SentiNet: Detecting Localized Universal Attacks Against Deep Learning Systems

***Abstract*—SentiNet is a novel detection framework for *localized universal attacks* on neural networks. These attacks restrict adversarial noise to contiguous portions of an image and are reusable with different images—constraints that prove useful for generating physically-realizable attacks. Unlike most other works on adversarial detection, SentiNet does not require training a model or preknowledge of an attack prior to detection. Our approach is appealing due to the large number of possible mechanisms and attack-vectors that an attack-specific defense would have to consider. By leveraging the neural network's susceptibility to attacks and by using techniques from model interpretability and object detection as detection mechanisms, SentiNet turns a weakness of a model into a strength. We demonstrate the effectiveness of SentiNet on three different attacks—i.e., data poisoning attacks, trojaned networks, and adversarial patches (including physically realizable attacks)—and show that our defense is able to achieve very competitive performance metrics for all three threats. Finally, we show that SentiNet is robust against strong adaptive adversaries, who build adversarial patches that specifically target the components of SentiNet's architecture.**

I. Introduction

Deep neural networks are susceptible to a variety of attacks aimed at causing misclassifications [28], [14], [4]. This has severe implications for the trustworthiness of deep learning models in security-critical decision-making situations. Defending against these attacks is challenging due to the wide variety of possible attack mechanisms and vectors, especially for models operating in the visual domain. In this work, we explore *localized* and *universal* attacks on visual classifiers and introduce SentiNet, a robust defense which detects adversarial inputs without requiring any specific model re-training or prior knowledge of the attack vector.

We focus on attacks that are *localized*, i.e., the adversarial region is constrained to a small contiguous portion of an image, and *universal*, i.e., attacks are image-agnostic. Inputs with these properties have proven useful for instantiating robust and physically realizable attacks that take the form of an adversarial object or sticker placed inside a visual scene [4], [21], [9], [10]. These classes of attacks typically use *unbounded* perturbations (i.e., without any specific ℓ_∞ or ℓ_2 constraint as in most digital attacks [28], [14]), to ensure that the attacks are robust to changes in viewpoint, lighting and other physical artifacts. Several such physical attacks have been demonstrated, with the goal of causing misclassifications when applied to arbitrary images with different class labels [4], [21]. A drawback of localized attacks is that they are generally visible and detectable by the human eye, but there are many situations where attacks can be deployed in autonomous settings or carefully disguised [4].

Our goal is to create a defense against localized universal attacks that is attack-vector agnostic. To this end, we identify unifying and necessary features of successful localized universal attacks—including physically realizable attacks—and develop SentiNet, a novel technique that exploits these attack behaviors to detect them. We start from the observation that localized universal attacks are designed to be robust to a variety of artifacts while generalizing to a large distribution of inputs (e.g., the adversarial patch of [4] is designed to work when applied to *any* input image). Our first insight is that a localized universal attacks' success relies on the use of "salient" features that strongly affect the model's classification on many different inputs. We thus consider techniques from *model interpretability* and *object detection* to discover highly salient contiguous regions of an input image. As we show, these techniques uncover adversarial image regions, as well as benign ones that strongly affect classification. In a second step, we exploit the strong robustness and generalization properties of malicious regions to distinguish them from benign regions with high saliency. Specifically, we overlay extracted regions on a large number of held-out clean images and test how often this results in a misclassification. Malicious regions are much more likely than benign regions to generate misclassifications and are thus detected by SentiNet. As we show in our evaluation of SentiNet, mounting an attack that evades detection requires lowering an adversarial region's saliency to a point where the region is no longer universal (i.e., it fails with high probability on random inputs)—even for a adaptive adversary with knowledge of SentiNet's architecture. We also validate SentiNet's effectiveness in a realistic physical setting, where it successfully detects a printed adversarial patch [4] with high reliability.

Contributions: To summarize, this paper makes the following contributions:

- We propose SentiNet[1], a new architecture that protects a neural network by using the same model to detect localized universal attacks.
- SentiNet uses a novel approach to detect a potential attack region using techniques developed for model visualization and object detection and feeds the attack deployed on multiple test images back to the network to perform attack classification.
- We evaluate SentiNet to protect networks against multiple attack vectors. While we primarily focus on adversarial patches, an extended version of our work [1] also shows SentiNet's effectiveness in detecting backdoors in poisoned or trojaned networks.

[1]Reviewers can inspect anonymously the code of SentiNet here https://www.dropbox.com/s/tc8oiyhhuq4oym3/sentinet.zip. We will publish the source code and datasets on GitHub once the peer review is concluded.

Fig. 1: Localized universal attack with an adversarial patch [4].

- We further evaluate SentiNet against a fully adaptive white-box adversary and present seven attacks against SentiNet's core components. We show that SentiNet is resistant against strong adversaries by demonstrating the robustness of each individual component.

II. Background

A. Threat Model

In this work, we assume a scenario where a DL system uses a pre-trained deep CNN model to classify sensor data. The goal of the adversary is to hijack the prediction of the model by providing a malicious image to the CNN model. In Section II-A1, we present the properties of the malicious inputs considered in this paper, and in Section II-A2, we present existing techniques to mount the attacks. Additionally, we assume an adversary that is aware of the defense mechanisms, and that can generate malicious images to bypass SentiNet. In Section II-A1, we present such an adversary.

1) Localized Universal Attacks: The goal of the adversary is to hijack the prediction of the model to gain control of the actions performed by the DL system by crafting a malicious object and placing it in the input image, e.g., inside the visual scene of the sensors. The malicious objects considered in this paper are *localized*, i.e., the object is constrained to a small contiguous region of the image. The objects of this paper are also *universal*, i.e., the object is created once and reused many times with different input images. Figure 1 shows an example of such universal, localized malicious objects.

2) Attack Vectors: The adversary can construct and use malicious objects in different ways, that we review in this section. For example, the adversary can "alter" the behavior of the network to respond to a malicious object before its deployment in a DL system. Examples of these attacks are trojaning [19], [21] or poisoning attacks [16], which we describe and evaluate in more detail in an extended version of this work [1]. Here, we focus on an adversary that causes misclassifications to uncompromised models by crafting ad-hoc malicious objects, e.g., adversarial patches [26], [20], [4], [9], [10], [3]. More specifically we consider an adversary that makes use of localized universal malicious objects, e.g., printed patches [26], [4], [10], [9] or 3D objects [3] (see, e.g., Figure 1), that can fool a model under real-world conditions such as lighting, sensor noise, and rotation.

3) Adaptive Adversary: Over the past years, many ideas have been proposed to protect neural networks from attacks. While some new ideas have moderately increased the robustness of neural networks (e.g., strong adversarial training [23]), many do not adequately protect networks against adversaries aware of the specific defense mechanism being used (see, e.g., [6], [2]). Accordingly, in this paper, we assume a strong white-box adversary that is fully aware of SentiNet, its architecture and mechanisms.

III. SentiNet

In this section, we present SentiNet. The goal of SentiNet is to identify adversarial inputs that will hijack the prediction of the neural network without assuming the knowledge of the attack vector beforehand, e.g., exploiting vulnerabilities of compromised and uncompromised networks. The core insight of SentiNet is to use the very behavior of adversarial misclassification to detect an attack.

The architecture of SentiNet is shown in Figure 2. First, SentiNet uses techniques from model interpretability and object detection to extract from an input scene x those regions that most highly influence the model prediction y (Section III-A). These regions likely contain the malicious object (if present) as well as benign salient regions. Then, SentiNet applies these extracted regions on a set of benign test inputs and observes the behavior of the model. Finally, SentiNet compares the synthetic behaviors with the known behavior of the model on benign inputs, to detect prediction hijacking (Section III-B).

A. Adversarial Object Localization

The first phase of our approach intends to localize on the given input the regions that might contain malicious objects. The idea is to identify the parts of the input x that contribute to the model prediction y. Because the attack is localized, we can hope to recover the true class of input x if we evaluate the model on a segmented input that contains no part of the attack. In the following, we look into the details of this phase. First, we present a segmentation-based approach to propose classes (see Figure 2.a). Then, starting from proposed classes and the given input, we generate a mask for x that may contain the malicious object (see Figure 2.b).

1) Class Proposal via Segmentation: The detection of the attack begins with the identification of a set of possible classes that may be predicted by the model f_m. The first of such classes is the actual prediction, i.e., $y = f_m(x)$. The other classes are identified by segmenting the input x and then evaluating the network on each segment. Algorithm 1 shows the algorithm to propose classes via input segmentation. Different approaches can be used to segment a given input x including sliding windows and network-based region proposals [25]. In our approach, we use the selective search image segmentation algorithm [29]. Selective search generates an exhaustive list of region proposals based on the patterns and edges found in natural scenes [29]. Then, we evaluate each proposed segment, i.e., $f(x_p)$, and return the k most confident predictions, where k is a configuration parameter of SentiNet. We exclude the primary class $y = f_m(x)$ from our choice of k classes. A general guideline towards selecting k is to set it to be slightly higher than the amount of classes that are present in the dataset per-image. In our case, Imagenet [8] has around 1.5 classes per image, and we set k to 2.

2) Mask Generation: Once the class proposal C is obtained, the second step of SentiNet consists in identifying the regions of x that most highly influence the predictions

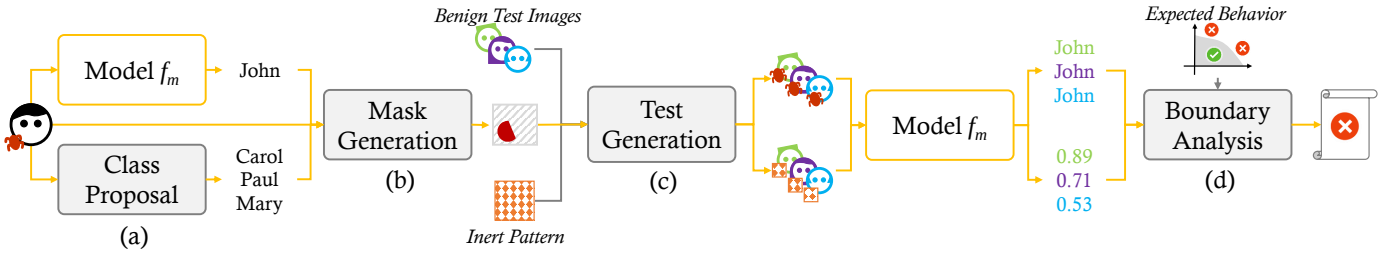


Fig. 2: Overview of the SentiNet architecture to protect a model f_m. The output and class proposals of an input are used to generate masks and image tests, which are then fed back into f_m to generate values for boundary analysis and attack classification.

Algorithm 1: ClassProposal

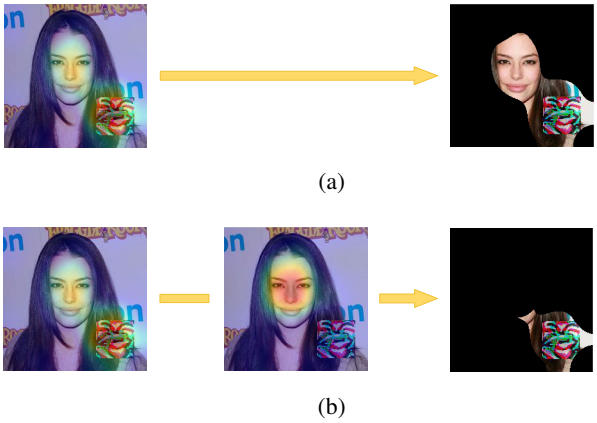
in : f_m – model;
x – input of f_m;
y – primary class, i.e., $y = f_m(x)$;
k – propositions
out : C - set of proposed classes and confidence ($|C| = k$)

P = `SelectiveSearch`(x);
$C = \{(y_p, conf_p) : \forall x_p \in P, (y_p, conf_p) = f_m(x_p) \wedge y_p \neq y\}$;
C = `TopConf`(C, k);
return C

C. To find these regions, we resort to techniques to explain and interpret model predictions.

A particularly suitable approach for our goal is Grad-CAM [7], a model-interpretation technique that identifies contiguous spatial regions of an input without requiring modifications to the original model. At a high level, Grad-CAM uses gradients computed in the final layers of a network to calculate the saliency of input regions. For class c, Grad-CAM calculates the gradients of the model's output y^c (the model's *logit* score for class c) with respect to each of the k feature maps A^k of the model's final pooling layer to obtain $\frac{\delta y^c}{\delta A^k}$. The mean gradient value of each filter map, or "neuron importance weight", is denoted $\alpha_c^k := \frac{1}{Z}\Sigma_i\Sigma_j \frac{\delta y^c}{\delta A^k}$. Finally, the feature maps A^k are weighted by their neuron importance and aggregated to obtain the final Grad-CAM output: $L^c_{Grad-CAM} := ReLU(\Sigma_k \alpha_c^k A^k)$. Here, $ReLU(x) = \max(x, 0)$ is the ReLU activation function [12] which retains only the positive gradient signals for class c. The output of Grad-CAM is a coarse heatmap of the positive importance of the image, usually at a lower resolution that the input image due to downsampling in the model's convolutional and pooling layers. Finally, masks are produced by binarizing the heatmap with a threshold of 15% of max intensity. We use this mask to segment salient regions for the next steps.

We generate masks with Grad-CAM as shown in Algorithm 2. We start by extracting the mask using Grad-CAM for the input x and prediction y. As Grad-CAM can identify salient areas for benign classes, the resulting mask may span over both benign and malicious regions. To improve the accuracy of the mask, we also extract a mask for each proposed class y_p. Then, we use these additional heatmaps to generate secondary masks to improve the original mask for the prediction y by subtracting common region, resulting in a set of masks that highlight only the localized attack and not the other salient regions.



(a)

(b)

Fig. 3: Differential mask generation to remove ambiguous areas. **(a)** Example of Grad-CAM mask using prediction y covering benign areas. **(b)** Example of precise mask by removing masks of class proposals C.

Algorithm 2: MaskGeneration

in : f_m – model;
x – input for f_m;
y, $conf$ – model prediction on x;
C – proposed classes
out : M - masks for candidate regions with the malicious object

$mask_y$ = `MaskGradCAM`(f_m, x, y);
$M = \{(mask_y - \texttt{MaskGradCAM}(f_m, x, y_p), conf_p)$
$: (y_p, conf_p) \in C\}$;
return $\{mask_y\} \cup M$

B. Attack Detection

Once we identified regions M of the inputs x that may be containing malicious inputs, we test the model f_m to determine whether any of the regions can hijack the expected predictions of a set of benign test inputs.

1) Test Generation: Once an input region is localized, SentiNet observes the effects the region has on the model to determine whether the region is adversarial or benign. To do so, SentiNet overlays the suspected region on a set of benign test images X, which are often shipped together with deployed models. Test images X are fed back into the network, where the number of fooled examples are counted. Intuitively, the more often an overlaid region causes a change in classification, the more likely the region is adversarial.

When the recovered region is small, this technique is

effective at distinguishing adversarial and benign inputs, as small benign objects cannot typically overwhelm a network's prediction. However, one problem of this approach is that a region (whether adversarial or benign) that covers a larger fraction of the input image will likely cause misclassifications when overlaid onto most other images because they occlude the original object.

To measure the extent to which an overlaid region causes misclassifications simply by occluding the original object, rather than by virtue of highly-salient features, we perform a second test where we *replace* the content of the overlaid region with an inert pattern of low saliency (e.g., Gaussian noise). We expect that adversarial regions will cause many misclassifications when overlaid on the test set, but have little effect on the network when replaced by an inert pattern before being overlaid. In contrast, we expect benign patterns to either cause few misclassifications, or to occlude objects and thus also disrupt the model when replaced by inert patterns. Algorithm 3 below shows the algorithm we use to test extracted regions. For each region, we check how many overlaid test images fooled the network, and how confident the model is when classifying images overlaid with inert patterns.

Algorithm 3: Testing

in : f_m – model;
x – input for f_m;
y – class of x;
M – proposed masks;
X – bening test images;
out : $Fooled$, $AvgConf$

$R = \{x * mask : mask \in M\}$;
IP = `InertPattern` (M);
X_R = `Overlay` (X, R);
X_{IP} = `Overlay` (X, IP);
$fooled_{y_R} = 0$, $avg_{conf_{IP}} = 0$;
for $x_R, x_{IP} \in X_R, X_{IP}$ **do**
  $(y_R, conf_R), (y_{IP}, conf_{IP}) = f(x_R), f(x_{IP})$;
  if $y_R == y$ **then**
    $fooled_{y_R}$ += 1
  $avg_{conf_{IP}}$ += $conf_{IP}$
$avg_{conf_{IP}} = \frac{avg_{conf_{IP}}}{|X|}$;
return $fooled_{y_R}, avg_{conf_{IP}}$;

2) Decision Boundary for Detection: With these two metrics (number of images fooled and average inert pattern confidence values) we can determine whether an input x is adversarial. A naive approach is to use thresholding based rules, but it is hard to determine how to set the thresholds and which metric holds more importance. Instead, we use metrics collected on clean examples to train a simple two-feature one-class classifier and classify outliers as adversarial.

IV. Evaluation

To demonstrate SentiNet's versatility, we evaluated its performance against a variety of existing attacks including poisoned networks, trojan triggers, and adversarial patches. For lack of space, we refer the interested reader to an extended version of our paper [1] for the results of this evaluation, in which we found that SentiNet attained an average true positive rate of 96.22% and an average true negative rate of 95.36% across a variety of attack scenarios.

V. Adaptive Attacks

Here, we focus on evaluating the more challenging and realistic threat model of a fully adaptive white-box adversary [5] that is aware of the presence of SentiNet, of its architecture and its inner workings. Such an adversary can attempt to create adversarial objects that simultaneously fool the target network while also compromising the core components of SentiNet, i.e., the mask generation, the class proposal, and the attack detection. In Section V-A, we present attacks aimed at fooling the mask generation phase. Then, in Section V-B, we evaluate attacks against the class proposal. Finally, in Section V-C, we present attacks against the attack detection phase.

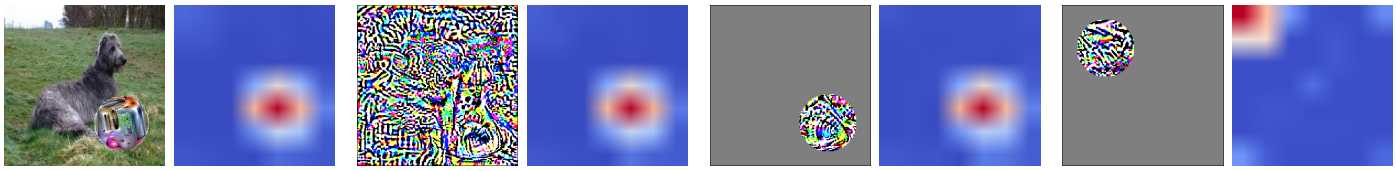
We evaluate SentiNet protecting an uncompromised network when processing adversarial patches. In our case, we use a VGG-16 [27] Imagenet-pretrained [8] network as our target network f_m.

A. Attacking the Region Proposals

Our defense is reliant on successfully localizing the adversarial region in an image. In our current framework, this is done using the Grad-CAM algorithm, which generates a heatmap of the salient regions leading to a classification. If an attacker can disrupt the Grad-CAM mechanism and avoid successful detection and localization, the subsequent components of the pipeline will fail. We observe that the Grad-CAM mechanism uses network back-propagation to measure region importance. As this mechanism is differentiable, an adversary could generate targeted gradient perturbations to control the position and size of the heatmap. In this section, we consider three of these attacks: Grad-CAM perturbation, misdirection of the mask creation, and mask minimization. As the analysis of this section shows, while manipulating the heatmap is feasible in theory, in our defense context heatmap manipulation is ineffective in practice.

1) Perturbing Grad-CAM: Consider the adversarial image and its Grad-CAM heatmap in Figure 4a for the target class "toaster". The heatmap in Figure 4a correctly identifies the area containing the patch. An adversary could try to generate a perturbation, i.e., an adversarial image, resulting in a visually identical heatmap. Such a perturbation can be created in practice as the Grad-CAM function $L^c_{GradCAM} = ReLU(\Sigma_k \alpha^k_c A^k)$ is differentiable, and the adversary can optimize an input on this function given a target class. For example, the adversary can use a standard Stochastic Gradient Descent Optimizer (SGD) on a VGG-16 network [27] to minimize a loss function calculated as the total difference between the current Grad-CAM output and the target Grad-CAM output, and iteratively add noise until the loss converges. An example of such a perturbation is shown in Figure 4b. This attack shows that Grad-CAM outputs can be precisely manipulated through gradient optimization. However, such an attack requires to feed the network with a perturbation of the size of the input of the network being protected, which is not part of our threat model (Section II-A).

2) Heatmap Misdirection: In this attack, we consider an adversary who intends to create an input image that tricks SentiNet to detect the adversarial object in a different region where the adversarial object is located. Let us assume the adversary intends to convince SentiNet that the adversarial



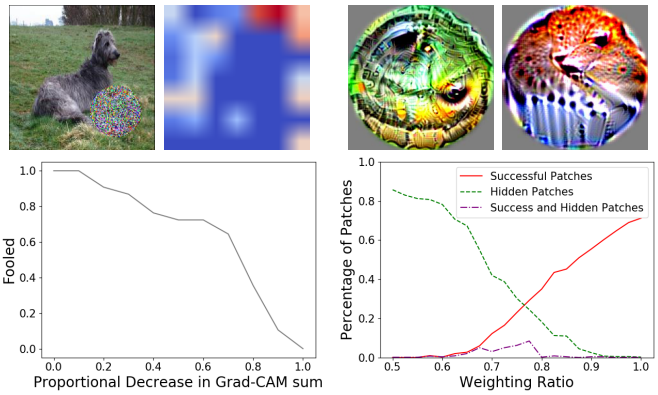
(a) Heatmap of the adv. patch (b) Full image perturbation (c) Localized perturbation (d) Misdirection

Fig. 4: Attacking region proposal. **(a)** The heatmap of the adversarial patch overlaid on an image of a dog. **(b)** Full-image perturbations can generate a visually identical heatmap. **(c)** Localized perturbation can generate the same heatmap of (a). **(d)** Localized perturbation fails to generate a similar heatmap.

object is in the position as indicated by the heatmap of Figure 4a. As we demonstrated earlier, an adversary allowed to add perturbational noise to an entire image can generate a heatmap at the desired position, but the adversary does not have full control of the whole image, rendering the attack infeasible. An attacker may try to generate a heatmap visually identical to the adversarial patch of Figure 4a by using a perturbation constrained to the same region. Figure 4c shows an example of such a perturbation obtained by modifying the loss function of the SGD to constrain the noise on the same region of interest. At this point, the adversary can try to generate a perturbation at a different location leaving the goal of the attack unchanged. An example of such an attempt is shown in Figure 4d. The perturbation is now on the top-left corner of the input image. However, the resulting heatmap is not shown in the desired position, i.e., bottom-right. Instead, the heatmap is positioned nearby the location of the perturbation, indicating that localized noise can only affect the corresponding Grad-CAM region.

3) Heatmap Minimization: An alternative strategy to fool SentiNet is to minimize the corresponding Grad-CAM region to the greatest extent possible to avoid detection. For example, the adversary can achieve that by modifying the loss function of SGD to optimize for minimal Grad-CAM output. The adversary can start from the adversarial patch image in Figure 5a and iteratively add perturbational noise to the region. Figure 5a, we show that as our loss converges, the Grad-CAM output is successfully minimized, avoiding detection of the patch. Our analysis shows that heatmap minimization, i.e., fooling SentiNet, is in tension with the attack success rate of the patch, i.e., fooling f_m. Figure 5a shows that as more noise is added to the patch, the success rate of the attack is reduced. This result suggests that Grad-CAM is capturing some inherent saliency of the region and cannot be minimized without reducing the attack effectiveness. The tension between the two objectives could be solved by creating a perturbation that optimizes both for targeted misclassification and for heatmap minimization. Let us consider an adversary that attempts to jointly generate a patch for both criteria by optimizing both objectives during every iteration. Also, to better explore the role played by each objective, we introduce a weighting parameter $\alpha \in [0, 1]$ between the two objectives (i.e., the misclassification objective has weight α, and the heatmap minimization has weight $1-\alpha$). To evaluate then attack, we generate 40 adversarial patches as described by Brown at al. [4] for a range of weighting values $\alpha \in [0, 1]$. For each patch, we select 100 images from the Imagenet test set [8] and apply the patch. In total, we generated 4,000 malicious images for each weighting ratio α. Then, we counted the patches that fool the model, the Grad-CAM[2], or both.

The results of our experiment are shown in Figure 5b. As patches work with slight variance, we averaged the attack result to give a better sense of how well a patch would perform. As can be seen in Figure 5b, there is an inverse relationship between the effectiveness of a patch in fooling f_m (in red) and the region proposal of SentiNet (dashed green), corroborating our previous findings. Additionally, we further plot the patches that successfully fulfill both criteria (in dashed purple), and find that the optimum value is where the ratios of successful misclassifications and hidden patches overlap, giving us patches that fulfill both criteria with an average success rate of 10%. This strongly suggests that minimizing the Grad-CAM output directly weakens the attack effectiveness and that it is difficult to optimize for both objectives jointly.



(a) Heatmap minimization (b) Minimization and attack

Fig. 5: Heatmap minimization attack. **(a)** Example of adversarial input minimizing the Grad-CAM output (top); Plot showing the tension between minimization and attack success. **(b)** Patches minimizing the heatmap and maximizing the attack success respectively (top); Plot of success of the patch vs. minimization of the heatmap.

4) Results on Attacking Region Proposals: To summarize, in this section, we presented three attacks against the Grad-CAM algorithm. Our evaluation shows that, while the Grad-CAM output can be perturbed with localized perturbations, misredirecting the Grad-CAM heatmap seems an infeasible

[2]We use a < 0.5 overlap ratio between the adversarial pixels and the Grad-CAM region.

attack strategy. Also, our results show that minimizing the heatmap size dramatically reduces the effectiveness of the attack. Accordingly, we conclude that attacking the Grad-CAM algorithm may bring a very little advantage to an adversary.

B. Attacking the Class Proposal

Another approach to avoid the localization of the adversarial region is attacking the class proposal step. In this section, we discuss attacks exploiting the algorithms implemented in the class proposal step. In particular, we concentrate on two types of attacks where the adversary intends avoiding detection by generating masks without the adversarial region or weakening the detection by creating inefficient masks.

1) Mask Reduction with Specialized Patch Sub-regions: Instead of avoiding detection, the adversary may try to weaken the detection of the patch by reducing the size of the mask of the primary class. The goal of the class proposal is to identify additional classes in the input image to remove ambiguous areas (see Figure 3a-b). Here, the attacker may try tricking the class proposal into identifying classes in sub-regions of the patch. Such an attack results in a mask containing only a portion of the entire patch, thus weakening the response of the model at test time and the detection of the attack. The creation of a localized universal patch exposing such behavior may be very challenging. Such a patch needs to have a region causing the activation of a class, say y', and, at the same time, another region for the target class y. Also, to carry this attack, the patch should cause a change in the prediction of the model when the region for y' is missing. Building a localized universal patch with such a non-linear dependency between the two regions may be hard, if not unfeasible, and we hope other researchers will explore whether such a type of attack is possible.

2) Avoiding Mask Reduction by Over-Segmentation: The adversary might attack the image segmentation algorithm, i.e., selective search [29], and trick the algorithm into returning many small segments outside of the malicious region. When processing small segments, the network may return predictions with low confidence which are ignored when generating the mask for the primary class (see `TopConf(`C`,` k`)` in Algorithm 1). As a result, the final mask will not be refined by removing the ambiguous areas thus weakening the response of the model at test time. However, attacking the selective search algorithm may not be a feasible strategy. Our inspection of the selective search algorithm revealed that an adversary controlling a limited region of the input image may not be able to influence the generation of a segment of arbitrary size and position. As a result, we believe that attacking the selective search algorithm may be unlikely.

C. Attacking the Attack Classification

We now consider attacks aimed at fooling the decision procedure of SentiNet, where extracted regions or inert patterns are overlaid onto benign test images to measure their generalizability. One possible attack in the context of poisoned or trojaned networks has the adversary manipulate the model so that it recognizes the inert pattern as a member of a targeted class. This would let the attacker bypass our defense as adversarial patterns will remain adversarial when overlaid with the "inert" pattern. This attack does not apply to the setting we consider in this short paper, where an adversary aims to evade an uncompromised network with adversarial patches. For the general case considered in an extended version of our work [1], we note that the choice of inert pattern can be made after the network has been trained, so one can test whether the chosen pattern affects the model. We also found that using patterns made up of random noise produces good results, which further hinders a training-time adversary's ability to target the specific pattern than will later be used at test-time.

D. Attack Size

The goal of SentiNet is to capture unreasonably salient attacks, designed to be small and unnoticeable. With this measure, SentiNet largely succeeds at detecting abnormal regions of images that deviate from patches of natural images, but not when modifying the majority of pixels in an image. As noted by Brown et al. [4], large images of an actual toaster will hijack the same prediction of an adversarial patch at a large enough size, raising an interesting question about what actually constitutes an "attack".

VI. Related Work

Detection techniques for attacks as a defensive measure have been proposed by many researchers. Safetynets [22] is designed to detect adversarial-noise based attacks and exploits the different activations adversarial perturbations produce to train a SVM classifier. Metzen et al. [17] use a similar approach by training a modified target classification network to detect adversarial perturbations. Feinman et al. [11] also trains a classifier to detect adversarial perturbational inputs based on the neural network features, while Gong et al. [13] introduces a classifier trained to detect adversarially-perturbed images. Magnet [24] trains a classifier on manifolds of normal examples to detect adversarial perturbations without prior knowledge of the attack. There are also some works designed at creating defenses that do not require training. Grosse et al. [15] uses statistical techniques to distinguish adversarial-perturbations outputs, while Hendrycks et al. [18] uses PCA to visualize differences in perturbed images. A survey by Yuan et al. [30] covers further detection defenses. All these works are only aimed at defending against adversarial perturbations whereas SentiNet can defend a network against other types of attacks, i.e., data poisoning and trojaning attacks.

VII. Conclusion

In this work, we introduce SentiNet, a framework for detecting localized universal attacks on Convolutional Neural Networks. Our method is notable because it only relies on the malicious behavior of an adversarial attack to perform classifications, without requiring prior knowledge of the attack vector. We further evaluate the robustness of SentiNet against strong adaptive adversaries by individually testing each component of our defense. We hope SentiNet inspires further approaches towards creating attack-agnostic defenses. Tailoring a defense towards a specific attack means unknown attacks cannot be captured, and also makes the system highly vulnerable to strong adaptive adversaries. We believe a similar approach can be used to detect other attacks by leveraging the same core concepts of identifying an attack from a model's weakness.

REFERENCES

[1] Anonymized for submission, "Sentinet: Detecting localized universal attacks against deep learning systems," https://www.dropbox.com/s/pr46pu4hqq4bphu/sentinet.pdf?dl=0, 2019.

[2] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," *arXiv preprint arXiv:1802.00420*, 2018.

[3] A. Athalye and I. Sutskever, "Synthesizing robust adversarial examples," *arXiv preprint arXiv:1707.07397*, 2017.

[4] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer, "Adversarial patch," *CoRR*, vol. abs/1712.09665, 2017. [Online]. Available: http://arxiv.org/abs/1712.09665

[5] N. Carlini, A. Athalye, N. Papernot, W. Brendel, J. Rauber, D. Tsipras, I. Goodfellow, and A. Madry, "On evaluating adversarial robustness," *arXiv preprint arXiv:1902.06705*, 2019.

[6] N. Carlini and D. Wagner, "Adversarial examples are not easily detected: Bypassing ten detection methods," in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. ACM, 2017, pp. 3–14.

[7] A. Chattopadhyay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, "Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks," *CoRR*, vol. abs/1710.11063, 2017. [Online]. Available: http://arxiv.org/abs/1710.11063

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[9] I. Evtimov, K. Eykholt, E. Fernandes, T. Kohno, B. Li, A. Prakash, A. Rahmati, and D. Song, "Robust physical-world attacks on machine learning models," *CoRR*, vol. abs/1707.08945, 2017. [Online]. Available: http://arxiv.org/abs/1707.08945

[10] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, F. Tramer, A. Prakash, T. Kohno, and D. Song, "Physical adversarial examples for object detectors," *arXiv preprint arXiv:1807.07769*, 2018.

[11] R. Feinman, R. R. Curtin, S. Shintre, and A. B. Gardner, "Detecting Adversarial Samples from Artifacts," *ArXiv e-prints*, Mar. 2017.

[12] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *AISTATS*, 2011.

[13] Z. Gong, W. Wang, and W. Ku, "Adversarial and clean data are not twins," *CoRR*, vol. abs/1704.04960, 2017. [Online]. Available: http://arxiv.org/abs/1704.04960

[14] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and Harnessing Adversarial Examples," *ArXiv e-prints*, Dec. 2014.

[15] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. D. McDaniel, "On the (statistical) detection of adversarial examples," *CoRR*, vol. abs/1702.06280, 2017. [Online]. Available: http://arxiv.org/abs/1702.06280

[16] T. Gu, B. Dolan-Gavitt, and S. Garg, "Badnets: Identifying vulnerabilities in the machine learning model supply chain," *CoRR*, vol. abs/1708.06733, 2017. [Online]. Available: http://arxiv.org/abs/1708.06733

[17] J. Hendrik Metzen, T. Genewein, V. Fischer, and B. Bischoff, "On Detecting Adversarial Perturbations," *ArXiv e-prints*, Feb. 2017.

[18] D. Hendrycks and K. Gimpel, "Early Methods for Detecting Adversarial Images," *ArXiv e-prints*, Aug. 2016.

[19] Y. Ji, X. Zhang, S. Ji, X. Luo, and T. Wang, "Model-reuse attacks on deep learning systems," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2018, pp. 349–363.

[20] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," *arXiv preprint arXiv:1607.02533*, 2016.

[21] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, and X. Zhang, "Trojaning attack on neural networks," in *NDSS*, 2018.

[22] J. Lu, T. Issaranon, and D. A. Forsyth, "Safetynet: Detecting and rejecting adversarial examples robustly," *CoRR*, vol. abs/1704.00103, 2017. [Online]. Available: http://arxiv.org/abs/1704.00103

[23] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=rJzIBfZAb

[24] D. Meng and H. Chen, "Magnet: a two-pronged defense against adversarial examples," *CoRR*, vol. abs/1705.09064, 2017. [Online]. Available: http://arxiv.org/abs/1705.09064

[25] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *CoRR*, vol. abs/1506.01497, 2015. [Online]. Available: http://arxiv.org/abs/1506.01497

[26] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. New York, NY, USA: ACM, 2016, pp. 1528–1540. [Online]. Available: http://doi.acm.org/10.1145/2976749.2978392

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

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

[29] J. R. Uijlings, K. E. Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *Int. J. Comput. Vision*, vol. 104, no. 2, pp. 154–171, Sep. 2013. [Online]. Available: http://dx.doi.org/10.1007/s11263-013-0620-5

[30] X. Yuan, P. He, Q. Zhu, R. R. Bhat, and X. Li, "Adversarial examples: Attacks and defenses for deep learning," *CoRR*, vol. abs/1712.07107, 2017. [Online]. Available: http://arxiv.org/abs/1712.07107