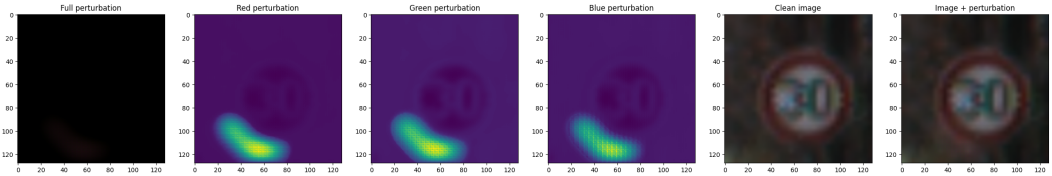


Figure 16: Example 2 of perturbation with 5 surrogate backbones, $\alpha = 1000$

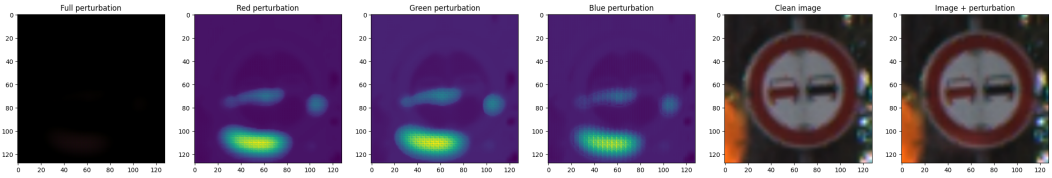


Figure 17: Example 3 of perturbation with 5 surrogate backbones, $\alpha = 1000$