Connecting the Dots: Detecting Adversarial Perturbations Using Context Inconsistency

Shasha Li[1], Shitong Zhu[1], Sudipta Paul[1],
Amit Roy-Chowdhury[1], Chengyu Song[1], Srikanth Krishnamurthy[1],
Ananthram Swami[2], and Kevin S Chan[2]

[1] University of California, Riverside, USA
{sli057, szhu014, spaul007}@ucr.edu,
{amitrc}@ece.ucr.edu, {csong, krish}@cs.ucr.edu
[2] US Army CCDC Army Research Lab
{ananthram.swami.civ, kevin.s.chan.civ}@mail.mil

Abstract. There has been a recent surge in research on adversarial perturbations that defeat Deep Neural Networks (DNNs) in machine vision; most of these perturbation-based attacks target object classifiers. Inspired by the observation that humans are able to recognize objects that appear out of place in a scene or along with other unlikely objects, we augment the DNN with a system that learns context consistency rules during training and checks for the violations of the same during testing. Our approach builds a set of auto-encoders, one for each object class, appropriately trained so as to output a discrepancy between the input and output if an added adversarial perturbation violates context consistency rules. Experiments on PASCAL VOC and MS COCO show that our method effectively detects various adversarial attacks and achieves high ROC-AUC (over 0.95 in most cases); this corresponds to over 20% improvement over a state-of-the-art context-agnostic method.

Keywords: object detection, adversarial perturbation, context

1 Introduction

Recent studies have shown that Deep Neural Networks (DNNs), which are the state-of-the-art tools for a wide range of tasks [10, 18, 25, 37, 44], are vulnerable to adversarial perturbation attacks [28, 49]. In the visual domain, such adversarial perturbations can be digital or physical. The former refers to adding (quasi-) imperceptible digital noises to an image to cause a DNN to misclassify an object in the image; the latter refers to physically altering an object so that the captured image of that object is misclassified. In general, adversarial perturbations are not readily noticeable by humans, but cause the machine to fail at its task.

To defend against such attacks, our observation is that the misclassification caused by adversarial perturbations is often *out-of-context*. To illustrate, consider the traffic crossing scene in Fig. 1; a stop sign often co-exists with a stop line, zebra crossing, street nameplate and other characteristics of a road intersection.

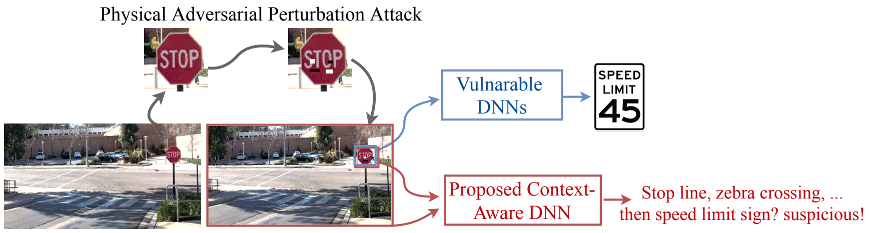

Physical Adversarial Perturbation Attack
Vulnerable DNNs
SPEED LIMIT 45
Proposed Context-Aware DNN
Stop line, zebra crossing, ...
then speed limit sign? suspicious!

Fig. 1. An example of how our proposed context-aware defense mechanism works. Previous studies [13, 43] have shown how small alterations (graffiti, patches etc.) to a stop sign make a vulnerable DNN classify it as a speed limit. We posit that a stop sign exists within the wider context of a scene (e.g., zebra crossing which is usually not seen with a speed limit sign). Thus, the scene context can be used to make the DNN more robust against such attacks.

Such co-existence relationships, together with the background, create a *context* that can be captured by human vision systems. Specifically, if one (physically) replaces the stop sign with a speed limit sign, humans can recognize the anomaly that the speed limit sign does not fit in the scene. If a DNN module can also learn such relationships (i.e., the context), it should also be able to deduce if the (mis)classification result (i.e., the speed limit sign) is out of context.

Inspired by these observations and the fact that context has been used very successfully in recognition problems, we propose to use *context inconsistency* to detect adversarial perturbation attacks. This defense strategy complements existing defense methods [17, 24, 33], and can cope with both digital and physical perturbations. To the best of our knowledge, it is the first strategy to defend object detection systems by considering objects "within the context of a scene."

We realize a system that checks for context inconsistencies caused by adversarial perturbations, and apply this approach for the defense of object detection systems; our work is motivated by a rich literature on context-aware object recognition systems [4, 11, 22, 35]. We assume a framework for object detection similar to [42], where the system first proposes many regions that potentially contain objects, which are then classified. In brief, our approach accounts for four types of relationships among the regions, all of which together form the context for each proposed region: a) regions corresponding to the same object (*spatial context*); b) regions corresponding to other objects likely to co-exist within a scene (*object-object context*; c) the regions likely to co-exist with the background (*object-background context*); and d) the consistency of the regions within the holistic scene (*object-scene context*). Our approach constructs a fully connected graph with the proposed regions and a super-region node which represents the scene. In this graph, each node has, what we call an associated context profile.

The *context profile* is composed of node features (i.e., the original feature used for classification) and edge features (i.e., context). Node features represent the region of interest (RoI) and edge features encode how the current region relates to other regions in its feature space representation. Motivated by the observation that the context profile of each object category is almost always unique, we use an auto-encoder to learn the distribution of the context profile of each category. In testing, the auto-encoder checks whether the classification result is consistent with the testing context profile. In particular, if a proposed region (say of class A) contains adversarial perturbations that cause the DNN of the object detector to misclassify it as class B, using the auto-encoder of class B to reconstruct the testing context profile of class A will result in a high reconstruction error. Based on this, we can conclude that the classification result is suspicious.

The main contributions of our work are the following.

- To the best of our knowledge we are the first to propose using context inconsistency to detect adversarial perturbations in object classification tasks.
- We design and realize a DNN-based adversarial detection system that automatically extracts context for each region, and checks its consistency with a learned context distribution of the corresponding category.
- We conduct extensive experiments on both digital and physical perturbation attacks with three different adversarial targets on two large-scale datasets - PASCAL VOC [12] and Microsoft COCO [32]. Our method yields high detection performance in all the test cases; the ROC-AUC is over 0.95 in most cases, which is 20-35% higher than a state-of-the-art method [47] that does not use context in detecting adversarial perturbations.

2 Related Work

We review closely-related work and its relationship to our approach.

Object Detection, which seeks to locate and classify object instances in images/videos, has been extensively studied [31, 34, 41, 42]. Faster R-CNN [42] is a state-of-the-art DNN-based object detector that we build upon. It initially proposes class-agnostic bounding boxes called region proposals (first stage), and then outputs the classification result for each of them in the second stage.

Adversarial Perturbations on Object Detection, and in particular physical perturbations targeting DNN-based object detectors, have been studied recently [6, 43, 48] (in addition to those targeting image classifiers [1, 13, 26]). Besides mis-categorization attacks, two new types of attacks have emerged against object detectors: the *hiding attack* and the *appearing attack* [6, 43] (see Section 3.1 for more details). While defenses have been proposed against digital adversarial perturbations in image classification, our work focuses on both digital and physical adversarial attacks on object detection systems, which is an open and challenging problem.

Adversarial Defense has been proposed for coping with digital perturbation attacks in the image domain. Detection-based defenses aim to distinguish perturbed images from normal ones. *Statistics based detection methods* rely on extracted features that have different distributions across clean images and perturbed ones [14, 19, 33]. *Prediction inconsistency based detection methods* process the images and check for consistency between predictions on the original images

Detection	Beyond MNIST CIFAR	Do not need perturbed samples for training	Extensibility to object detection
PCAWhiten [19]	✗	✓	✗, PCA is not feasible on large regions
GaussianMix [14]	✗	✗	✗, Fixed-sized inputs are required
Steganalysis [33]	✓	✗	✗, Unsatisfactory performance on small regions
ConvStat [38]	✗	✗	✓
SafeNet [36]	✓	✗	✓
PCAConv [29]	✓	✗	✗, Fixed-sized inputs are required
SimpleNet [16]	✗	✗	✓
AdapDenoise [30]	✓	✗	✓
FeatureSqueeze [47]	✓	✓	✓

Table 1. Comparison of existing detection-based defenses; since FeatureSqueeze [47] meets all the basic requirements of our approach, it is used as a baseline in the experimental analysis.

and processed versions [30, 47]. *Other methods train a second binary classifier* to distinguish perturbed inputs from clean ones [29, 36, 38]. However many of these are effective only on small and simple datasets like MNIST and CIFAR-10 [5]. Most of them need large amounts of perturbed samples for training, and very few can be easily extended to region-level perturbation detection, which is the goal of our method. Table 1 summarizes the differences between our method and the other defense methods; we extend FeatureSqueeze [47], considered a state-of-the-art detection method, which squeezes the input features by both reducing the color bit depth of each pixel and spatially smoothening the input images, to work at the region-level and use this as a baseline (with this extension its performance is directly comparable to that of our approach).

Context Learning for Object Detection has been studied widely [4, 11, 21, 40, 45]. Earlier works that incorporate context information into DNN-based object detectors [9, 15, 39] use object relations in post-processing, where the detected objects are re-scored by considering object relations. Some recent works [7, 27] perform sequential reasoning, i.e., objects detected earlier are used to help find objects later. The state-of-the-art approaches based on recurrent units [35] or neural attention models [22] process a set of objects using interactions between their appearance features and geometry. Our proposed context learning framework falls into this type, and among these, [35] is the one most related to our work. We go beyond the context learning method to define the context profile and use context inconsistency checks to detect attacks.

3 Methodology

3.1 Problem Definition and Framework Overview

We propose to detect adversarial perturbation attacks by recognizing the context inconsistencies they cause, i.e., by connecting the dots with respect to whether the object fits within the scene and in association with other entities in the scene.
Threat Model. We assume a strong white-box attack against the two-stage

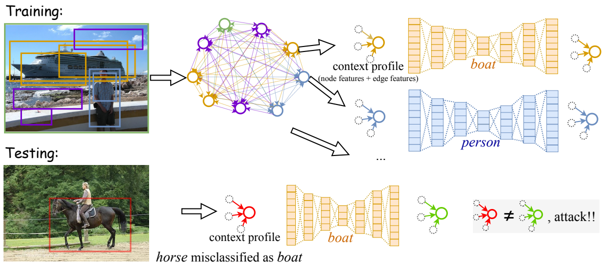


Fig. 2. Training phase: a fully connected graph is built to connect the regions of the scene image – details in Fig. 3; context information relating to each object category is collected and used to train auto-encoders. Testing phase: the context profile is extracted for each region and input to the corresponding auto-encoder to check if it matches with benign distribution.

Faster R-CNN model where both the training data and the parameters of the model are known to the attacker. Since there are no existing attacks against the first stage (i.e., region proposals), we do not consider such attacks. The attacker's goal is to cause the second stage of the object detector to malfunction by adding digital or physical perturbations to *one* object instance/background region. There are three types of attacks [6, 43, 48]:

- *Miscategorization attacks* make the object detector miscategorize the perturbed object as belonging to a different category.
- *Hiding attacks* make the object detector fail in recognizing the presence of the perturbed object, which happens when the confidence score is low or the object is recognized as background.
- *Appearing attacks* make the object detector wrongly conclude that the perturbed background region contains an object of a desired category.

Framework Overview. We assume that we can get the region proposal results from the first stage of the Faster R-CNN model and the prediction results for each region from its second stage. We denote the input scene image as I and the region proposals as $R_I = [r_1, r_2, ..., r_N]$, where N is the total number of proposals of I. During the training phase, we have the ground truth category label and bounding box for each r_i, denoted as $S_I = [s_1, s_2, ..., s_N]$. The Faster R-CNN's predictions on proposed regions are denoted as $\tilde{S}_I$. Our goal as an attack detector is to identify perturbed regions from all the proposed regions.

Fig. 2 shows the workflow of our framework. We use a structured DNN model to build a fully connected graph on the proposed regions to model the context of a scene image. We name this as Structure ContExt ModEl, or `SCEME` in short. In `SCEME`, we combine the node features and edge features of each node r_i, to form its context profile. We use auto-encoders to detect context inconsistencies as outliers. Specifically, during the training phase, for each category, we train a separate auto-encoder to capture the distribution of the benign context profile

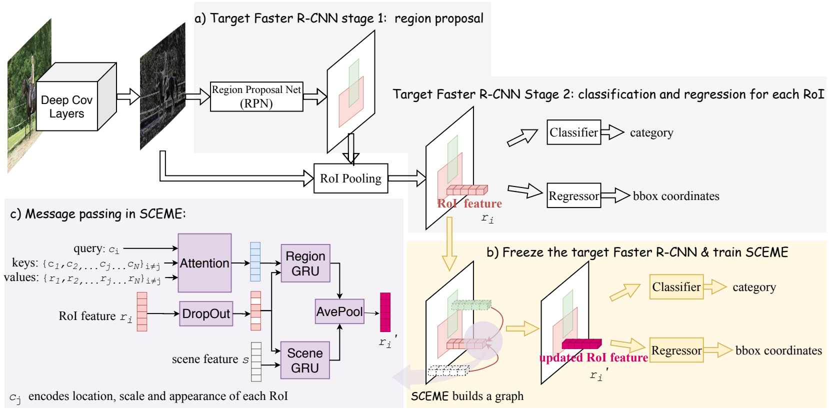

a) Target Faster R-CNN stage 1: region proposal
Deep Cov Layers
Region Proposal Net (RPN)
RoI Pooling
Target Faster R-CNN Stage 2: classification and regression for each RoI
Classifier
category
Regressor
bbox coordinates
RoI feature
r_i
c) Message passing in SCEME:
query: c_i
keys: $\{c_1, c_2, \dots c_j \dots c_N\}_{i \neq j}$
values: $\{r_1, r_2, \dots r_j \dots r_N\}_{i \neq j}$
Attention
Region GRU
RoI feature r_i
DropOut
AvePool
r_i'
scene feature s
Scene GRU
c_j encodes location, scale and appearance of each RoI
b) Freeze the target Faster R-CNN & train SCEME
Classifier
category
updated RoI feature
r_i'
Regressor
bbox coordinates
SCEME builds a graph

Fig. 3. (a) The attack target model, the Faster R-CNN, is a two-stage detector. (b) SCEME is built upon the proposed regions from the first stage of the Faster R-CNN, and updates the RoI features by message passing across regions. (c) Zooming in on SCEME shows how it fuses context information into each RoI, by updating RoI features via Region and Scene GRUs.

of that category. We also have an auto-encoder for the background category to detect hiding attacks. During testing, we extract the context profile for each proposed region. We then select the corresponding auto-encoder based on the prediction result of the Faster R-CNN model and check if the testing context profile belongs to the benign distribution. If the reconstruction error rate is higher than a threshold, we posit that the corresponding region contains adversarial perturbations. In what follows, we describe each step of SCEME in detail.

3.2 Constructing SCEME

In this subsection, we describe the design of the fully connected graph and the associated message passing mechanism in SCEME. Conceptually, SCEME builds a fully connected graph on each scene image. Each node is a region proposal generated by the first stage of the target object detector, plus the scene node. The initial node features, r_i, are the RoI pooling features of the corresponding region. The node features are then updated ($r_i \rightarrow r_i'$) using message passing from other nodes. After convergence, the updated node features r_i' are used as inputs to a regressor towards refining the bounding box coordinates and a classifier to predict the category, as shown in Fig. 3(b). Driven by the object detection objective, we train SCEME and the following regressor and classifier together. We freeze the weights of the target Faster R-CNN during the training. To force SCEME to rely more on context information instead of the appearance information (i.e., node features) when performing object detection, we apply a dropout function [20] on the node features before inputing into SCEME, during the training phase. At the end of training, SCEME should be able to have better object detection performance than the target Faster R-CNN since it explicitly uses the context information from other regions to update the appearance features of each region via message passing. This is observed in our implementation.

We use Gated Recurrent Units (GRU) [8] with attention [2] as the message passing mechanism in SCEME. For each proposed region, relationships with other regions and the whole scene form four kinds of context:

- *Same-object context*: for regions over the same object, the classification results should be consistent;
- *Object-object context*: co-existence, relative location, and scale between objects are usually correlated;
- *Object-background context*: the co-existence of the objects and the associated background regions are also correlated;
- *Object-scene context*: when considering the whole scene image as one super region, the co-existence of objects in the entire scene are also correlated.

To utilize object-scene context, the scene GRU takes the scene node features s as the input, and updates $r_i \rightarrow r_{scene}$. To utilize the other kinds of context, since we have no ground truth about which object/background the regions belong to, we use attention to learn what context category to utilize from different regions. The query and key (they encode information like location, appearance, scale, etc.) pertaining to each region are defined similar to [35]. Comparing the relative location, scale and co-existence between the query of the current region and the keys of all the other regions, the attention system assigns different attention scores to each region, i.e., it updates r_i, utilizing different amount of information from $\{r_j\}_{j \neq i}$. Thus, r_j is first weighted by the attention scores and then all r_j are summed up as the input to the Region GRU to update $r_i \rightarrow r_{regions}$ as shown in Fig. 3(c). The corresponding output, $r_{regions}$ and r_{scene}, are then combined via the average pooling function to get the final updated RoI feature vector r'.

3.3 Context Profile

In this subsection, we describe how we extract a context profile in SCEME. Recall that a context profile consists of node features r and edge features, where the edge features describe how r is updated. Before introducing the edge features that we use, we describe in detail how message passing is done with GRU [8].

A GRU is a memory cell that can remember the initial node features r and then fuse incoming messages from other nodes into a meaningful representation. Let us consider the GRU that takes the feature vector v (from other nodes) as the input, and updates the current node features r. Note that r and v have the same dimensions since both are from RoI pooling. GRU computes two gates given v and r, for message fusion. The reset gate γ_r drops or enhances information in the initial memory based on its relevance to the incoming message v. The update gate γ_u controls how much of the initial memory needs to be carried over to the next memory state, thus allowing a more effective representation. In other words, γ_r and γ_u are two vectors of the same dimension as r and v, which are learned by

Algorithm 1: SCEME: Training phase

```
Input  : $\{R_I, S_I, \tilde{S}_I\}_{I \in TrainSet}$
Output : SCEME, $AutoEncoder_c$ for each object category $c$, and $thresh_{err}$
SCEME ← TrainSCEME( $\{R_I, S_I\}_{I \in TrainSet}$)
ContextProfiles[$c$] = [] for each object category c
for each $R_I = [r_1, r_2, ...]$ do
    $X_I = [x_1, x_2, ...] \leftarrow$ ExtractContextProfiles(SCEME, $R_I$)
    for each region, its prediction, and its context profile $\{r_j, \tilde{s_j}, x_j\}$ do
        $\tilde{c} \leftarrow$ GetPredictedCategory($\tilde{s_j}$)
        ContextProfiles[$\tilde{c}$] ← ContextProfiles[$\tilde{c}$] + $x_j$
    end
end
for each category $c$ do
    $AutoEncoder_c \leftarrow$ TrainAutoEncoder($ContextProfiles[c]$)
end
$thresh_{err}$ = GetErrThreshold($\{AutoEncoder_c\}$)
return SCEME, $\{AutoEncoder_c\}$, $thresh_{err}$
```

the model to decide what information should be passed to the next memory state given the current memory state and the incoming message. Therefore, we use the gate vectors as the edge features in the context profile. There are, in total, four gate feature vectors from both the Scene GRU and the Region GRU. Therefore, we define the context profile of a proposed region as $x = [r, \gamma_{u1}, \gamma_{u2}, \gamma_{r1}, \gamma_{r2}]$.

3.4 AutoEncoder for Learning Context Profile Distribution

In benign settings, all context profiles of a given category must be similar to each other. For example, stop sign features exist with features of road signs and zebra crossings. Therefore, the context profile of a stop sign corresponds to a unique distribution that accounts for these characteristics. When a stop sign is misclassified as a speed limit sign, its context profile should not fit with the distribution corresponding to that of the speed limit sign category.

For each category, we use a separate auto-encoder (architecture shown in the supplementary material) to learn the distribution of its context profile. The input to the auto-encoder is the context profile $x = [r, \gamma_{u1}, \gamma_{u2}, \gamma_{r1}, \gamma_{r2}]$. A fully connected layer is first used to compress the node features (r) and edge features ($[\gamma_{u1}, \gamma_{u2}, \gamma_{r1}, \gamma_{r2}]$) separately. This is followed by two convolution layers, wherein the node and edge features are combined to learn the joint compression. Two fully connected layers are then used to further compress the joint features. These layers form a bottleneck that drives the encoder to learn the true relationships between the features and get rid of redundant information. SmoothL1Loss, as defined in [23, 46], between the input and the output is used to train the auto-encoder, which is a common practice.

Once trained, we can detect adversarial perturbation attacks by appropriately thresholding the reconstruction error. Giving a new context profile during testing, if a) the node features are not aligned with the corresponding distribution of benign node features, or b) the edge features are not aligned with the corresponding distribution of benign edge features, or c) the joint distribution between the

Algorithm 2: SCEME: Testing phase

```
Input  : $R_I$, $\tilde{S}_I$, SCEME, $\{AutoEncoder_c\}$, $thresh_{err}$
Output : perturbed regions $PerturbedSet$
$PerturbedSet = []$
$X_I$ = ExtractContextProfiles(SCEME, $R_I$)
for each region, its prediction, and its context profile $\{r_j, \tilde{s_j}, x_j\}$ do
    $\tilde{c} \leftarrow$ GetPredictedCategory($\tilde{s_j}$)
    err = GetAutoEncoderReconErr($AutoEncoder_{\tilde{c}}, x_j$)
    if $err > thresh_{err}$ then
        region $\leftarrow$ GetRegion($\tilde{s_j}$)
        $PerturbedSet \leftarrow PerturbedSet +$ region
end
return $PerturbedSet$
```

node features and the edge features is violated, the auto-encoder will not be able to reconstruct the features using its learned distribution/relation. In other words, a reconstruction error that is larger than the chosen threshold would indicate either an appearance discrepancy or a context discrepancy between the input and output of the auto-encoder.

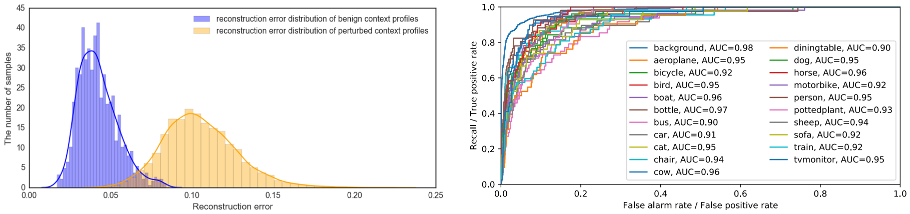
An overview of the approach (training and testing phases) is captured in Algorithms 1 and 2.

4 Experimental Analysis

We conduct comprehensive experiments on two large-scale object detection datasets to evaluate the proposed method, SCEME, against six different adversarial attacks, viz., digital miscategorization attack, digital hiding attack, digital appearing attack, physical miscategorization attack, physical hiding attack, and physical appearing attack, on Faster R-CNN (the general idea can be applied more broadly). We analyze how different kinds of context contribute to the detection performance. We also provide a case study for detecting physical perturbations on stop signs, which has been used widely as a motivating example.

4.1 Implementation Details

Datasets. We use both PASCAL VOC [12] and MS COCO [32]. PASCAL VOC contains 20 object categories. Each image, on average, has 1.4 categories and 2.3 instances [32]. We use *voc07trainval* and *voc12trainval* as training datasets and the evaluations are carried out on *voc07test*. MS COCO contains 80 categories. Each image, on average, has 3.5 categories and 7.7 instances. *coco14train* and *coco14valminusminival* are used for training, and the evaluations are carried out on *coco14minival*. Note that COCO has few examples for certain categories. To make sure we have enough number of context profiles to learn the distribution, we train 11 auto-encoders for the 11 categories that have the largest numbers of extracted context profiles. Details are provided in the supplementary material.
Attack Implementations. For digital attacks, we use the standard iterative fast gradient sign method (IFGSM) [26] and constrain the perturbation location



(a) (b)

Fig. 4. (a) Reconstruction errors of benign aeroplane context profiles are generally smaller than those of the context profiles of digitally perturbed objects that are misclassified as an aeroplane. (b) Thresholding the reconstruction error, we get the detection ROC curves for all the categories on PASCAL VOC dataset.

within the ground truth bounding box of the object instance. Because our defense depends on contextual information, it is not sensitive to how the perturbation is generated. We compare the performance against perturbations generated by a different method (FGSM) in the supplementary material. We use the physical attacks proposed in [13, 43], where perturbation stickers are constrained to be on the object surface; the color of the stickers should be printable, and the pattern of the stickers should be smooth. For evaluations on a large scale, we do not print or add stickers physically; we add them digitally onto the scene image. This favors attackers since they can control how their physical perturbations are captured.

Defense Implementation. Momentum optimizer with momentum 0.9 is used to train `SCEME`. The learning rate is 5e-4 and decays every 80k iterations at a decay rate of 0.1. The training finishes after 250k iterations. Adam optimizer is used to train auto-encoders. The learning rate is 1e-4 and reduced by 0.1 when the training loss stops decreasing for 2 epochs. Training finishes after 10 epochs.

4.2 Evaluation of Detection Performance

Evaluation Metric. We extract the context profile for each proposed region, feed it to its corresponding auto-encoder and threshold the reconstruction error to detect adversarial perturbations. Therefore, we evaluate the detection performance at the region level. Benign/negative regions are the regions proposed from clean objects; perturbed/positive regions are the regions relating to perturbed objects. We report Area Under Curve (AUC) of Receiver Operating Characteristic Curve (ROC) to evaluate the detection performance. Note that there can be multiple regions of a perturbed object. If any of these regions is detected, it is a successful perturbation detection. For hiding attacks, there is a possibility of no proposed region; however, it occurs rarely (less than 1%).

Visualizing the Reconstruction Error. We plot the reconstruction error of benign aeroplane context profiles and that of digitally perturbed objects that are misclassified as an aeroplane. As shown in Fig. 4(a), the context profiles of perturbed regions do not conform with the benign distribution of aeroplanes'

Method	Digital Perturbation			Physical Perturbation		
	Miscateg	Hiding	Appearing	Miscateg	Hiding	Appearing
Results on PASCAL VOC:						
FeatureSqueeze [47]	0.724	0.620	0.597	0.779	0.661	0.653
Co-occurGraph [3]	0.675	-	-	0.810	-	-
SCEME (node features only)	0.866	0.976	0.828	0.947	0.964	0.927
SCEME	0.938	0.981	0.869	0.973	0.976	0.970
Results on MS COCO:						
FeatureSqueeze [47]	0.681	0.682	0.578	0.699	0.687	0.540
Co-occurGraph [3]	0.605	-	-	0.546	-	-
SCEME (node features only)	0.901	0.976	0.810	0.972	0.954	0.971
SCEME	0.959	0.984	0.886	0.989	0.968	0.989

Table 2. The detection performance (ROC-AUC) against six different attacks on PASCAL VOC and MS COCO dataset

context profiles and cause larger reconstruction errors. This test validates our hypothesis that the context profile of each category has a unique distribution. The auto-encoder that learns from the context profile of class A will not reconstruct class B well.

Detection Performance. Thresholding the reconstruction error, we plot the ROC curve for "aeroplane" and other object categories tested on PASCAL VOC dataset, in Fig. 4(b). The AUCs for all 21 categories (including background) are all over 90%. This means that all the categories have their unique context profile distributions, and the reconstruction error of their auto-encoders effectively detect perturbations. The detection performance results, against six attacks on PASCAL VOC and MS COCO, are shown in Tab. 2. Three baselines are considered.

- *FeatureSqueeze* [47]. As discussed in Tab. 1, many existing adversarial perturbation detection methods are not effective beyond simple datasets. Most require perturbed samples while training, and only few can be extended to region-level perturbation detection. We extend FeatureSqueeze, one of the state-of-the-art methods, that is not limited by these, for the object detection task. Implementation details are provided in the supplementary material.
- *Co-occurGraph* [3]. We also consider a non-deep graph model where co-occurrence context is represented, as a baseline. We check the inconsistency between the relational information in the training data and testing images to detect attacks. Details are in the supplementary material. Note that the co-occurrence statistics of background class cannot be modeled, and so this approach is inapplicable for detecting hiding and appearing attacks.
- *SCEME (node features only)*. Only node features are used to train the auto-encoders (instead of using context profiles with both node features for region representation and edge features for contextual relation representation). Note that the node features already implicitly contain context information since, with Faster R-CNN, the receptive field of neurons grows with depth and eventually covers the entire image. We use this baseline to quantify the improvement we achieve by explicitly modeling context information with SCEME.

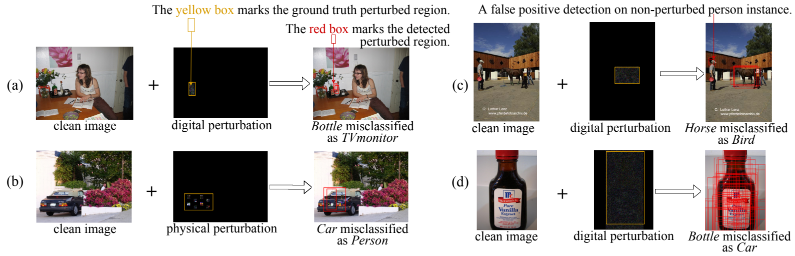

The yellow box marks the ground truth perturbed region.
The red box marks the detected perturbed region.
A false positive detection on non-perturbed person instance.
(a)
+
clean image
digital perturbation
Bottle misclassified as TVmonitor
(b)
+
clean image
physical perturbation
Car misclassified as Person
(c)
+
clean image
digital perturbation
Horse misclassified as Bird
(d)
+
clean image
digital perturbation
Bottle misclassified as Car

Fig. 5. A few interesting examples. `SCEME` successfully detects both digital and physical perturbations as shown in (a) and (b). (c) shows that the horse misclassification affects the context profile of person and leads to false positive detection on the person instance. (d) Appearance information and spatial context are used to successfully detect perturbations.

Our method `SCEME`, yields high AUC on both datasets and for all six attacks; many of them are over 0.95. The detection performance of `SCEME` is consistently better than that of FeatureSqueeze, by over 20%. Compared to Co-occurGraph, the performance of our method in detecting miscategorization attacks, is better by over 15%. Importantly, `SCEME` is able to detect hiding and appearing attacks and detect perturbations in images with one object, which is not feasible with Co-occurGraph. Using node features yields good detection performance and further using edge features, improves performance by up to 8% for some attacks.

Examples of Detection Results. We visualize the detected perturbed regions for both digital and physical miscategorization attack in Fig. 5. The reconstruction error threshold is chosen to make the false positive rate 0.2%. `SCEME` successfully detects both digital and physical perturbations as shown in Fig. 5(a)and(b). The misclassification of the perturbed object could affect the context information of another coexisting benign object and lead to a false perturbation detection on the benign object as shown in Fig. 5(c). We observe that this rarely happens. In most cases, although some part of the object-object context gets violated, the appearance representation and other context would help in making the right detection. When there are not many object-object context relationships as shown in Fig. 5(d), appearance information and spatial context are mainly used to detect a perturbation.

4.3 Analysis of Different Contextual Relations

In this subsection, we analyze what roles different kinds of context features play. **Spatial context** consistency means that nearby regions of the same object should yield consistent prediction. We do two kinds of analysis. The first one is to observe the correlations between the adversarial detection performance and the number of regions proposed by the target Faster R-CNN for the perturbed object. Fig. 6(a) shows that the detection performance improves when more

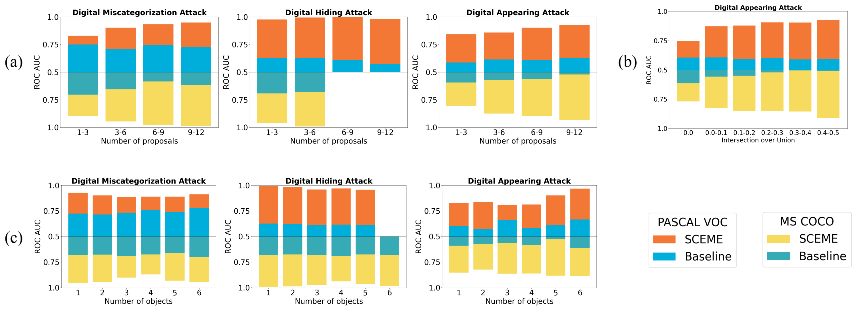


Fig. 6. Subfigures are diverging bar charts. They start with ROC-AUC = 0.5 and diverge in both upper and lower directions: upper parts are results on PASCAL VOC and lower parts are on MS COCO. For each dataset, we show both the results from the FeatureSqueeze baseline and `SCEME`, using overlay bars. (a) The more the regions proposed, the better our detection performs, as there is more utilizable spatial context; (b) the larger the overlapped region between the "appearing object" and another object, the better our detection performs, as the spatial context violation becomes larger and detectable (we only analyze the appearing attack here); (c) the more the objects, the better our detection performs generally, as there is more utilizable object-object context (performance slightly saturates at first due to inadequate spatial context).

regions are proposed for the object and this correlation is not observed for the baseline method (for both datasets). This indicates that spatial context plays a role in perturbation detection. Our second analysis is on appearing attacks. If the "appearing object" has a large overlap with one ground truth object, the spatial context of that region will be violated. We plot in Fig. 6(b) the detection performance with respect to the overlap between the appearing object and the ground truth object, measured by Intersection over Union (IoU). We observe that the more these two objects overlap, the more likely the region is detected as perturbed, consistent with our hypothesis.

Object-object context captures the co-existence of objects and their relative position and scale relations. We test the detection performance with respect to the number of objects in the scene images. As shown in Fig. 6(c), in most cases, the detection performance of `SCEME` first drops or stays stable, and then improves. We believe that the reason is as follows: initially, as the number of objects increases, the object-object context is weak and so is the spatial context as the size of the objects gets smaller with more of them; however, as the number of objects increases, the object-object context dominates and performance improves.

4.4 Case Study on Stop Sign

We revisit the stop sign example and provide quantitative results to validate that context information helps defend against perturbations. We get 1000 per-

False Positive Rate	0.1%	0.5%	1%	5%	10%
Recall of FeatureSqueeze [47]	0	0	0	3%	8%
Recall of SCEME (node features only)	33%	52%	64%	83%	91%
Recall of SCEME	54%	67%	74%	89%	93%

Table 3. Recall for detecting perturbed stop signs at different false positive rate.

turbed stop sign examples, all of which are misclassified by the Faster RCNN, from the COCO dataset. The baselines and SCEME, are tested for detecting the perturbations. If we set a lower reconstruction error threshold, we will have a better chance of detecting the perturbed stop signs. However, there will be higher false positives, which means wrong categorization of clean regions as perturbed. Thus, to compare the methods, we constrain the threshold of each method so as to meet a certain *False Positive Rate* (FPR), and compute the *recall* achieved, i.e., out of the 1000 samples, how many are detected as perturbed? The results are shown in Tab. 3. FeatureSqueeze [47] cannot detect any perturbation until a FPR 5% is chosen. SCEME detects 54% of the perturbed stop signs with a FPR of 0.1%. Further, compared to its ablated version (that only uses node features), our method detects almost twice as many perturbed samples when the FPR required is very low (which is the case in many real-world applications).

5 Conclusions

Inspired by how humans can associate objects with where and how they appear within a scene, we propose to detect adversarial perturbations by recognizing context inconsistencies they cause in the input to a machine learning system. We propose SCEME, which automatically learns four kinds of context, encompassing relationships within the scene and to the scene holistically. Subsequently, we check for inconsistencies within these context types, and flag those inputs as adversarial. Our experiments show that our method is extremely effective in detecting a variety of attacks on two large scale datasets and improves the detection performance by over 20% compared to a state-of-the-art, context agnostic method.

Acknowledgments

This research was partially sponsored by ONR grant N00014-19-1-2264 through the Science of AI program, and by the U.S. Army Combat Capabilities Development Command Army Research Laboratory under Cooperative Agreement Number W911NF-13-2-0045 (ARL Cyber Security CRA). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Combat Capabilities Development Command Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

1. Athalye, A., Engstrom, L., Ilyas, A., Kwok, K.: Synthesizing robust adversarial examples. arXiv preprint arXiv:1707.07397 (2017)
2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
3. Bappy, J.H., Paul, S., Roy-Chowdhury, A.K.: Online adaptation for joint scene and object classification. In: European Conference on Computer Vision. pp. 227–243. Springer (2016)
4. Barnea, E., Ben-Shahar, O.: Exploring the bounds of the utility of context for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7412–7420 (2019)
5. Carlini, N., Wagner, D.: Adversarial examples are not easily detected: Bypassing ten detection methods. In: Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security. pp. 3–14 (2017)
6. Chen, S.T., Cornelius, C., Martin, J., Chau, D.H.P.: Shapeshifter: Robust physical adversarial attack on faster r-cnn object detector. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 52–68. Springer (2018)
7. Chen, X., Gupta, A.: Spatial memory for context reasoning in object detection. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4086–4096 (2017)
8. Cho, K., Van Merriënboer, B., Bahdanau, D., Bengio, Y.: On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259 (2014)
9. Choi, M.J., Torralba, A., Willsky, A.S.: A tree-based context model for object recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence **34**(2), 240–252 (2011)
10. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
11. Dvornik, N., Mairal, J., Schmid, C.: Modeling visual context is key to augmenting object detection datasets. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 364–380 (2018)
12. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International Journal of Computer Vision **88**(2), 303–338 (2010)
13. Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., Prakash, A., Kohno, T., Song, D.: Robust physical-world attacks on deep learning visual classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1625–1634 (2018)
14. Feinman, R., Curtin, R.R., Shintre, S., Gardner, A.B.: Detecting adversarial samples from artifacts. arXiv preprint arXiv:1703.00410 (2017)
15. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE Transactions on Pattern Analysis and Machine Intelligence **32**(9), 1627–1645 (2009)

16. Gong, Z., Wang, W., Ku, W.S.: Adversarial and clean data are not twins. arXiv preprint arXiv:1704.04960 (2017)
17. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572 (2014)
18. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778 (2016)
19. Hendrycks, D., Gimpel, K.: Early methods for detecting adversarial images. arXiv preprint arXiv:1608.00530 (2016)
20. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.: Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580 (2012)
21. Hollingworth, A.: Does consistent scene context facilitate object perception? Journal of Experimental Psychology: General **127**(4), 398 (1998)
22. Hu, H., Gu, J., Zhang, Z., Dai, J., Wei, Y.: Relation networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3588–3597 (2018)
23. Huber, P.J.: Robust estimation of a location parameter. In: Breakthroughs in Statistics, pp. 492–518. Springer (1992)
24. Jia, X., Wei, X., Cao, X., Foroosh, H.: Comdefend: An efficient image compression model to defend adversarial examples. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6084–6092 (2019)
25. Jin, D., Gao, S., Kao, J.Y., Chung, T., Hakkani-tur, D.: Mmm: Multi-stage multi-task learning for multi-choice reading comprehension. arXiv preprint arXiv:1910.00458 (2019)
26. Kurakin, A., Goodfellow, I., Bengio, S.: Adversarial examples in the physical world. arXiv preprint arXiv:1607.02533 (2016)
27. Li, J., Wei, Y., Liang, X., Dong, J., Xu, T., Feng, J., Yan, S.: Attentive contexts for object detection. IEEE Transactions on Multimedia **19**(5), 944–954 (2016)
28. Li, S., Neupane, A., Paul, S., Song, C., Krishnamurthy, S.V., Roy-Chowdhury, A.K., Swami, A.: Stealthy adversarial perturbations against real-time video classification systems. In: NDSS (2019)
29. Li, X., Li, F.: Adversarial examples detection in deep networks with convolutional filter statistics. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 5764–5772 (2017)
30. Liang, B., Li, H., Su, M., Li, X., Shi, W., Wang, X.: Detecting adversarial image examples in deep neural networks with adaptive noise reduction. IEEE Transactions on Dependable and Secure Computing (2018)
31. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2980–2988 (2017)
32. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European Conference on Computer Vision. pp. 740–755. Springer (2014)

33. Liu, J., Zhang, W., Zhang, Y., Hou, D., Liu, Y., Zha, H., Yu, N.: Detection based defense against adversarial examples from the steganalysis point of view. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4825–4834 (2019)
34. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: European Conference on Computer Vision. pp. 21–37. Springer (2016)
35. Liu, Y., Wang, R., Shan, S., Chen, X.: Structure inference net: Object detection using scene-level context and instance-level relationships. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6985–6994 (2018)
36. Lu, J., Issaranon, T., Forsyth, D.: Safetynet: Detecting and rejecting adversarial examples robustly. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 446–454 (2017)
37. McCool, C., Perez, T., Upcroft, B.: Mixtures of lightweight deep convolutional neural networks: Applied to agricultural robotics. IEEE Robotics and Automation Letters **2**(3), 1344–1351 (2017)
38. Metzen, J.H., Genewein, T., Fischer, V., Bischoff, B.: On detecting adversarial perturbations. arXiv preprint arXiv:1702.04267 (2017)
39. Mottaghi, R., Chen, X., Liu, X., Cho, N.G., Lee, S.W., Fidler, S., Urtasun, R., Yuille, A.: The role of context for object detection and semantic segmentation in the wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 891–898 (2014)
40. Oliva, A., Torralba, A., Castelhano, M.S., Henderson, J.M.: Top-down control of visual attention in object detection. In: Proceedings 2003 International Conference on Image Processing (Cat. No. 03CH37429). vol. 1, pp. I–253. IEEE (2003)
41. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 779–788 (2016)
42. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems. pp. 91–99 (2015)
43. Song, D., Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Tramer, F., Prakash, A., Kohno, T.: Physical adversarial examples for object detectors. In: 12th USENIX Workshop on Offensive Technologies (WOOT 18) (2018)
44. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2818–2826 (2016)
45. Torralba, A.: Contextual priming for object detection. International Journal of Computer Vision **53**(2), 169–191 (2003)
46. Xie, J., Yang, J., Ding, C., Li, W.: High accuracy individual identification model of crested ibis (nipponia nippon) based on autoencoder with self-attention. IEEE Access **8**, 41062–41070 (2020)

47. Xu, W., Evans, D., Qi, Y.: Feature squeezing: Detecting adversarial examples in deep neural networks. arXiv preprint arXiv:1704.01155 (2017)
48. Zhao, Y., Zhu, H., Liang, R., Shen, Q., Zhang, S., Chen, K.: Seeing isn't believing: Towards more robust adversarial attack against real world object detectors. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. pp. 1989–2004 (2019)
49. Zhu, S., Wang, Z., Chen, X., Li, S., Iqbal, U., Qian, Z., Chan, K.S., Krishnamurthy, S.V., Shafiq, Z.: A4: Evading learning-based adblockers. arXiv preprint arXiv:2001.10999 (2020)