Carpet-bombing patch: attacking a deep network without usual requirements

Pol Labarbarie, Adrien Chan-Hon-Tong, Stéphane Herbin, Milad Leyli-Abadi

Carpet-bombing patch: attacking a deep network without usual requirements

Pol Labarbarie
IRT SystemX and ONERA/DTIS, Université Paris-Saclay
Palaiseau, France
firstname.name@irt-systemx.fr

Adrien Chan-Hon-Tong and Stéphane Herbin
ONERA/DTIS, Université Paris-Saclay
Palaiseau, France
firstname.name@onera.fr

Milad Leyli-Abadi
IRT SystemX
Palaiseau, France
firstname.name@irt-systemx.fr

Abstract

Although deep networks have shown vulnerability to evasion attacks, such attacks have usually unrealistic requirements. Recent literature discussed the possibility to remove or not some of these requirements. This paper contributes to this literature by introducing a carpet-bombing patch attack which has almost no requirement. Targeting the feature representations, this patch attack does not require knowing the network task. This attack decreases accuracy on Imagenet, mAP on Pascal Voc, and IoU on Cityscapes without being aware that the underlying tasks involved classification, detection or semantic segmentation, respectively. Beyond the potential safety issues raised by this attack, the impact of the carpet-bombing attack highlights some interesting property of deep network layer dynamic.

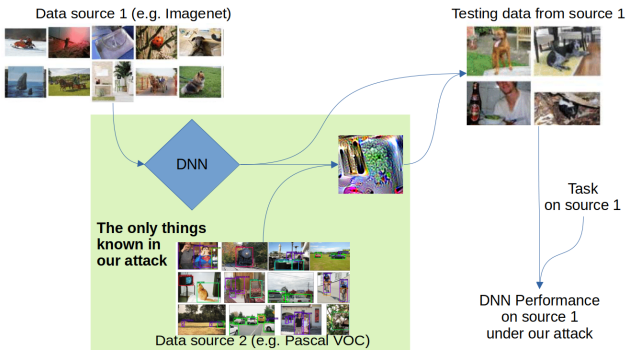


Figure 1. Carpet-bombing patch. Our attack only requires a deep network and a proxy data source to design an adversarial patch. It does not requires neither data from targeted source nor even the knowledge of the task on these data. Yet it could strongly decrease performance on this task.

1. Introduction

Deep neural networks (DNNs) have given state-of-the-art results in most computer vision tasks, including image classification [16], semantic segmentation [21], and object detection [29]. Due to their complexity, DNNs have showed vulnerability to adversarial examples, *i.e.*, small perturbations of their inputs designed to fool them [2, 33]. Such vulnerability has motivated researchers to try to develop more robust DNNs [23], as well as to prove that they are robust [5]. Other research has been dedicated to the design of more powerful attacks [17] or the development of different class of attacks, e.g., patch attacks and universal attacks [3, 35]. Although such attacks cause safety issues, they also reveal interesting properties about DNNs and their internal structure. For example, invisible noise attacks have highlighted that DNNs sometimes focus on high-frequency [13]. Also, [15, 22, 26] have offered connections between adversarial ML and broader topics like network interpretability.

In the same way, we propose in this paper several experiments which highlight that the deep network features are not activated alike by adversarial *clutter* or adversarial *foreground* even when optimised to do so (following [15] which introduces the idea to target middle features).

As major contribution, we show how these experiments led us to find a new kind of adversarial patch attack which requires less prior information than previous attacks as highlighted by Figure 1 and Table 1. Built to disrupt the feature representations of network encoder, this new patch could affect multiple tasks at the same time. For this reason,

Adversarial attacks	Network	Target image or dataset	Task	Access to pixel
[2, 33] (original)	+	+	+	+
[12, 36] (blackbox)	≈	+	+	+
[24] (universal)	+	≈	+	+
[25, 31] (patch)	+	≈	+	–
[15, 30] (feature)	–	+	≈	+
Ours	+	–	–	–

Table 1. Typology of adversarial attack depending on the requirement of the attack (less requirement is obviously better). Those requirements are prior knowledge or the fact that the attack is performed on pixel values (not feasible in real world). Our attack is the first with almost not requirement: it is clearly the first tested without knowledge of the task prior (and it also does not require neither the target nor the dataset nor access to pixel).

this new attack we named carpet-bombing patch, should interest (at least) the safety community. More broadly, it raises questions about intermediary layer dynamics.

To explain the interest (from a safety point of view) in our patch, we briefly present a typology of the different attacks in the following. The original adversarial attacks present the following requirements:

- the knowledge of the targeted testing data,
- the knowledge of the targeted task,
- the knowledge of the targeted network (architecture, weights and training data),
- and, the access to the pixel value (i.e. the attack is performed **after** image acquisition and not directly in the physical world).

Many adversarial attacks try to remove one or more of the above-mentioned requirements. For example, *black box* usually refers to the fact that network is unknown[1] [27], *universal* usually refers to the fact that the attack is not specific to some target [24] and *patch-attacks* are more likely to be printable in real-world [3].

In this paper, we introduce a new attack that requires almost no prior: this attack does not require knowing the task (to our knowledge, this setting has not been explored), and at the same time, it also does not require knowing the target or to access the pixel. Finally, it offers a moderate effect in blackbox setting. This is the main limitation of this attack regarding recent works like [15, 39]. Yet it offers a stronger effect in whitebox setting and is more physically plausible.

The paper is organised as follows: The related works are described in section 2 followed by the scientific story of the proposed attack in section 3. The experiments proving the harmfulness of this attack are presented in section 4. Finally, the conclusions are provided in section 5.

[1]Despite it covers situations where queries of the network are allowed, but inner variables of the network are not known, or, the situation where no queries are allowed usually called *transferable attacks*.

2. Related works

2.1. Transferable invisible adversarial attacks

At the beginning, the community tried to build transferable adversarial noises by directly targeting the white box model loss [10, 20]. Some works improve transferability by adding a momentum [8] or by building an ensemble of white box models on which the attack is built [20, 37]. However, such methods show low transferability due to overfitting of the source model. Zhou et al. [39] reduce the overfitting by introducing a regulariser that maximises the distance between natural images and adversarial noises in the feature space representation. Rozsa et al. [30], and Inkawhich et al. [15] propose not to target the white box model loss, but exclusively the feature representations. They propose to minimise the L_2 distance between a target point and a source in the feature space for a chosen layer. The source point is often the feature representation of a certain class. This procedure is sensitive to the choice of the target and shows low scalability to larger models, and dataset such as Imagenet [7]. To better represent the target class in the feature space, Inkawhich et al. [14] propose to model the class-wise feature distributions of the white box model. Instead of targeting one single layer, they suggest to attack multiple layers.

2.2. Adversarial patch attacks (APAs)

Classification: APAs were introduced first for image classification by Brown et al. [3]. Instead of finding a small additive adversarial noise, they constrained the optimization procedure to a small part of the image but allowed it to be unconstrained in magnitude. They produced a patch capable of fooling multiple ImageNet classification models in digital or physical domains (just by printing the patch).

Object detection: Attacking object detectors was explored in several works working on different applications. In the beginning, patches were directly applied on the struck object. The first two works on patch-based attacks had targeted stop signs [4, 32]. They produced stickers, when applied, can fool YOLOv2 or Faster RCNN. Thys et al. [34] were the first to create a patch causing the disappearance of people when it was applied on them. These works focused on designing a patch that overlaps the targeting object to change its class or suppress detection.

On the other hand, contextual patch attacks are patches which without overlapping with the object of interest can blind the detector. They were first explored by Liu et al. [19]. Their patch, named Dpatch, showed transferability over patch position, network architecture, and dataset. However, their patches are never clipped to the image range, which is unsuitable for real-world applications. Lee et al. [18] studied the Dpatch attack in feasible physical condi-

tions and compared it to their new attack. They outperformed the Dpatch method and showed real-time attack success. The success of those attacks consists of adding a salient patch in the image producing false positives. Saha et al. [31] were among the first to develop attacks and defence for contextual adversarial patches. They introduce the idea of removing false positives on the patch to measure mainly the contextual effect of patches.

Semantic segmentation: The first paper introducing real-world APAs targeting semantic segmentation models is [25]. The work presents a novel loss function that, when used, leads to powerful attacks in both digital and real-world scenarios. Unlike patches applied to classification or object detection, the authors showed that semantic segmentation models are not easily corruptible.

As is common throughout the APAs, patches are designed to use the model loss function as the target objective. For comparison, targeting the feature space of the model encoder, we develop a new patch capable of fooling multiple tasks which shows similar performance when attacking with task knowledge. However, we find out that this kind of patches do not keep the model transferability property as is the case with invisible noises.

3. Another look to feature attacks

3.1. Trying to get the best of the two worlds

As recalled in related works, the best attacks (from a requirement point of view) were, on one hand, patch attacks [31], not requiring to access pixel value (and image target), and, on the other hand, feature-based adversarial attacks [14, 15], which are model transferable. The starting point of our work was to try to combine the best of these two attacks.

More formally, let F a given pre-trained neural network performing a certain task. We denote by f the encoder part of F and denote by $\mathbb{L} = \{1, ..., L\}$ the set of the L layers of f. One encoder f is often used in multiple F, each performing a different task. Let δ be the adversarial perturbation that can be whether an invisible adversarial noise or an APA. Adversarial noises are obtained by optimising without constraint in space but with a constraint for a certain L_p norm of δ (i.e $||\delta||_p \leq \varepsilon$ for $\varepsilon > 0$). On the other hand, APA is optimised with constraint in space but without constraint in norm. For both of them, to make a realistic attack, we enforce that pixels of the perturbed input or of the APA are in the $[0, 1]$ range. Transferable invisible adversarial noises are generally obtained by the following optimisation procedure:

$$\arg\max_{\delta} \mathcal{L}_{task}(F, x, y, \delta) + \eta \mathcal{L}_{feature}(F, x, \delta), \quad (1)$$

where x is the input image, y is the label associated with the task and $\eta > 0$ is the weight corresponding to the contribution of the feature loss. We have two terms: the first one, $\mathcal{L}_{task}$ can be the model loss or a loss derived from the task. This loss disturbs directly the model task. The second term is a feature disruptive loss. This loss generally enforces that the feature map of the perturbed input differs from the original inputs. The adversarial patches are designed with the objective:

$$\arg\max_{\delta} \mathcal{L}_{task}(F, x, y, \delta). \quad (2)$$

In the case of classification, we have $\mathcal{L}_{task} = \mathcal{L}_{cross-entropy}$. Concerning the object detection, the task loss can be general like $\mathcal{L}_{task} = \mathcal{L}_{object-detector}$ or can be more specific such as $\mathcal{L}_{task} = \mathcal{L}_{cross-entropy}$ and for semantic segmentation, we have $\mathcal{L}_{task} \simeq \mathcal{L}_{cross-entropy}$. The transferable invisible attack and the patch attack involve a task loss term that implies knowledge of the task. Our proposed carpet-bombing attack, inspired by the feature disruptive term of transferable noises, is described by the following formula

$$\arg\max_{\delta} \mathcal{L}_{feature}(f, x, \delta), \quad (3)$$

with

$$\mathcal{L}_{feature}(f, x, \delta) = \sum_{l \in \mathbb{L}} \sum_{k \in K} ||(f_l(x_\delta)_k - f_l(x)_k) \odot m_l||_2, \quad (4)$$

where $f_l(x_\delta)_k$ is the k-th feature map of layer l of f for the corrupted image x_δ, m_l the binary mask which is 0 on the patch and 1 everywhere else and $\odot$ is the element-wise product. If δ is an invisible noise we have

$$x_\delta = x + \delta \quad (5)$$

and if, δ is a patch, we have

$$x_\delta = x \odot (1 - m) + \delta \odot m, \quad (6)$$

where m is a binary mask that is 1 on the patch and 0 everywhere else. Attacking features of f instead of F makes the patch independent of the task considered by F. Hence, we can generate one patch capable of fooling multiple F that are based on the same f. It can correspond to the scenario where the attacker does not know the task or when multiple tasks use the same encoder f.

3.2. Intriguing behaviour of noise vs patch

Consequently, following [15], we adapt the idea of feature attack across a patch rather than a noise. Nevertheless, we observe very different behaviour between these two attacks, although they are basically designed for the same objective. We propose in this section to study the effects of the

constraint on both the obtained attack and the feature space. We also conduct several experiments to force the patch to have the same effect on features as invisible noise and inversely.

3.2.1 Attack setting

Herein, we present the attacking procedure applied to invisible noises and designed patches. Our goal is to design universal contextual attacks. We provide more details in the following concerning these attributes.

Universal: We follow [24,31] and learn a universal attack (invisible noise or a patch) on the training dataset that works across the unseen test dataset. Instead of finding a specific adversarial δ for each image x, we optimise δ through iterating over training images. To do so, we sample two subsets of images from the test dataset that the network is designed for: one to train our patch and the other to evaluate it.

Contextual effects: The objective is to design patches that deteriorate the performance metric by impacting the whole feature map. It should not be sensitive to its placement over an object or the rate of false alarms. For image classification, instead of placing the patch in the middle of the image, we fix the patch at the top-left corner *i.e.* pixel $(5, 5)$. For object detection, we extract images wherein objects do not overlap with the patch, which are placed at the top-left corner. Secondly, we remove detections overlapping the patch. Finally, since there are no objects of interest for semantic segmentation, we follow the setup of [25]. The patch is centred in the scene, and metrics are not measured on patch pixels.

For both adversarial noises and patches, we solve their corresponding optimisation problem and clip them to $[0, 1]$. Clipping ensures that the perturbed image is always maintained in the distribution of original images, and the produced patches are more realistic and do not contain *inf* values. When not specified, the patch is initialised as an all-zeros tensor, and we launch the optimisation process for 100 steps, where 1 step corresponds to 1000 iterations. SGD is used as an optimiser with a momentum of 0.9, a minibatch size of 1, and 10 iterations per minibatch. We evaluate the performance of attacks in the same condition as during the training phase.

3.2.2 Comparative results

First, we evaluate the effects of both attacks on the training model and the hidden model (no knowledge of weights or architecture). In our experiments, we use models pretrained on ImageNet-1K [7]: ResNet50 (R50) as the white-box model and ResNet18 (R18) as the hidden model (exactly like in [15]). For patches, we use the procedure detailed in Sec. 3.2 and for invisible noises, following [15], we use iterative gradient sign attack with momentum (TMIFGSM) [8] for 100 steps. Both attacks are targetting uniquely layer 4 of R50 *i.e.*, $\mathbb{L} = \{L\}$. We split the ImageNet-1K test set into two subsets. We train attacks on 40000 images and evaluate them on 10000 images. Once attacks are learned, we apply them to the testing set. Table 2 shows that patches have a better severity but present less transferability than invisible noises.

	Whitebox (R50)	Hidden model (R18)
Clean	76.06	70.14
Noise	16.13	**33.19**
Patch	**0.69**	62.99

Table 2. Accuracies (%) on 10000 ImageNet images for both white box and hidden model under adversarial attacks designed to break internal feature map.

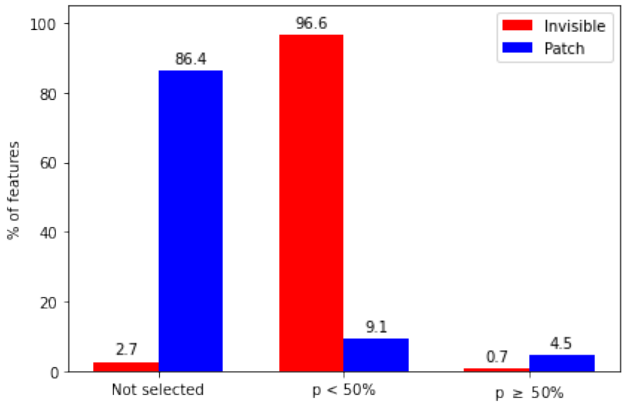


Figure 2. Classification of features for both attacks depending on their frequency of top disruption evaluated on 1000 ImageNet images. The fifty top attacked features for the L_2-norm are extracted for each image. The label "Not selected" correspond when the feature does not appear once in the top attacked features through 1000 images.

From this result, we can ask ourselves; *What is making adversarial noises transferable?*

3.2.3 Attacked features

To find whether or not attacked features are responsible for the transferability of the attack, we extract "top attacked features" from both types of attacks. To do so, we pass 1000 cleaned and corrupted images to the white box model and extract features from layer 4 of R50. We measure the L_2-distance between cleaned and attacked features for each image and save only the top fifty attacked features (*ie* largest norm). We observe from figure 2 that patches are essentially targeting a limited set of features ($\approx 13\%$) and focusing on

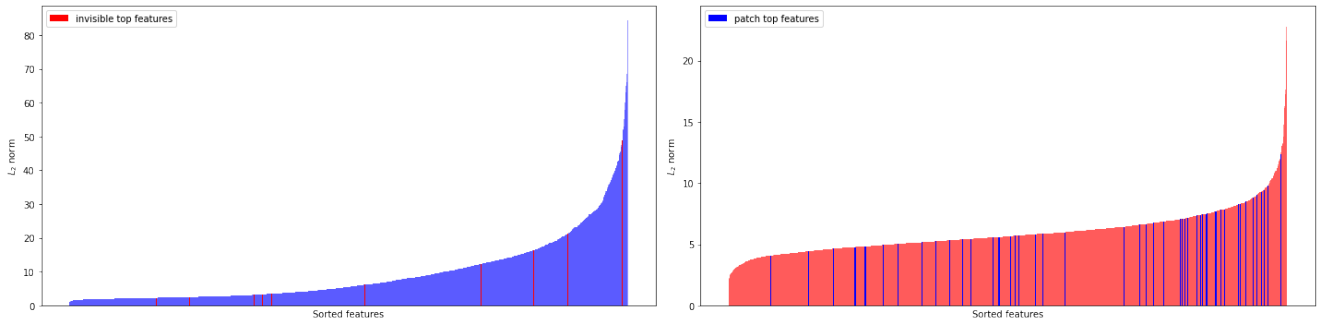


Figure 3. Sorted bar chart of the average L_2-distance between cleaned and attacked features over 1000 ImageNet images. On the left, images are corrupted by patch and on the right, by invisible adversarial noise. The top attacked features for invisible noise are highlighted on the left and on the right for the patch.

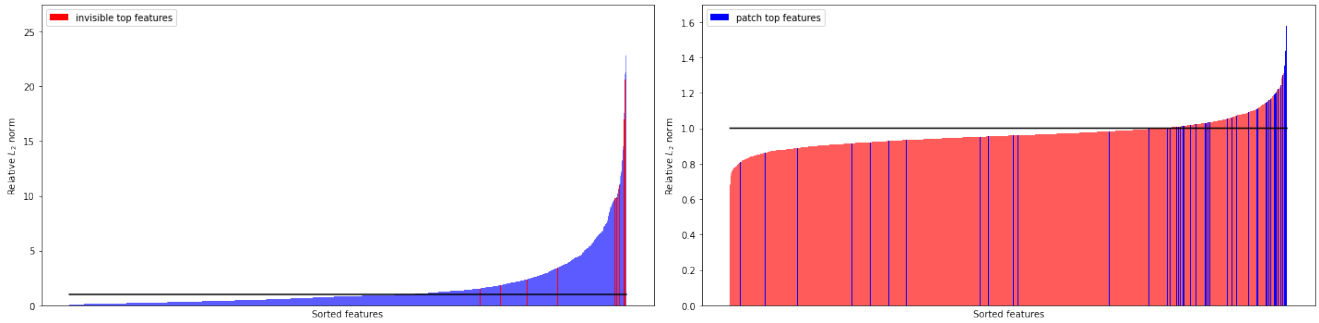


Figure 4. Sorted bar chart of the average relative L_2-distance between forced and unforced attack. The average is done on 1000 ImageNet images. On the left, images are corrupted by patch and on the right, by invisible adversarial noise. The top attacked features for invisible noise are highlighted on the left and on the right for the patch. Those top features are used to design the forced attack.

a more significant part of features than invisible noises (4.5 % against 0.7 %). The latter targets a more extensive set of features but in a random fashion (96.6 % for $p < 50\%$). In figure 3, we plot the sorted average L_2 distance between clean and attacked images for each feature and highlight the top attacked features ($p \geq 50\%$) in red for the invisible attack and in blue for the APA. We observe that patch attacks significantly impact features more than invisible noise. We retrieve the same dynamic as in figure 2: patches focus on a smaller set of features than adversarial noises. By looking at the dynamic of the graph, invisible noises seem to have a diffusive effect on the features and patches show a sharper effect. Interesting to note that they are attacking different features by default.

That brings us to ask: *Are the invisible noises transferable due to the fact that they target these top features? Can we force a patch to only target those features? Is this operation make the patch transferable? Are some features insensitive to a particular attack?*

3.2.4 Mimetic attack

We extract the most attacked features for both adversarial attacks ($p \geq 50\%$), and we denote them, K_{patch} and K_{inv} respectively. Now, we resolve (4) the same way as before, except we replace K by K_{inv} when designing the patch and by K_{patch} when designing an invisible attack. Figure 4 plots the relative L_2 norm when forced attacks target specific features. We highlight the previously top-attacked features for each attack. This graph shows that the patch can attack top invisible features (augmentation from previous trials by a factor of $\simeq 10$ of the attack on those features). It indicates the fact that patches are somehow capable of disrupting top-attacked invisible features. On the other hand, invisible noises could not target other features. Most values around the value one indicate the difficulty of constructing noises targeting a selected set of features. However, does it affect the performance of both models? In Table 2, we report performance when we constrain attacks. Both attacks demonstrate less effectiveness on the white box than in default mode. However, concerning the hidden model, we do not observe a gain when targeting invisible features and *vice versa*.

3.2.5 Spatial impact:

At this point, one could wonder if there is a difference between feature perturbation created by noise or patch, as the perturbation designed by an adversarial noise works on different models, which is not the case for an adversarial patch. One possible explanation is that, the spatial patterns of both perturbations are structurally different (even if this is not visible when considering the spatial average of a given feature channel).

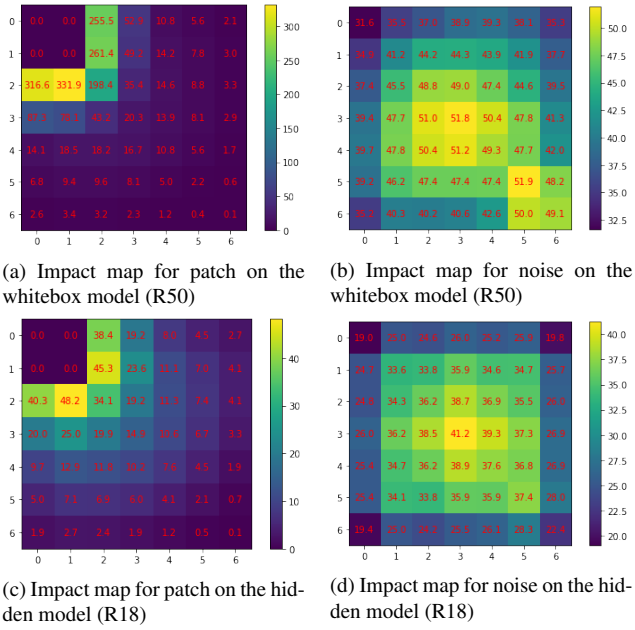
We propose to study the spatial impact of attacks. We apply attacks built by default (*i.e.,* no constraint on the attacked features) on 1000 images and measure the average L_2-distance between attacked features and clean features in each cell of the feature map. We obtain a map representing the spatial impact of attacks (Fig. 5). The heat maps of attack impact are shown in top figures when applied to the white box model (Fig. 5a, 5b) and on the bottom, the relative map when applied on the hidden and the white box model (Fig. 5c, 5d). From these plots (Fig. 5a, 5b), it could be clearly seen that patches are attacking their neighbourhood area and the adversarial noises are diffused over all features. We see that patches have a broader impact on attacked cells than noises. But is this significant impact capable of transfer to the hidden model? From plots 5c and 5d, we observe that the patch effect on its neighbourhood almost disappears. Contrarily, concerning the invisible noises, their effect decreases despite their significant impact.

3.3. Transfer between related networks

Seeing the behaviour of the patch attack on a hidden model, we consider different but closer models. For this purpose, we consider a Darknet-19 [28] classifier trained on Imagenet, and we finetune it into a Yolo on PASCAL VOC. We save a snapshot of the Darknet-19 being finetuned every 300 iterations and measure the impact of the original patch designed to target the Imagenet. Precisely, we choose to target last layer *i.e.,* $\mathbb{L} = \{L\}$). We pass 1000 ImageNet images and extract the top attacked features.

As a result, figure 6 shows the evolution of patch feature impact for Darknet-19 and evolutionary versions of YoloV2. This graph shows that patch impact decreases very quickly during YoloV2 training. First, the amplitude of the attack strongly decreases. Then, we observe that the patch has a significant pattern drift considering most attacked features after 600 iterations. So, even when targeting the same model with related weights, the original patch quickly becomes ineffective (we check that it is possible to design a specific patch for each snapshot).

These experiments also point out that the features not considered by the attack in the original model can become the most important. At this point, it means either weight changes too quickly so that the initial relationship between the two networks is irrelevant or that the network importantly reorganises the influence of features.



(a) Impact map for patch on the whitebox model (R50)

(b) Impact map for noise on the whitebox model (R50)

(c) Impact map for patch on the hidden model (R18)

(d) Impact map for noise on the hidden model (R18)

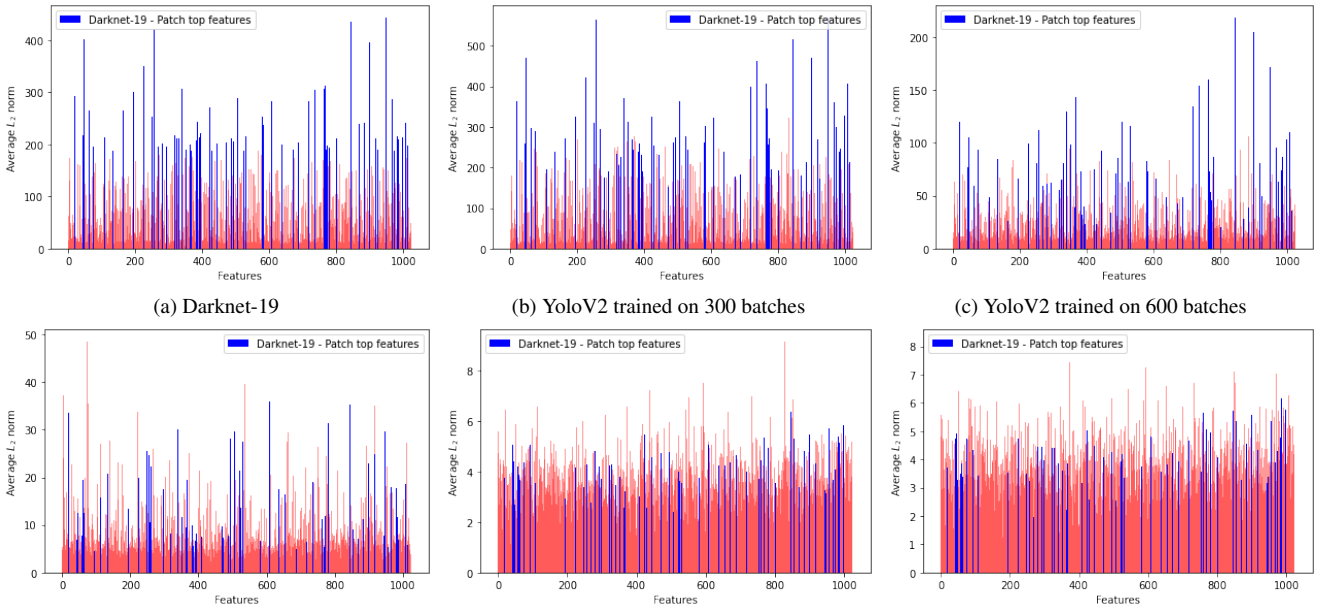
Figure 5. Impact map obtained by averaging the L_2-distance over features in cells. On the top row are represented maps for the different attacking procedures (patch or invisible) for the whitebox model (R50) and on the bottom row for the hidden model (R18).

3.4. Carpet-bombing patch

Seeing previous experiments, it seems that combining the best of the two worlds does not provide satisfying results: the conversion proposed in [15] into a patch attack (to remove additional requirements) does not inherit from [14, 15] property. Nevertheless, even though our patch attack provides only a moderate effect in black box setting, a second look at the noise-vs-patch experiments reveals some interesting properties.

First, our patch attack is much more powerful than noise based for modifying the feature map in white box setting: in this setting, the feature norm modification is eight times more intense with the patch than noise (in absolute value). This is why we call our patch a carpet bombing patch: it heavily modifies the targeted feature maps, eventually producing an output modification. Moreover, we observe that this heavy perturbation offers some interesting features:

- like most patch attacks, it does not need to know the target image, but even stronger, it can be designed from a proxy dataset;
- it does not need to know the underlying task;
- and as a patch attack, it does not need to access the pixel.



(a) Darknet-19 (b) YoloV2 trained on 300 batches (c) YoloV2 trained on 600 batches

(d) YoloV2 trained on 900 batches (e) YoloV2 trained on 30000 batches (f) YoloV2 final version

Figure 6. Patch impact evolution during the fine-tuning of Darknet-19 to YoloV2, from top-left to bottom-right. For each step, a bar chart of the average L_2-distance between cleaned and attacked features for the last layer of Darknet-19. The average is computed over 1000 images. Highlighted the top attacked features by patch attack designed to disrupt Darknet-19 last layer features.

To our knowledge, those three features have never been observed simultaneously: standard attacks rely on the task loss and even [15] which targets features only consider classification, and crafting a patch on proxy data source has never been explored before (of course, the two data sources can not be too different).

Despite combining network transferable attack [15] and patch attack [31] was not satisfying, we still obtained a threatening attack (described in 3.1) whose performances are detailed in the next section.

4. Experiments

In the previous section, we have shown a first contribution[2] by comparing the behaviour of *adversarial noise* and *adversarial shape*. Yet, the main contribution, oriented toward the safety community, is the design of a new adversarial patch attack with even fewer requirements than previous ones (as pointed in table 1). Numerous experiments are performed in this section highlighting that the proposed attack requires neither the underlying task, the target, or even the knowledge of the exact data source.

4.1. Datasets

We report our results for image classification task on the commonly-used dataset ImageNet [7], for object detection task on PASCAL VOC [9] and for semantic segmentation task on Cityscapes [6] which is a popular dataset for urban semantic segmentation.

Briefly, ImageNet is a set of 1M high resolution images (256x256 pixels) tagged with 1000 labels. Pascal VOC is a set of 50K large images (some above 512x512 pixels) containing a few objects from 10 classes. Finally, Cityscapes consists of a few thousand high-resolution images (1024×2048) taken from a car while driving (there are 2975 images for training and 500 for validation).

4.2. Eliminating the requirement of knowing the task

In this section, our goal is to design a patch capable of fooling multiple tasks without any underlying knowledge of them. In the following, we explain each task and the corresponding results.

4.2.1 Image classification

We design our patch to attack the backbone of the well-known ResNet-50 [11] from Pytorch Model Zoo. This model has been pretrained on ImageNet-1K [7]. We split the ImageNet-1K test set into training and test sets. The patch is fixed at the top-left corner i.e., pixel $(5, 5)$ with a dimension 50×50. We solve Equation (4) using the previously explained procedure (see Sec. 3.2) and choose to target only layer 4. We compare our attack to the well-known patch attack for classification [3].

[2] potentially interesting for all computer vision community

Task	Clean		SOTA attack		Ours	
Classification (Acc)	76.06		0.17 [3]		0.69	
Detection (mAP)	72.77		52.29 [31]		59.04	
Segmentation (mIOU/mAcc)	69.00	78.00	44.59	54.54 [25]	43.11	54.81

Table 3. Comparison of performance (%) under our attack and state-of-the-art task patch attacks for image classication, object detection and semantic segmentation. SOTA is different for each task, while our attack is unaware of the underlying task.

For this task, the model accuracy has dropped nearly to 0% (Tab 3). This result is impressive since our patch has been designed without any knowledge of the underlying task. It can deteriorate the feature map so the network can not exploit it.

4.2.2 Object detection

Following the methodology and the conditions used in [31], we compare our attack against their universal blindness attack, where both methods target YoloV2 architecture [28]. We sample from the PASCAL VOC [9] test dataset, two subsets of images that do not overlap with the patch. Each image is rescaled to 416×416 dimensions, and we fixed each patch to 100×100 dimensions at the top-left corner. Again, we solve Equation (4) using the previously mentioned procedure (Sec. 3.2). For the encoder part (f) of YoloV2 architecture (F), Darknet-19 is considered. We target the last layer of Darknet-19, i.e., $\mathbb{L} = \{L\}$. Once the patches are designed and learned by optimising the Equation (4) and [31], we evaluate them with the same emplacement as during the training phase. In evaluation setting, we set the confidence threshold of YoloV2 architecture to 0.0005, the Non-Maximum Suppression (NMS) to 0.45, and the Intersection Over Union (IOU) to 0.5.

The corresponding results are shown in Table 3. Our attacking performance is interesting because this performance is reached without introducing false alarms to the patch. The obtained result proves the weakness of object detectors, which indicates that the disruption of feature representations can influence the decision of a complicated task like detection.

4.2.3 Semantic segmentation

We compare the performance of the proposed patch against the recent state-of-the-art patch attacks designed for semantic segmentation task [25] and using Cityscapes dataset [6]. We use the same settings as in [25] to compare. For patch training, we randomly sample 250 images from the training set and to evaluate the impact of patches, we use the entire validation set. For the sake of comparison, we select BiSeNet [38], one of the state-of-the-art real-time semantic segmentation models. We target the two last layers of the Context Path module, i.e., $\mathbb{L} = \{L-1, L\}$. Since we are not using Expectation Over Transformation (EOT) [1], patches are placed at the middle part of images following [25] and have a dimensionality of 300×600 pixels. We use Adam optimiser with a learning rate of 0.5 and run the optimisation process over 200 epochs. The evaluation has been done with the same emplacement of image as during the training phase.

Table 3 shows that we obtain similar results to state-of-the-art segmentation attacks. Disrupting features of one module of the model seems to degrade the performance highly.

Task	Clean	ImageNet $\rightarrow \mathcal{D}$	PASCALVOC $\rightarrow \mathcal{D}$	Cityscapes $\rightarrow \mathcal{D}$
Classification (Acc) - R50	76.06	0.69	0.26	0.48
Detection (mAP) - YoloV2	72.77	64.33	59.04	63.10

Table 4. Comparison of performance (%) under our attack when the targeted dataset is not known. The targeted dataset $\mathcal{D}$ is ImageNet and PASCALVOC for classification and detection, respectively.

4.3. Removing the requirement of knowing the data distribution

Finally, we test whether or not our attack could rely on data from the targeted distribution. In a real-life scenario, a hacker would often have access to the underlying target model than to the data on which the model was built. We evaluate the impact of our attack when trained on a completely different dataset. We considered two tasks and three datasets. For image classification, our patch is built on PASCALVOC [9], or on Cityscapes [6] to sway R50 trained on ImageNet [7]. To target YoloV2 architecture [28] trained on PASCALVOC [9], we design our attack on ImageNet [7] or on Cityscapes [6]. We used the optimisation procedure described in section 3.2. Once patches are learned, we apply them to the data on which the model is trained. For clarity, we report the performance of models when attacked by patches designed on the targeted distribution.

We show impressive attacking results for both tasks (Tab 4). Our patch decreases the performance near to 0%, and by 9 % points for classification and detection, respectively. We demonstrate similar results for classification when the patch is directly learned on the targeted distribution. And for detection, mean average precision (mAP) falls significantly independently from the fact that the patch is designed on ImageNet, Cityscapes or PASCALVOC.

To the best of our knowledge, such a level of degradation without knowledge of the task, the target (or the dataset of the target) and without direct access to the image pixel has never been reported before.

5. Conclusion

This paper introduces a new evasion attack targeting deep networks, which can be crafted without access to targeted datum (or data distribution), targeted tasks and that could plausibly be produced in the real world. Such easily reproducible attack should be considered for safety reasons. Beyond the proposed attack, this paper also reports interesting experiments which highlight the difference between adversarial noise and adversarial patch. These results may interest more broadly than the strict attack-defence game and should be deeply studied in future works.

Acknowledgements

This work has been supported by the French government under the France 2030 program, as part of the SystemX Technological Research Institute.

References

[1] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. In *International conference on machine learning*, pages 284–293. PMLR, 2018. 8

[2] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 387–402. Springer, 2013. 1, 2

[3] Tom B Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial patch. *arXiv preprint arXiv:1712.09665*, 2017. 1, 2, 7, 8

[4] Shang Tse Chen, Cory Cornelius, Jason Martin, and Duen Horng Polo Chau. Shape shifter: Robust physical adversarial attack on faster r-cnn object detector: Recognizing outstanding. *D. Research*, 2019. 2

[5] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pages 1310–1320. PMLR, 2019. 1

[6] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. 7, 8

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 2, 4, 7, 8

[8] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018. 2, 4

[9] Mark Everingham, Andrew Zisserman, Christopher KI Williams, Luc Van Gool, Moray Allan, Christopher M Bishop, Olivier Chapelle, Navneet Dalal, Thomas Deselaers, Gyuri Dorkó, et al. The pascal visual object classes challenge 2007 (voc2007) results. 2008. 7, 8

[10] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 2

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 7

[12] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *International Conference on Machine Learning*, pages 2137–2146. PMLR, 2018. 2

[13] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. *Advances in neural information processing systems*, 32, 2019. 1

[14] Nathan Inkawhich, Kevin Liang, Binghui Wang, Matthew Inkawhich, Lawrence Carin, and Yiran Chen. Perturbing across the feature hierarchy to improve standard and strict blackbox attack transferability. *Advances in Neural Information Processing Systems*, 33:20791–20801, 2020. 2, 3, 6

[15] Nathan Inkawhich, Wei Wen, Hai Helen Li, and Yiran Chen. Feature space perturbations yield more transferable adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7066–7074, 2019. 1, 2, 3, 4, 6, 7

[16] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012. 1

[17] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016. 1

[18] Mark Lee and Zico Kolter. On physical adversarial patches for object detection. *arXiv preprint arXiv:1906.11897*, 2019. 2

[19] Xin Liu, Huanrui Yang, Ziwei Liu, Linghao Song, Hai Li, and Yiran Chen. Dpatch: An adversarial patch attack on object detectors. *SafeAI 2019 (AAAI Workshop on Artificial Intelligence Safety)*, 2018. 2

[20] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016. 2

[21] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 1

[22] Giulio Lovisotto, Nicole Finnie, Mauricio Munoz, Chaithanya Kumar Mummadi, and Jan Hendrik Metzen. Give me your attention: Dot-product attention considered harmful for adversarial patch robustness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15234–15243, 2022. 1

[23] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 1

[24] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1765–1773, 2017. 2, 4

[25] Federico Nesti, Giulio Rossolini, Saasha Nair, Alessandro Biondi, and Giorgio Buttazzo. Evaluating the robustness of semantic segmentation for autonomous driving against real-world adversarial patch attacks. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2280–2289, 2022. 2, 3, 4, 8

[26] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 427–436, 2015. 1

[27] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016. 2

[28] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017. 6, 8

[29] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. 1

[30] Andras Rozsa, Manuel Günther, and Terranee E Boult. Lots about attacking deep features. In *2017 IEEE International Joint Conference on Biometrics (IJCB)*, pages 168–176. IEEE, 2017. 2

[31] Aniruddha Saha, Akshayvarun Subramanya, Koninika Patil, and Hamed Pirsiavash. Role of spatial context in adversarial robustness for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 784–785, 2020. 2, 3, 4, 7, 8

[32] Dawn Song, Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Florian Tramer, Atul Prakash, and Tadayoshi Kohno. Physical adversarial examples for object detectors. In *12th USENIX workshop on offensive technologies (WOOT 18)*, 2018. 2

[33] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013. 1, 2

[34] Simen Thys, Wiebe Van Ranst, and Toon Goedemé. Fooling automated surveillance cameras: adversarial patches to attack person detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 0–0, 2019. 2

[35] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. *Advances in Neural Information Processing Systems*, 33:1633–1645, 2020. 1

[36] Jonathan Uesato, Brendan O'donoghue, Pushmeet Kohli, and Aaron Oord. Adversarial risk and the dangers of evaluating against weak attacks. In *International Conference on Machine Learning*, pages 5025–5034. PMLR, 2018. 2

[37] Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L Yuille. Improving transferability of adversarial examples with input diversity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2730–2739, 2019. 2

[38] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 325–341, 2018. 8

[39] Wen Zhou, Xin Hou, Yongjun Chen, Mengyun Tang, Xiangqi Huang, Xiang Gan, and Yong Yang. Transferable adversarial perturbations. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 452–467, 2018. 2