IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Evolutionary Algorithm-based images, humanly indistinguishable and adversarial against Convolutional Neural Networks: efficiency and filter robustness

RALUCA CHITIC[1], ALI OSMAN TOPAL[1], AND FRANCK LEPRÉVOST.[1],

[1]University of Luxembourg, House of Numbers, 6, avenue de la Fonte, L-4364 Esch-sur-Alzette, G-D of Luxembourg (e-mail: Raluca.Chitic@uni.lu, Aliosman.Topal@uni.lu, Franck.Leprevost@uni.lu)

Corresponding author: Raluca Chitic (e-mail: Raluca.Chitic@uni.lu).

ABSTRACT Convolutional neural networks (CNNs) have become one of the most important tools for image classification. However, many models are susceptible to adversarial attacks, and CNNs can perform misclassifications. In previous works, we successfully developed an EA-based black-box attack that creates adversarial images for the *target scenario* that fulfils two criteria. The CNN should classify the adversarial image in the target category with a confidence ≥ 0.95, and a human should not notice any difference between the adversarial and original images. Thanks to extensive experiments performed with the CNN $\mathcal{C}$ = VGG-16 trained on the CIFAR-10 dataset to classify images according to 10 categories, this paper, which substantially enhances most aspects of [1], addresses four issues. (1) From a *pure* EA point of view, we highlight the conceptual originality of our algorithm $\text{EA}_d^{\text{target},\mathcal{C}}$, versus the classical EA approach. The competitive advantage obtained was assessed experimentally during image classification. (2) We then measured the intrinsic performance of the EA-based attack for an extensive series of ancestor images. (3) We challenged the filter resistance of the adversarial images created by the EA for five well-known filters. (4) We proceed to the creation of natively filter-resistant adversarial images that can fool humans, CNNs, and CNNs composed with filters.

INDEX TERMS Convolutional neural network, evolutionary algorithm, black-box attack, image classification, adversarial perturbation, filter.

I. INTRODUCTION

IN 2012, Krizhevsky et al. [2] presented the outstanding performance of convolutional neural networks (CNNs) on a very difficult image classification task [3]. Since then, as computing capacity has increased, CNNs have become the main driver of many computer vision applications, including image classification [4]–[7], facial recognition [8], [9], malware detection [10]–[12], email spam filters [13], speech recognition [14], [15], robotics [16], and self-driving cars [17], [18]. Despite their increasing power and the variety of applications, CNNs are susceptible to deception. In the context of image classification, by analogy with *Trompe-l'œil* that challenges humans' visual perception, a CNN can be led to misclassification of objects in an image. The generic attack to create such specially crafted *adversarial images* consists of adding some appropriate noise to a legitimate input, leading the network to label the new input in a different category than expected [13], [19], [20].

White-box and *black-box* attacks differ according to the level of knowledge about the addressed CNN at the disposal of the attacker. In the former case, the attacker has a complete knowledge of the CNN model, its design and its parameters. Gradient-based attacks [21] make use of the CNN parameters to calculate the optimal direction in which to modify the image, such that it becomes adversarial. The situation is opposite in the black-box case, in which the attacker's knowledge is scarcely limited to the size of the images handled by the CNN, and to classification values outputted by the CNN for ad-hoc queries (but without any information about how these values are obtained). Within black-box attacks, transfer-based methods [21] use the collected query information to create a substitute model that is similar to the targeted CNN. Gradient-based methods are used to attack the substitute model, which leads to adversarial images that transfer to the target CNN. Another type of black-box attacks are score-based methods [21], which do not try to infer the CNN's parameters, and hence do not calculate any gradients; they only make use of the CNN's predicted output probabilities for either all, or a subset of object classes.

Starting from an original image labeled by a CNN as representing an object in a specific category, methods that creates adversarial images may adopt different scenarios. For instance, a *targeted* attack creates an adversarial image that the CNN misclassifies as belonging to an a priori particular predefined class, different from the original one. A different scenario is addressed by *untargeted* attacks that only require the CNN to misclassify the adversarial image as belonging to any class whatsoever, provided that this class differs from the original one.

Although efficient against a CNN, the perturbations added to an original image to create an adversarial image may be highly noticeable for a human eye, as illustrated in Fig. 1 (a), (b), (c), and (d). Our evolutionary algorithm-based black-box, targeted attack $\text{EA}_d^{\text{target},\mathcal{C}}$ (introduced in [22], [23], see also [24]) differs from existing techniques in this respect. Not only does our evolutionary algorithm (EA) efficiently produce adversarial images that deceive the targeted CNN model $\mathcal{C}$ with high accuracy, but the perturbations added by our algorithm to the original image are not perceptible to the human eye, as shown in Fig.1e (original image in the first row, our adversarial image in the second row).

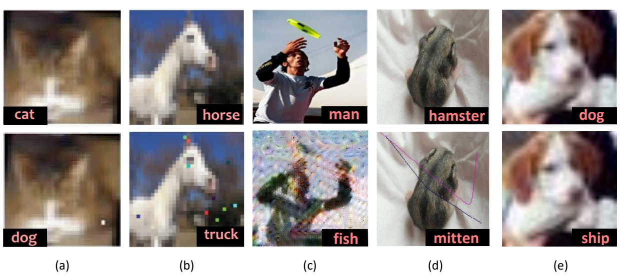


FIGURE 1: The images in the first row represent the original images and in the second row the adversarial images and their respective class labels that are created by (a) One-Pixel attack [25], (b) Few-Pixels attack [26], (c) Fooling Transfer Net (FTN) [27], (d) Scratch that! [28], and (e) our EA-based attack [22], [23] .

II. MOTIVATION

The purpose of this study, which substantially enhances most aspects of [1], is to address four issues. Broadly, the first two deal with explaining our choices for the design of the EA and proving its success as an adversarial attack. Rather than simply introducing an adversarial attack, in the latter two issues we also study its potential vulnerability to filter defenses and adapt the attack so as to assure its robustness even in cases when the attacker is not aware of the defense.

More specifically, the above-mentioned four issues are the following: (1) Conceptual originality and competitive advantage of our algorithm $\text{EA}_d^{\text{target},\mathcal{C}}$, (2) intrinsic performance of this EA-based attack, (3) filter resistance of the adversarial images created by the EA, and (4) creation of natively filter-resistant adversarial images. Before being more specific, let us point out that all experiments in this paper are performed with distance $d = L_2$ for the CNN $\mathcal{C}$ = VGG-16 [6], [29] trained on the CIFAR-10 [30] dataset to classify images according to 10 categories, and mainly address the *target scenario*, but also, to a lesser extent, the *untargeted scenario* (see Section III).

The issue (1) (see Section IV) relates to a series of conceptual differences, from a "pure" evolutionary algorithm point of view (hence independent of the task to perform, to some extent), between our adapted version and the classical EA approach [31]. To assess the practical impact of these conceptual differences, we analyzed their respective performances in creating adversarial images for a demanding definition of a successful attack. Indeed, for the (c_a, c_t) target scenario performed on an ancestor $\mathcal{A}$ classified by VGG-16 in c_a, we require these algorithms to create in less than 7000 generations an adversarial image $\mathcal{D}$ classified by VGG-16 as belonging to $c_t \neq c_a$ with a c_t-label value ≥ 0.95, while remaining so close to $\mathcal{A}$ that a human would not notice any difference between the adversarial and the ancestor. The first outcome of this study is that our "adapted EA" significantly outperforms the "classic EA" approach at creating such adversarial images for all considered cases, since it requires between 8% and 25% fewer generations to terminate successfully.

We then address issue (2) by a thorough and extended efficiency study of our EA-based attack with the "adapted EA" version (see Section V). In the first series of experiments with one ancestor per category of CIFAR-10, we performed 10 independent runs per ancestor per target category, leading to a total of 900 attacks. The algorithm $\text{EA}_{L_2}^{\text{target},\text{VGG-16}}$ achieves a 100% success rate (all ancestor-target categories are achieved for at least one of the 10 runs performed on each ancestor), requiring between 290 and 2793 generations on average, depending on the (c_a, c_t) target scenario. To better assess the importance of the choice of the ancestor in a given category c_a, and the impact of the seed value used for a specific run, we extend these experiments. In a second series of experiments, we randomly selected 50 distinct ancestors for each of the 10 categories of CIFAR-10 and altogether ran 4500 attacks for the target scenario. In this case, our algorithm achieved a success rate of 98%, requiring between 461 and 1717 generations on average. Moreover, both series of experiments show that a run of $\text{EA}_{L_2}^{\text{target,VGG-16}}$ has more than 96% (actually 96.56% for the former, and 98.06% for the latter series) to terminate successfully, and to create images that fool humans and VGG-16 trained on CIFAR-10, despite our demanding requirements for a successful termination.

Issues (3) and (4) (addressed in Sections VI and VII, respectively) deserve to be put into the following broader perspective. Let $\mathcal{A}$ be an image classified by a CNN $\mathcal{C}$ in some category c_a, and $\mathcal{D}$ be an adversarial image; for the *target scenario*, $\mathcal{C}$ classifies in a distinct category c_t (at this stage, the type of attack that leads to $\mathcal{D}$ does not matter). One now considers a function $\mathcal{F}$ that acts on such images to create images $\mathcal{F}(\mathcal{A})$ and $\mathcal{F}(\mathcal{D})$ of the size handled by the CNN (which coincides with the same common size of $\mathcal{A}$ and $\mathcal{D}$ in the present case). How does the CNN classify these new images? Does $\mathcal{F}(\mathcal{D})$ remain adversarial, or does the composition $\mathcal{C} \circ \mathcal{F}$ (that consists in putting $\mathcal{F}$ ahead of $\mathcal{C}$) protect $\mathcal{C}$ against the attack? If this latter case holds, can one adapt the attack to create images that fool not only $\mathcal{C}$, but also the $\mathcal{F}$-enhanced CNN $\mathcal{C} \circ \mathcal{F}$? If yes, would such images, adversarial for $\mathcal{C} \circ \mathcal{F}$, be adversarial as well for $\mathcal{C} \circ \mathcal{G}$ for $\mathcal{G} \neq \mathcal{F}$, and hence have the capability to fool the same CNN $\mathcal{C}$ but enhanced by other functions $\mathcal{G}$?

Among the different meaningful functions $\mathcal{F}$ one could think of in this context, we undertake the study for filters. Indeed, daily used in image processing, filters substantially impact the visual appearance of images for a human eye on the one hand, and potentially affect the classification process of a trained CNN on the other hand. It is therefore tempting to check whether adding filters may prevent CNNs from misclassification, or reduce this risk to some extent, when facing an adversarial image. Additionally, one may also want to evaluate the quality of adversarial images by their capacity to mimic the ancestor's image behavior when exposed to filters.

For reasons given in Section VI, in which issue (3) is discussed, we proceed to the selection of five filters, namely the inverse filter (F_1), the Gaussian blur filter (F_2), the median filter (F_3), the unsharp mask filter (F_4), and the F_5 combination of the two last filters. With each of them, we filter the ancestor $\mathcal{A}_a$ and the adversarial images $\mathcal{D}_{a,t}(\mathcal{A}_a)$ created by the $\text{EA}_{L_2}^{\text{target,VGG-16}}$ algorithm in Section V. VGG-16 was then challenged with the filtered images. The values of a series of specifically designed indicators led to two conclusions. On the one hand, the Inverse, and the Unsharp mask filters are significantly inefficient against our EA, because, for instance, 95% of the adversarial images filtered by F_4 remain adversarial for the *target scenario*, and 95% remain adversarial for the *untargeted scenario* (in a relaxed sense to be made precise in this Section). *A contrario*, the other filters, especially the combination F_5, render our EA-based attack less effective for both the *target* and the *untargeted scenario*.

This led us to address the final issue in (4). For a filter F, we conceived a filter-enhanced F-fitness function (see Section VII), and the corresponding algorithm $\text{EA}_{L_2,F}^{\text{target,VGG-16}}$, obtained from $\text{EA}_{L_2}^{\text{target,VGG-16}}$, by updating the fitness function accordingly. For the reasons given in Section VII, we select $F = F_5$, and allocate to $\text{EA}_{L_2,F_5}^{\text{target,VGG-16}}$ the task to create adversarial images that are moreover natively immune against filter F_5. In other words, these adversarial images simultaneously fool $\mathcal{C}$ and $\mathcal{C} \circ F_5$ for $\mathcal{C} =$ VGG-16 for the *target scenario* (still with the demanding target label value ≥ 0.95), while remaining so close to the ancestor that no human eye would notice any difference. We performed similar experiments for issue (2). The first series of 900 attacks (one ancestor per ancestor category, 10 independent runs for each $(c_a(\mathcal{A}_a), c_t)$ scenario) shows that $\text{EA}_{L_2,F_5}^{\text{target,VGG-16}}$ achieves a success rate of 96.66% (three combinations were not achieved), and that the probability that it terminated successfully for a given run was 95.77%, requiring an average between 798 and 2746 generations for the successful $(c_a(\mathcal{A}_a), c_t)$ considered. In a second series of 4500 attacks performed with 50 different ancestors per category, $\text{EA}_{L_2,F_5}^{\text{target,VGG-16}}$ showed a success rate of 88%, with between 1250 and 2404 generations on average.

We complete study (4) by exploring whether an adversarial image, constructed by $\text{EA}_{L_2,F_5}^{\text{target,VGG-16}}$ to fool both $\mathcal{C}$ and $\mathcal{C} \circ F_5$, would also be adversarial against $\mathcal{C} \circ F_k$ for the other filters $F_1, F_2, F_3, and F_4$ for $\mathcal{C} =$ VGG-16. Our study shows that it is so for F_3 and F_4 with (depending on the *target* or *untargeted scenario*) between 83% and 89% of the images remaining adversarial against these filters. 56% of theses images were also adversarial for F_1 for the *untargeted scenario*, while this percentage dropped to 23% for F_2. Therefore, the $\text{EA}_{L_2,F_5}^{\text{target,VGG-16}}$ attack, designed to be robust against $\mathcal{C}$ and $\mathcal{C} \circ F_5$ for $\mathcal{C} =$VGG-16, is also robust to a significant extent against all individual filters for the $untargeted scenario$.

Section VIII summarizes the conclusions of this case study, and provides a series of research directions.

III. THE TARGET SCENARIO ON VGG-16 TRAINED ON CIFAR-10

Although applicable to any CNN trained at image classification on any dataset, we instantiate our approach on the concrete case of VGG-16 [6] trained on CIFAR-10 [30].

A. VGG-16 TRAINED ON CIFAR-10

The CIFAR-10 dataset encompasses $50,000$ training images, and $10,000$ test images of size $32 \times 32 \times 3$, meaning that each image has a width and height of 32 pixels, and each pixel has a color resulting from the three RGB values. Once trained, VGG-16 sorts images according to the 10 categories c_i of CIFAR-10 listed in the 2^{nd} row of Table 1, composed of 4 "Object" categories (c_1, c_2, c_9, c_{10}), and of 6 "Animal" categories (c_i for $3 \leq i \leq 8$). The 4^{th} row of Table 1 displays the original ancestor images $\mathcal{A}_a$ in the categories c_a (and their respective c_a-label values, see below) used throughout this paper, and the 3^{rd} row give their reference number in the test set of CIFAR-10.

In practice, an input image $\mathcal{I}$ given to VGG-16 trained on CIFAR-10 is processed through 16 layers to produce a classification output vector:

$$\mathbf{o}_{\mathcal{I}} = (\mathbf{o}_{\mathcal{I}}[1], \cdots, \mathbf{o}_{\mathcal{I}}[10])\,, \tag{1}$$

where $0 \leq \mathbf{o}_{\mathcal{I}}[i] \leq 1$ for $1 \leq i \leq 10$, and $\sum_{i=1}^{10} \mathbf{o}_{\mathcal{I}}[i] = 1$. Each c_i-label value $\mathbf{o}_{\mathcal{I}}[i]$ measures the probability that image $\mathcal{I}$ belongs to category c_i. Consequently, an image $\mathcal{I}$ is classified as belonging to category c_k if $k = \arg\max_{1 \leq i \leq 10}(\mathbf{o}_{\mathcal{I}}[i])$. The higher the label value $\mathbf{o}_{\mathcal{I}}[k]$, the higher the confidence that $\mathcal{I}$ represents an object of category c_k.

B. TARGETED AND UNTARGETED SCENARIOS

The *target* scenario consists of first choosing two different categories, $c_t \neq c_a$, among the 10 categories of CIFAR-10. Then, one is given an ancestor image $\mathcal{A}$ labeled by VGG-16 as belonging to c_a. Finally, one constructs an adversarial image $\mathcal{D}$, classified by VGG-16 as belonging to c_t, although $\mathcal{D}$ remains so close to $\mathcal{A}$ that a human would likely classify $\mathcal{D}$ as belonging to c_a or even be unable to distinguish $\mathcal{D}$ from $\mathcal{A}$. The classification threshold value is set at $\tau = 0.95$, meaning that such a $\mathcal{D}$ has achieved its purpose if $\mathbf{o}_{\mathcal{D}}[t] \geq 0.95$. We shall also encounter in Section VI the slightly different *untargeted* scenario. In this case, an adversarial image $\mathcal{D}$ is still required to be similar to $\mathcal{A}$ for a human eye, but one only requires that VGG-16 classifies $\mathcal{D}$ as belonging to a category $c \neq c_a$, in the limited sense that the label value of c outputted by VGG-16 for $\mathcal{D}$ is the largest among all label values, and is strictly larger than the label value of c_a. In particular, an image adversarial for the *target* scenario is also adversarial for the *untargeted* scenario, but the inverse may not be true.

IV. "ADAPTED_EA" VERSUS "CLASSIC_EA"

Our evolutionary algorithm $\text{EA}_d^{\text{target},\mathcal{C}}$ (see [1], [22]–[24], [32]) is a black-box, targeted attack that constructs adversarial images against a CNN $\mathcal{C}$ in the sense sketched in Subsection III-B, where d is a metric assessing the proximity for a human eye between the evolved images and the original image.

In this section, we show, from a "pure" evolutionary algorithm point of view, that $\text{EA}_d^{\text{target},\mathcal{C}}$ presents a series of important and substantial differences compared to the approach classically ([31]) adopted for EAs performing similar tasks, and we prove that these differences lead to a comparative advantage in terms of performance. First, we examine these differences from a conceptual point of view, meaning independently of any specific task. For simplicity, we refer to our version as "adapted_EA" and to its classical version as "classic_EA". We then compared the performances of these algorithms for the task consisting of fooling VGG-16 trained on CIFAR-10 for image recognition in the *target scenario*. In other words, these algorithms are given the task of evolving an ancestor image $\mathcal{A}$ into an adversarial image $\mathcal{D}$ fulfilling the conditions described in Subsection III-B. We specify the parameters of the EAs, and run the algorithms for four different ancestor/target combinations.

All experiments in this paper were implemented in Python 3.7 with the NumPy [33] library. For the filter experiments in Sections VI and VII, we used the OpenCV implementation library [34]. Keras [35] was used to load and run the VGG-16 [6] model. The experiments were performed on nodes with NVIDIA Tesla V100 GPGPUs of the IRIS HPC Cluster at the University of Luxembourg [36].

A. CONCEPTUAL DIFFERENCES BETWEEN "ADAPTED_EA" AND "CLASSIC_EA"

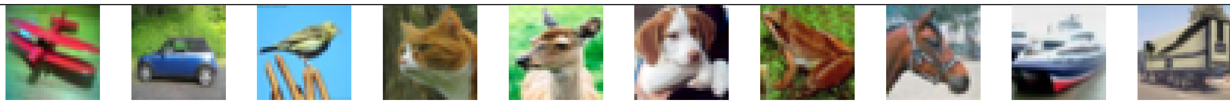
To illustrate the differences between our version ("adapted_EA") and the classic version ("classic_EA", as described in [31]) of an EA, let us provide their respective algorithmic pseudo-codes. We assume that both have a fixed population size, which remains constant geneation for generation. For both, we set the initial population as made of identical copies of the considered ancestor. Based on our experiments, we considered a population size of 160 as the best trade-off in terms of speed and accuracy.

Algorithm 1 "Classic_EA" algorithm pseudo code, the population size = N

```
BEGIN
    t = 0
    INITIALISE population P(t = 0) = A × N;
    EVALUATE P(t = 0);
    while isNotTerminated() do
        SELECT:
        P_e(t) = P(t).selectElites(10-20% of N); / elites
        RECOMBINE / MUTATE:
        P_c(t) = reproduction(P_e(t)); / off-springs;
        mutate(P_c(t));
        EVALUATE:
        P(t + 1) = evaluate(P_c(t), P(t));
        t = t + 1
    end
END
```

The main difference between "classic_EA" (as described in Algorithm 1) and our version (as described in Algorithm 2) is the process of selection, recombination and mutation. In

TABLE 1: For $1 \leq a \leq 10$, the image $\mathcal{A}_a$ (and its reference number n^o in the test set of CIFAR-10) classified by VGG-16 in the category c_a, with its corresponding c_a-label values. These images are used as ancestor in most of our experiments.

a	1	2	3	4	5	6	7	8	9	10
c_a	plane	car	bird	cat	deer	dog	frog	horse	ship	truck
n^o	281	82	67	91	455	16	29	17	1	76
$\mathcal{A}_a$	 0.6900	0.9999	0.9999	0.9998	0.9999	0.9996	0.9999	0.9998	0.9996	0.9984

Algorithm 2 "Adapted_EA" algorithm pseudo, the population size = N

1: BEGIN
2: $t = 0$
3: *INITIALISE* population $P(t = 0) = \mathcal{A} \times N$;
4: *EVALUATE* $P(t = 0)$;
5: **while** *isNotTerminated()* **do**
6: *SELECT:*
7: $P_e(t) = P(t).selectElites(N_e = 10)$; / elites
8: $P_w(t) = P(t).selectWorsts(N/2)$; / "didn't make it"
9: $Pm(t) = P(t) - (P_e(t) U P_w(t))$; //middle-class
10: *RECOMBINE / MUTATE:*
11: $P_{keep}(t) = P(t).randomSelect(N/2 - N_e)$ U $P_e(t)$;
12: $mutate(P_{keep}(t)$ U $P_m(t))$;
13: $P_c(t) = reproduction(P_{keep}(t), P_m(t))$;
14: *EVALUATE:*
15: $P(t + 1) = evaluate(P_e(t), P_c(t))$;
16: $t = t + 1$
17: **end**
18: END

"classic_EA", the best 10-20% of the population are selected as elites (hence between 16 and 32 individuals), and new offsprings are generated with these elites by recombination and mutation . Then the last 10-20% (idem) of the population is eliminated, and only these 10-20% are updated at each generation (see line 6-10 in Algorithm 1). However, in our version, the number of elites is set to the first 10 individuals; then, the algorithm starts to modify the whole rest (150 individuals) of the population by eliminating, mutating, and recombining with elites just after the first generation.

B. THE EA PARAMETERS

The task on which we evaluate the performance of both approaches is the construction of adversarial images for CNNs. Although our algorithm $\text{EA}_d^{\text{target},\mathcal{C}}$ is efficient for a series of CNNs, here we make our point for the instantiation $\text{EA}_{L_2}^{\text{target,VGG-16}}$ of this algorithm (Algorithm 2) and of its classical EA version (Algorithm 1), for $\mathcal{C}$ = VGG-16 trained on CIFAR-10, and for the metric $d = L_2$. Starting from a common ancestor image $\mathcal{A}$ of size $32 \times 32 \times 3$ labeled by VGG-16 as belonging to c_a, and from a target category $c_t \neq c_a$, the specific parameters and choices of the algorithms are as follows:

Population initialization. Both algorithms start the search with the same initial population set, made of 160 identical replicas of the ancestor image $\mathcal{A}$.

Evaluation - Fitness Function. This operation is performed on each individual image ind of a given generation g_p *via* the fitness function $fit_{L_2}(ind, g_p)$ that assesses a dual goal: the evolution of ind towards the target category c_t, and its proximity to ancestor $\mathcal{A}$, measured by using the L_2-norm:

$$fit_{L_2}(ind, g_p) = A(g_p, ind)\mathbf{o}_{ind}[c_t] - B(g_p, ind)L_2(ind, \mathcal{A}) \geq 0, \quad (2)$$

where the quantities $A(g_p, ind), B(g_p, ind) \geq 0$ weight and balance the dual goal. The L_2-norm is used to calculate the difference between the pixel values of the ancestor and the considered image ind:

$$L_2(ind, \mathcal{A}) = \sum_{p_j} |ind[p_j] - \mathcal{A}[p_j]|^2, \quad (3)$$

where p_j is the pixel in the j^{th} position, and $0 \leq ind[p_j], \mathcal{A}[p_j] \leq 255$ are the corresponding pixel values of the images ind and $\mathcal{A}$. Concretely, for any generation g_p, one sets $B(g_p, ind) = 10^{-5}$. The value of $A(g_p, ind)$ depends on $\mathbf{o}_{ind}[c_t]$ (note that $\log_{10} \mathbf{o}_{ind}[c_t] \leq 0$).

$$A(g_p, ind) = 10^{-\log_{10} \mathbf{o}_{ind}[c_t]} \quad (4)$$

Selection, Recombination, Mutation. The fitness function of each individual in the population is computed (starting with the first generation made of the initial population).

In adapted_EA, the population is sorted into three groups depending on their fitness values in the selection process. The elite, which is composed of the 10 individuals with the best fitness values. The lower class, "didn't make it", is the last half of the population, while the remaining 70 individuals constitute the middle class (line 6-9 in Algorithm 2). To replace the lower class, a "keep" group is created by combining elites and 70 random individuals from the previous population (line 11 in Algorithm 2). then mutation and cross-over is applied to the entire population (except the elites) to increase the exploration capability of the algorithm (line 12-13 in Algorithm 2). During these processes, only the elites pass unchanged to the next generation. Finally, these operations lead to 160 descendant images composing the individuals of the new generation subject to the next round of evaluation (line 14 in Algorithm 2).

The algorithms used the same parameters and techniques for the mutation and crossover operations as described in [22], [32]. In a nutshell, pixel mutations (only the value of a channel of a pixel location is changed) are performed by randomly choosing (with a power law) the number of pixels to be mutated and modifying their values in the range ± 3 in the two versions of EA used here. And cross-overs are essentially obtained by swapping a rectangular area at a uniformly random location between two individuals. The values of the rectangle's length and width are chosen uniformly random and can be between 1 and 30. Cross-overs are performed on a single channel, chosen uniformly random.

For a fair competition, the same seed value was applied to the adapted_EA and classic_EA to ensure fair competition. For the mutation, this seed impacts the location of the pixels and the magnitude (within a range defined below) of the modifications they undergo. For the cross-over, the seed impacts which individuals form pairs, as well as the location and size of the interchanged regions.

Termination condition. For each version of the EA, this loop is repeated until a descendant image is created in less than 7000 generations (this maximum number of iterations is a reasonable trade-off, based on our experiments), which is classified as the target category c_t with a probability ≥ 0.95, while remaining so close to $\mathcal{A}$ that a human would not notice any difference between it and $\mathcal{A}$ (and *a fortiori* would classify this descendant image still as belonging to the original category c_a). This defines a **successful termination**, in which case one notes $\mathcal{D}_{a,t}(\mathcal{A})$ as the adversarial image resulting from $\text{EA}_{L_2}^{\text{target,VGG-16}}$ (Algorithm 2) run on $\mathcal{A}$, and $\mathcal{D}_{a,t}^{\text{classic}}(\mathcal{A})$ as the result of the classic (Algorithm 1) version of the EA also run on $\mathcal{A}$. Otherwise, the algorithm **terminates without success**.

Therefore, the algorithms terminate after 7000 generations at the latest, regardless of whether they have succeeded in creating such an adversarial image.

C. EXPERIMENTAL COMPARISON OF "ADAPTED_EA" WITH "CLASSIC_EA"

We experimentally compared the efficiency of both versions of the EA for four ancestor/target pairs of categories Animal/Animal, Object/Object, Animal/Object, and Object/Animal.

Concretely, the Animal ancestor categories are bird and dog, with image $\mathcal{A}_3$ as ancestor for the bird category c_3, and $\mathcal{A}_6$ as ancestor for the dog category c_6 taken from Table 1. Similarly, the Object ancestor categories are plane and ship, with images $\mathcal{A}_1$ as the ancestor for the plane category c_1, and image $\mathcal{A}_9$ as the ancestor for the ship category c_9.

With these ancestors, we performed 10 independent runs of the algorithms for each of the following combinations: the bird/cat pair (Animal/Animal), plane/truck pair (Object/Object), dog/car pair (Animal/Object), and ship/horse pair (Object/Animal).

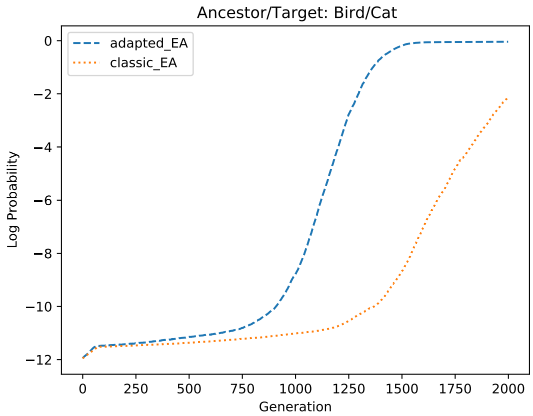
Performance comparison. In all cases, the 10 independent runs of each algorithm succeeded in (far) less than 7000 generations. Table 2 lists the minimum number of generations (min_{gen}), maximum number of generations (max_{gen}), and mean generations ($mean_{gen}$) obtained over the 10 independent runs of each algorithm. The convergence graph, plotted in Figure 2, shows the convergence speed of both algorithms for all cases. The horizontal axis of these graphs is the number of generations, and the vertical axis is the average log probability of the target category obtained for these 10 independent runs.

TABLE 2: Comparison of classic_EA and adapted_EA in generating adversarial images for the *target scenario* for 4 different Ancestor/Target combinations ($\mathcal{A}_a$ is the ancestor image in c_a used in the experiments) to fool VGG-16 trained on CIFAR-10. The results are over the 10 independent runs of each algorithm.

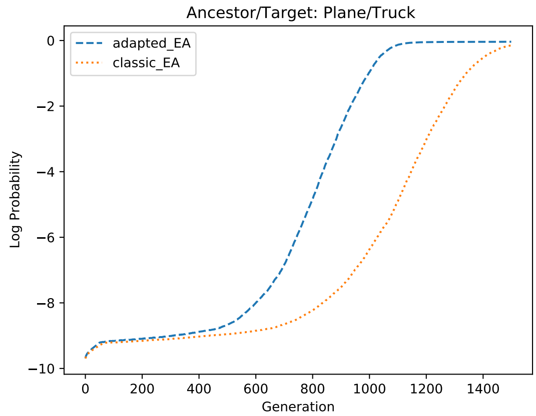
Ancestor/Target	Algorithms	min_{gen}	max_{gen}	$mean_{gen}$
bird ($\mathcal{A}_3$)/cat	classic_EA	1726	2433	2172.9
	adapted_EA	1353	2177	1629.2
plane ($\mathcal{A}_1$)/truck	classic_EA	1311	1810	1547.5
	adapted_EA	1050	1439	1194.8
dog ($\mathcal{A}_6$)/car	classic_EA	1132	1334	1199.3
	adapted_EA	811	1050	907.0
ship ($\mathcal{A}_9$)/horse	classic_EA	1972	3412	2582.1
	adapted_EA	1543	3171	2377.8

Results and Discussion. As can be seen in Table 2, "adapted_EA" outperforms "classic_EA" in all cases. The former requires fewer generations than the latter to obtain adversarial images with a confidence of 0.95. Figure 2 confirms that "adapted_EA" converges faster than "classic_EA". The graphs indicate that both algorithms apparently exhaust most of their generations to find the correct regions and/or pixels to modify. Once done, their learning curves accelerate drastically, still with "adapted_EA" leading the race against "classic_EA".

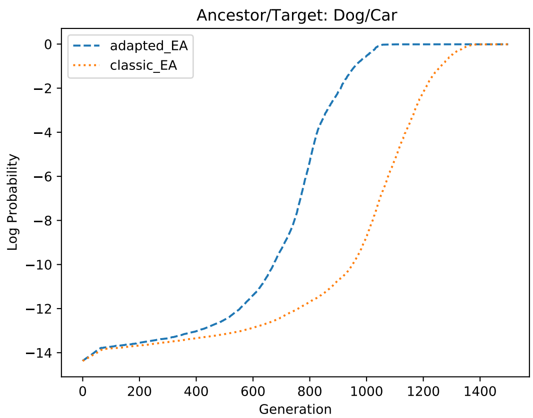
Although both algorithms start the search with the same 160 identical images, their respective performances differ substantially, as a consequence of their distinct updating process of the population. Indeed, "adapted_EA" starts these updates for the whole population, except for the elite individuals passed unchanged to the next generation, and does so right after the 1st generation. However, "classic_EA" only updates 20% of its population in each generation. Changing only 32 individuals, as opposed to changing 150 individuals, makes it much slower for the classic version compared to its adapted competitor. These results not only legitimize the choices made in our earlier work ([1], [22]–



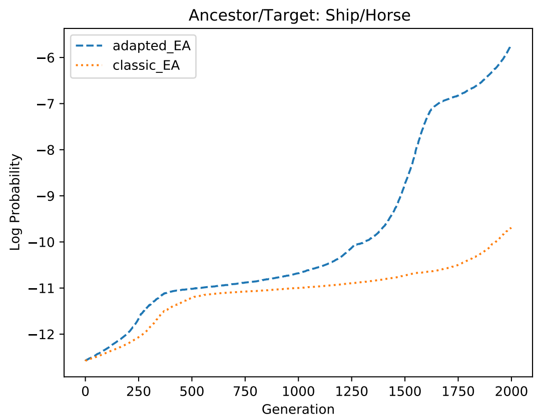
(a)



(b)



(c)



(d)

FIGURE 2: Convergence characteristics of classic_EA and adapted_EA for different ancestor/target pairs. These experiments are performed with the ancestors $\mathcal{A}_3$ (in the bird category), $\mathcal{A}_6$ (dog), $\mathcal{A}_1$ (plane) and $\mathcal{A}_9$ (ship) taken from Table 1.

[24], [32]), but also provide some evidence that for similar exploration problems with a starting point made of the same individuals (hence, not only for the construction of images that are adversarial for a CNN), the generic selection and mutation process adopted in "adapted_EA" (algorithm 2) shortens the learning period of the algorithm and enhances the convergence speed.

We complete this comparative analysis by assessing the potential differences in behavior between the adversarial images created by each version of the EA. To this purpose, we computed the Kullback-Leibler divergence [37] between the probability densities derived from the normalized histograms of the pixel modifications induced by each of them. In all cases, the values (averaged over the ten independent runs) of the Kullback-Leibler divergences were negligible (they vary between $2.24e-04$ and $5.17e-03$), indicating that the noise created by one version of the EA significantly differs from the noise created by the other. Hence, while both versions of the EA create adversarial images, the modifications introduced by each of them differ strongly, although both these modifications introduced by each EA on the one hand, as well as their differences on the other hand, are not perceptible by a human.

V. THE ADVERSARIAL IMAGES OBTAINED BY $\text{EA}_{L_2}^{\text{target,VGG-16}}$

As a result of Section IV, from now on we only consider the "adapted_EA" version of our evolutionary algorithm, namely $\text{EA}_{L_2}^{\text{target,VGG-16}}$. For an ancestor $\mathcal{A}_a$ in a category c_a, and the target scenario for category c_t, one defines $\mathcal{D}_{a,t}(\mathcal{A}_a) = \text{EA}_{L_2}^{\text{target,VGG-16}}(\mathcal{A}_a, c_t)$, provided that the algorithm terminates successfully. One writes more simply $\mathcal{D}_{a,t}$, or even $\mathcal{D}_t$, if there is no ambiguity about the choice of the ancestor $\mathcal{A}_a$ chosen in category c_a (*mutatis mutandis* in Sections VI and VII).

A. WITH ONE ANCESTOR PER CATEGORY

From Table 1, we pick the ancestor image $\mathcal{A}_a$ in category c_a and perform 10 independent runs (with random seed values) of $\text{EA}_{L_2}^{\text{target,VGG-16}}$ for all nine possible target categories $c_t \neq c_a$.

An example of the quality of the obtained adversarial images is highlighted by the comparison between the dog ancestor $\mathcal{A}_6$ of Table 1, and its corresponding 9 evolved adversarial images $\mathcal{D}_t$, with $t \neq 6$ (obtained after the first of the 10 independent runs of the EA) as shown in Figure 3. More generally, Figure 11 (Appendix A) contains the adversarial images obtained by the first successful run out of the ten independent runs of $\text{EA}_{L_2}^{\text{target,VGG-16}}$ for each of the ancestor images in Table 1, and Table 5 (Appendix A) gives their respective label values.

This example illustrates that by slightly changing many pixels instead of heavily changing a few pixels, our approach enhances the indistinguishability between the adversarial image and the ancestor image. In particular, our method differs substantially from [25], [26], [28], where a small fraction of pixels is changed, but at the cost of being noticeable for a human without difficulty (see Figure 1, Section I).

For the ancestor image $\mathcal{A}_a$ (from Table 1) in category c_a specified in its a^{th} row, the t^{th} column of Figure 4 gives the average number of generations required by $\text{EA}_{L_2}^{\text{target,VGG-16}}$ to terminate, computed over 10 independent runs. In the four ancestor/target combinations, this number is followed by a symbol $(\star x)$ or $(\star x, \ddagger y)$. These symbols indicate that the algorithm did not achieve the $\tau = 0.95$ threshold value within 7000 generations for x of the 10 runs, and therefore terminated without success for the corresponding seed values. The c_t-label values of the corresponding best descendant images remained stuck at a local optimum < 0.95, whose *quality* is also indicated by the symbol. In the case of symbol $(\star x)$, this local optimum was quite close to 0.95 (not less than 0.9370 actually; we call *quasi-adversarial* the corresponding images produced by $\text{EA}_{L_2}^{\text{target,VGG-16}}$). In the case of symbol $(\star x, \ddagger y)$, the complementary number y specifies the number of runs among the x unsuccessful runs for which the local optimum remained very low (between circa 10^{-4} to 10^{-5}).

For each $1 \leq a \leq 10$, the "Row Average" value, displayed in the rightmost column of the a^{th} row, indicates the average number of generations required to perform our attack on ancestor $\mathcal{A}_a$ in category c_a for all $c_t \neq c_a$ (*Mutatis mutandis* the "Column Average" value displayed in the bottom row of the t^{th} column).

Our EA showed a success rate of 100 % since all possible target categories were achieved with at least one of the ten runs for the considered ancestors. Still, some attacks are easier than others. The ancestor image for which $\text{EA}_{L_2}^{\text{target,VGG-16}}$ requires the least amount of effort in general is the $horse$ ancestor image $\mathcal{A}_8$, and $bird$ (c_3) is the easiest target category regardless of the ancestor category (with the considered ancestor images at least). At the other end of the scale are the $deer$ ancestor image $\mathcal{A}_5$ and the $bird$ ancestor image $\mathcal{A}_3$ for which $\text{EA}_{L_2}^{\text{target,VGG-16}}$ requires the largest amount of effort in general, while dog (c_6), $truck$ (c_{10}) and $ship$ (c_9) are the hardest target categories. These correspond precisely to the categories (and the ancestors) for which some runs of $\text{EA}_{L_2}^{\text{target,VGG-16}}$ terminated without having created an appropriate adversarial image within 7000 generations. Indeed, out of the altogether 900 attacks (10 runs for each of the 90 ancestor/target combinations) performed by $\text{EA}_{L_2}^{\text{target,VGG-16}}$, Figure 4 shows that only 31 did not succeed. It is worth noting the homogeneity and non-diversity of the quality of rare unsuccessful cases. For such an unsuccessful (c_a, c_t) combination, either the local optimum is close to the $\tau = 0.95$ value for all failed cases (this occurs for the nine unsuccessful runs of the $(bird\ (\mathcal{A}_4), dog)$ combination), or it is very far from this threshold value for all failed cases (this occurs for the 22 unsuccessful runs with the $deer$ ($\mathcal{A}_5$) ancestor for the car, $ship$ and $truck$ targets).

Therefore, as a consequence of this study with one ancestor $\mathcal{A}_a$ per category c_a, our experiments show that the probability that $\text{EA}_{L_2}^{\text{target,VGG-16}}$ terminates successfully for a given run is 96.56%, and that its termination requires between 290 and 2793 generations on average.

As for the run time of $\text{EA}_{L_2}^{\text{target,VGG-16}}$, Figure 12 in the Appendix A gives the average time (in seconds) required by all successfully completed ancestor/target combinations, with one ancestor per category. On average, it takes 79.77 seconds to generate a successful adversarial image.

B. WITH 50 DISTINCT ANCESTORS PER CATEGORY

To further evaluate the efficiency of our attack beyond the case of a single ancestor $\mathcal{A}_a$ per category c_a, as described in Subsection V-A, and to assess the importance of a specific ancestor chosen in a given category, we considered 50 distinct images taken randomly (from the CIFAR-10 testing set) in each of the 10 categories c_a. Unlike the 10 independent runs per ancestor of Subsection V-A, we considered that running $\text{EA}_{L_2}^{\text{target,VGG-16}}$ with one single run per ancestor was enough to make our point. So in total, we performed $50 \times 10 \times 9 = 4500$ attacks with $\text{EA}_{L_2}^{\text{target,VGG-16}}$. Figure 5, which summarizes the outcome of this experiment, is to be interpreted in a similar way as Figure 4, with the difference that the averages are computed over the 50 ancestors per category c_a. Note also that the $(\star x)$ and $(\star x, \ddagger y)$ symbols added to some cell values for a given (c_a, c_t) scenario have a different interpretation in Figure 5 compared to Figure 4, since they apply globally to *different ancestors* here, as opposed to applying to different runs performed on the *same ancestor* in Figure 4.

Performance differs again from one category to another. The ancestor categories for which $\text{EA}_{L_2}^{\text{target,VGG-16}}$ requires

FIGURE 3: From the left, comparison of the ancestor $\mathcal{A}_6$ in the 6^{th} position with the adversarial images $\mathcal{D}_t$ in the t^{th} position ($t \neq 6$). VGG-16 classifies $\mathcal{A}_6$ in the *dog* category with probability 0.9996386, and classifies $\mathcal{D}_t$ in the target category c_t with probability ≥ 0.95.

	plane	car	bird	cat	deer	dog	frog	horse	ship	truck	Row Average
plane ($\mathcal{A}_1$)		64	275	2188	702	613	337	798	147	1108	692
car ($\mathcal{A}_2$)	1095		451	1246	768	1545	543	676	422	725	830
bird ($\mathcal{A}_3$)	1080	665		1823	925	6921(*9)	559	2719	872	1092	1850
cat ($\mathcal{A}_4$)	494	341	250		263	217	113	411	526	555	352
deer ($\mathcal{A}_5$)	2700	6233(*5, ‡5)	343	460		239	834	712	6683(*8, ‡8)	6939(*9, ‡9)	2793
dog ($\mathcal{A}_6$)	879	882	460	129	938		397	280	971	545	609
frog ($\mathcal{A}_7$)	690	520	295	488	717	536		834	927	685	632
horse ($\mathcal{A}_8$)	454	221	300	204	223	303	371		309	228	290
ship ($\mathcal{A}_9$)	318	182	1291	432	2599	1502	823	2065		484	1077
truck ($\mathcal{A}_{10}$)	145	663	383	1411	437	864	919	271	292		598
Column Average	872	1085	449	931	841	1415	544	974	1238	1373	

FIGURE 4: $\text{EA}_{L_2}^{\text{target,VGG-16}}$'s performance on all possible ancestor/target combinations with one ancestor per category. The rows give the ancestor category c_a (and the specific ancestor $\mathcal{A}_a$ in c_a), the columns indicate the target class c_t, and the cell values indicate the average number of generations required by $\text{EA}_{L_2}^{\text{target,VGG-16}}$ to terminate, computed on 10 independent runs.

	plane	car	bird	cat	deer	dog	frog	horse	ship	truck	Row Average
plane		1201 (*2, ‡2)	284	606	415	859 (*1)	752 (*1, ‡1)	858	304	1042 (*3, ‡3)	702
car	1807 (*5, ‡5)		1740 (*3, ‡3)	2618 (*8, ‡8)	1751 (*3, ‡3)	2154 (*4, ‡4)	1492 (*2, ‡2)	2425 (*8, ‡8)	988 (*1, ‡1)	478	1717
bird	416	1029 (*1, ‡1)		376	390	537	397	679	575	844 (*1, ‡1)	583
cat	653 (*1)	703	358		381	152	234	321	834	519	462
deer	762 (*2, ‡2)	1459 (*4, ‡3)	208	290		274	382	269	855 (*1, ‡1)	1139 (*4, ‡4)	626
dog	772 (*1, ‡1)	799	319	203	492		344	392	609	686	513
frog	527	646	306	302	321	463		588	532	466	461
horse	1343 (*2, ‡2)	1869 (*4, ‡3)	851	692	310	325	1085 (*1, ‡1)		1679 (*4, ‡4)	2252 (*9, ‡9)	1156
ship	454	708	890	1044 (*2, ‡2)	684 (*1, ‡1)	1246 (*1)	734	1319 (*2, ‡1)		639	858
truck	576	495 (*1, ‡1)	813	912 (*1, ‡1)	1059 (*1, ‡1)	1077 (*2, ‡1)	994	908	395		803
Column Average	812	990	641	783	645	787	713	862	752	896	

FIGURE 5: $\text{EA}_{L_2}^{\text{target,VGG-16}}$'s performance on all possible ancestor/target combinations with 50 distinct ancestors per category. The rows give the ancestor category c_a, the columns indicate the target class c_t. The cell values give the average number of generations required by $\text{EA}_{L_2}^{\text{target,VGG-16}}$ to terminate, and computed on one run performed on each of the 50 ancestors in the category c_a.

the least amount of effort in general are the $frog$, cat, and dog categories. In addition, $\text{EA}_{L_2}^{\text{target,VGG-16}}$ achieves the target categories $bird$ and $deer$ fairly fast, regardless of the ancestor categories. Conversely, the ancestor categories car and $horse$ are those for which $\text{EA}_{L_2}^{\text{target,VGG-16}}$ requires the largest amount of effort in general, while the car and the $truck$ are the hardest target categories.

In this context, the comparison of these results with those of Figure 4 shows the relevance for $\text{EA}_{L_2}^{\text{target,VGG-16}}$'s performance of the specific ancestor image chosen in a given category c_a. Indeed, while, for instance, the specific ancestor $\mathcal{A}_8$ in the $horse$ category was optimal in a sense (achieving all possible target categories in 290 generations on average), this property did not extend to the $horse$ category as a whole as just seen. *A contrario*, for instance, the combination $(deer, truck)$ with the ancestor $\mathcal{A}_5$ in the $deer$ category was (with 6939 generations on average) the toughest to achieve among all trials of Subsection V-A, it is reasonably easy to achieve in general (with 1139 generations on average) with the 50 ancestors chosen for our experiment.

Finally, out of the 4500 trials performed by $\text{EA}_{L_2}^{\text{target,VGG-16}}$, only 87 did not terminate successfully. Therefore, this experiment provides heuristic evidence that one run of $\text{EA}_{L_2}^{\text{target,VGG-16}}$ has a probability of 98.06% to terminate successfully. To better assess the strength of the failed cases, we ran again the 87 unsuccessful cases 10 times with different seed values: out of them, 28 succeeded in less than 10 runs, while 59 did not. This result, together with the fact that our algorithm required between 461 and 1717 generations on average in this case, and compared to the outcome of the similar experiments performed in the previous Subsection V-A with other ancestors, further sustains the impact of the specific ancestor $\mathcal{A}_a$ taken in a given category c_a, and of the seed value used to run the EA. It also shows that the success rate of our attack, namely the capacity for $\text{EA}_{L_2}^{\text{target,VGG-16}}$ to terminate successfully for at least one of ten runs out of a small number of trials, is $\geq 98.68\%$.

VI. ROBUSTNESS OF $\text{EA}_{L_2}^{\text{target,VGG-16}}$ AGAINST FILTERS

For the reasons given in the introduction to this paper (Section I), the study undertaken in this section essentially amounts to checking whether adding filters may prevent VGG-16 from misclassification, or may reduce this risk to some extent, when facing an adversarial image created by $\text{EA}_{L_2}^{\text{target,VGG-16}}$.

A. SELECTION OF FILTERS

Although a large list of filters exists, we focus on the following four filters that have a significant impact on images [38, Chapters 7 and 8].

The *inverse filter* F_1 replaces all the colours by their complementary colours. This operation is performed pixel for pixel by subtracting the RGB value $(255, 255, 255)$ of white by the RGB value of that pixel.

The *Gaussian blur filter* F_2 uses a Gaussian distribution to calculate the kernel, $G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$, where x is the distance from the origin on the x-axis, y is the distance from the origin on the y-axis and σ is the standard deviation of the Gaussian distribution. By design, the process gives more priority to the pixels in the center, and blurs around it with a lesser impact as one moves away from the center.

The *median filter* F_3 is used to reduce noise and artifacts in a picture. Although under some conditions it can reduce noise while preserving the edges, this does not really occur for small images such as those considered here. In general, one selects a pixel and computes the median of all the surrounding pixels.

The *unsharp mask filter* F_4 enhances the sharpness and contrast of the images. The unsharp-masked image is obtained by blurring a copy of the image using a Gaussian blur, which is then weighted and subtracted from the original image.

Any filter F, or any combination of filters $F_{i_1}, F_{i_2}, \cdots, F_{i_k}$ operating successively (in that order) on an image $\mathcal{I}$, creates a filtered image $F(\mathcal{I})$ or $F_{i_k} \circ \cdots \circ F_{i_2} \circ F_{i_1}(\mathcal{I})$.

We make use of these four filters $F_1, F_2, F_3,$, and F_4 either individually or as the combination $F_5 = F_3 \circ F_4$. The reason for the choice of the latter $F_3 \circ F_4$ is that F_4 is used to amplify and highlight detail, while F_3 is used to remove noise from an image without removing detail. Therefore, a combination of these filters can remove the noise created by the EA while maintaining a high level of detail. Moreover, because the computations are performed on images of size 32×32, we take a filter size $f = 1$ for F_1 and $f = 3$ for the others.

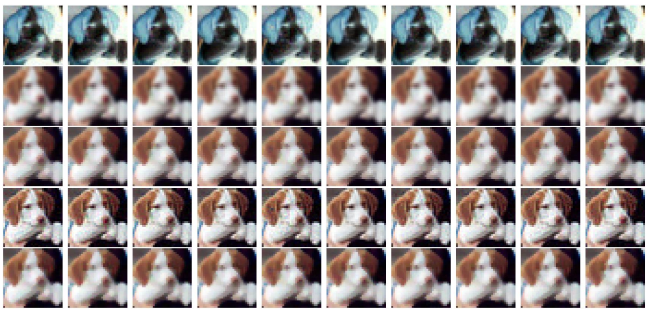


FIGURE 6: Comparison of the impact of filters on the ancestor $\mathcal{A}_6$ and on the adversarial images $\mathcal{D}_t$. The k^{th} row represents $F(\mathcal{D}_t)$ in t^{th} position (with $\mathcal{D}_6 = \mathcal{A}_6$), where $F = F_k$ for $1 \leq k \leq 5$.

For each $F = F_k$, $1 \leq k \leq 5$, we then challenge VGG-16 with these 100 filtered images $F(\mathcal{A}_a)$ and $F(\mathcal{D}_{a,t}(\mathcal{A}_a))$.

The complete classification and the corresponding label values output by VGG-16 for $F(\mathcal{A}_a)$ and $F(\mathcal{D}_{a,t}(\mathcal{A}_a))$ for the five considered filters and for all (c_a, c_t) combinations are given in Tables 6 to 10 (Appendix A). In these tables, an image is classified as belonging to category c if c has the largest label value outputted by VGG-16 among all categories.

B. INDICATORS ADDRESSING THE ROBUSTNESS OF FILTERED ADVERSARIALS

Filters differ substantially in their individual capacities to sustain the adversarial component of the filtered $F(\mathcal{D}_{a,t}(\mathcal{A}_a))$. Additionally, it may also happen that VGG-16 classifies $F(\mathcal{A}_a)$ in a category different from the ancestor category c_a. Since in this section (and the next one) we consider that the classification of an image in a given category c means that the label value given by VGG-16 for c is the largest among all possible categories, we relax the formulation of the *target scenario* accordingly; in this context, one does not necessarily require a target label value exceeding the threshold value of 0.95, but only asks that it is the largest one. The formulation of the *untargeted scenario* in the filtered context, made precise below in this subsection, requires paying attention to the potential difference between the categories c_a and $c_{F(\mathcal{A}_a)}$.

The following indicators quantitatively assess the aforementioned issues for each filter F_k, with the considered ancestors and adversarial images. These indicators take integer values, and we specify their theoretical bounds (which clearly depend on the number 10 of ancestors, and on the number 9 of target categories considered in this study).

For each $1 \leq a \leq 10$, we first define $\rho_k(\mathcal{A}_a)$ as the number of target categories c_t such that VGG-16 classifies $F_k(\mathcal{D}_{a,t}(\mathcal{A}_a))$ (including potentially $\mathcal{D}_{a,a}(\mathcal{A}_a) = \mathcal{A}_a$) back to the ancestor category c_a. One computes $\Sigma_k = \sum_{a=1}^{10} \rho_k(\mathcal{A}_a) \in [0, 100]$.

One sets $\delta_k(\mathcal{A}_a) = 1$ if $\rho_k(\mathcal{A}_a) = 10$, that is, if the filtered ancestor and all filtered adversarial images are classified back to the ancestor category. Otherwise $\delta_k(\mathcal{A}_a) = 0$. One computes $\Delta_k = \sum_{a=1}^{10} \delta_k(\mathcal{A}_a) \in [0, 10]$.

One sets $\mu_k(\mathcal{A}_a) = 0$ if VGG-16 classifies $F_k(\mathcal{A}_a)$ back to c_a, and $\mu_k(\mathcal{A}_a) = 1$ if it does not. One defines $\mathcal{M}_k = \sum_{a=1}^{10} \mu_k(\mathcal{A}_a) \in [0, 10]$.

Of interest for the *target scenario* is $\tau_k(\mathcal{A}_a)$, the number of $t \neq a$ for which $F_k(\mathcal{D}_{a,t}(\mathcal{A}_a))$ is classified as belonging to c_t (namely those that "really succeed"), and its sum $\mathcal{T}_k = \sum_{a=1}^{10} \tau_k(\mathcal{A}_a) \in [0, 90]$.

Finally, we consider $\widetilde{\tau}_k(a)$ to assess the *untargeted scenario*: $\widetilde{\tau}_k(a)$ counts the number of $t \neq a$ for which $F_k(\mathcal{D}_{a,t}(\mathcal{A}_a))$ is classified as belonging to $c \neq c_{F_k(\mathcal{A}_a)}$. One computes its sum $\widetilde{\mathcal{T}}_k = \sum_{a=1}^{10} \widetilde{\tau}_k(\mathcal{A}_a) \in [0, 90]$.

Observe *en passant* that the inequality $\mathcal{T}_k \leq \widetilde{\mathcal{T}}_k$ may theoretically not hold (as opposed to what happens in the absence of any filter, where the corresponding inequality necessarily holds). The reason is that one considers $c_t \neq c_a$ for the left-hand side of the inequality, and $c \neq c_{F_k(\mathcal{A}_a)}$ for the right-hand side. Since the quantities c_a and $c_{F_k(\mathcal{A}_a)}$ may differ, the set whose number of elements is $\mathcal{T}_k$ may not be included in the set whose number of elements is $\widetilde{\mathcal{T}}_k$.

C. ROBUSTNESS ANALYSIS OF THE ADVERSARIAL $\mathcal{D}_{A,T}(\mathcal{A}_A)$ AGAINST FILTERS

Let us now proceed to the analysis of Table 3, which provides these quantities resulting from, and summarizing Tables 6 to 10 (Appendix A).

TABLE 3: Indicator values assessing the robustness of adversarial images $\mathcal{D}_{a,t}(\mathcal{A}_a)$ against filters. For each ancestor $\mathcal{A}_a$, computation of $(\rho_k(\mathcal{A}_a), \delta_k(\mathcal{A}_a), \mu_k(\mathcal{A}_a))$ in the 1st row, and of $(\tau_k(\mathcal{A}_a), \widetilde{\tau}_k(\mathcal{A}_a))$ in the 2nd row. The last two rows give the sums $\sum_{a=1}^{10}$ of these quantities.

$\mathcal{A}_a$ \ k	1	2	3	4	5
$\mathcal{A}_1$	(10,1,0)	(0,0,1)	(2,0,0)	(0,0,1)	(7,0,0)
	(0,0)	(2,7)	(1,8)	(9,8)	(1,3)
$\mathcal{A}_2$	(1,0,0)	(3,0,0)	(9,0,0)	(3,0,0)	(10,1,0)
	(1,9)	(2,7)	(0,1)	(7,7)	(0,0)
$\mathcal{A}_3$	(7,0,0)	(10,1,0)	(10,1,0)	(1,0,0)	(10,1,0)
	(1,3)	(0,0)	(0,0)	(9,9)	(0,0)
$\mathcal{A}_4$	(3,0,0)	(10,1,0)	(10,1,0)	(1,0,0)	(10,1,0)
	(1,7)	(0,0)	(0,0)	(8,9)	(0,0)
$\mathcal{A}_5$	(1,0,1)	(10,1,0)	(10,1,0)	(1,0,0)	(10,1,0)
	(3,7)	(0,0)	(0,0)	(9,9)	(0,0)
$\mathcal{A}_6$	(0,0,1)	(0,0,1)	(1,0,1)	(1,0,0)	(4,0,0)
	(3,5)	(1,0)	(1,1)	(9,9)	(1,6)
$\mathcal{A}_7$	(5,0,0)	(10,1,0)	(10,1,0)	(2,0,0)	(10,1,0)
	(1,5)	(0,0)	(0,0)	(8,8)	(0,0)
$\mathcal{A}_8$	(0,0,1)	(10,1,0)	(10,1,0)	(1,0,0)	(10,1,0)
	(3,7)	(0,0)	(0,0)	(9,9)	(0,0)
$\mathcal{A}_9$	(6,0,0)	(1,0,1)	(8,0,0)	(1,0,0)	(8,0,0)
	(2,4)	(2,2)	(1,2)	(9,9)	(1,2)
$\mathcal{A}_{10}$	(0,0,1)	(0,0,1)	(10,1,0)	(1,0,0)	(10,1,0)
	(1,1)	(1,0)	(0,0)	(9,9)	(0,0)
$(\Sigma_k, \Delta_k, \mathcal{M}_k)$	(33, 1, 4)	(54, 5, 4)	(80, 6, 1)	(12, 0, 1)	(89, 7, 0)
$(\mathcal{T}_k, \widetilde{\mathcal{T}}_k)$	(16, 48)	(8, 16)	(3, 12)	(86, 86)	(3, 11)

Looking at Σ_k shows that, although all filters $F_1, \cdots, F_5$ bring some filtered images back to c_a, the unsharp mask (F_4) and the inverse (F_1) filters are less efficient in this regard. In contrast, the three other filters bring back a majority of the filtered images back to c_a. The median (F_3) filter and foremost the combination (F_5) of the Unsharp and Median filters are highly effective, since more than 80% of all filtered images are classified back to c_a. The three filters $F = F_2, F_3$, and F_5 are also those that bring all filtered images back to c_a for 5 (in the case of F_2), 6 (in the case of F_3) and 7 (in the case of F_5) ancestors, including *a fortiori* the filtered ancestor.

Consistently, the consideration of $\mathcal{T}_k$ and of $\widetilde{\mathcal{T}}_k$ shows that $\text{EA}_{L_2}^{\text{target,VGG-16}}$ resists highly efficiently against the Unsharp mask filter F_4, as 95 % (86 out of 90) filtered images remain adversarial for the *target scenario* (with target label values no less than 0.5505, see Table 9), and 95 % (86 out of 90) filtered images are adversarial for the *untargeted scenario*. Our EA is also significantly efficient against the inverse filter

F_1, as 17 % (16/90) filtered images remain adversarial for the *target scenario* (with target label values ≥ 0.4415, see Table 6), and 53 % (48/90) are adversarial for the *untargeted scenario*.

On the other hand, the Gaussian blur (F_2), the median (F_3), and the median and unsharp combined (F_5) filters are effective to a far larger extent against $\text{EA}_{L_2}^{\text{target,VGG-16}}$, with F_3 and F_5 being particularly efficient at removing the adversarial property of the descendant images. Indeed, only three filtered adversarial images (hence 3 % of all filtered) remain adversarial for the target scenario for each of these two filters (with target label values ≥ 0.4978 for F_3 and ≥ 0.8131 for F_5; see Tables 8 and 10). For the *untargeted scenario* finally, the proportion of filtered images that are adversarial drops to 13 % (12/90) for F_3, and to 12 % (11/90) for F_5.

This study proves that the inverse (F_1) and the unsharp mask (F_4) filters are largely inefficient against our EA, but that the Gaussian (F_2), and foremost, the median (F_3) and the combination ($F_5 = F_3 \circ F_4$) of the unsharp mask and the median filters render our EA-based attack significantly less effective for both the *targeted scenario* and the *untargeted scenario*, at least with the ancestor images considered.

VII. THE FILTER-ENHANCED F-FITNESS FUNCTION

The results of the previous section lead to the conception of a new fitness function that *natively* forces the EA to create adversarial images that remain adversarial (in the sense of Subsection III-B) once filtered. For a filter F, the filtered-enhanced F-fitness function is obtained as the following variant of the fitness function defined in Equation (2):

$$fit_{L_2}^{F}(ind, g_p) = A(g_p, ind)(\mathbf{o}_{ind}[c_t] + \mathbf{o}_{F(ind)}[c_t]) - B(g_p, ind)L_2(ind, \mathcal{A}), \quad (5)$$

where the component $\mathbf{o}_{F(ind)}[c_t]$ measures the probability that an individual filtered with F is classified as the target category. One obtains $\text{EA}_{L_2,F}^{\text{target,VGG-16}}$ from $\text{EA}_{L_2}^{\text{target,VGG-16}}$ by updating the fitness function accordingly. The termination and termination with success criteria are the same as in Subsection IV-B.

Since $F_5 = F_3 \circ F_4$ is not only highly efficient against $\text{EA}_{L_2}^{\text{target,VGG-16}}$, but is the filter that reverts the largest proportion (89 %) of images $\mathcal{D}_{a,t}(\mathcal{A}_a)$ back to c_a, we limit this study to this case.

A. RUNNING $\text{EA}_{L_2,F_5}^{\text{target,VGG-16}}$ WITH ONE ANCESTOR PER CATEGORY

For $1 \leq a \leq 10$, one performs 10 independent runs of $\text{EA}_{L_2,F_5}^{\text{target,VGG-16}}$ on the ancestor $\mathcal{A}_a$ in category c_a given in Table 1. If $\text{EA}_{L_2,F_5}^{\text{target,VGG-16}}$ terminates successfully, one writes $\mathcal{D}_{a,t}^{F_5}(\mathcal{A}_a)$ for the first adversarial image obtained by $\text{EA}_{L_2,F_5}^{\text{target,VGG-16}}$ in less than 7000 generations. By construction, this image and its by F_5 filtered version are classified by VGG-16 as belonging to the target category c_t with probability ≥ 0.95, while remaining so close to $\mathcal{A}_a$ for a human eye that no one would notice any difference.

Figure 7 pictures the adversarial images $\mathcal{D}_{6,t}^{F_5}(\mathcal{A}_6)$ obtained that way for the dog ancestor $\mathcal{A}_6$ (all first runs succeeded for the dog ancestor).

For the ancestor image $\mathcal{A}_a$ (taken from Table 1) in category c_a specified in its a^{th} row, the t^{th} row of Figure 8 gives the average number of generations required by $\text{EA}_{L_2,F_5}^{\text{target,VGG-16}}$ to terminate, computed over 10 independent runs. With a terminology adapted from the one used in Figure 4, this number is followed by a symbol $(\star x, \ddagger y, \dagger z)$ in 5 of the 90 cells. The occurrence of this symbol means that the algorithm did not terminate successfully for x out of the 10 runs (i.e., the average value = 7000 if $x = 10$). Not succeeding means that the c_t-label value of the most performing descendant images $\mathcal{D}$ or of the filtered image $F_5(\mathcal{D})$ is stuck at a local optimum < 0.95. The symbols $\ddagger y$ and $\dagger z$ measure the quality of these local optima. $\ddagger y$ (respectively, $\dagger z$) counts the number of runs among the x unsuccessful ones for which the local optimum for descendant $\mathcal{D}$ (respectively, $F_5(\mathcal{D})$) remained very low (between 10^{-3} and 10^{-6}).

Of the 900 performed runs, 38 did not terminate successfully, and 3 out of the 90 possible ancestor/target scenarios were not achieved, namely the pairs $(plane(\mathcal{A}_1), deer)$, $(bird(\mathcal{A}_3), car)$, $(horse(\mathcal{A}_8), ship)$. Therefore, the experiments show a success rate of $\text{EA}_{L_2,F_5}^{\text{target,VGG-16}}$ of 96.66%, and a probability that the algorithm terminates successfully for a given run of 95.77%.

Comparing Figure 8 to Figure 4, when all 10 runs terminate successfully for both $\text{EA}_{L_2}^{\text{target,VGG-16}}$ and $\text{EA}_{L_2,F_5}^{\text{target,VGG-16}}$ for an $(ancestor(\mathcal{A}_a), target)$ pair (83 cases altogether), the latter algorithm usually requires more generations than the former on average (with three notable exceptions, namely the $(ship(\mathcal{A}_9), deer)$, the $(ship(\mathcal{A}_9), dog)$ and the $(truck(\mathcal{A}_{10}), cat)$ pairs for which it needs 10%, 18% and 13% fewer generations). The fact that, for the 80 remaining pairs, $\text{EA}_{L_2,F_5}^{\text{target,VGG-16}}$ requires between 1.12 and 3.87 (depending on the pair considered) times more generations than $\text{EA}_{L_2}^{\text{target,VGG-16}}$ to terminate successfully is not surprising since there are 3 and no longer 2 criteria to fulfill.

For all 87 combinations $(ancestor(\mathcal{A}_a), target)$ for which $\text{EA}_{L_2,F_5}^{\text{target,VGG-16}}$ terminated successfully in at least one of the 10 independent runs, Figure 13 (Appendix B) displays the first adversarial image $\mathcal{D}_{a,t}^{F_5}(\mathcal{A}_a)$ obtained by $\text{EA}_{L_2,F_5}^{\text{target,VGG-16}}$ (with $\mathcal{D}_{a,t}^{F_5}(\mathcal{A}_a) = \mathcal{A}_a$ repeated on the diagonal for the sake of consistency and comparison), and Table 11 (Appendix B) gives the corresponding label values.

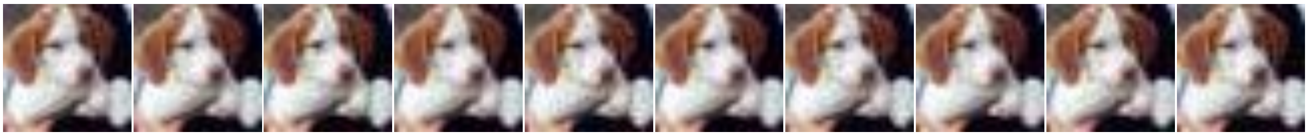


FIGURE 7: From left to right, comparison of the ancestor $\mathcal{A}_6$ in the 6[th] position with the adversarial images $\mathcal{D}^{F_5}_{6,t}(\mathcal{A}_6)$ in the t^{th} position ($t \neq 6$).

	plane	car	bird	cat	deer	dog	frog	horse	ship	truck	Row Average
plane ($\mathcal{A}_1$)		169	435	2455	7000 (*10, ‡0, †10)	866	512	1330	174	1562	1611
car ($\mathcal{A}_2$)	1600		562	1704	1170	1911	832	1640	655	1010	1231
bird ($\mathcal{A}_3$)	1320	7000 (*10, ‡0, †10)		1508	1197	2201	5132 (*7, ‡0, †7)	3440	1111	1808	2746
cat ($\mathcal{A}_4$)	1468	1320	459		559	359	266	839	1016	1466	861
deer ($\mathcal{A}_5$)	2931	4797 (*1, ‡1, †0)	723	810		431	1098	1217	2266	2924	1910
dog ($\mathcal{A}_6$)	1582	1275	723	189	2171		775	573	1440	1365	1121
frog ($\mathcal{A}_7$)	1761	1574	576	1074	1262	1037		1863	1775	1695	1401
horse ($\mathcal{A}_8$)	768	503	475	450	435	814	753		7000 (*10, ‡0,†10)	391	1287
ship ($\mathcal{A}_9$)	475	262	1333	740	2333	1226	1186	3279		811	1293
truck ($\mathcal{A}_{10}$)	225	1011	638	1224	706	1391	1051	436	503		798
Column Average	1347	1990	658	1128	1870	1137	1289	1624	1771	1448	

FIGURE 8: $\text{EA}^{\text{target,VGG-16}}_{L_2,F_5}$'s performance on all possible ancestor/target combinations. The rows give the ancestor categories c_a (and the specific ancestor $\mathcal{A}_a$ in c_a), the columns indicate the target class c_t, and the cell values give the average number of generations required by $\text{EA}^{\text{target,VGG-16}}_{L_2,F_5}$ to terminate, computed on 10 independent runs.

The competition time of $\text{EA}^{\text{target,VGG-16}}_{L_2,F_5}$ is given in Figure 14 (see the Appendix B). It shows the average time (in seconds) required by all successfully completed ancestor/target combinations, with one ancestor per category. On average, it takes 242.7 seconds to generate a successful adversarial image, which is almost 3 times slower that $\text{EA}^{\text{target,VGG-16}}_{L_2}$.

B. RUNNING $\text{EA}^{\textit{target, VGG-16}}_{L_2,F_5}$ WITH 50 ANCESTORS PER CATEGORY

For the sake of completeness, we performed the same experiments as in Subsection V-B with the same 500 ancestor images (50 ancestor images per ancestor category), but with $\text{EA}^{\text{target,VGG-16}}_{L_2,F_5}$ instead of $\text{EA}^{\text{target,VGG-16}}_{L_2}$. Figure 9 shows the outcome. Of the 4500 attacks, 543 were unsuccessful; hence, the success rate of $\text{EA}^{\text{target,VGG-16}}_{L_2,F_5}$ was 88%, and required between 1250 and 2404 generations on average.

Comparing Figure 4 with Figure 8 and Figure 5 with Figure 9 shows that $\text{EA}^{\text{target,VGG-16}}_{L_2,F_5}$ usually requires more generations than $\text{EA}^{\text{target,VGG-16}}_{L_2}$ to construct adversarial images, which is to be expected since $\text{EA}^{\text{target,VGG-16}}_{L_2,F_5}$ must satisfy not two, but three conditions.

C. ROBUSTNESS OF $\mathcal{D}^{F_5}_{A,T}(\mathcal{A}_A)$ AGAINST VGG-16$\circ F_K$ FOR ALL FILTERS

Using again the images of Figure 13 (Appendix B) obtained as described in Subsection VII-A, the ancestor $\mathcal{A}_a$ and the corresponding adversarial images $\mathcal{D}^{F_5}_{a,t}(\mathcal{A}_a)$ were then tested against all five filters of Subsection VI-A. Figure 10 shows the outcome of this process for the dog ancestor $\mathcal{A}_6$ and the adversarial images $\mathcal{D}^{F_5}_{6,t}(\mathcal{A}_6)$.

These filtered images are given to VGG-16 for classification (see Appendix B, Table 11 for F_5, and Table 12 for F_1, F_2, F_3, and F_4, with $\mathcal{D}^{F_5}_{a,a}(\mathcal{A}_a) = \mathcal{A}_a$ to ease the notations).

Mutatis mutandis, Table 4 is obtained in a similar way as in Table 3. Note that the upper bounds of the indicators are impacted by the fact that three combinations $(c_a(\mathcal{A}_a), c_t)$ were not achieved. Indeed, one has $0 \leq \rho^{F_5}_k(\mathcal{A}_a) \leq 9$ for $a = 1, 3, 8$, and $0 \leq \rho^{F_5}_k(\mathcal{A}_a) \leq 10$ otherwise. One writes $\delta^{F_5}_k(\mathcal{A}_a) = 1$ if the filtered ancestor and all filtered adversarial images are classified back to the ancestor category whenever possible. Consistently, one has $0 \leq \tau^{F_5}_k(\mathcal{A}_a), \widetilde{\tau}^{F_5}_k(\mathcal{A}_a) \leq 8$ for $a = 1, 3, 8$, and $0 \leq \tau^{F_5}_k(\mathcal{A}_a), \widetilde{\tau}^{F_5}_k(\mathcal{A}_a) \leq 9$ otherwise. As a consequence, one has $0 \leq \Sigma^{F_5}_k \leq 97$, $0 \leq \Delta^{F_5}_k, \mathcal{M}^{F_5}_k \leq 10$, and $0 \leq \mathcal{T}^{F_5}_k, \widetilde{\mathcal{T}}^{F_5}_k \leq 87$.

Table 4 clearly shows that the produced images are not only adversarial for F_5, but also for F_3 and F_4 to a large extent for the *target scenario* (88% and 84%, respectively), and for the *untargeted scenario* (89% and 88% respectively).

	plane	car	bird	cat	deer	dog	frog	horse	ship	truck	Row Average
plane		3423 (*17, ‡1, †16)	1038 (*4, ‡0, †4)	1914 (*8, ‡0, †8)	2348 (*12, ‡0, †12)	2438 (*15, ‡0, †15)	1920 (*6, ‡0, †6)	2816 (*15, ‡0, †15)	2190 (*8, ‡1, †8)	3546 (*22, ‡1, †21)	2404
car	1829 (*1, ‡0, †1)		1552 (*1, ‡1, †0)	2168 (*2, ‡2, †0)	1964 (*3, ‡1, †2)	1786 (*2, ‡0, †2)	1898 (*2, ‡2, †0)	1187	2814 (*7, ‡5, †2)	1066 (*2, ‡1, †1)	1807
bird	1505 (*2, ‡0, †2)	3280 (*11, ‡2, †8)		2112 (*10, ‡0, †10)	1821 (*8, ‡0, †8)	1491 (*5, ‡1, †4)	1514 (*7, ‡0, †7)	2144 (*7, ‡0, †7)	2246 (*10, ‡0, †10)	3254 (*15, ‡0, †15)	2152
cat	1406 (*1, ‡0, †1)	3132 (*15, ‡2, †12)	894 (*3, ‡0, †3)		2260 (*12, ‡0, †12)	855 (*4, ‡0, †4)	1842 (*10, ‡0, †10)	1745 (*7, ‡0, †7)	2820 (*13, ‡0, †13)	3876 (*21, ‡0, †21)	2092
deer	1367 (*3, ‡1, †3)	3680 (*18, ‡12, †8)	535 (*1, ‡0, †1)	870 (*2, ‡0, †2)		723	1362 (*5, ‡0, †5)	1189 (*4, ‡0, †4)	1485 (*2, ‡0, †2)	2524 (*8, ‡3, †7)	1526
dog	1842 (*5, ‡0, †5)	2911 (*12, ‡0, †12)	1027 (*4, ‡0, †4)	603 (*2, ‡0, †2)	1856 (*8, ‡0, †8)		2120 (*11, ‡0, †11)	1781 (*7, ‡0, †7)	2419 (*10, ‡0, †10)	3028 (*14, ‡0, †14)	1954
frog	1481	3712 (*16, ‡8, †8)	613	734	895 (*2, ‡0, †2)	904		1961 (*3, ‡1, †2)	1775 (*4, ‡0, †4)	2583 (*10, ‡1, †9)	1629
horse	1419 (*1, ‡0, †1)	2687 (*5, ‡0, †5)	779	1215 (*2, ‡0, †2)	956 (*3, ‡0, †3)	997 (*2, ‡0, †2)	1866 (*7, ‡0, †7)		1783 (*2, ‡0, †2)	2959 (*10, ‡1, †9)	1629
ship	1218 (*2, ‡0, †2)	2724 (*9, ‡0, †9)	1222 (*1, ‡0, †1)	1494 (*2, ‡0, †2)	2649 (*13, ‡0, †13)	1710 (*2, ‡1, †1)	2431 (*12, ‡0, †12)	2056 (*4, ‡2, †2)		2490 (*11, ‡0, †11)	1999
truck	1355 (*2, ‡0, †2)	1180 (*4, ‡0, †4)	978	1199 (*2, ‡1, †1)	1584 (*3, ‡0, †3)	1337 (*1, ‡1, †0)	1302 (*2, ‡0, †2)	1157 (*1, ‡0, †1)	1157 (*1, ‡0, †1)		1250
Column Average	1491	2970	960	1368	1815	1360	1806	1782	2077	2814	

FIGURE 9: 500 attacked images with 50 samples per ancestor class. Rows correspond to source classes, columns correspond to target classes, and cell values correspond to the average number of generations needed by $\text{EA}_{L_2,F_5}^{\text{target,VGG-16}}$ to terminate.

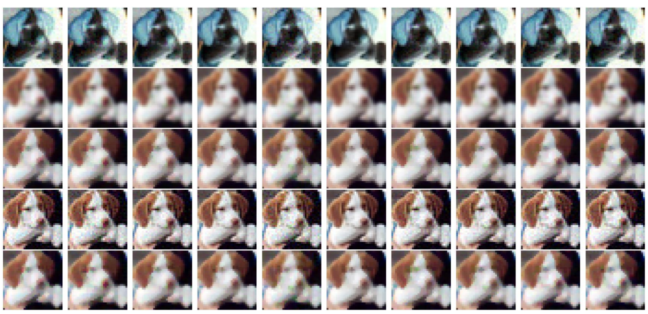


FIGURE 10: Impact of filters on the ancestor $\mathcal{A}_6$ and adversarial images $\mathcal{D}_{6,t}^{F_5}(\mathcal{A}_6)$. The k^{th} row represents $F(\mathcal{D}_{6,t}^{F_5}(\mathcal{A}_6))$ in t^{th} position (with $\mathcal{D}_{6,6}^{F_5}(\mathcal{A}_6) = \mathcal{A}_6$), where $F = F_k$ for $1 \leq k \leq 5$.

Additionally, 56% of these images were efficient against F_1 for the *untargeted scenario*, while this percentage dropped to 23% with F_2.

This study shows that the $\text{EA}_{L_2,F_5}^{\text{target,VGG-16}}$ attack, designed to be robust against F_5, is also robust to some significant extent against all individual filters considered for the *untargeted* scenario, and the Gaussian filter (F_2) is the most efficient at removing the adversarial character of the constructed images.

VIII. CONCLUSION

This study, which substantially complements our previous works [1], [22]–[24], [32], successfully addresses the four issues raised in the introduction. First, we proved that the conceptual originality of our generic evolutionary algorithm leads to a competitive advantage in terms of performance compared to the classical EA approach. Then, an extensive experimental study showed the intrinsic efficiency of our algorithm, $\text{EA}_{L_2}^{\text{target,VGG-16}}$, at constructing adversarial images for the *target scenario* performed against VGG-16 with

TABLE 4: Indicator values assessing the robustness of adversarial images $\mathcal{D}_{a,t}^{F_5}(\mathcal{A}_a)$ against filters. For each ancestor $\mathcal{A}_a$, computation of $(\rho_k^{F_5}(\mathcal{A}_a), \delta_k^{F_5}(\mathcal{A}_a), \mu_k^{F_5}(\mathcal{A}_a))$ in the 1st row, and of $(\tau_k^{F_5}(\mathcal{A}_a), \widetilde{\tau}_k^{F_5}(\mathcal{A}_a))$ in the 2nd row. The last two rows give the sums $\sum_a$ of these quantities for all possible a.

$\mathcal{A}_a$ \ k	1	2	3	4	5
$\mathcal{A}_1$	(9,1,0) (0,0)	(0,0,1) (2,6)	(1,0,0) (8,8)	(0,0,1) (8,7)	(1,0,0) (8,8)
$\mathcal{A}_2$	(1,0,0) (1,9)	(2,0,0) (4,8)	(1,0,0) (9,9)	(4,0,0) (6,6)	(1,0,0) (9,9)
$\mathcal{A}_3$	(6,0,0) (2,3)	(9,1,0) (0,0)	(6,0,0) (3,3)	(1,0,0) (8,8)	(1,0,0) (8,8)
$\mathcal{A}_4$	(5,0,0) (2,5)	(9,0,0) (1,1)	(1,0,0) (8,9)	(1,0,0) (5,9)	(1,0,0) (9,9)
$\mathcal{A}_5$	(0,0,1) (3,8)	(10,1,0) (0,0)	(1,0,0) (8,9)	(1,0,0) (9,9)	(1,0,0) (9,9)
$\mathcal{A}_6$	(0,0,1) (3,6)	(0,0,1) (1,0)	(0,0,1) (7,6)	(1,0,0) (9,9)	(1,0,0) (9,9)
$\mathcal{A}_7$	(6,0,0) (2,4)	(8,0,0) (2,2)	(1,0,0) (9,9)	(5,0,0) (5,5)	(1,0,0) (9,9)
$\mathcal{A}_8$	(0,0,1) (1,8)	(6,0,0) (2,3)	(1,0,0) (8,8)	(1,0,0) (7,8)	(1,0,0) (8,8)
$\mathcal{A}_9$	(6,0,0) (1,4)	(1,0,1) (2,2)	(2,0,0) (8,8)	(3,0,0) (7,7)	(1,0,0) (9,9)
$\mathcal{A}_{10}$	(0,0,1) (1,2)	(0,0,1) (2,1)	(1,0,0) (9,9)	(1,0,0) (9,9)	(1,0,0) (9,9)
$(\Sigma_k^{F_5}, \Delta_k^{F_5}, \mathcal{M}_k^{F_5})$	(33,1,4)	(45,2,4)	(15,0,1)	(18,0,1)	(10,0,0)
$(\mathcal{T}_k^{F_5}, \widetilde{\mathcal{T}}_k^{F_5})$	(16,49)	(16,23)	(77,78)	(73,77)	(87,87)

images from CIFAR-10. We then challenged the adversarial images obtained against a series of filters, and finally designed a variant $\text{EA}_{L_2,F}^{\text{target,VGG-16}}$ of the EA, specifically designed to fool VGG-16 and VGG-16 composed with a filter F, and demonstrated the efficiency of the produced adversarial images not only against the specific chosen filter, but also against other filters.

The results of this paper lead to a series of additional studies. First, from a *pure* EA point of view, we intend to look for methods to accelerate our algorithm, including early warnings that indicate a high probability of unsuccess of a given run. In this line of thought and more specifically for the construction of adversarial images, other efficiency improvement methods will be studied, such as the restriction of the zones on which the EA should focus its noise creation, or the search for optimized paths between c_a and c_t for a given ancestor via auxiliary categories. Second, since the small 32×32 images in this study are naturally *grainy*, we intend to apply our attack to larger images, not only those of ImageNet [3], but foremost high resolution images arising from different horizons (e.g., satellite, medical, or artistic images), which may lead to combinations $\mathcal{C} \circ \mathcal{F}$ for functions $\mathcal{F}$ that are no longer filters. Moreover, we plan to replace the $L2$ distance with $L - infinity$ in order to check the attack performance against state-of-the-art defense methods. Finally, our EA-based attack can potentially be extended to other domains (natural language processing, speech recognition, etc.) beyond the computer vision applications mentioned in the first paragraph of the Introduction.

.

APPENDIX A

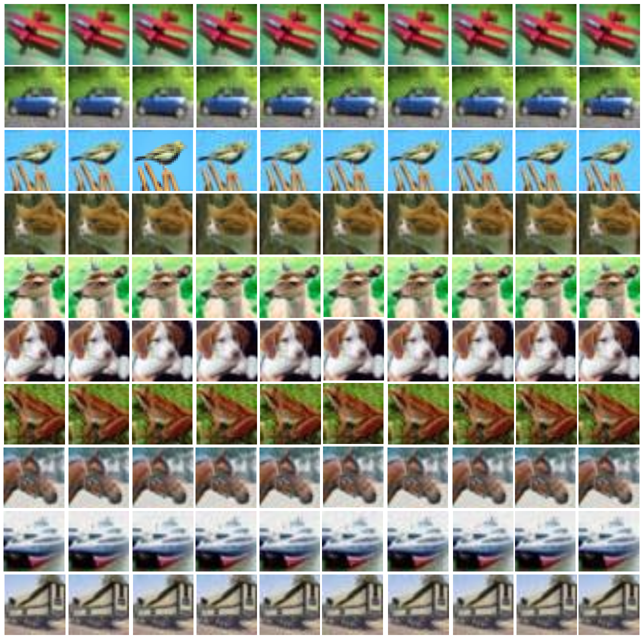


FIGURE 11: For $1 \leq a \leq 10$, the image on the diagonal of the a^{th} row is the ancestor $\mathcal{A}_a$ (recovered from Table 1) classified by VGG-16 as belonging to the category c_a, and the picture in the t^{th} column, with $t \neq a$, is the adversarial picture $\mathcal{D}_{a,t}(\mathcal{A}_a) = \text{EA}_{L_2}^{\text{target,VGG-16}}(\mathcal{A}_a, c_t)$ classified by VGG-16 as belonging to c_t, obtained after the first of the 10 independent runs.

	plane	car	bird	cat	deer	dog	frog	horse	ship	truck	Row Average
plane ($\mathcal{A}_1$)		6.95	29.8	239.64	76.77	66.76	36.68	87.57	16.24	121.35	75.75
car ($\mathcal{A}_2$)	120.97		49.84	135.63	83.33	167.08	58.77	73.18	45.88	78.52	90.36
bird ($\mathcal{A}_3$)	117.02	72		197.42	100.12	671.95	60.51	293.8	94.43	118.03	191.70
cat ($\mathcal{A}_4$)	53.38	37.04	27.14		28.55	23.56	12.33	44.51	61.44	68.2	39.57
deer ($\mathcal{A}_5$)	198.95	402.95	25.35	33.99	34.57	17.64	61.12	52.2	399.42	470.01	169.62
dog ($\mathcal{A}_6$)	64.45	64.7	33.65	9.51	68.77	66.78	29.1	20.57	71.36	40.18	46.91
frog ($\mathcal{A}_7$)	50.77	38.17	21.7	35.94	52.91	39.47	39.02	61.47	68.14	50.37	45.80
horse ($\mathcal{A}_8$)	33.4	16.26	22.1	15.04	16.39	22.36	27.35		22.8	16.73	21.38
ship ($\mathcal{A}_9$)	23.39	13.39	94.87	31.79	183.15	97.4	53.05	132.95	139.16	31.24	80.04
truck ($\mathcal{A}_{10}$)	9.4	42.86	24.76	91	28.18	55.68	59.34	17.5	18.89	18.47	36.61
Column Average	75	77	37	88	67	123	44	87	94	101	

FIGURE 12: $\text{EA}_{L_2}^{\text{target,VGG-16}}$'s performance on all possible ancestor/target combinations with one ancestor per category. The rows give the ancestor category c_a (and the specific ancestor $\mathcal{A}_a$ in c_a), the columns indicate the target class c_t, and the cell values indicate the average number of seconds required by $\text{EA}_{L_2}^{\text{target,VGG-16}}$ to terminate successfully, computed on 10 independent runs, with only the successful runs being considered.

TABLE 5: For $\mathcal{C}$ = VGG-16, each of the cell in $(a,t)^{\text{th}}$-position contains a pair (maximum label value, corresponding class) given by $\mathcal{C}$ for $\mathcal{D}_{a,t}(\mathcal{A}_a)$ (with $\mathcal{D}_{a,a}(\mathcal{A}_a) = \mathcal{A}_a$).

	plane c_1	car c_2	bird c_3	cat c_4	deer c_5	dog c_6	frog c_7	horse c_8	ship c_9	truck c_{10}
plane($\mathcal{A}_1$)	0.6900 plane	0.9506 car	0.9501 bird	0.9500 cat	0.9501 deer	0.9500 dog	0.9502 frog	0.9501 horse	0.9537 ship	0.9531 truck
car($\mathcal{A}_2$)	0.9519 plane	0.9999 car	0.9546 bird	0.9515 cat	0.9534 deer	0.9509 dog	0.9508 frog	0.9606 horse	0.9509 ship	0.9502 truck
bird($\mathcal{A}_3$)	0.9502 plane	0.9505 car	0.9999 bird	0.9501 cat	0.9511 deer	0.9509 dog	0.9517 frog	0.9523 horse	0.9506 ship	0.9510 truck
cat($\mathcal{A}_4$)	0.9514 plane	0.9507 car	0.9512 bird	0.9998 cat	0.9510 deer	0.9519 dog	0.9543 frog	0.9503 horse	0.9514 ship	0.9552 truck
deer($\mathcal{A}_5$)	0.9524 plane	0.9501 car	0.9507 bird	0.9514 cat	0.9999 deer	0.9545 dog	0.9520 frog	0.9501 horse	0.9560 ship	0.9510 truck
dog($\mathcal{A}_6$)	0.9516 plane	0.9502 car	0.9529 bird	0.9518 cat	0.9501 deer	0.9996 dog	0.9502 frog	0.9512 horse	0.9508 ship	0.9518 truck
frog($\mathcal{A}_7$)	0.9519 plane	0.9528 car	0.9501 bird	0.9530 cat	0.9521 deer	0.9527 dog	0.9999 frog	0.9529 horse	0.9521 ship	0.9515 truck
horse($\mathcal{A}_8$)	0.9502 plane	0.9523 car	0.9503 bird	0.9568 cat	0.9521 deer	0.9510 dog	0.9521 frog	0.9998 horse	0.9587 ship	0.9514 truck
ship($\mathcal{A}_9$)	0.9504 plane	0.9543 car	0.9581 bird	0.9506 cat	0.9500 deer	0.9516 dog	0.9517 frog	0.9505 horse	0.9996 ship	0.9504 truck
truck($\mathcal{A}_{10}$)	0.9525 plane	0.9532 car	0.9518 bird	0.9517 cat	0.9557 deer	0.9511 dog	0.9516 frog	0.9507 horse	0.9517 ship	0.9984 truck

TABLE 6: For $\mathcal{C}$ = VGG-16, the cell in $(a, t)^{\text{th}}$-position gives (top part) the c_a-label value and the c_t-label value, and (bottom part) the maximum label value and corresponding class of $\mathcal{C} \circ F_1$ for $\mathcal{D}_{a,t}(\mathcal{A}_a)$ (with $\mathcal{D}_{a,a}(\mathcal{A}_a) = \mathcal{A}_a$).

	plane c_1	car c_2	bird c_3	cat c_4	deer c_5	dog c_6	frog c_7	horse c_8	ship c_9	truck c_{10}
plane ($\mathcal{A}_1$)	0.9923	0.9870	0.9830	0.8888	0.5932	0.9845	0.9857	0.9436	0.9007	0.9806
	0.9923	1.56e-03	5.16e-04	4.17e-04	1.71e-04	9.59e-04	1.92e-03	8.26e-05	9.18e-02	2.52e-05
	0.9923	0.9870	0.9830	0.8888	0.5932	0.9845	0.9857	0.9436	0.9007	0.9806
	plane	plane	plane	plane	plane	plane	plane	plane	plane	plane
car ($\mathcal{A}_2$)	2.77e-04	0.7608	1.71e-04	5.46e-05	1.09e-04	7.36e-04	6.81e-04	1.63e-04	2.60e-05	3.29e-03
	0.1501	0.7608	1.44e-04	1.82e-03	4.02e-05	2.29e-04	1.00e-04	1.08e-05	0.9993	8.08e-03
	0.8491	0.7608	0.9958	0.9962	0.9965	0.9837	0.9969	0.9864	0.9993	0.9877
	ship	car	ship	ship	ship	ship	ship	ship	ship	ship
bird ($\mathcal{A}_3$)	0.2966	0.7862	0.9996	0.2070	0.8838	0.9665	0.8639	0.4902	0.4071	0.5798
	5.15e-02	0.1254	0.9996	8.18e-04	1.38e-04	2.49e-04	0.1120	5.17e-04	0.5643	1.63e-03
	0.4964	0.7862	0.9996	0.6009	0.8838	0.9665	0.8639	0.4902	0.5643	0.5798
	ship	bird	bird	ship	bird	bird	bird	bird	ship	bird
cat ($\mathcal{A}_4$)	0.1906	4.82e-02	4.13e-02	0.9176	0.6035	0.1058	0.5140	0.1909	1.27e-02	2.49e-02
	0.1682	2.36e-02	2.79e-04	0.9176	5.37e-03	1.72e-03	4.64e-02	6.87e-02	0.9800	1.78e-02
	0.5977	0.8971	0.9162	0.9176	0.6035	0.8079	0.5140	0.4871	0.9800	0.9054
	ship	ship	ship	cat	cat	ship	cat	frog	ship	ship
deer ($\mathcal{A}_5$)	3.90e-05	7.41e-03	9.00e-02	5.71e-04	0.3245	0.7423	2.16e-03	0.1149	7.83e-03	8.01e-04
	0.9985	4.49e-02	2.39e-03	0.9982	0.3245	1.23e-02	3.97e-02	7.53e-04	0.4939	3.41e-02
	0.9985	0.8634	0.8982	0.9982	0.5838	0.7423	0.6883	0.7741	0.4939	0.9616
	plane	plane	cat	cat	plane	deer	cat	cat	ship	cat
dog ($\mathcal{A}_6$)	3.08e-04	9.21e-03	5.23e-03	1.69e-03	1.32e-03	0.0014	7.82e-02	1.03e-02	4.41e-03	5.18e-03
	8.74e-04	1.28e-03	1.18e-04	0.9977	1.80e-04	0.0014	0.4415	3.46e-05	1.19e-02	0.5093
	0.9925	0.6375	0.9727	0.9977	0.9380	0.9983	0.4415	0.9794	0.8320	0.5093
	truck	cat	cat	cat	truck	cat	frog	cat	truck	truck
frog ($\mathcal{A}_7$)	0.5602	0.9198	9.37e-02	0.4898	0.2230	0.2092	0.9140	0.2013	9.97e-04	0.4658
	0.4272	5.41e-03	4.93e-04	0.4423	1.90e-03	5.25e-02	0.9140	1.20e-03	0.9941	1.37e-02
	0.5602	0.9198	0.7740	0.4898	0.7097	0.3961	0.9140	0.4221	0.9941	0.4658
	frog	frog	plane	frog	cat	cat	frog	cat	ship	frog
horse ($\mathcal{A}_8$)	3.06e-05	3.20e-04	1.83e-04	1.89e-04	1.46e-04	5.03e-05	1.09e-05	0.0004	2.27e-05	1.27e-04
	0.9316	2.43e-03	2.29e-02	9.12e-03	4.38e-04	0.8593	7.05e-03	0.0004	0.9470	6.98e-03
	0.9316	0.4645	0.5039	0.3962	0.6209	0.8593	0.8860	0.7479	0.9470	0.8123
	plane	plane	dog	plane	plane	dog	plane	dog	ship	plane
ship ($\mathcal{A}_9$)	0.9136	0.9445	0.1834	0.9034	7.89e-03	1.71e-03	0.9801	3.43e-02	0.9865	0.9242
	7.62e-02	4.45e-04	1.30e-03	8.70e-02	0.6532	0.9306	1.79e-03	1.04e-02	0.9865	3.55e-04
	0.9136	0.9445	0.7320	0.9034	0.6532	0.9306	0.9801	0.9311	0.9865	0.9242
	ship	ship	plane	ship	deer	dog	ship	cat	ship	ship
truck ($\mathcal{A}_{10}$)	2.35e-05	2.68e-04	3.16e-05	1.43e-04	6.35e-05	4.79e-05	6.68e-04	3.60e-04	1.38e-04	0.0001
	0.9994	4.12e-05	4.41e-04	5.57e-05	1.43e-04	4.18e-05	4.62e-05	1.65e-03	1.30e-02	0.0001
	0.9994	0.9971	0.9970	0.9941	0.9946	0.9941	0.8366	0.9947	0.9865	0.9973
	plane	plane	plane	plane	plane	plane	ship	plane	plane	plane

TABLE 7: For $\mathcal{C}$ = VGG-16, the cell in $(a,t)^{\text{th}}$-position gives (top part) the c_a-label value and the c_t-label value, and (bottom part) the maximum label value and corresponding class of $\mathcal{C} \circ F_2$ for $\mathcal{D}_{a,t}(\mathcal{A}_a)$ (with $\mathcal{D}_{a,a}(\mathcal{A}_a) = \mathcal{A}_a$).

	plane c_1	car c_2	bird c_3	cat c_4	deer c_5	dog c_6	frog c_7	horse c_8	ship c_9	truck c_{10}
plane ($\mathcal{A}_1$)	0.2591	0.2052	0.1978	0.1256	0.2017	0.1885	0.1907	0.1415	9.10e-02	0.2188
	0.2591	0.5786	0.1627	4.95e-03	5.82e-04	1.89e-02	3.08e-02	1.51e-04	0.7307	3.94e-05
	0.4463	0.5786	0.3455	0.5941	0.6175	0.4134	0.4057	0.7010	0.7307	0.5850
	car	car	ship	ship	ship	ship	car	ship	ship	ship
car ($\mathcal{A}_2$)	5.29e-03	0.9988	9.92e-02	5.78e-03	0.4091	0.1245	5.40e-02	1.37e-02	3.80e-02	0.9989
	3.23e-04	0.9988	0.7795	4.33e-02	7.96e-04	9.59e-04	0.5196	6.14e-05	3.73e-02	1.76e-04
	0.8311	0.9988	0.7795	0.7865	0.4091	0.7438	0.5196	0.9224	0.8378	0.9989
	bird	car	bird	bird	car	bird	frog	bird	bird	car
bird ($\mathcal{A}_3$)	0.9996	0.9998	0.9998	0.9997	0.9997	0.9997	0.9998	0.9996	0.9997	0.9998
	1.98e-04	5.04e-06	0.9998	1.13e-04	5.52e-06	4.45e-05	1.72e-05	2.11e-05	3.80e-05	5.91e-06
	0.9996	0.9998	0.9998	0.9997	0.9997	0.9997	0.9998	0.9996	0.9997	0.9998
	bird	bird	bird	bird	bird	bird	bird	bird	bird	bird
cat ($\mathcal{A}_4$)	0.9876	0.9959	0.9743	0.9992	0.9955	0.9691	0.9983	0.9723	0.9968	0.9917
	4.65e-06	7.52e-07	8.52e-04	0.9992	5.96e-04	3.02e-02	2.37e-04	4.62e-05	8.10e-06	9.37e-07
	0.9876	0.9959	0.9743	0.9992	0.9955	0.9691	0.9983	0.9723	0.9968	0.9917
	cat	cat	cat	cat	cat	cat	cat	cat	cat	cat
deer ($\mathcal{A}_5$)	0.9997	0.9985	0.9989	0.9988	0.9998	0.9983	0.9996	0.9992	0.9985	0.9997
	8.28e-06	1.35e-06	9.30e-04	7.09e-04	0.9998	1.49e-03	1.73e-05	1.43e-04	5.85e-06	1.02e-06
	0.9997	0.9985	0.9989	0.9988	0.9998	0.9983	0.9996	0.9992	0.9985	0.9997
	deer	deer	deer	deer	deer	deer	deer	deer	deer	deer
dog ($\mathcal{A}_6$)	2.17e-04	3.40e-03	2.52e-03	1.48e-04	3.64e-03	3.14e-04	3.71e-04	8.51e-04	1.95e-04	2.33e-04
	1.34e-05	2.26e-06	5.67e-05	0.9998	1.88e-05	3.14e-04	8.29e-06	1.63e-05	2.77e-06	2.94e-06
	0.9997	0.9965	0.9973	0.9998	0.9962	0.9996	0.9995	0.9990	0.9997	0.9997
	cat	cat	cat	cat	cat	cat	cat	cat	cat	cat
frog ($\mathcal{A}_7$)	0.9994	0.9997	0.9995	0.9977	0.9980	0.9941	0.9998	0.9982	0.9995	0.9996
	1.90e-05	8.46e-06	2.25e-04	1.21e-03	8.21e-05	4.55e-03	0.9998	4.94e-05	6.97e-05	7.29e-06
	0.9994	0.9997	0.9995	0.9977	0.9980	0.9941	0.9998	0.9982	0.9995	0.9996
	frog	frog	frog	frog	frog	frog	frog	frog	frog	frog
horse ($\mathcal{A}_8$)	0.7487	0.9692	0.8900	0.9062	0.9967	0.9568	0.9164	0.9997	0.9792	0.9758
	4.02e-04	7.37e-05	9.74e-02	8.47e-02	1.94e-03	3.46e-03	7.37e-04	0.9997	4.28e-04	2.19e-05
	0.7487	0.9692	0.8900	0.9062	0.9967	0.9568	0.9164	0.9997	0.9792	0.9758
	horse	horse	horse	horse	horse	horse	horse	horse	horse	horse
ship ($\mathcal{A}_9$)	1.73e-03	4.33e-02	5.67e-02	1.85e-02	0.8242	0.1091	8.23e-02	7.87e-03	0.3924	1.55e-03
	0.9894	0.7334	1.25e-03	3.34e-04	2.25e-05	8.34e-04	1.54e-04	7.16e-05	0.3924	4.34e-03
	0.9894	0.7334	0.8965	0.6402	0.8242	0.8392	0.4956	0.8441	0.4525	0.9214
	plane	car	plane	plane	ship	plane	plane	plane	plane	plane
truck ($\mathcal{A}_{10}$)	1.61e-03	6.03e-04	1.57e-03	1.05e-04	7.27e-04	4.96e-03	9.52e-03	3.86e-03	1.02e-03	5.62e-03
	0.9955	2.88e-04	2.79e-03	1.88e-04	3.04e-02	4.73e-04	2.58e-04	0.1820	9.07e-05	5.62e-03
	0.9955	0.9873	0.9920	0.9931	0.9670	0.9184	0.9876	0.8099	0.9974	0.9919
	plane	plane	plane	plane	plane	plane	plane	plane	plane	plane

TABLE 8: For $\mathcal{C}$ = VGG-16, the cell in $(a,t)^{\text{th}}$-position gives (top part) the c_a-label value and the c_t-label value, and (bottom part) the maximum label value and corresponding class of $\mathcal{C} \circ F_3$ for $\mathcal{D}_{a,t}(\mathcal{A}_a)$ (with $\mathcal{D}_{a,a}(\mathcal{A}_a) = \mathcal{A}_a$).

	plane	car	bird	cat	deer	dog	frog	horse	ship	truck
	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}
plane ($\mathcal{A}_1$)	0.7298	0.6388	0.3414	0.1388	0.3576	0.2870	0.3931	0.2510	0.1163	0.3362
	0.7298	6.08e-02	0.1755	8.80e-04	3.35e-04	3.20e-02	1.14e-02	2.19e-04	0.8616	2.08e-05
	0.7298	0.6388	0.4205	0.8034	0.5857	0.5863	0.5051	0.6781	0.8616	0.6479
	plane	plane	ship	ship	ship	ship	ship	ship	ship	ship
car ($\mathcal{A}_2$)	0.3643	0.9997	0.9734	0.9666	0.9858	0.8656	0.9075	0.9986	0.9607	0.9997
	1.23e-03	0.9997	1.79e-02	1.44e-03	7.33e-05	5.59e-04	5.22e-02	8.15e-06	1.97e-03	5.86e-05
	0.5191	0.9997	0.9734	0.9666	0.9858	0.8656	0.9075	0.9986	0.9607	0.9997
	bird	car	car	car	car	car	car	car	car	car
bird ($\mathcal{A}_3$)	0.9998	0.9998	0.9999	0.9998	0.9998	0.9998	0.9999	0.9998	0.9998	0.9998
	7.36e-05	3.88e-06	0.9999	5.44e-05	4.91e-06	2.73e-05	1.26e-05	1.22e-05	2.02e-05	4.76e-06
	0.9998	0.9998	0.9999	0.9998	0.9998	0.9998	0.9999	0.9998	0.9998	0.9998
	bird	bird	bird	bird	bird	bird	bird	bird	bird	bird
cat ($\mathcal{A}_4$)	0.9971	0.9977	0.9873	0.9994	0.9980	0.9969	0.9986	0.9916	0.9975	0.9963
	2.73e-06	1.95e-06	9.16e-03	0.9994	1.09e-04	2.56e-03	2.80e-04	1.33e-04	1.16e-05	1.81e-06
	0.9971	0.9977	0.9873	0.9994	0.9980	0.9969	0.9986	0.9916	0.9975	0.9963
	cat	cat	cat	cat	cat	cat	cat	cat	cat	cat
deer ($\mathcal{A}_5$)	0.9994	0.9995	0.9998	0.9996	0.9998	0.9991	0.9999	0.9995	0.9986	0.9997
	1.02e-05	7.05e-07	1.73e-05	7.95e-05	0.9998	8.14e-04	1.17e-05	2.09e-04	4.97e-06	1.04e-06
	0.9994	0.9995	0.9998	0.9996	0.9998	0.9991	0.9999	0.9995	0.9986	0.9997
	deer	deer	deer	deer	deer	deer	deer	deer	deer	deer
dog ($\mathcal{A}_6$)	0.2027	0.8235	0.4543	1.37e-02	0.1518	0.2668	1.11e-02	0.1644	5.96e-02	3.49e-02
	1.70e-05	3.99e-06	1.20e-03	0.9861	2.06e-05	0.2668	2.52e-05	4.93e-05	4.62e-06	4.38e-06
	0.7969	0.8235	0.5443	0.9861	0.8479	0.7329	0.9886	0.8352	0.9401	0.9649
	cat	dog	cat	cat	cat	cat	cat	cat	cat	cat
frog ($\mathcal{A}_7$)	0.9997	0.9998	0.9998	0.9993	0.9997	0.9994	0.9998	0.9994	0.9997	0.9997
	1.74e-05	1.30e-05	4.40e-05	4.63e-04	1.08e-05	2.41e-04	0.9998	1.72e-05	5.84e-05	6.97e-06
	0.9997	0.9998	0.9998	0.9993	0.9997	0.9994	0.9998	0.9994	0.9997	0.9997
	frog	frog	frog	frog	frog	frog	frog	frog	frog	frog
horse ($\mathcal{A}_8$)	0.9965	0.9985	0.9958	0.9988	0.9997	0.9992	0.9891	0.9998	0.9974	0.9994
	6.08e-05	1.93e-05	3.63e-03	1.24e-04	1.37e-04	5.16e-05	2.94e-05	0.9998	2.45e-05	8.11e-06
	0.9965	0.9985	0.9958	0.9988	0.9997	0.9992	0.9891	0.9998	0.9974	0.9994
	horse	horse	horse	horse	horse	horse	horse	horse	horse	horse
ship ($\mathcal{A}_9$)	0.5341	0.4759	0.8869	0.7291	0.9987	0.7917	0.4933	0.6350	0.9942	0.2402
	0.3343	0.4978	8.15e-04	3.15e-04	3.33e-06	2.92e-04	1.03e-04	5.64e-05	0.9942	1.07e-02
	0.5341	0.4978	0.8869	0.7291	0.9987	0.7917	0.4933	0.6350	0.9942	0.6106
	ship	car	ship	ship	ship	ship	ship	ship	ship	car
truck ($\mathcal{A}_{10}$)	0.9685	0.9701	0.5662	0.6816	0.7751	0.7993	0.8764	0.8361	0.6649	0.9924
	2.94e-02	2.13e-02	4.58e-03	7.19e-04	0.1100	1.29e-04	2.88e-04	8.23e-04	7.35e-03	0.9924
	0.9685	0.9701	0.5662	0.6816	0.7751	0.7993	0.8764	0.8361	0.6649	0.9924
	truck	truck	truck	truck	truck	truck	truck	truck	truck	truck

TABLE 9: For $\mathcal{C}$ = VGG-16, the cell in $(a,t)^{\text{th}}$-position gives (top part) the c_a-label value and the c_t-label value, and (bottom part) the maximum label value and corresponding class of $\mathcal{C} \circ F_4$ for $\mathcal{D}_{a,t}(\mathcal{A}_a)$ (with $\mathcal{D}_{a,a}(\mathcal{A}_a) = \mathcal{A}_a$).

	plane c_1	car c_2	bird c_3	cat c_4	deer c_5	dog c_6	frog c_7	horse c_8	ship c_9	truck c_{10}
plane($\mathcal{A}_1$)	0.4425	1.10e-02	1.62e-02	5.08e-03	6.28e-03	1.16e-02	1.27e-02	2.14e-03	4.610e-02	4.84e-03
	0.4425	0.9871	0.9689	0.9843	0.9813	0.9566	0.9610	0.9815	0.9146	0.9792
	0.5497	0.9871	0.9689	0.9843	0.9813	0.9566	0.9610	0.9815	0.9146	0.9792
	car	car	bird	cat	deer	dog	frog	horse	ship	truck
car($\mathcal{A}_2$)	9.97e-03	0.9999	0.8717	0.1439	0.3505	9.05e-02	0.2418	6.16e-02	0.7124	3.99e-02
	0.9879	0.9999	0.1196	0.8162	0.6083	0.8912	0.7558	0.9287	0.2840	0.9597
	0.9879	0.9999	0.8717	0.8162	0.6083	0.8912	0.7558	0.9287	0.7124	0.9597
	plane	car	car	cat	deer	dog	frog	horse	car	truck
bird($\mathcal{A}_3$)	1.63e-03	1.75e-03	0.9999	5.26e-03	8.46e-03	1.27e-02	4.31e-03	8.68e-03	1.49e-03	3.86e-03
	0.9710	0.9903	0.9999	0.9268	0.9705	0.9505	0.9952	0.9505	0.9916	0.9606
	0.9710	0.9903	0.9999	0.9268	0.9705	0.9505	0.9952	0.9505	0.9916	0.9606
	plane	car	bird	cat	deer	dog	frog	horse	ship	truck
cat($\mathcal{A}_4$)	8.86e-03	4.34e-04	3.31e-02	0.9998	6.74e-03	4.28e-02	2.03e-03	7.95e-02	3.50e-03	5.92e-04
	0.8439	0.9948	0.7611	0.9998	0.9860	7.94e-02	0.9979	0.6060	0.9079	0.9833
	0.8439	0.9948	0.7611	0.9998	0.9860	0.8764	0.9979	0.6060	0.9079	0.9833
	plane	car	bird	cat	deer	frog	frog	horse	ship	truck
deer($\mathcal{A}_5$)	1.65e-04	9.22e-05	3.67e-02	2.14e-03	0.9999	2.27e-02	3.95e-04	1.00e-02	1.27e-03	1.16e-03
	0.9932	0.9970	0.9630	0.9902	0.9999	0.9771	0.9987	0.9852	0.9957	0.9951
	0.9932	0.9970	0.9630	0.9902	0.9999	0.9771	0.9987	0.9852	0.9957	0.9951
	plane	car	bird	cat	deer	dog	frog	horse	ship	truck
dog($\mathcal{A}_6$)	3.01e-03	6.08e-05	3.49e-03	7.11e-02	8.47e-04	0.9998	6.87e-04	2.71e-03	4.09e-04	2.47e-05
	0.9524	0.9985	0.9830	0.9286	0.9943	0.9998	0.9960	0.9960	0.9974	0.9994
	0.9524	0.9985	0.9830	0.9286	0.9943	0.9998	0.9960	0.9960	0.9974	0.9994
	plane	car	bird	cat	deer	dog	frog	horse	ship	truck
frog($\mathcal{A}_7$)	7.74e-02	1.94e-02	9.23e-02	0.2083	0.1896	0.4448	0.9999	0.5326	8.68e-02	4.41e-02
	0.9017	0.9796	0.9075	0.7900	0.8091	0.5505	0.9999	0.4461	0.9092	0.9519
	0.9017	0.9796	0.9075	0.7900	0.8091	0.5505	0.9999	0.5326	0.9092	0.9519
	plane	car	bird	cat	deer	dog	frog	frog	ship	truck
horse($\mathcal{A}_8$)	5.42e-03	5.40e-03	5.30e-03	1.65e-02	8.67e-03	1.40e-02	1.94e-04	0.9998	1.63e-02	6.68e-03
	0.9515	0.9768	0.8715	0.8458	0.9852	0.9342	0.9958	0.9998	0.9648	0.9316
	0.9515	0.9768	0.8715	0.8458	0.9852	0.9342	0.9958	0.9998	0.9648	0.9316
	plane	car	bird	cat	deer	dog	frog	horse	ship	truck
ship($\mathcal{A}_9$)	0.2174	0.1769	3.05e-02	2.13e-02	6.81e-03	0.2438	5.26e-03	3.09e-02	0.9997	6.52e-02
	0.6631	0.8214	0.9155	0.9712	0.8909	0.6297	0.9929	0.9414	0.9997	0.9095
	0.6631	0.8214	0.9155	0.9712	0.8909	0.6297	0.9929	0.9414	0.9997	0.9095
	plane	car	bird	cat	deer	dog	frog	horse	ship	truck
truck($\mathcal{A}_{10}$)	0.1588	1.10e-02	2.70e-02	4.66e-03	0.2818	1.23e-02	6.82e-03	0.1163	6.94e-02	0.9993
	0.8403	0.9869	0.9666	0.9878	0.6789	0.9517	0.9914	0.7095	0.9270	0.9993
	0.8403	0.9869	0.9666	0.9878	0.6789	0.9517	0.9914	0.7095	0.9270	0.9993
	plane	car	bird	cat	deer	dog	frog	horse	ship	truck

TABLE 10: For $\mathcal{C}$ = VGG-16 and $F_5 = F_3 \circ F_4$, the cell in $(a,t)^{\text{th}}$-position gives (top part) the c_a-label value and the c_t-label value, and (bottom part) the maximum label value and corresponding class of $\mathcal{C} \circ F_5$ for $\mathcal{D}_{a,t}(\mathcal{A}_a)$ (with $\mathcal{D}_{a,a}(\mathcal{A}_a) = \mathcal{A}_a$).

	plane c_1	car c_2	bird c_3	cat c_4	deer c_5	dog c_6	frog c_7	horse c_8	ship c_9	truck c_{10}
plane($\mathcal{A}_1$)	0.8817	0.8366	0.5261	0.2653	0.5060	0.5797	0.5169	0.3472	0.1666	0.6224
	0.8817	4.33e-02	0.1688	9.90e-04	4.08e-04	3.43e-02	1.12e-02	3.96e-04	0.8131	3.38e-05
	0.8817	0.8366	0.5261	0.6487	0.5060	0.5797	0.5169	0.5715	0.8131	0.6224
	plane	plane	plane	ship	plane	plane	plane	ship	ship	plane
car($\mathcal{A}_2$)	0.9637	0.9998	0.9907	0.9971	0.9980	0.9907	0.9935	0.9994	0.9961	0.9997
	2.85e-04	0.9998	7.17e-03	3.92e-04	2.70e-05	1.01e-04	3.75e-03	7.29e-06	3.66e-04	6.04e-05
	0.9637	0.9998	0.9907	0.9971	0.9980	0.9907	0.9935	0.9994	0.9961	0.9997
	car	car	car	car	car	car	car	car	car	car
bird($\mathcal{A}_3$)	0.9998	0.9998	0.9999	0.9997	0.9998	0.9998	0.9999	0.9998	0.9998	0.9998
	8.86e-05	3.74e-06	0.9999	6.25e-05	6.93e-06	3.01e-05	1.42e-05	1.25e-05	2.32e-05	4.46e-06
	0.9998	0.9998	0.9999	0.9997	0.9998	0.9998	0.9999	0.9998	0.9998	0.9998
	bird	bird	bird	bird	bird	bird	bird	bird	bird	bird
cat($\mathcal{A}_4$)	0.9990	0.9984	0.9853	0.9997	0.9987	0.9982	0.9990	0.9833	0.9983	0.9978
	3.46e-06	2.06e-06	1.13e-02	0.9997	7.19e-05	9.95e-04	2.55e-04	2.29e-04	1.57e-05	2.76e-06
	0.9990	0.9984	0.9853	0.9997	0.9987	0.9982	0.9990	0.9833	0.9983	0.9978
	cat	cat	cat	cat	cat	cat	cat	cat	cat	cat
deer($\mathcal{A}_5$)	0.9982	0.9959	0.9996	0.9995	0.9992	0.9974	0.9999	0.9988	0.9958	0.9995
	1.71e-05	1.77e-06	1.43e-05	5.01e-05	0.9992	2.49e-03	1.28e-05	6.28e-04	7.19e-06	1.58e-06
	0.9982	0.9959	0.9996	0.9995	0.9992	0.9974	0.9999	0.9988	0.9958	0.9995
	deer	deer	deer	deer	deer	deer	deer	deer	deer	deer
dog($\mathcal{A}_6$)	0.3989	0.9915	0.8812	0.1154	0.1433	0.9148	7.58e-02	0.6196	0.4921	0.1617
	2.67e-05	1.96e-06	2.37e-03	0.8843	2.35e-05	0.9148	5.47e-05	7.70e-05	6.71e-06	1.04e-05
	0.6006	0.9915	0.8812	0.8843	0.8563	0.9148	0.9238	0.6196	0.5074	0.8379
	cat	dog	dog	cat	cat	dog	cat	dog	cat	cat
frog($\mathcal{A}_7$)	0.9998	0.9998	0.9998	0.9997	0.9998	0.9996	0.9999	0.9997	0.9998	0.9997
	2.03e-05	3.11e-05	6.01e-05	1.86e-04	1.38e-05	1.14e-04	0.9999	1.07e-05	4.63e-05	1.33e-05
	0.9998	0.9998	0.9998	0.9997	0.9998	0.9996	0.9999	0.9997	0.9998	0.9997
	frog	frog	frog	frog	frog	frog	frog	frog	frog	frog
horse($\mathcal{A}_8$)	0.9924	0.9953	0.9900	0.9989	0.9998	0.9991	0.9945	0.9998	0.9963	0.9995
	8.50e-05	4.29e-05	9.55e-03	8.48e-05	7.39e-05	4.19e-05	2.76e-05	0.9998	2.31e-05	1.99e-05
	0.9924	0.9953	0.9900	0.9989	0.9998	0.9991	0.9945	0.9998	0.9963	0.9995
	horse	horse	horse	horse	horse	horse	horse	horse	horse	horse
ship($\mathcal{A}_9$)	0.7220	0.1382	0.5000	0.7736	0.9983	0.6808	0.4648	0.6148	0.9945	0.4792
	4.71e-02	0.8536	6.50e-04	3.24e-04	3.35e-06	3.04e-04	8.79e-05	3.85e-05	0.9945	9.87e-03
	0.7220	0.8536	0.5000	0.7736	0.9983	0.6808	0.4955	0.6148	0.9945	0.4792
	ship	car	ship	ship	ship	ship	car	ship	ship	ship
truck($\mathcal{A}_{10}$)	0.9894	0.9847	0.5815	0.9414	0.9378	0.9084	0.9752	0.9194	0.9244	0.9987
	9.42e-03	1.34e-02	2.45e-03	3.01e-04	2.02e-02	1.71e-04	1.62e-04	5.57e-04	7.97e-03	0.9987
	0.9894	0.9847	0.5815	0.9414	0.9378	0.9084	0.9752	0.9194	0.9244	0.9987
	truck	truck	truck	truck	truck	truck	truck	truck	truck	truck

APPENDIX B

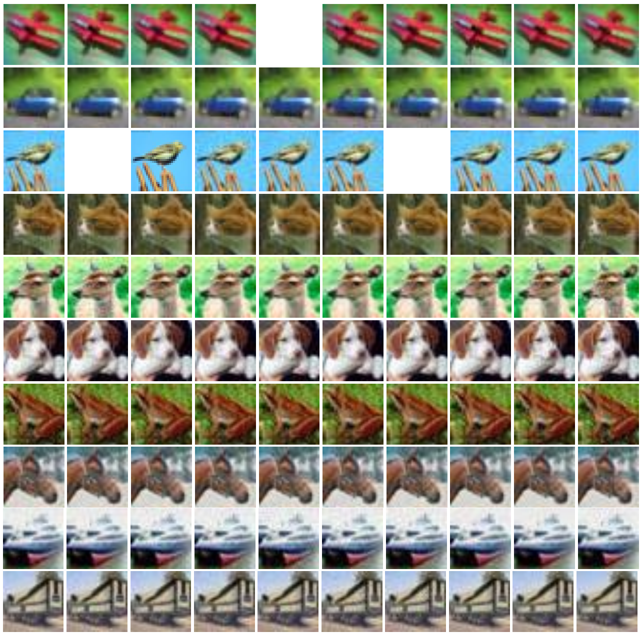


FIGURE 13: For $1 \leq a \leq 10$, the image on the diagonal at the $(a,a)^{th}$ position is the ancestor $\mathcal{A}_a$ (recovered from Table 1) classified by VGG-16 as belonging to the category c_a. The picture in the $(a,t)^{th}$ position, with $t \neq a$, is the adversarial picture $\mathcal{D}^{F_5}_{a,t}(\mathcal{A}_a) = \text{EA}^{\text{target,VGG-16}}_{L_2,F_5}(\mathcal{A}_a, c_t)$ obtained after the first successful run of the algorithm. Both images $\mathcal{D}^{F_5}_{a,t}(\mathcal{A}_a)$ and $F_5(\mathcal{D}^{F_5}_{a,t}(\mathcal{A}_a))$ are classified by VGG-16 as belonging to c_t with a c_t-label value ≥ 0.95. The 3 fully empty pictures correspond to the $(ancestor(\mathcal{A}_a), target)$ combinations for which the algorithm did not terminate successfully for any of the 10 runs.

	plane	car	bird	cat	deer	dog	frog	horse	ship	truck	Row Average
plane ($\mathcal{A}_1$)		35.76	91.12	506.3		175.48	103.65	273.09	34.85	311.95	191.53
car ($\mathcal{A}_2$)	320.99		113.00	341.12	234.18	382.84	166.51	328.07	131.11	202.12	246.66
bird ($\mathcal{A}_3$)	261.66			302.21	241.56	441.72	472.76	705.55	225.94	372.27	377.96
cat ($\mathcal{A}_4$)	309.18	278.29	96.88		117.97	75.48	55.62	151.89	184.48	266.03	170.65
deer ($\mathcal{A}_5$)	531.52	823.99	129	144.76		77.16	196.59	217.64	405.2	523.14	338.78
dog ($\mathcal{A}_6$)	283.08	251.62	155.09	40.63	472.17		171.14	126.53	268.26	253.78	224.70
frog ($\mathcal{A}_7$)	387.93	346.47	126.77	236.8	243.1	188.35		336.26	318.62	304.77	276.56
horse ($\mathcal{A}_8$)	138.08	90.46	85.51	81.00	79.06	147.76	135.58			70.54	103.50
ship ($\mathcal{A}_9$)	85.7	47.26	239.83	133.41	420.04	220.47	239.54	667.64		164.96	246.54
truck ($\mathcal{A}_{10}$)	40.81	182.2	115.01	233.75	135.08	264.78	200.74	83.13	95.88		150.15
Column Average	262.11	257.01	128.02	224.44	242.90	219.34	193.57	321.09	208.04	274.40	

FIGURE 14: $\text{EA}^{\text{target,VGG-16}}_{L_2,F_5}$'s performance on all possible ancestor/target combinations with one ancestor per category. The rows give the ancestor category c_a (and the specific ancestor $\mathcal{A}_a$ in c_a), the columns indicate the target class c_t, and the cell values indicate the average number of seconds required by $\text{EA}^{\text{target,VGG-16}}_{L_2}$ to terminate successfully, computed on 10 independent runs, with only the successful runs being considered.

TABLE 11: For $\mathcal{C}$ = VGG-16, each of the two parts of the cell in $(a, t)^{\text{th}}$-position contains a pair (maximum label value, corresponding class) given by $\mathcal{C}$ (top) and by $\mathcal{C} \circ F_5$ (bottom) for $\mathcal{D}^{F_5}_{a,t}(\mathcal{A}_a)$ (with $\mathcal{D}^{F_5}_{a,a}(\mathcal{A}_a) = \mathcal{A}_a$) whenever applicable (3 cells are empty).

	plane c_1	car c_2	bird c_3	cat c_4	deer c_5	dog c_6	frog c_7	horse c_8	ship c_9	truck c_{10}
plane($\mathcal{A}_1$)	0.69, plane	0.98, car	0.95, bird	0.95, cat		0.95, dog	0.95, frog	0.95, horse	0.95, ship	0.95, truck
	0.88, plane	0.95, car	0.98, bird	0.97, cat		0.98, dog	0.98, frog	0.97, horse	0.98, ship	0.98, truck
car($\mathcal{A}_2$)	0.95, plane	0.99, car	0.95, bird	0.95, cat	0.95, deer	0.95, dog	0.95, frog	0.95, horse	0.96, ship	0.95, truck
	0.99, plane	0.99, car	0.99, bird	0.99, cat	0.99, deer	0.99, dog	0.99, frog	0.99, horse	0.99, ship	0.99, truck
bird($\mathcal{A}_3$)	0.95, plane		0.99, bird	0.95, cat	0.95, deer	0.95, dog	0.95, frog	0.95, horse	0.95, ship	0.95, truck
	0.99, plane		0.99, bird	0.98, cat	0.99, deer	0.99, dog	0.99, frog	0.97, horse	0.99, ship	0.97, truck
cat($\mathcal{A}_4$)	0.95, plane	0.95, car	0.95, bird	0.99, cat	0.95, deer	0.95, dog	0.95, frog	0.95, horse	0.95, ship	0.95, truck
	0.99, plane	0.99, car	0.99, bird	0.99, cat	0.99, deer	0.99, dog	0.99, frog	0.99, horse	0.99, ship	0.99, truck
deer($\mathcal{A}_5$)	0.95, plane	0.95, car	0.95, bird	0.95, cat	0.99, deer	0.95, dog	0.95, frog	0.95, horse	0.95, ship	0.95, truck
	0.99, plane	0.99, car	0.99, bird	0.99, cat	0.99, deer	0.99, dog	0.99, frog	0.99, horse	0.99, ship	0.99, truck
dog($\mathcal{A}_6$)	0.95, plane	0.95, car	0.95, bird	0.95, cat	0.95, deer	0.99, dog	0.95, frog	0.95, horse	0.95, ship	0.95, truck
	0.99, plane	0.99, car	0.99, bird	0.99, cat	0.99, deer	0.91, dog	0.99, frog	0.99, horse	0.99, ship	0.99, truck
frog($\mathcal{A}_7$)	0.95, plane	0.95, car	0.95, bird	0.95, cat	0.95, deer	0.95, dog	0.99, frog	0.95, horse	0.95, ship	0.95, truck
	0.99, plane	0.99, car	0.99, bird	0.99, cat	0.99, deer	0.99, dog	0.99, frog	0.99, horse	0.99, ship	0.99, truck
horse($\mathcal{A}_8$)	0.95, plane	0.95, car	0.95, bird	0.96, cat	0.95, deer	0.95, dog	0.95, frog	0.99, horse		0.95, truck
	0.99, plane	0.99, car	0.99, bird	0.99, cat	0.99, deer	0.99, dog	0.99, frog	0.99, horse		0.99, truck
ship($\mathcal{A}_9$)	0.95, plane	0.95, car	0.95, bird	0.95, cat	0.95, deer	0.95, dog	0.95, frog	0.95, horse	0.99, ship	0.95, truck
	0.99, plane	0.99, car	0.99, bird	0.99, cat	0.99, deer	0.99, dog	0.99, frog	0.99, horse	0.99, ship	0.99, truck
truck($\mathcal{A}_{10}$)	0.95, plane	0.95, car	0.95, bird	0.95, cat	0.95, deer	0.95, dog	0.95, frog	0.95, horse	0.95, ship	0.99, truck
	0.99, plane	0.99, car	0.99, bird	0.99, cat	0.99, deer	0.99, dog	0.99, frog	0.99, horse	0.99, ship	0.99, truck

TABLE 12: For $\mathcal{C}$ = VGG-16, each of the 4 parts of the cell in $(a,t)^{\text{th}}$-position contains a pair (maximum label value, corresponding class) given, respectively from the top to the bottom, by $\mathcal{C} \circ F_1$, $\mathcal{C} \circ F_2$, $\mathcal{C} \circ F_3$, and $\mathcal{C} \circ F_4$ for $\mathcal{D}^{F_5}_{a,t}(\mathcal{A}_a)$ (with $\mathcal{D}^{F_5}_{a,a}(\mathcal{A}_a) = \mathcal{A}_a$) whenever applicable.

	plane c_1	car c_2	bird c_3	cat c_4	deer c_5	dog c_6	frog c_7	horse c_8	ship c_9	truck c_{10}
plane($\mathcal{A}_1$)	0.99, plane 0.44, car 0.72, plane 0.54, car	0.95, plane 0.81, car 0.82, car 0.99, car	0.97, plane 0.35, ship 0.92, bird 0.98, bird	0.54, plane 0.58, ship 0.64, cat 0.98, cat		0.65, plane 0.50, ship 0.87, dog 0.96, dog	0.95, plane 0.35, car 0.90, frog 0.96, frog	0.92, plane 0.57, ship 0.40, horse 0.96, horse	0.87, plane 0.78, ship 0.98, ship 0.90, ship	0.97, plane 0.47, ship 0.48, truck 0.96, truck
car($\mathcal{A}_2$)	0.97, ship 0.61, bird 0.99, plane 0.97, plane	0.76, car 0.99, car 0.99, car 0.99, car	0.92, ship 0.97, bird 0.99, bird 0.95, car	0.99, ship 0.80, cat 0.99, cat 0.95, cat	0.99, ship 0.83, bird 0.99, deer 0.85, car	0.55, ship 0.80, bird 0.91, dog 0.49, car	0.99, ship 0.56, frog 0.99, frog 0.86, frog	0.99, ship 0.89, bird 0.99, horse 0.70, horse	0.99, ship 0.59, ship 0.99, ship 0.56, ship	0.99, ship 0.99, car 0.99, truck 0.90, truck
bird($\mathcal{A}_3$)	0.77, bird 0.99, bird 0.96, plane 0.92, plane		0.99, bird 0.99, bird 0.99, bird 0.99, bird	0.48, cat 0.99, bird 0.65, bird 0.96, cat	0.89, bird 0.99, bird 0.75, deer 0.96, deer	0.74, ship 0.99, bird 0.64, dog 0.93, dog	0.84, bird 0.99, bird 0.91, bird 0.99, frog	0.99, bird 0.99, bird 0.91, bird 0.98, horse	0.83, ship 0.99, bird 0.93, bird 0.98, ship	0.56, bird 0.99, bird 0.97, bird 0.95, truck
cat($\mathcal{A}_4$)	0.32, plane 0.93, cat 0.45, bird 0.55, frog	0.61, cat 0.93, cat 0.95, car 0.97, car	0.51, ship 0.97, cat 0.99, bird 0.78, frog	0.91, cat 0.99, cat 0.99, cat 0.99, cat	0.95, frog 0.96, cat 0.99, deer 0.63, deer	0.49, cat 0.67, dog 0.99, dog 0.92, frog	0.61, cat 0.98, cat 0.97, frog 0.99, frog	0.51, cat 0.67, cat 0.99, horse 0.68, frog	0.50, ship 0.99, cat 0.81, ship 0.88, ship	0.98, frog 0.97, cat 0.61, truck 0.98, truck
deer($\mathcal{A}_5$)	0.96, plane 0.99, deer 0.93, plane 0.98, plane	0.65, car 0.87, deer 0.75, bird 0.98, car	0.99, cat 0.98, deer 0.98, bird 0.98, bird	0.99, cat 0.98, deer 0.99, cat 0.97, cat	0.58, plane 0.99, deer 0.99, deer 0.99, deer	0.46, cat 0.99, deer 0.76, dog 0.96, dog	0.97, cat 0.99, deer 0.99, frog 0.99, frog	0.77, cat 0.99, deer 0.97, horse 0.97, horse	0.46, cat 0.99, deer 0.99, ship 0.98, ship	0.95, cat 0.99, deer 0.77, truck 0.96, truck
dog($\mathcal{A}_6$)	0.79, truck 0.99, cat 0.94, plane 0.63, plane	0.81, frog 0.99, cat 0.88, car 0.65, car	0.96, cat 0.99, cat 0.99, bird 0.96, bird	0.99, cat 0.99, cat 0.99, cat 0.93, cat	0.93, frog 0.98, cat 0.88, deer 0.97, deer	0.99, cat 0.99, cat 0.73, cat 0.99, dog	0.90, frog 0.99, cat 0.75, frog 0.99, frog	0.89, cat 0.99, cat 0.99, horse 0.95, horse	0.99, truck 0.99, cat 0.69, cat 0.94, ship	0.83, truck 0.99, cat 0.58, cat 0.99, truck
frog($\mathcal{A}_7$)	0.93, plane 0.99, frog 0.99, plane 0.81, plane	0.88, frog 0.99, frog 0.99, car 0.95, car	0.61, frog 0.99, frog 0.98, bird 0.93, bird	0.80, frog 0.63, frog 0.99, cat 0.62, frog	0.74, cat 0.97, deer 0.99, deer 0.67, frog	0.76, cat 0.63, frog 0.99, dog 0.81, frog	0.91, frog 0.99, frog 0.99, frog 0.99, frog	0.56, frog 0.56, horse 0.99, horse 0.63, frog	0.89, ship 0.99, frog 0.99, ship 0.63, ship	0.93, frog 0.99, frog 0.99, truck 0.95, truck
horse($\mathcal{A}_8$)	0.74, plane 0.95, horse 0.99, plane 0.88, plane	0.71, plane 0.97, horse 0.99, car 0.97, car	0.60, plane 0.88, bird 0.99, bird 0.98, bird	0.80, plane 0.60, cat 0.99, cat 0.95, cat	0.78, bird 0.98, horse 0.99, deer 0.95, deer	0.39, bird 0.89, horse 0.99, dog 0.79, bird	0.48, plane 0.61, cat 0.99, frog 0.99, frog	0.74, dog 0.99, horse 0.99, horse 0.99, horse		0.58, plane 0.98, horse 0.9546, truck 0.97, truck
ship($\mathcal{A}_9$)	0.87, ship 0.99, plane 0.99, plane 0.49, plane	0.99, ship 0.77, car 0.99, car 0.70, car	0.69, cat 0.96, plane 0.99, bird 0.46, ship	0.89, ship 0.91, plane 0.96, cat 0.96, cat	0.74, car 0.67, ship 0.85, deer 0.62, deer	0.53, dog 0.95, plane 0.99, dog 0.24, ship	0.99, ship 0.80, plane 0.59, ship 0.98, frog	0.38, cat 0.98, plane 0.99, horse 0.55, horse	0.98, ship 0.45, plane 0.99, ship 0.99, ship	0.65, ship 0.93, plane 0.76, truck 0.96, truck
truck($\mathcal{A}_{10}$)	0.99, plane 0.99, plane 0.99, plane 0.81, plane	0.59, plane 0.99, plane 0.99, car 0.94, car	0.99, plane 0.98, plane 0.97, bird 0.97, bird	0.68, ship 0.96, plane 0.99, cat 0.85, cat	0.92, plane 0.97, plane 0.99, deer 0.91, deer	0.71, plane 0.96, plane 0.99, dog 0.97, dog	0.96, ship 0.95, plane 0.99, frog 0.99, frog	0.98, plane 0.58, horse 0.99, horse 0.58, horse	0.96, plane 0.99, plane 0.97, ship 0.91, ship	0.99, plane 0.99, plane 0.99, truck 0.99, truck

REFERENCES

[1] R. Chitic, N. Deridder, N. Bernard, and F. Leprévost, "Robustness of adversarial images against filters," in *Communications in Computer and Information Science, International Conference on Optimization and Learning (OLA) 2021*, vol. 1443. Springer, 2021.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, pp. 1097–1105, 2012. [Online]. Available: https://papers.nips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 248–255. [Online]. Available: https://ieeexplore.ieee.org/document/5206848

[4] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2818–2826. [Online]. Available: https://ieeexplore.ieee.org/document/7780677

[5] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2017*, 2017, pp. 4700–4708. [Online]. Available: https://ieeexplore.ieee.org/document/8099726

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014. [Online]. Available: https://arxiv.org/abs/1409.1556

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[8] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition: A convolutional neural-network approach," *IEEE Transactions on Neural Networks*, vol. 8, no. 1, pp. 98–113, 1997. [Online]. Available: https://ieeexplore.ieee.org/document/554195

[9] S. M. Hizam, W. Ahmed, M. Fahad, H. Akter, I. Sentosa, and J. Ali, "User behavior assessment towards biometric facial recognition system: A sem-neural network approach," *arXiv preprint arXiv:2106.03371*, 2021.

[10] N. Papernot, P. McDaniel, A. Sinha, and M. Wellman, "Towards the science of security and privacy in machine learning," *CoRR*, vol. abs/1611.03814, 2016. [Online]. Available: https://arxiv.org/abs/1611.03814

[11] K. Grosse, N. Papernot, P. Manoharan, M. Backes, and P. McDaniel, "Adversarial examples for malware detection," in *European symposium on research in computer security 2017*. Springer, 2017, pp. 62–79. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-66399-9_4

[12] J. Jeon, J. H. Park, and Y.-S. Jeong, "Dynamic analysis for iot malware detection with convolution neural network model," *IEEE Access*, vol. 8, pp. 96 899–96 911, 2020.

[13] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli, "Evasion attacks against machine learning at test time," in *Machine Learning and Knowledge Discovery in Databases*, ser. LNAI, vol. 8190. Springer, 2013, pp. 387–402. [Online]. Available: https://www.springer.com/gp/book/9783642409936

[14] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *CoRR*, vol. abs/1609.08114, 2016. [Online]. Available: https://arxiv.org/abs/1609.08144

[15] S. Kwon *et al.*, "A cnn-assisted enhanced audio signal processing for speech emotion recognition," *Sensors*, vol. 20, no. 1, p. 183, 2020.

[16] L. Caltagirone, M. Bellone, L. Svensson, and M. Wahde, "Lidar–camera fusion for road detection using fully convolutional neural networks," *Robotics and Autonomous Systems*, vol. 111, pp. 125–131, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0921889018300496?via%3Dihub

[17] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang *et al.*, "End to end learning for self-driving cars," *CoRR*, vol. abs/1604.07316, 2016. [Online]. Available: https://arxiv.org/abs/1604.07316

[18] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012. [Online]. Available: https://ieeexplore.ieee.org/document/6296526

[19] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *CoRR*, vol. abs/1312.6199, 2013. [Online]. Available: https://arxiv.org/abs/1312.6199

[20] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 39–57. [Online]. Available: https://ieeexplore.ieee.org/document/7958570

[21] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *CoRR*, vol. abs/1801.00553, 2018. [Online]. Available: http://arxiv.org/abs/1801.00553

[22] R. Chitic, N. Bernard, and F. Leprévost, "Evolutionary algorithms deceive humans and machines at image classification: An extended proof of concept on two scenarios," *Journal of Information and Telecommunication*, vol. 5, pp. 121–143, 2021, http://dx.doi.org/10.1080/24751839.2020.1829388.

[23] R. Chitic, N. Bernard, and F. Leprévost, "A proof of concept to deceive humans and machines at image classification with evolutionary algorithms," in *Asian Conference on Intelligent Information and Database Systems*. Springer, 2020, pp. 467–480.

[24] N. Bernard and F. Leprévost, "How evolutionary algorithms and information hiding deceive machines and humans for image recognition: A research program," in *Proceedings of the OLA'2019 International Conference on Optimization and Learning (Bangkok, Thailand, Jan 29-31, 2019)*. Springer, Heidelberg, 2019, pp. 12–15.

[25] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 5, pp. 828–841, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8601309

[26] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2016, pp. 372–387. [Online]. Available: https://ieeexplore.ieee.org/document/7467366

[27] J. Wu, "Generating adversarial examples in the harsh conditions," *CoRR*, vol. abs/1908.11332, 2020. [Online]. Available: https://arxiv.org/abs/1908.11332

[28] M. Jere, L. Rossi, B. Hitaj, G. Ciocarlie, G. Boracchi, and F. Koushanfar, "Scratch that! An evolution-based adversarial attack against neural networks," *CoRR*, vol. abs/1912.02316, 2019. [Online]. Available: https://arxiv.org/abs/1912.02316

[29] Y. Geifman, "cifar-vgg," 2018, https://github.com/geifmany/cifar-vgg.

[30] A. Krizhevsky, V. Nair, and G. Hinton, "The CIFAR-10 dataset," 2009. [Online]. Available: https://www.cs.toronto.edu/~kriz/cifar.html

[31] A. E. Eiben and J. E. Smith, *Introduction to evolutionary computing*. Springer, 2003. [Online]. Available: https://www.springer.com/gp/book/9783642072857

[32] N. Bernard and F. Leprévost, "Evolutionary algorithms for convolutional neural network visualisation," in *High Performance Computing – 5th Latin American Conference, 2018 (Bucaramanga, Colombia, Sep 23-28, 2018)*, ser. Communications in Computer and Information Science, vol. 979. Springer, Heidelberg, 2018, pp. 18–32. [Online]. Available: https://www.springerprofessional.de/high-performance-computing/16587152

[33] T. E. Oliphant, *A guide to NumPy*. Trelgol Publishing USA, 2006, vol. 1.

[34] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000, https://www.drdobbs.com/open-source/the-opencv-library/184404319.

[35] F. Chollet, "Keras," GitHub code repository, 2015-2020, https://github.com/fchollet/keras.

[36] S. Varrette, P. Bouvry, H. Cartiaux, and F. Georgatos, "Management of an academic HPC cluster: The UL experience," in *Proceedings of the 2014 International Conference on High Performance Computing & Simulation (HPCS 2014)*. IEEE, 2014, pp. 959–967.

[37] S. Kullback and R. Leibler, "On information and sufficiency," *The Annals of Mathematical Statistics*, vol. 22, pp. 79–86, 1951. [Online]. Available: https://www.jstor.org/stable/2236703

[38] J. S. Lim, *Two-Dimensional Signal and Image Processing*. Prentice Hall, 1990.

RALUCA CHITIC is a third-year Ph.D. student in computer science at the University of Luxembourg. She holds a BSc degree in physics and an MSc degree in artificial intelligence. Her current focus is on computer vision, explainable neural networks and evolutionary algorithms.

ALI OSMAN TOPAL received his B.S. degree in Electrical and Electronic Engineering from Gaziantep University, Turkey, in 2000, and his M.S. and Ph.D. degrees in Computer Engineering from Epoka University, Tirana, Albania, in 2017. He is currently a researcher, lecturer, and post-doc in computer science at the University of Luxembourg.

His research interests lie in the area of artificial intelligence, ranging from theory to implementation. His research on evolutionary algorithms has been published in prestigious journals. He is currently focusing on computer vision, deep learning, and XAI.

FRANCK LEPRÉVOST has been a professor at the University of Luxembourg since 2003. He was its vice-president from 2005 to 2015, in charge of organization, international relations, and rankings. He received his Ph.D. in mathematics from University Paris 7 in 1992. Before joining the University of Luxembourg, he held academic positions in France (CNRS, Paris, University Joseph Fourier, Grenoble) and Germany (Max-Planck Institut ür Mathematik, Bonn, Technische Universität Berlin). He spent a sabbatical year (2016) at Polytech, Saint Petersburg (Russia). His research interests include pure mathematics, security of telecommunication and cryptology, evolutionary algorithms and deep learning, international relations and academic leadership. He is the author of 60 publications and three scientific books.