UNIVERSITY OF CALIFORNIA
RIVERSIDE

Spatial and Motion Context for Adversarial Attacks on Vision Systems

A Dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Computer Science

by

Shasha Li

June 2021

Dissertation Committee:

Dr. Srikanth V. Krishnamurthy, Co-Chairperson
Dr. Amit Roy Chowdhury, Co-Chairperson
Dr. Chengyu Song
Dr. Vagelis Papalexakis

The Dissertation of Shasha Li is approved:

Committee Co-Chairperson

Committee Co-Chairperson

University of California, Riverside

Acknowledgments

First and foremost, I thank my advisors Dr. Srikanth V. Krishnamurthy and Dr. Amit Roy Chowdhury who were instrumental in defining the path of my research. Dr. Krishnamurthy, who respected my opinions a lot, never saved his compliments when I achieved something and always was my solid backup when I was stuck. Dr. Krishnamurthy has instilled in me a strong interdisciplinary academic profile and helped me cultivate my research skills. I am grateful for Dr. Chowdhury's guidance and support. I am especially thankful for his discussions which inspired me to always relate my research to its broader literature background.

I am grateful to each member of my dissertation committee; they have provided me extensive personal and professional guidance. I would like to thank Dr. Chengyu Song for working closely together and sharing his knowledge on security. Without his expertise in security, I would not have had the confidence to finish this dissertation.

This dissertation would not have been possible without the generous financial support from our funding agencies and the university. This work was partially supported by the U.S. Army Research Laboratory under cooperative agreement number W911NF-13-2-0045, partially sponsored by an ONR grant N00014-19-1-2264 through the Science of AI program, and partially sponsored by the DARPA TMVD program through Award # HR00112090096. In addition, I am grateful to the university for supporting my graduate studies through the Dean's Distinguished Fellowship Award and my last quarter through the Dissertation Year Program Award.

On a personal note, it was my pleasure to meet my labmates from diverse backgrounds. I will miss those afternoons when we stood in a circle with coffee or tea in hand talking about the progress of research experiments or hot news in our own countries. I would also like to thank other

friends I met during my doctoral studies; our friendship has made my doctoral life colorful. I wish to wholeheartedly thank my two special friends. One is Shuyang, who is an outstanding machine learning researcher, the one who comforts me the most in the world, and the one who became my husband in December of 2020. The other one is Jingle, an orange tabby cat who was born when I first arrived in California, the one who accompanies me on days and nights when I am with smiles and tears.

Finally, I am honored to be part of a loving and caring family. My parents, when they were young, dreamed of getting higher education but they never had a chance due to financial reasons. They were more than supportive when I decided to fly thirteen hours away from them and seek a doctoral program in the United States. I am deeply grateful to them for rooting the eagerness for knowledge in my mind and encouraging me to become who I am today. This dissertation is dedicated to them.

To my parents for all the support.

ABSTRACT OF THE DISSERTATION

Spatial and Motion Context for Adversarial Attacks on Vision Systems

by

Shasha Li

Doctor of Philosophy, Graduate Program in Computer Science
University of California, Riverside, June 2021
Dr. Srikanth V. Krishnamurthy, Co-Chairperson
Dr. Amit Roy Chowdhury, Co-Chairperson

Deep Neural Networks (DNNs) have achieved state-of-the-art performance on a wide range of tasks, thus are increasingly deployed in real-world applications. However, recent studies have found that DNNs are vulnerable to carefully crafted perturbations that are imperceptible to human eyes but fool DNNs into making incorrect predictions. Since then, an arms race between the generation of adversarial perturbation attacks and defenses to thwart them has taken off. This dissertation pursues important directions in this regard and discovers a series of adversarial attack methods and proposes an adversarial defense strategy.

The dissertation starts with white-box attacks where an adversary has full access to the victim DNN model, including model parameters and training settings, and proposes white-box attack methods against face recognition models and real-time video classification models. However, in most real-world attacks, the adversary only has partial information about the victim models, such as the predicted labels. In such black-box attacks, the attacker can send queries to the victim model to collect the corresponding labels, and thereby estimate the gradients needed for curating the adversarial perturbations. A query-efficient black-box video attack method is proposed by pa-

rameterizing the temporal structure of the gradient search space with geometric transformations. The new method exposes vulnerabilities of diverse video classification models and achieves new state-of-the-art attack results.

In addition to attack methods, a defense strategy utilizing context consistency check is proposed, which is inspired by the observation that humans can recognize objects that appear out of place in a scene. By augmenting DNN models with a system that learns context consistency rules during training and checks for the violations of the same during testing, the proposed approach effectively detects various adversarial attacks, with a detection rate over 20% better than the state-of-the-art context-agnostic methods.

In summary, the dissertation reveals several DNN models' vulnerabilities to adversarial attacks in both white-box and black-box attack settings. The proposed adversarial attack methods can be used as benchmarks to evaluate the robustness of image/video models, and are expected to stimulate studies on adversarial image/video defense. An adversarial defense strategy is proposed to enhance the robustness of DNN models

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Deep Neural Networks (DNNs) have achieved state-of-the-art performance on a wide range of computer vision tasks, e.g., face recognition [157,201,221,240,244], object detection [153, 158,214,216] and video classification [129,129,253,253,256,256], thus are increasingly deployed in real-world scenarios, even in security-severe applications such as face recognition based payment systems [78], self driving cars [130, 230] and video surveillance systems [239]. However, recent studies [18, 19, 87, 110, 243] found that DNNs are vulnerable to carefully crafted perturbations that are imperceptible to human eyes but fool DNNs to make incorrect predictions. Since then, an arms race between the generation of adversarial perturbation attacks and the defenses to thwart them has taken off. This dissertation focuses on revealing the vulnerability of DNN models under various attack settings by proposing a series of adversarial attack strategies. In addition, the dissertation aims to improve the robustness and security of current DNN models and propose an adversarial defense strategy.

In the following paragraphs, we explain different attack settings and how the adversarial

attack strategies proposed in the dissertation apply to them.

In the **white-box** attack setting, the adversary has complete knowledge and access to the victim DNN model, including the architecture and parameters of the models and the data distribution used to train the models. In this setting, the attacker is able to compute the gradient of the pre-defined adversarial loss function to generate the adversarial perturbations. We investigate the vulnerability of face recognition models and video classification models in such white-box attack setting in Chapter. 2 and Chapter. 3 separately. Specifically, Chapter. 2 presents a systematic, wide-ranging measurement study of the vulnerability of DNN-based face recognition systems. Experiments show that arbitrary impersonation attacks, wherein an arbitrary attacker impersonates an arbitrary target identity, are hard if imperceptibility of perturbations is an auxiliary goal. Factors such as skin color, gender, and age, impact the ability to carry out an attack on a specific target victim, to different extents. After the study on static perturbations on image inputs, Chapter. 3 moves to adversarial perturbations on real-time video streams. With a specially designed structure that counts for temporal poisoning, the proposed attack method fools the video classification system to mis-classify the target(malicious) actions with rates over 80% whenever the actions are present in real-time video streams.

In the **black-box** attack setting, the adversary can only query the victim model to collect the corresponding prediction results and thereby estimate the gradients needed for curating the adversarial examples. This is the common attack setting in real-word attacks, but also a more challenging setting. Chapter. 4 reveals the vulnerability of widely employed video classification models in such black-box setting. It is demonstrated that the accounting for the temporal dimension is important for the gradient estimation in query-efficient black-box video attacks. By parameterizing

the temporal structure of the the gradient search space with simple geometric transformations, the proposed method achieves extremely high attack success rates with fewer number of queries than previous state-of-the art methods.

In addition to the above adversarial attack methods, Chapter. 5 proposes a defense strategy by utilizing context information. 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. The proposed approach effectively detects various adversarial attacks, with a detection rate over 20% better than the state-of-the-art context-agnostic methods.

In summary, this dissertation studies the vulnerability of deep image models and deep video models, and reveals the DNN models' vulnerability to adversarial attacks in both white-box and black-box attack settings. A series of adversarial attack strategies are proposed, which can be used as benchmarks to evaluate the robustness of image/video models, and are expected to stimulate the study on adversarial image/video defense. An adversarial defense strategy is proposed in the dissertation, which can be used to enhance the robustness of DNN models.

Chapter 2

Measurement-driven Security Analysis of Imperceptible Impersonation Attacks

2.1 Abstract

The emergence of Internet of Things (IoT) brings about new security challenges at the intersection of cyber and physical spaces. One prime example is the vulnerability of Face Recognition (FR) based access control in IoT systems. While previous research has shown that Deep Neural Network (DNN)-based FR systems (FRS) are potentially susceptible to imperceptible impersonation attacks, the potency of such attacks in a wide set of scenarios has not been throughly investigated. In this chapter, we present the first systematic, wide-ranging measurement study of the exploitability of DNN-based FR systems using a large scale dataset. We find that arbitrary impersonation attacks, wherein an arbitrary attacker impersonates an arbitrary target, are hard if imperceptibility is an auxiliary goal. Specifically, we show that factors such as skin color, gender, and

age, impact the ability to carry out an attack on a specific target victim, to different extents. We also study the feasibility of constructing universal attacks that are robust to different poses or views of the attacker's face. Our results show that finding a universal perturbation is a much harder problem from the attacker's perspective. Finally, we find that the perturbed images do not generalize well across different DNN models. This suggests security countermeasures that can dramatically reduce the exploitability of DNN-based FR systems.

Key Words: face recognition, imperceptible adversarial perturbation, Internet of Things

2.2 Introduction

Face-recognition-based biometric authentication has become very popular in Internet of Things (IoT) [174, 204, 284]. In fact, according to the International Biometric Group (IBG), face is the second most widely deployed biometric in terms of market share, right after fingerprints [175]. The most noteworthy applications using face recognition include opening doors [114], activating personalized services by automated identification of users, e.g., smart TV program selector or pervasive software such as Microsoft's Kinect [174, 302].

Face Recognition Systems (FRSs) are typically trained on known faces, and use the trained model to classify test cases (i.e., when a human presents herself to a camera). The deep learning paradigm has seen significant proliferation in FRSs due to its ability to provide high recognition accuracy [201, 221, 244].

Due to the ubiquity of FRSs in security-critical applications, their security and reliability have drawn attention and various attacks have been showcased. Early presentation attacks [7, 67, 281] impersonate a victim's identity by presenting a fake face to FRSs, which could be in the

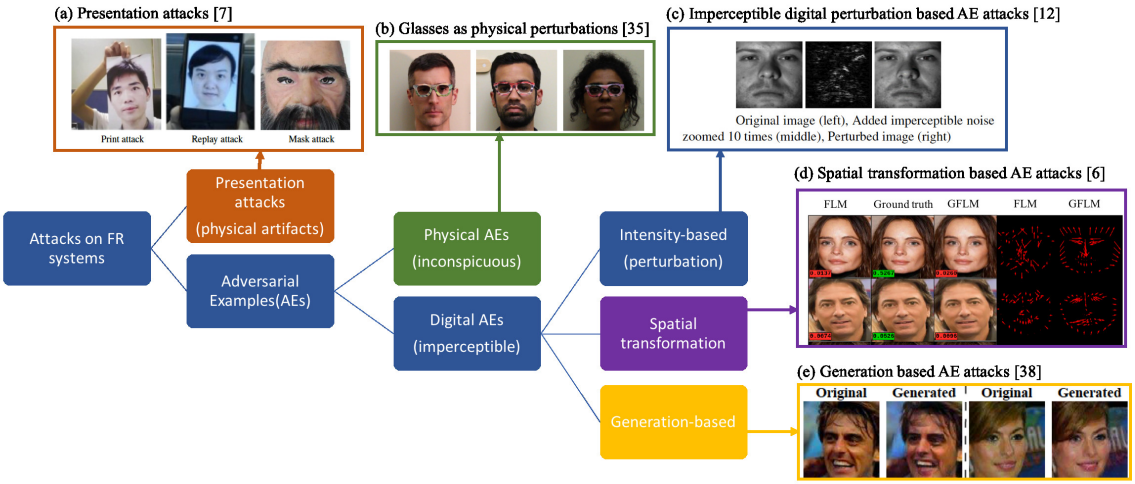


Figure 2.1: Various attacks on Face Recognition Systems. We focus on intensity-based AE attacks in our analysis since they are the kind of attacks explored the most in the literature. Intensity-based AE attacks are fast to carry out and are proven to have high attack success rates.

form of photographs, replayed videos, 3D masks etc., as shown in Fig. 2.1(a). It has been recently shown that Deep Neural Networks (DNNs) are vulnerable to adversarial examples [88, 137, 243]. Adversarial examples are generated in such a manner that humans cannot notice adversarially induced perturbations and correctly classify the images, but the perturbations cause FRSs to misclassify them. Many attack methods [52, 56, 89, 236] have been proposed to generate adversarial examples for impersonation attacks, among which, intensity-based adversarial examples (Fig. 2.1(c)) can be quickly generated and are effective against a variety of FRSs [88, 137]. Intensity-based impersonation attacks add imperceptible perturbation to the original face images such that the FRSs misclassify the perturbed face images (adversarial examples) to be that of the victim.

While we defer a detailed discussion of related work to § 2.3, we find that none of previous efforts perform an in depth study on the scope and effectiveness of such intensity-based impersonation attacks (referred to as impersonation attacks from hereon). In other words, there seems to be no answer yet to the question "Can an arbitrary attacker impersonate an arbitrary victim easily?"

The key term here is *easily*. Specifically, if an attacker were able to add arbitrary amount of perturbations to her own image, she certainly could impersonate any victim. However, this would cause the attacker to be stand out, i.e., her actions could be perceived by observers as strange or even suspicious. Thus, the perturbation has to be imperceptible—the perturbation used must be small and inconspicuous. The question that is of interest therefore becomes "Can the perturbations be kept small in general settings?".

Towards answering this question, we undertake an in depth, systematic measurement study of the exploitability of DNN-based FRSs, using a very large scale dataset of about 2.6 million images. Our measurement study demonstrates that several factors influence the imperceptibility of impersonation attacks. We also find that it is more difficult to fool systems if the attacker has to account for the variability in her pose/orientation and other environmental conditions such as lighting, or use the perturbations generated from one DNN model to attack a different model. Based on the measurements, we suggest security countermeasures that could significantly enhance the security of FR based IoT access control. In brief, our contributions in this chapter are :

- We perform an extensive measurement study which shows that the efficacy/imperceptibility of impersonation attacks depend on several factors such as gender, skin color and age. We quantify the extent to which each of these factors affect the attack.
- We perform an in-depth measurement study to understand the feasibility of constructing universal perturbations that make the attack robust to different poses or facial orientations of the attacker. We find that this is much harder in practice from the attacker's perspective.
- We show that the use of multiple DNNs for performing FR (check faces across DNN models) can render imperceptible impersonation attacks almost infeasible.

2.3 Related Work

2.3.1 DNNs based FRSs

A lot of efforts have targeted the design of highly accurate FRSs. Traditional methods applied hand-crafted features like edges and texture descriptors [60, 142, 143, 200], which have been used for a long time. Due to the convenience of obtaining large training data and the availability of inexpensive computing power and memory, the trend towards replacing the traditional methods by deep learning methods is increasing. Deep Convolutional Neural Networks (DCNNs) can automatically extract high level representative features from large datasets and have been shown to be invariant to illumination variations, brightness variations, age variations and/or facial orientation [8]. Today the state-of-the-art FR algorithms are almost all based on end-to-end DCNNs [157, 201, 221, 240, 244]. We use VGG-Face [201] in our analysis. VGG-Face is a 39-layer DCNN, and is one of the most well-known and highly accurate face recognition systems.

2.3.2 Presentation Attacks

It is generally believed that DNN-based FRSs have extremely high recognition accuracy, even better than humans. However, this is based on the implicit assumption that attackers do not actively attempt to fool the system. Recently however, there have been extensive efforts reported in the literature on attacks targeting FRSs [65, 67, 79, 83, 281].

Many early approaches used by attackers to spoof a FRS, are based on using fake target faces, which is termed *presentation attack*. In general, attackers hold a non-real face of a target person in front of the camera to evade the FRS. The attackers could use photographs [7, 149], replayed videos [40, 292], dummy faces (such as 3D masks) [67, 134], or 3D virtual reality facial

(a) original **(b)** impersonation **(c)** impersonation **(d)** impersonation **(e)** impersonation

Figure 2.2: Impersonation attacks using the Fast Landmark Manipulation (FLM) method proposed in [52]. (a) shows the original image; (b)-(e) show four impersonation attacks, within each the left image is the adversarial example and the target identity is shown in the right image.

models displayed on a screen [281] as shown in Fig. 2.1(a). While these methods are shown to successfully lead to attacker misclassification as the target identities, such attacks, they however require the attacker to overtly indulge in action that may seem strange or even suspicious to nearby observers.

2.3.3 Adversarial Examples for FRSs

More recently, general DNN-based classifiers [146, 243, 299] have been shown to be vulnerable to adversarial example attacks. Adversarial Examples (AEs) refer to perturbed inputs, which are correctly classified by humans, but misclassified by machine learning systems. In [224, 226], the authors demonstrate the potential of using adversarial examples to conduct real face attacks on FRSs, i.e., the attackers use their own faces to mount attacks. By wearing special glasses (physical perturbations), the attacker's face can be misclassified by the DDN as shown in Fig. 2.1(b).

In addition to physical AE attacks , various digital AE attack approaches have been proposed, which can be categorized into three kinds as follows.

- *Intensity-based AE attacks.* Imperceptible Perturbations are added to the images to change the intensity of each pixel as shown in Fig. 2.1(c). [88] hypothesizes that DNNs are vulnerable to AE attacks because of their linear nature and thus proposed the fast gradient sign method (FGSM) for efficiently generating perturbations. [137] extends the FGSM method by apply-

ing it multiple times with a small step size. [182] uses a norm minimization based formulation, termed DeepFool, to search for adversarial perturbations by casting it as an optimization problem. [31] introduces new gradient based attack algorithms that are more effective in terms of the adversarial success rates. We use [137] in our analysis since it can generate adversarial perturbations very fast, which is the key requirement for large-scale analysis (needed to generate these perturbations), and at the same time, it achieves very high attack success rates compared to other fast methods.

- *Spatial transformation based AE attacks.* As opposed to manipulating the pixel values, perturbations generated through spatial transformation could result in large L_p distance measures, but are perceptually realistic as shown in Fig. 2.1(d). [273] estimates the displacement field for all pixel locations in the input images. [52] first detects key landmarks of the faces and the displacement field is only defined for the key landmarks.

- *Generation-based AE attacks.* [236] utilize generative models to generate fake face images as shown in Fig. 2.1(e), which are visually similar to the original face images, thus hard to cause noticability; at the same time, these have similar feature representations as the target faces, and are thus recognized as the target individuals.

There are two different kinds of attack goals viz.:

- Dodging, where the attacker seeks to have one face misidentified as any other different face.

- Impersonation, where the attacker seeks to have one face classified as a specific target victim's face, which is harder than the dodging attacks.

While dodging attacks are of interest in evading surveillance, impersonation attacks, which are

(a) $\sigma = 0$ **(b)** $\sigma = 0.828$ **(c)** $\sigma = 1.042$ **(d)** $\sigma = 1.541$ **(e)** $\sigma = 2.074$ **(f)** $\sigma = 2.716$ **(g)** $\sigma = 5.00$

Figure 2.3: Perturbed images for different levels of perturbation σ. In (g) with large value of $\sigma = 2.7160$, patterns are visible on the forehead, left cheek and nose. (The patterns are more visible in color version.)

much more targeted, are of more relevance to IoT security. Attackers can leverage this method to gain unauthorized entry, for instance, by bypassing a smart locking mechanism. Our work thus focuses on impersonation attacks. The spatial transformation based attacks, that is, Fast Landmark Manipulation Method (FLM) and Grouped Fast Landmark Manipulation Methods (GFLM), are proposed for realizing dodging attacks. We extend these two methods to the impersonation attack. We observe that FLM gives largely deformed facial images as shown in Fig. 2.2, which is not imperceptible at all. GFLM, which aims to generate more natural adversarial examples, fails in all the four impersonation attacks. Therefore, it is evident that these types of attacks are not appropriate for impersonation and thus, we do not perform additional measurements on such spatial transformation based attack methods.

We focus on intensity-based AE attacks in our analysis since they are the kind of attacks explored the most in the literature. Intensity-based AE attacks are fast to carry out and have been proven to have extremely high attack success rates. Unlike prior works which simply showcase the possibility of such attacks, we do extensive measurements to provide a detailed view of the potency of such attacks in various scenarios and unearth various factors that affect this potency.

(a) **(b)** **(c)**

Figure 2.4: Perturbed images with restrictions on the location of pixels to be perturbed. In (a), all pixels are to be perturbed, $\sigma = 2.7160$. In (b), only left half of the image is allowed to be perturbed to achieve the same goal as (a). In (c), only top left quarter is allowed to be perturbed to achieve the same goal as (a).

2.4 Imperceptible Impersonation Attack

To ensure that an impersonation attack is imperceptible (i.e., does not raise suspicion for human observers), the attackers should modify the faces such that visibility of the modifications is minimal. In this section, we describe the attack model and how the magnitude of the perturbation are meatured. The lower the magnitude of the perturbation, the higher the imperceptibility [224, 243].

2.4.1 Attack Model

We assume that the attacker mounts the impersonation attack after the system has been trained. This implies that the adversary cannot ”poison” the FRS by altering training data or by injecting mislabeled data. Rather, the adversary can only alter the composition of input images based on the knowledge of the underlying DNN model. Our attack model is consistent with IoT access control attack scenarios where the attacker cannot tamper with the manufacturing of the commercial smart devices. In this chapter, we mainly focus on a white-box model in which the attacker knows the DNN architecture and the parameters of the FRSs being attacked. This is supported by the fact that it is possible to train local models that can infer the functionality of the target FRSs [229] and carry transfer attacks to the target FRSs. However, in Section 2.5.5, we also examine a black-box

model by evaluating how well the perturbed images generated for one model can be successful in the impersonation attack on another model.

2.4.2 Perturbation Vector

Suppose the finite set of people's identities (i.e., labels) to be detected by the FRS is $\mathcal{C}$, with $|\mathcal{C}| = N$. Further, suppose that each input image is given as an RGB vector $\mathbf{x}$ and the ground truth label of $\mathbf{x}$ is given by $c_x \in \{1, 2, \cdots, N\}$.

A DNN-based FRS implements a high-dimensional non-linear function which maps an input $\mathbf{x}$ to an output probability vector $f(\mathbf{x})$ of length N, where each element in the output vector represents the probability that $\mathbf{x}$ matches the corresponding label. In addition, the label that corresponds to the largest entry in $f(\mathbf{x})$ is output as the recognition result. Consequently, a correct recognition result is realized when c_xth entry of $f(\mathbf{x})$ is the maximum entry. Thus, the ideal output $f(\cdot)$ is a one-hot vector, i.e., only the c_xth entry has value 1 and all the other entries are zero.

To impersonate a target c_t, the attacker with an input image vector $\mathbf{x}_a$ thus finds a perturbation vector $\mathbf{r}$ such that c_tth entry of $f(\mathbf{x}_a + \mathbf{r})$ is the maximum one. To measure the error in the output of the FRS with the adversarial input $\mathbf{x}_a + \mathbf{r}$, we adopt the *softmaxloss* score [201]. For an input vector $\mathbf{x}_a$ and a given label c_t, the *softmaxloss* function is defined as:

$$softmaxloss(f(\mathbf{x}_a), c_t) = -\log(\frac{e^{<h_{c_t}, f(\mathbf{x}_a)>}}{\sum_{c=1}^{N} e^{<h_c, f(\mathbf{x}_a)>}}), \tag{2.1}$$

where $< \cdot, \cdot >$ denotes inner product between two vectors and h_c is the one-hot vector corresponding to label c. Note that the value of *softmaxloss* score is low when the DNN outputs the label as c_t and high otherwise. The attacker's goal is to achieve a $softmaxloss(f(\mathbf{x}_a + \mathbf{r}), c_t)$ that is low enough such that c_xth entry of $f(\mathbf{x})$ is the maximum entry, while minimizing $||\mathbf{r}||$. In other words,

Algorithm 1 Computing perturbation vector.

Input : image $\mathbf{x}_a$, target identity c_t

Output: Output: impersonation perturbation $\mathbf{r}$

Initialize $\mathbf{r} \leftarrow \mathbf{0}$

do

  $\Delta\mathbf{r} = \text{argmin } softmaxloss(f(\mathbf{x}_a + \mathbf{r} + \Delta\mathbf{r}), c_t)$

  Quantize the additional perturbation: $\Delta\mathbf{r}' \leftarrow \Delta\mathbf{r}$

  Update the perturbation: $\mathbf{r} \leftarrow \mathbf{r} + \Delta\mathbf{r}'$

while $\mathbf{x}_a + \mathbf{r}$ *is not recognized as* c_t;

the attacker solves the following optimization problem.

$$\mathbf{r}^* = \arg\min_{\mathbf{r}} softmaxloss(f(\mathbf{x}_a + \mathbf{r}), c_t) + \alpha||\mathbf{r}||. \tag{2.2}$$

In (2.2), α is weight factor used to balance impersonation error and imperceptibility. As discussed in § 2.3, BIM algorithm [137] as shown in Algorithm 1 is used to solve this optimization problem.

2.4.3 Measuring Imperceptibility

Using (2.2), the attacker can always find perturbation vectors that allow desired misclassification of input vectors. However, the produced attack image, i.e., $\mathbf{x}_a + \mathbf{r}^*$ is not guaranteed to be "imperceptible" to humans. In other words, the perturbation vector could be too large. This would cause the produced perturbed image to be quite distinguishable from the original attacker image. To quantify the effectiveness of the attack in various settings, we measure per pixel per color channel magnitude of perturbation using the root mean square error (RMSE) between the original

and perturbed images. In particular, suppose that the images are of width W, height H and number of color channels D. Let the total number of dimensions in an image vector be $M = W \times H \times D$. Given an input and perturbed image vectors $\mathbf{x}, \mathbf{x}' \in \{0, 1, \cdots, 255\}^M$, the RMSE (we also use the term "noise level") is given by the following.

$$\sigma(\mathbf{x}, \mathbf{x}') = \sqrt{\frac{1}{M} \sum_{i=1}^{M} (x(i) - x'(i))^2}, \tag{2.3}$$

where $x(i)$ is the ith component of $\mathbf{x}$, and σ is in pixel-value units, where $\sigma \in [0, 255]$.

To get a sense of what values of σ renders a perturbed attack image easy to identify, we show images of an attacker with varying levels of perturbation in Fig. 2.3. We note that for $\sigma > 2$, it is easy to identify the noisy pixels in the perturbed images.

2.4.4 Physical Imperceptibility

If the attackers want to realize this perturbation physically (via using various paraphernalia such as dummy faces, or 3D-printed glasses), the amount of perturbation will need to be limited in terms of either (a) the maximum number of pixels to which the noise is added, or (b) the locations of those pixels [224], or (c) both. In Fig. 2.4, we study the effects of such limitations. We fix the attacker image and a target label, and then find the adversarial images when the entire image can be perturbed, as well as when only the left half and top left quarters of the image pixels are to be perturbed. As shown in the figure, the noise level increases significantly and the pattern is perceptible. Thus, one can expect the attack to be much harder in these cases. In the rest of the chapter, we only consider scenarios in which the full attacker image is subject to perturbation. This reflects a worst case scenario analysis from the defender's perspective. Even in this scenario, we show that it can be hard for an attacker to launch the attack in all possible scenarios.

(a) Impersonating A.J. Buckley, $\sigma = 0.88$ **(b)** Impersonating Adam Buxton, $\sigma = 1.65$ **(c)** Impersonating Boris Kodjoe, $\sigma = 2.26$

Figure 2.5: Different noise levels needed for Micheal Crichton to impersonate three different identities. It is rather easy for Micheal Crichton to impersonate A.J. Buckley; and hard to impersonate Boris Kodjoe

	1.	2.	3.	4.	5.	6.	7.	8.	9.	10.
a.	**2.55**	**2.28**	1.26	1.23	1.89	1.95	**3.20**	**3.15**	**3.17**	1.57
b.	**2.12**	1.87	2.00	**2.45**	1.70	**2.04**	**2.38**	1.98	**2.39**	**2.65**
c.	**2.14**	1.30	1.56	1.39	1.63	**2.61**	**2.27**	**5.96**	**3.03**	1.35
d.	**2.75**	1.99	1.78	1.97	1.54	1.81	**2.09**	1.78	1.66	1.98

Figure 2.6: Noise level σ required for an attacker (a-d) to impersonate a target (1-10). It is easier for the considered attackers (who are all male with pale skin color) to impersonate targets who are also male with pale skin color, as compared to impersonating other targets. The noise levels needed to impersonate target 1 are all large. Impersonating targets 6-10 seems to require larger noise levels.

2.5 Experiments

In this section, we detail the results of our measurement study towards getting an in depth understanding of the practicality of imperceptible impersonation attacks on DNN-based FRS and the factors that influence such attacks.

2.5.1 Experimental Setup

The FRS used in our experiments is VGG-Face [201], one of the most well-known and highly accurate face recognition systems as discussed in § 2.3. The analysis is based on the VGG-

Face dataset [201], which contains $N = 2622$ identities of celebrities, and approximately 1000 facial images per identity; this translates to a total of about 2.6 million images.

2.5.2 Case Studies

To begin with, we use the face image of Micheal Crichton as the attacking image,(i.e., the input) and study whether some targeted individuals are harder than others to impersonate with the attacking image. Fig. 2.5 shows the minimum perturbations needed for the attacking image to impersonate three different individuals. We observe that it is rather easy for Micheal Crichton to impersonate A.J. Buckley. However, when it comes to impersonating Boris Kodjoe, the perturbation gets larger and is noticeable by human.

For a more general case study, Fig. 2.6 shows the noise level σ needed to achieve a successful attack by each attacker depicted on the column, to impersonate each target depicted on the row. It is clear that, different attackers need different values of σ to successfully impersonate different targets. Interestingly, the patterns of large perturbations (marked in red) seen in Fig. 2.6 suggest that it is easier for the considered attackers (e.g., who are all male with pale skin color) to impersonate targets who are also male with pale skin color, as compared to impersonating other targets. In addition, the noise levels needed to impersonate target 1 are all large, which is possibly due to a difference in gender. Furthermore, we see that impersonating targets 6-10 seems to require larger noise levels. This can be attributed to differences in skin color, age, or a combination of both. This motivates our study to further examine the impact of these factors in Section § 2.5.3.

Having performed the above preliminary studies, we next look at the statistical distribution of the ability of an attacker to impersonate different targets, subject to a constraint on the noise

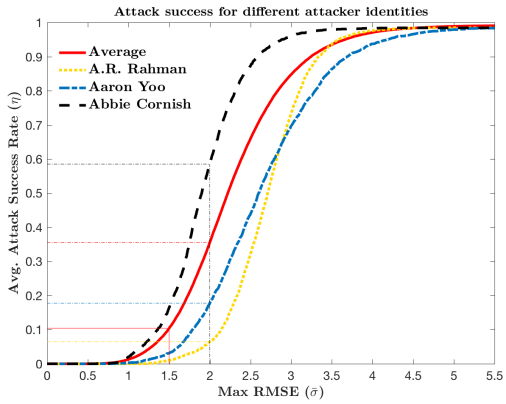


Figure 2.7: Impersonation attack performance. Abbie Cornish (female, white, young) can more successfully impersonate others, on average, compared to A.R. Rahman (male, Indian, young) and Aaron Yoo (male, Asian, young).

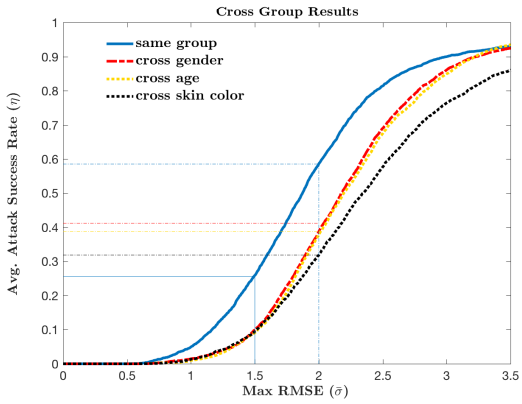


Figure 2.8: Cross group impersonation attack performance. It is easier for an attacker to impersonate a target identity having the same attributes (gender, skin color, age). Impersonation across different skin color is the most hardest.

level $\sigma \leq \bar{\sigma}$. We define the attack success rate $\eta(\bar{\sigma})$ as the percentage of target labels which an attacker can impersonate for a given $\bar{\sigma}$. In Fig. 2.7, we show the success rates η for three different attackers impersonating all other remaining labels in the VGG-Face dataset. One can see that Abbie Cornish (female, white, young) can more successfully impersonate others, on average, compared to A.R. Rahman (male, Indian, young) and Aaron Yoo (male, Asian, young). For example, with the threshold $\bar{\sigma} = 2$, Abbie Cornish can successfully impersonate 58% of all the labels while A.R. Rahman achieves a success rate of only 6.5% and Aaron Yoo achieves a success rate of 17.7%. This could be attributed to the fact that the VGG-Face dataset contains more white people than people of other races. We observe that the gender distribution is almost balanced in the dataset.

To get aggregate results, we randomly sample the VGG-Face dataset to get a 100-identity subset $\mathcal{S}$. We fix each identity in $\mathcal{S}$ as a specific attacker, and then find the perturbation vector with each of the remaining labels in $\mathcal{S}$ as targets, and we compute $\eta(\bar{\sigma})$ for each attacker for a range of values of $\bar{\sigma}$. We then repeat this experiment 10 times and compute the average attack success rate

across attackers in all the samples. The results show that, on average, it is not easy for an attacker to impersonate *any* target identity. In particular, with $\bar{\sigma} = 1.5$, the success rate is only 10.4%. We take a deeper look into how the success rate breaks down within different groups of people in the following section.

2.5.3 Factors That Influence the Attack

Next, we take a closer look at the extent to which various factors, discussed in § 2.5.2, influence an attacker's ability to carry out an imperceptible impersonation attack. Specifically, we consider different groups of identities based on gender, skin color, and age attributes. We manually label the dataset to produce four groups: (a) white young male (100 identities), (b) white young female (100 identities), (c) black young male (69 identities), and (d) white old male (100 identities). We do not consider other attribute combinations, such as black young female, or white old female, because the majority of the images in the VGG-Face dataset are for white skin color, and young people. For group (c), we only have 69 identities due to limitedness of data points matching such attributes. To reduce errors in labeling, each of the authors of the chapter manually labeled the dataset independently and we considered only the images with unanimously common labels in our group dataset. In addition, when labeling, we discard an identity whenever its attributes are hard to label manually.

To investigate the impact of the aforementioned attributes on the imperceptible impersonation attack, we conduct four experiments based on the four groups:

- Take people in group (a) as attackers trying to impersonate the other people in group (a); this case reflects *same group* impersonation measurements;

- take people in group (a) as attackers trying to impersonate people in group (b); this case represents *cross gender* impersonation measurements;
- take people in group (a) as attackers trying to impersonate people in group (c); this case reflects *cross skin color* impersonation measurements;
- take people in group (a) as attackers trying to impersonate people in group (d); this case counts for *cross age* impersonation measurements.

In Fig. 2.8, we plot the average attack success rate versus different perturbation constraint $\bar{\sigma}$, for each of the four aforementioned experiments. We note that it is easier for an attacker to impersonate a target identity having the same attributes (gender, skin color, age). For the same group experiment, with the threshold $\tilde{\sigma} = 1.5$, the success rate is 25.65%. Recall that the aggregate success rate in § 2.5.2 is only 10.4%. Moreover, as shown in the figure, it is relatively easier for an attacker to impersonate a target with a different gender or age than to impersonate a target with different skin color. For example, with the threshold $\tilde{\sigma} = 2$, the success rate for cross skin color is only 31.85% while the success rate for cross age and gender are around 40%. These results seem consistent with (and can be explained by) observations that have been previously reported in computer vision literature [42, 128, 227, 251]. Specifically, these papers show that in several scenarios, shape and texture cues suffer from degradation (affecting age or gender) and the color feature becomes dominant [287]. Thus, we conclude that the VGG-Face model relies less on features such as shape and texture as compared to color.

2.5.4 Universal Perturbation Results

In a realistic setting, an attacker may want one universal perturbation to impersonate the target identity for all the face images captured in different settings such as pose, camera angle, and lighting conditions. In order to launch the impersonation attack in the presence of these variations, an attacker will need to find a single perturbation vector $\tilde{\mathbf{r}}$ that allows misclassification of a set of his/her own images $\mathcal{X}_a$ of size K, to the target victim label, thus accounting for as many conditions as possible. In other words, the attacker needs to construct a vector $\tilde{\mathbf{r}}$ such that $f(\mathbf{x}_a + \tilde{\mathbf{r}}) = c_t$, $\forall \mathbf{x}_a \in \mathcal{X}_a$ for some given target label c_t.

The approach for calculating $\tilde{\mathbf{r}}$ is similar to the one described in § 2.4.2. The only difference is that now the objective function changes to the following.

$$\tilde{\mathbf{r}} = \arg\min_{\mathbf{r}} \sum_{\mathbf{x}_a \in \mathcal{X}_a} softmaxloss(f(\mathbf{x}_a + \mathbf{r}), c_t) + \alpha||\mathbf{r}||. \tag{2.4}$$

The condition to stop the iterations now requires that all K images to be misclassified as the target label, upon adding on the same perturbation vector.

In Fig. 2.9, we show an example of an attacker with three images, $\mathcal{X}_a = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$. In Fig. 2.10, we show the output perturbed image $\mathbf{x}_1 + \tilde{\mathbf{r}}$ when $\tilde{\mathbf{r}}$ is computed using only image $\{\mathbf{x}_1\}$, images $\{\mathbf{x}_1, \mathbf{x}_2\}$, and all the images $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$, respectively. It is evident that the attacker image is more perceptible as more attack images are considered in computing the universal perturbation vector $\tilde{\mathbf{r}}$.

In Fig.2.11, we plot the average success rate for an attacker employing universal perturbation. Here, we randomly sample 100 identities from the VGG-Face dataset and let them imper-

Figure 2.9: A set of face images of Micheal Crichton. $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$

Figure 2.10: Universal perturbations visualization. Three perturbations are universal to different number of attacking images. Left: universal for $\{\mathbf{x}_1\}$, $\sigma = 1.7509$; Middle: universal for $\{\mathbf{x}_1, \mathbf{x}_2\}$, $\sigma = 3.6027$; Right: universal for $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$, $\sigma = 7.8877$. The perturbations are more perceptible as more attacking images are considered in computing the universal perturbation vector.

sonate each other. We conduct this experiment 10 times and average the results. The results show that the success rate is strictly decreasing when a universal perturbation vector is required to perturb multiple attacker images. More importantly, the attackers' ability to impersonate a given target is significantly reduced with even slight increases in K. For example, the success rate with threshold $\tilde{\sigma} = 2$ is 39.9% for $K = 1$ (the case considered in § 2.5.2 and § 2.5.3). However, when we increase K to 2, the success rate drops dramatically to 2.28% and the success rate when $K = 3$ is only 0.6%, which suggests that the impersonations can almost fail all the time, if the attacker seeks to be imperceptible.

2.5.5 Cross Model Measurements

Recently, it has been shown that adversary examples that are successfully misclassified by one trained DNN model can also cause misclassifications in other (different) DNN models that have different hyperparameters [180, 243]. However, it is unclear whether different models could misclassify the perturbed images to the same target classes, which is the key characteristic for

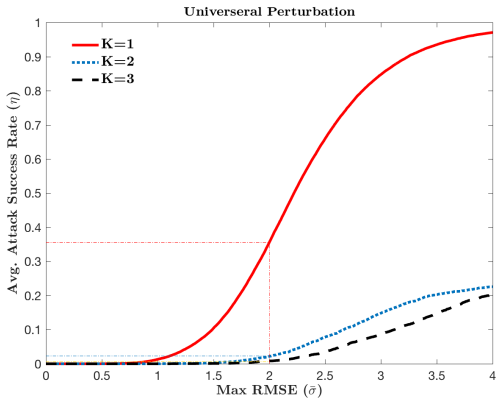


Figure 2.11: Universal perturbation impersonation attack performance. K is the number of attacking images to generate the universal perturbations. The attackers' ability to impersonate given targets is significantly reduced when the perturbations are required to be universal to multiple attacking images.

determining if white-box impersonation attacks can easily extend to black-box attacks. To check whether our perturbed images targeting impersonation generalize across different DNN models, we fine-tune the AlexNet DNN [136] on the VGG-Face dataset, and test the impersonation attack success ratio on the AlexNet model using the perturbed images generated using VGG-Face model.

We test $10,000$ perturbed images generated by VGG-Face, and find that *none* are classified as the victims by the AlexNet, but most of them are misclassified by the AlexNet. This significant result indicates that impersonation attacks do not easily transfer across different DNN models. It will be extremely hard for the attacker to use the perturbation vectors to fool a DNN model different from the one used to generate them. Thus cross model validation could significantly enhance the robustness of face recognition based access control in IoT systems.

2.5.6 Detecting and removing perturbations

Finally, we test whether de-noising [203] (which could be used by an IoT access control system) affects the potency of the attack. Three standard de-noising filters are considered in our

experiments: average filter, median filter, and Wiener filter. We test 100 different perturbed images, and find that all of them are still misclassified as the targets. This suggests that de-noising does not hurt the attack. This is because de-noising filters assume a certain pattern of noise, which is unlikely to be what is used by the attacker for generating the perturbations.

We conclude that traditional noise detection and de-noising algorithms are not helpful in countering the imperceptible impersonation attack since the perturbation generated is structured.

2.5.7 Summary of Results

Below is a summary of our take-aways based on the results in § 2.5.2 to § 2.5.6. **(a)** DNNs are vulnerable to adversary examples. However, in contrast to recent work in the literature, we find that the average success rates of the imperceptible impersonation attack are low. **(b)** Attackers can achieve better success rates by choosing targets with similar attributes; in particular choosing targets with same skin color helps. **(c)** When variations, such as pose, camera angle and lighting conditions are considered, the attack is significantly less successful. **(d)** Perturbed images do not generalize well across different DNN models. **(e)** Current noise estimation and de-noising methods do not adversely impact the imperceptible impersonation attack.

2.6 Conclusion

The security of face recognition is an important toptic as face recognition is more and more used in IoT access control. In this chapter, we perform an in-depth measurement study of the generality and efficacy of imperceptible impersonation attacks that have recently gained popularity. Our study is done using a very large dataset. We find that it is hard for a given adversary to imper-

sonate an arbitrary target victim without making perceptible changes to her face. Further, we show that several factors such as age, race and gender of the attacker and victim influence the efficacy of the attack and we quantify the impact of each. We also show that, in a realistic scenario where the attacker seeks to be robust to different poses or variations in environmental conditions, the attack becomes more difficult or even impossible. Based on this, we suggest the use of cross-model verifications as well as multi-views, which can potentially counter such attacks very effectively.

Acknowledgments

This research was partially sponsored by the U.S. Army Combat Capabilities Development Command Army Research Laboratory and was accomplished 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 here on.

Chapter 3

Stealthy Adversarial Perturbations Against Real-Time Video Classification Systems

3.1 Abstract

Recent research has demonstrated the brittleness of machine learning systems to adversarial perturbations. However, the studies have been mostly limited to perturbations on images and more generally, classification tasks that do not deal with real-time stream inputs. In this chapter we ask ”Are adversarial perturbations that cause misclassification in real-time video classification systems possible, and if so what properties must they satisfy?” Real-time video classification systems find application in surveillance applications, smart vehicles, and smart elderly care and thus, misclassification could be particularly harmful (e.g., a mishap at an elderly care facility may be

missed). Video classification systems take video clips as inputs and these clip boundaries are not deterministic. We show that perturbations that do not take "the indeterminism in the clip boundaries input to the video classifier" into account, do not achieve high attack success rates. We propose novel approaches for generating 3D adversarial perturbations (perturbation clips) that exploit recent advances in generative models to not only overcome this key challenge but also provide stealth. In particular, our most potent 3D adversarial perturbations cause targeted activities in video streams to be misclassified with rates over 80%. At the same time, they also ensure that the perturbations leave other (untargeted) activities largely unaffected making them extremely stealthy. Finally, we also derive a single-frame (2D) perturbation that can be applied to every frame in a video stream, and which in many cases, achieves extremely high misclassification rates.

3.2 Introduction

Deep Neural Networks (DNN) based real-time video classification systems are being increasingly deployed in real world scenarios. Examples of applications include video surveillance [239], self driving cars [130], health-care [260], etc. To elaborate, video surveillance systems capable of automated detection of "targeted" human activities or behaviors (e.g., accident, violence), can trigger alarms (upon detection) and drastically reduce information workloads on human operators. Without the assistance of DNN-based classifiers, human operators will need to simultaneously monitor footage from a large number of video sensors. This can be a difficult and exhausting task, and comes with the risk of missing behaviors of interest and slowing down decision cycles. In self-driving cars, video classification has been used to understand pedestrian actions and make navigation decisions [130]. Similar applications can be envisaged in the Army Next Generation Combat

Vehicle (NGCV). Real-time video classification systems have also been deployed for automatic "fall detection" in elderly care facilities [260], and detection of abnormal actions around automated teller machines [254]. All of these applications directly relate to the physical security or safety of people and property. Thus, stealthy attacks on such real-time video classification systems are likely to cause unnoticed pecuniary loss and compromise personal safety. Note that while objects can be detected or distinguished by examining the individual frames in a video (akin to object detection on images), many activities can only be recognized or distinguished by considering a sequence of frames holistically (i.e., a clip consisting of multiple frames).

Recent studies have shown that virtually all DNN-based systems are vulnerable to well-designed adversarial inputs [86, 181, 182, 224, 243], which are also referred to as *adversarial examples*. Szegedy *et al.* [243], showed that adversarial perturbations that are hardly perceptible to humans can cause misclassification in DNN-based image classifiers. Goodfellow *et al.* [87], analyzed the potency of realizing adversarial samples in the physical world. Moosavi *et al.* [181], and Mopuri *et al.* [184], introduced the concept of "image-agnostic" perturbations. Recent efforts by Hosseini et al. [104], and Wei et al. [265], explore adversarial perturbations on videos. However, they are limited in that their attack models do not work on real-time video classification systems (more details in § 3.10).

The high level question that we try to address in this chapter is: *"Is it possible to launch stealthy attacks against DNN-based real-time video classification systems by adding adversarial perturbations on a video stream, and if so how?"* In contrast with the aforementined prior work, attacking a real-time video classifier poses new challenges that were not all previously identified or addressed. *First*, because video streams are collected in real-time, the corresponding perturbations

also need to be generated on-the-fly with the same frame rate which can be extremely computationally intensive. *Second*, to make the attack stealthy, attackers would want to add perturbations on the video in such a way that they will only cause misclassification for the targeted (possibly malicious) activities, while keeping the classification of other activities unaffected. In a real-time video stream, since the activities change across time, it is hard to identify online and in one-shot [70], the target frames on which to add perturbations (and thereby ensure that the other untargeted activities are not affected). *Third*, real-time video classifiers use video clips (a set of frames) as inputs [70, 254] (i.e., as video is captured, it is broken up into clips and each clip is fed to the classifier). This introduces two additional hyper-parameters viz., the *length* of a clip and the *boundaries* (i.e., beginning and ending) of a clip. Even if attackers are aware of the length of each clip, it is hard to predict the boundaries of the clips as they are non-deterministic. This is problematic because when the attacker generated perturbations are applied to the wrong frame within a clip (i.e., perturbation for frame 1 of a clip being applied to frame 2 of that clip), the perturbations may not work as expected. (Please see Figure 3.4 and the associated discussion for more details).

In this chapter, our first objective is to investigate how to generate adversarial perturbations against real-time video classification systems by overcoming the above challenges. We resolve the real-time challenge by using *universal perturbations* (UP) [181]. UPs are universal in the sense that a UP is not specific to one input example, but works on any input example from the same distribution as that of the training data. Universal perturbations affect the classification results by using just a (single) set of perturbations generated off-line. Because they work on unseen inputs, they preclude the need for intensive on-line computations to generate perturbations for every incoming video clip. To generate such universal perturbations, we leverage generative DNN models.

However, adding universal perturbations to all clips of the video can cause misclassification of all the activities in the video stream. This may expose the attack since the results may be abnormal (e.g., many people performing rare actions). It may even cause activities from other classes to be mis-classified as the target class. To make the attack stealthy, we introduce the novel concept of *dual purpose universal perturbations*, which we define as universal perturbations which only cause misclassification of activities belonging to the target class, while minimizing, or ideally, having no effect on the classification results for activities belonging to the other classes.

Dual purpose universal perturbations by themselves do not provide high success rates because of the nondeterminism of the clip boundaries. To be more specific, let l be the length of a clip input to the classifier, and $p = \{p_1, p_2, \ldots, p_l\}$ be the perturbations for a clip of frames $x = \{f_1, f_2, \ldots, f_l\}$; then input $x' = \{f_1 \oplus p_1, f_2 \oplus p_2, \ldots, f_l \oplus p_l\}$, where $\oplus$ denotes pixel-wise addition, would yield misclassification but other combinations like $x'' = \{f_1 \oplus p_l, f_2 \oplus p_1, \ldots, f_l \oplus p_{l-1}\}$ (where p_l in the latter expression refers to the last frame in the previous clip) may not cause misclassification. To solve this problem, we introduce a new type of perturbation that we call the *Circular Dual Purpose Universal Perturbation (C-DUP)*. The C-DUP is a 3D perturbation which is effective on a video stream even in the presence of a temporal misalignment between the perturbation clips and the input video clips. Specifically, any cyclic permutations of a C-DUP perturbation clip are also still valid perturbations. For example, both $\{f_1 \oplus p_l, f_2 \oplus p_1, \ldots, f_l \oplus p_{l-1}\}$ and $\{f_1 \oplus p_{l-1}, f_2 \oplus p_l, \ldots, f_l \oplus p_{l-2}\}$ can cause expected misclassification. Because of this property, C-DUP works even if the sequential concatenation of two broken up parts of two consecutive perturbation clips, is added to an input video clip as a perturbation clip. To generate C-DUPs, we make significant changes to the baseline generative model used to generate universal perturbations. In

particular, we add a new unit to generate circular perturbations, that is placed between the generator and the fixed discriminator (as discussed later). We demonstrate that the C-DUP is very stable and effective in achieving real-time stealthy attacks on video classification systems.

Finally, to better understand the effect of the temporal dimension, we also investigate the feasibility of attacking the classification systems using a simple and light 2D perturbation (frame level instead of clip level) which is applied across all the frames of a video. By tweaking our generative model, we are able to generate such perturbations which we name as *2D Dual Purpose Universal Perturbations (2D-DUP)*. These perturbations work well on a sub-set of videos, but not all. We will discuss the reasons for this when we describe these 2D attacks in § 3.7.4.

Our Contributions: In brief, our contributions are:

- We provide a comprehensive analysis of the challenges in crafting adversarial perturbations for real-time video classifiers. We empirically identify what we call the boundary effect phenomenon in generating adversarial perturbations against video streams (see § 3.7.2). In a nutshell, the boundary effect arises because of the nondeterminism of the boundaries of the clips input to the video classification system.
- We design and develop a generative framework to craft two types of stealthy adversarial perturbations against real-time video classifiers, viz., the circular dual purpose universal perturbation (C-DUP) and the 2D dual purpose universal perturbation (2D-DUP). These perturbations are agnostic to (a) the content the video streams capture (i.e., are universal) and (b) the clip boundaries within the streams.
- We demonstrate the potency of our adversarial perturbations using two different video datasets. In particular, the UCF101 dataset captures coarse-grained activities (human actions such as ap-

plying eye makeup, bowling, drumming) [238]. The Jester dataset captures fine-grained activities (hand gestures such as sliding hand left, sliding hand right, turning hand clockwise, turning hand counterclockwise) [55]. We are able to launch stealthy attacks on both datasets with over a 80 % misclassification rate, while ensuring that the other classes are correctly classified with relatively high accuracy.

3.3 Background

In this section, we provide the background relevant to our work. Specifically, we discuss how a real-time video classification system works and what standard algorithms are currently employed for action recognition.

3.3.1 Real-time Video-based Classification Systems

DNN based video classification systems are being increasingly deployed in real-world scenarios. Examples include fall detection in elderly care [80], abnormal event detection on campuses [257,258], security surveillance for smart cities [259], and self-driving cars [130,131]. Given an input real-time video stream, which may contain one or more known actions, the goal of a video classification system is to correctly recognize the sequence of the performed actions. Real-time video classification systems commonly use a *sliding window* to extract video clips and use the clips as inputs to a classifier to analyze the video stream [70, 254]. The classifier computes an output score for each class in each sliding window. The sliding window moves with a stride. Moving in concert with the sliding window, one can generate "score curves" for each action class. Note that the scores for all the action classes evolve with time. The score curves are then smoothed (to remove

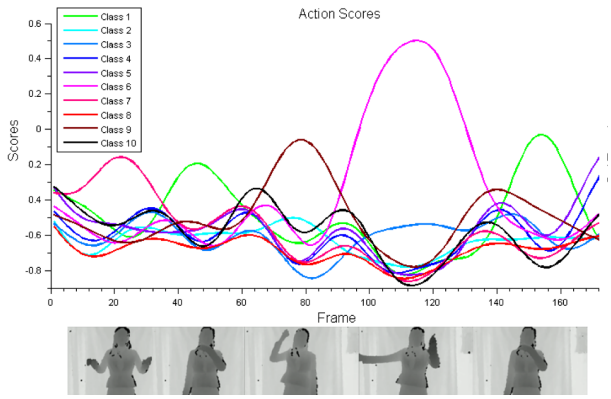


Figure 3.1: This figure [70] illustrates the score curves computed by a video classifier with a sliding window for every class. Real-time video classification systems use these score curves to do online action recognition.

noise) as shown in Figure 3.1. With the smoothed score curves, the on-going actions are predicted online. From the figure one can see that, the real-time video classification system is fooled if one can make the classifier output a low score for the true class in each sliding window; with this, the true actions will not be recognized.

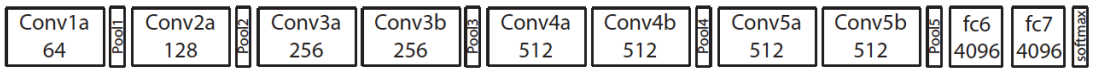


Figure 3.2: The C3D architecture [253]. C3D net has 8 convolution, 5 max-pooling, and 2 fully connected layers, followed by a softmax output layer. All 3D convolution kernels are $3 \times 3 \times 3$ with a stride [253] of 1 in both spatial and temporal dimensions. The number of filters is shown in each box. The 3D pooling layers are represented as pool1 to pool5. All pooling kernels are $2 \times 2 \times 2$, except for pool1, which is $1 \times 2 \times 2$. Each fully connected layer has 4096 output units.

3.3.2 The C3D Classifier

Convolutional neural networks (CNNs) are being increasingly applied in video classification. Among these, spatio-temporal networks like C3D [253] and two-stream networks like I3D [32] outperform other network structures [91, 100]. However, two-stream networks require optical flow extraction as preprocessing. Without the requirement of non-trivial pre-processing on the video

stream, spatio-temporal networks are more efficient and suitable for real-time applications; among these, C3D is the start-of-art model [91].

Given its desirable attributes and popularity, without loss of generality, we use the C3D model as our attack target in this chapter. The C3D model is based on 3D ConvNet (a 3D convolutional neural network or CNN) [129, 253, 256], which is very effective in modeling temporal information (because it employs 3D convolution and 3D pooling operations). The architecture and hyperparamters of C3D are shown in Figure 3.2. The input to the C3D classifier is a clip consisting of 16 consecutive frames. This means that upon using C3D, the sliding window size is 16. Both the height and the width of each frame are 112 pixels and each frame has 3 (RGB) channels. The last layer of C3D is a softmax layer that provides a classification score with respect to each class.

3.4 Threat Model and Datasets

In this section, we describe our threat model. We also provide a brief overview of the datasets we chose for validating our attack models.

3.4.1 Threat Model

We consider a white-box model for our attack, i.e., the adversary has access to the training datasets used to train the video classification system, and has knowledge of the deep neural network model used in the real-time classification system. We assume that the datasets are trusted. We also assume that the adversary is capable of injecting perturbations in the real-time video stream. In particular, we assume the adversary to be a man-in-the-middle that can intercept and add perturbations to streaming video [139], or that it could have previously installed a malware that is able to add

perturbation prior to classification [191].

We assume that the adversaries seek to be stealthy i.e., they want the system to only misclassify malicious actions without affecting the recognition of the other actions. So, we consider two attack goals. *First*, given a target class, we want all the clips from this class to be misclassified by the real-time video classifier. *Second*, for all the clips from other (non-target) classes, we want the classifier to correctly classify them.

We point out here that a man-in-the-middle attacker will be unable to simply replace the streaming video with static frames or pre-recorded video and yet achieve the required stealthiness. This is because of two reasons. First, the attacker has no a priori knowledge about "when" a targeted action occurs. For example, an attacker with malicious intent may want to misclassify the action of an elderly person falling down at a smart elderly care center that is monitored by multiple cameras (e.g., [205]). Since the attacker does not know when and where exactly an elderly person will fall down, it has to replace the video streams from *all* the cameras with the pre-recorded video of elderlies doing something else (e.g., walking) *for extended periods or ideally all the time*. However, it is hard to guarantee that the replaced videos are visually similar to the real-time environment (e.g., people and their actions, weather) and replaying videos out of context may be noticeable. In addition, it is possible that the attacker may be capable of delaying the video by a short period to inject targeted perturbations against specific activities; however, while such an approach can eliminate universal and stealth requirements, it cannot overcome the boundary effect and cannot obviate the corresponding computation needed for online perturbation generation. Second, the attacker also has to replace the actions of multiple people involved at the facility captured with the multiple cameras. In other words, a large number of replacement videos capturing a large set of people at the facility

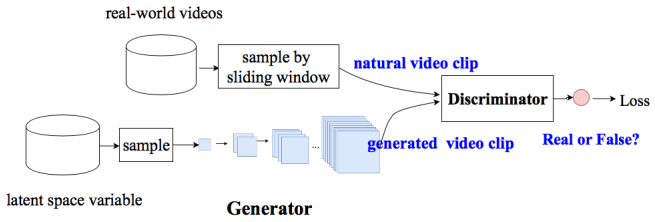
will be necessary. If the replaced videos show the same person at different locations, or people who are not at the facility, this will be noticeable. Applying perturbations on the video will enable the attacker to stealthily misclassify only the specific activity relating to the falling an elderly, keeping all other actions unaffected. Furthermore, the imperceptibility of these perturbations will not cause any human operator to notice anything overtly wrong.

3.4.2 Our Datasets

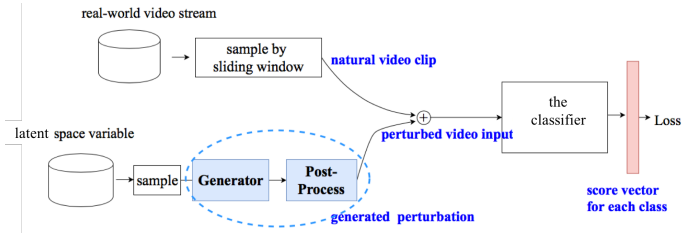
We use the human action recognition dataset UCF-101 [238] and the hand gesture recognition dataset 20BN-JESTER dataset (Jester) [55] to validate our attacks on video classification systems. We use these two datasets because they represent two kinds of classification, i.e., coarse-gained and fine-grained action classification.

The UCF 101 dataset: The UCF 101 dataset used in our experiments is the standard dataset collected from Youtube. It includes 13320 videos from 101 human action categories (e.g., applying lipstick, biking, blow drying hair, cutting in the kitchen etc.). The videos collected in this dataset have variations in camera motion, appearance, background, illumination conditions etc. Given the diversity it provides, we consider the dataset to validate the feasibility of our attack model on coarse-grained actions. There are three different (pre-existing) splits [238] in the dataset; we use split 1 for both training and testing, in our experiments. The training set includes 9,537 video clips and the testing set includes 3,783 video clips.

The Jester dataset: The 20BN-JESTER dataset (Jester) is a recently collected dataset with hand gesture videos. These videos are recorded by crowd-sourced workers performing 27 kinds of gestures (e.g., sliding hand left, sliding two fingers left, zooming in with full hand, zooming out with



(a) GAN Architecture



(b) Our Architecture

Figure 3.3: We use a GAN-like architecture for the generative model. However, our architecture is different from GAN in the following aspects: 1) The discriminator is a pre-trained classifier we attack, whose goal is to classify videos, and not to distinguish between the natural and synthesized inputs; 2) The generator generates perturbations, and not direct inputs to the discriminator, and the perturbed training inputs are fed to discriminator; 3) The learning objective is to let the discriminator misclassify the perturbed inputs.

full hand etc.). We use this dataset to validate our attack with regard to fine-grained actions. Since this dataset does not currently provide labels for the testing set, we withhold a subset of the training set as our validation set and use the validation set for testing. The training set has 148,092 short video clips and our testing set has 14,787 short video clips.

3.5 Generating Perturbations for Real-time Video Streams

From the adversary's perspective, we first consider the challenge of attacking a real-time video stream. In brief, when attacking an image classification system, the attackers usually take the following approach. First, they obtain the *target* image that is to be attacked with its true label.

Next, they formulate a optimization problem wherein they try to compute the "minimum" noise that is to be added (towards imperceptibility) in order to cause a mis-classification of the target. The formulation takes into account the function of the classifier, the input image, and its true label. Backpropagation is commonly used to solve this optimization problem [87, 137, 173].

In the context of real-time video classification, the video is not available to the attackers a priori. Thus, they will need to create perturbations that can effectively perturb an incoming video stream, whenever a *target class* is present. Generation of online perturbations based on an incoming video stream would have an associated cost of $O(f \times b \times n)$ where, f is frame rate, b is cost of one backpropagation on the DNN, and n is the number of backpropagations needed to solve the optimization problem.

Our approach is to compute the perturbations offline and apply them online, and thus, the online computation cost is $O(1)$. Since we cannot predict what is captured in the video, we need perturbations which work with unseen inputs. A type of perturbation that satisfies this requirement is called the Universal Perturbation (UP), which has been studied in the context of generating adversarial samples against image classification systems [181, 184]. In particular, Mopuri *et al.*, have developed a generative model that learns the space of universal perturbations for images using a GAN-like architecture. Inspired by this work, we develop a similar architecture, but make modifications to suit our objective. Our goal is to generate adversarial perturbations that fool the discriminator instead of exploring the space for diverse UPs. In addition, we retrofit the architecture to handle video inputs. Our architecture is depicted in Figure 3.3b. It consists of three main components: 1) a 3D generator which generates universal perturbations (clips); 2) a post-processor, which for now does not do anything but is needed to solve other challenges described in subsequent

sections; and 3) a pre-trained discriminator for video classification, viz., the C3D model described in § 3.3.2.

The 3D generator in our model is configured to use 3D deconvolution layers and provide 3D outputs as shown in Figure 3.8. Specifically, it generates a clip of perturbations, whose size is equal to the size of the video clips taken as input by the C3D classifier. To generate universal perturbations, the generator first takes a noise vector z from a latent space. Next, It maps z to a perturbation clip p, such that, $G(z) = p$. It then adds the perturbations on a training clip x (denote the set of inputs from the training class as X) to obtain the perturbed clip $x + p$. Let $c(x)$ be the true label of x. This perturbed clip is then input to the C3D model which outputs the score vector $Q(x+p)$ (for the perturbed clip). The classification should ensure that the highest score corresponds to the true class ($c(x)$ for input x) in the benign setting. Thus, the attacker seeks to generate a p such that the C3D classifier outputs a low score to the $c(x)$th element in the Q vector (denoted as $Q_{c(x)}$) for $x + p$. In other words, this means that after applying the perturbation, the probability of mapping x to class $c(x)$ is lower than the probability that it is mapped to a different class (i.e., the input activity is not correctly recognized).

We seek to make this perturbation clip p "a universal perturbation", i.e., adding p to any input clip belonging to the target class would cause misclassification. This means that we seek to minimize the sum of the cross-entropy loss over all the training data as per Equation 3.1. Note that the lower the cross-entropy loss, the higher the divergence of the predicted probability from the true label [112].

$$\underset{G}{\text{minimize}} \quad \sum_{x \in X} -\log[1 - Q_{c(x)}(x + G(z)] \tag{3.1}$$

When the generator is being trained, for each training sample, it obtains feedback from the discriminator and adjusts its parameters to cause the discriminator to misclassify that sample. It tries to find a perturbation that works for every sample from the distribution space known to the discriminator. At the end of this phase, the attacker will have a generator that outputs universal perturbations which can cause the misclassification on any incoming input sample from the same distribution (as that of the training set). However, as discussed next, just applying the universal perturbations alone will not be sufficient to carry out a successful attack. In particular, the attack can cause unintended clips to be misclassified as well, which could compromise our stealth requirement as discussed next in §3.6.

3.6 Making Perturbations Stealthy

Blindly adding universal perturbations will affect the classification of clips belonging to other non-targeted classes. This may raise alarms, especially if many of these misclassifications are mapped on to rare actions. Thus, while causing the target class to be misclassified, the impact on the other classes must be imperceptible. This problem can be easily solved when dealing with image recognition systems since images are self-contained entities, i.e., perturbations can be selectively added to target images only. However, video inputs change temporally and an action captured in a set of composite frames may differ from that in the subsequent frames. It is thus hard to a priori identify (choose) the frames relating to the target class, and add perturbations specifically to them. For example, consider a case with surveillance in a grocery store. If attackers seek to misclassify an action related to shoplifting and cause this action to go undetected, they are unlikely to have precise knowledge of the exact time when the action will occur and be captured by the video

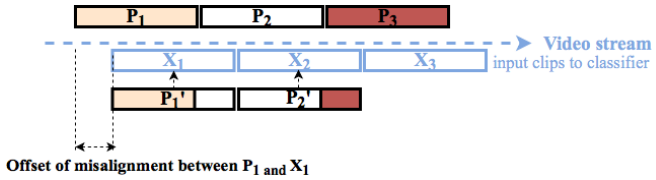
activity recognition system. Adding universal perturbations blindly in this case, could cause misclassifications of other actions (e.g., other benign customer actions may be mapped onto shoplifting actions thus triggering alarms). A similar example may be construed with respect to the elderly care system described in § 3.4.1; here, the attacker has no way of knowing a priori when an elderly falls.

Since it is hard (or even impossible) to a priori identify the frame(s) that capture the intended actions and choose them for perturbation, the attackers need to add perturbations to each frame. However, to make these perturbations furtive, they need to ensure that the perturbations added only mis-classify the target class while causing other (non-targeted) classes to be classified correctly. We name this unique kind of universal perturbations as "Dual-Purpose Universal Perturbations" or DUP for short.

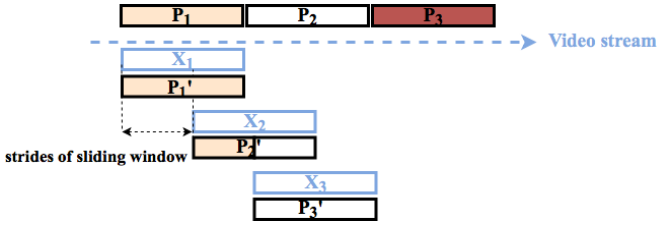
In order to realize DUPs, we have to guarantee that for the input clip x_t, if it belongs to the target class (denote the set of inputs from the target class as T), the C3D classifier returns a low score with respect to the correct class $c(x_t)$, i.e., $Q_{c(x_t)}$. For all input clips x_s that belong to other (non-target) classes (denote the set of inputs from non-target classes as S, thus, $S = X - T$), the model returns high scores with regard to their correct mappings ($Q_{c(x_s)}$). To cause the generator to output DUPs, we refine the optimization problem in Equation 3.1 as shown in Equation 3.2:

$$\begin{aligned} \underset{G}{\text{minimize}} \quad & \lambda \times \sum_{x_t \in T} -\log[1 - Q_{c(x_t)}(x_t + G(z))] \\ & + \sum_{x_s \in S} -\log[Q_{c(x_s)}(x_s + G(z))] \end{aligned} \tag{3.2}$$

The first term in the equation again relates to minimizing the cross-entropy of the target class, while the second term maximizes the cross-entropy relating to each of the other classes. The parameter λ is the weight applied with regard to the misclassification of the target class. For attacks



(a) Misalignment when the starting position of a clip input to the classifier, is not aligned with what the attacker assumes. Because of this, the perturbation added to input clip X_1 is a concatenation of two partial perturbations from P_1 and P_2.



(b) Misalignment can occur even if the starting position is aligned when a small stride is used. Here, the stride of the sliding window is half the clip size. This causes a misalignment because of which, the perturbation added to input clip X_2 is a concatenation of two partial perturbations from P_1 and P_2.

Figure 3.4: Two cases that can potentially cause misalignment between perturbation clips and the input clips to the classifier. The first parts of both figures represent the temporal sequence of generated perturbation clips. The lower parts of both figures capture the temporal sequence of input clips tested by the video classifier and the perturbation clips added to them.

where stealth is more important, we may use a smaller λ to guarantee that the emphasis on the misclassification probability of the target class is reduced while the classification of the non-target classes are affected to the least extent possible.

3.7 Impact of Nondeterministic Clip Boundaries

In this section, we first discuss why directly applying existing methods to generate perturbations against video streams do not work. Subsequently, we propose a new set of perturbations that do work (and are very effective) on video streams.

3.7.1 Misalignment due to Nondeterministic Clip Boundaries

The input to the video classifier is a clip composed of a sequence of frames. Given any input clip, the previously described attack methods (UP and DUP) can generate a perturbation clip that can be added to that input clip. As discussed in § 3.3.1, an input clip is controlled by a sliding window which in turn is defined by three hyper-parameters: the *window size* l, the *sliding stride* o, and the *starting position* f_{start}. Because f_{start} is non-deterministic, the clip boundaries of an input to the classifier in a real-time video classification system are also *nondeterministic*. As a result, even for white-box attackers, they cannot know a priori the clip boundaries (the consecutive frames in a video stream belonging to an input clip) used by the video classifier.

The nondeterminism in the clip boundaries is likely to cause *a misalignment* between the perturbation clips generated by the attacker and the input clips used by the classifier. Figure 3.4 depicts two cases where misalignment happens even with the attacker-friendly white-box scenario. The first row shows three perturbation clips P_1, P_2 and P_3 generated by the attacker [1]. The second row shows three input clips X_1, X_2 and X_3 used by the classifier. The clips in the two sequences are not aligned because the starting point of the sliding window is different from that of the perturbation clip. Consequently, the perturbation applied to input clip X_1 is actually a concatenation of the latter part of P_1 and the first part of P_2 (a perturbation P_1').

In a second case, as shown in Figure 3.4b, the perturbation clip P_2 and the input clip X_2 are not aligned because the stride of the sliding window is smaller than the window size. This smaller stride is commonplace in video classification systems as discussed in [32, 70, 253, 254].

3.7.2 The Boundary Effect

[1] For UP and DUP, $P_1 = P_2 = P_3$.

Because C3D utilizes a 3D CNN, we find via empirical experiments that when there is a misalignment between the perturbation clip and the input clip, it can cause significant degradations in the attack success rates, even for universal perturbations. For example, considering Figure 3.4a, the DUP P_1 should work on any input clip; however, the actual applied perturbation clip P_1' (which is the concatenation of two partial broken up perturbations) is less likely to work. We refer to this phenomenon as the boundary effect.

To formalize the boundary effect problem, let us consider a video stream represented by $\{\ldots, f_{i-2}, f_{i-1}, f_i, f_{i+1}, f_{i+2}, \ldots\}$ where, f_i represents the ith frame. The perturbation generated by G to cause a misclassification of the clip $\{f_i, f_{i+1}, \ldots, f_{i+l-1}\}$ (say $\{p_1, p_2, \ldots, p_l\}$) will be different from the one generated for a temporally staggered clip $\{f_{i-1}, f_i, \ldots, f_{i+l-2}\}$ (true for previously designed perturbations including UP and DUP). In other words, the perturbed clip $\{f_{i-1} \oplus p_1, f_1 \oplus p_2, \ldots, f_{i+l-2} \oplus p_l\}$ is unlikely to be effective in achieving misclassification.

To exemplify this problem, we perform extensive evaluations of existing established methods with regard to attacking the C3D model. In particular, we use the APIs from the CleverHans repository [195] to generate video perturbations. We experiment with several methods from CleverHans, including the most recent ones (e.g., CarliniWagnerL2 and DeepFool). The results presented in this chapter are based on the basic iteration method [137] with default parameters and all the videos in the UCF-101 testing set. We point out here that results based on all the other methods in the repository are very similar. We consider different boundaries for the clips in the videos (temporally staggered versions of the clips) and generate perturbations for each staggered version. Note that the sliding window size for C3D is 16 and thus, there are 16 staggered versions. We choose a candidate frame, and compute the correlations between the perturbations added in the different stag-

gered versions. Specifically, the perturbations are tensors and the normalized correlation between two perturbations is the inner product of the unit-normalized tensors representing the perturbations.

We represent the average normalized correlations in the perturbations (computed across all frames in the testing set) for two locations in the matrix shown in Figure 3.5. The row index and the column index represent the location of the frames in the two staggered clips. For example, the entry corresponding to $\{7, 7\}$ represents the case where the frame considered was the 7^{th} frame in the two clips, (actually, here it is the same clip). In this case, clearly the correlation is 1.00. However, we see that the correlations are much lower if the positions of the same frame in the two clips (two staggered versions) are different. As an example, consider the entry $\{5, 9\}$ which corresponds to the case where a frame is the fifth position in clip 1, and the same frame is at the ninth position in clip 2: the average normalized correlation between the two added perturbations is 0.39, which indicates that the perturbations that CleverHans adds in the two cases are quite different.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	1.00	0.40	0.25	0.24	0.25	0.23	0.22	0.21	0.22	0.20	0.21	0.20	0.18	0.18	0.24	0.26
	2	1.00	0.32	0.30	0.27	0.27	0.24	0.24	0.22	0.24	0.22	0.23	0.19	0.21	0.24	0.25
		3	1.00	0.38	0.28	0.34	0.31	0.26	0.28	0.24	0.27	0.23	0.26	0.22	0.19	0.17
			4	1.00	0.36	0.36	0.30	0.34	0.27	0.30	0.26	0.28	0.25	0.25	0.19	0.18
				5	1.00	0.40	0.42	0.34	0.39	0.31	0.35	0.29	0.28	0.22	0.21	0.19
					6	1.00	0.39	0.43	0.34	0.40	0.31	0.34	0.25	0.25	0.20	0.19
						7	1.00	0.40	0.43	0.40	0.30	0.34	0.30	0.23	0.21	0.18
							8	1.00	0.39	0.43	0.34	0.38	0.26	0.26	0.19	0.19
								9	1.00	0.41	0.43	0.34	0.34	0.25	0.22	0.20
									10	1.00	0.39	0.42	0.30	0.30	0.22	0.21
										11	1.00	0.40	0.37	0.28	0.25	0.21
											12	1.00	0.36	0.33	0.25	0.24
												13	1.00	0.38	0.28	0.24
													14	1.00	0.36	0.28
														15	1.00	0.47
															16	1.00

Figure 3.5: The average normalized correlation matrix computed with perturbations generated using the basic iteration API from CleverHans. The rows and columns represent the location of a frame in the two clips. The value represents the correlation between perturbations on the same frame but generated when that frame located in different positions (indicated by the row and column indices) in the two temporally staggered clips.

In Figure 3.6, we show the average magnitude of perturbations added (over all frames

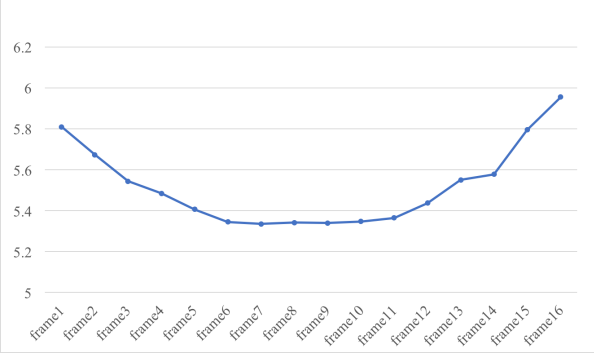


Figure 3.6: Magnitude of perturbation on each frame: The abscissa is the frame position, and the ordinate is the magnitude of average perturbation on the frame. (The attack seeks to misclassify a given video clip from UCF 101 dataset.)

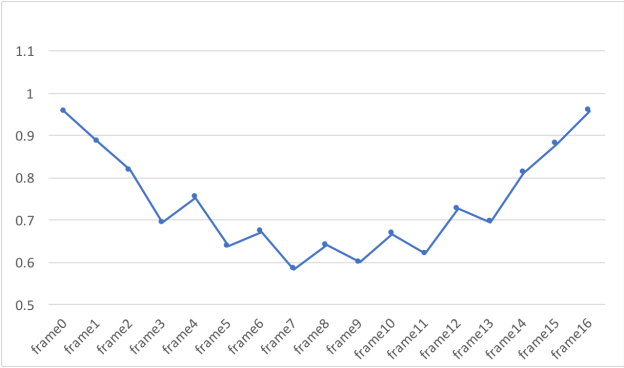


Figure 3.7: Attack success rate when there is mismatch. The abscissa is the offset between the clip generating perturbation and the clip tested. The ordinate is the attack success rate. (Attack aims to misclassify a given video clip from UCF 101 dataset.)

and all videos), when the target frame is at different locations within a clip. The abscissa depicts the frame position, and the ordinate represents the magnitude of the average perturbation. While the difference in the magnitude of perturbations added to two frames that are close to each other in terms of position (e.g., adjacent frames) within the clip, is small (this is because such frames are similar), the magnitude of perturbations added to frames that are distant in terms of location could potentially be quite different (because such frames could be quite dissimilar).

We further showcase the impact of the boundary effect by measuring the degradation in attack efficacy due to mismatches between the anticipated start point when the perturbation is

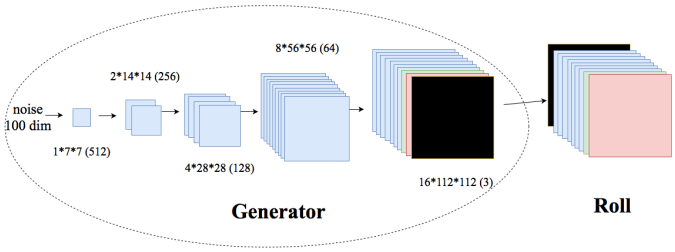


Figure 3.8: This figure illustrates the Generator and Roll for generating C-DUP. 1) The generator takes a noise vector as input, and outputs a perturbation clip with 16 frames. Note that the number of temporal dimensions with the C3D model is 16. The output size for each layer is shown as temporal dimension $\times$ horizontal spatial dimension $\times$ vertical spatial dimension $\times$ number of channels. 2) The roll part shifts the perturbation clip by some offset. The figure shows one example where we roll the front black frame to the back.

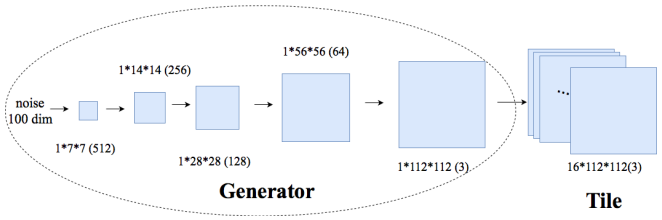


Figure 3.9: This figure illustrates the Generator and Tile for generating 2D-DUP. 1) The generator takes a noise vector as input, and outputs a single-frame perturbation. 2) The tile part constructs a perturbation clip by repeating the single-frame perturbation generated 16 times.

generated and the actual start point when classifying the clip (as shown in Figure 3.4a). Figure 3.7 depicts the results. The abscissa is the offset between the generated (intended) perturbation clip and the input clip used in classification. We can see that as the distance between the two start points increases, the attack success rate initially degrades but increases again as the the tested perturbation clip (a concatenation clip) is closer or more similar (has a better overlap) to the intended perturbation clip. For example, if the offset is 15, the perturbation clip added (concatenation clip) is offset by a single frame compared to the original (intended) perturbation clip.

3.7.3 Circular Dual-Purpose Universal Perturbation

To cope with the boundary effect, we develop a novel extension to the generative DNN model to significantly modify the DUPs proposed in § 3.6 to compose what we call "Circular Dual-Purpose Universal Perturbations (C-DUP)."

Let us suppose that the size of the sliding window is 16. Then, the DUP clip P includes 16 frames (of perturbation), denoted by $\{p_1, p_2, \ldots, p_{16}\}$. Since P is a clip of universal perturbations, we launch the attack by repeatedly adding perturbations on each consecutive clip consisting of 16 frames, in the video stream. One can visualize that we are generating a perturbation stream which can be represented as $\{p_1, p_2, \ldots, p_{15}, p_{16}, p_1, p_2, \ldots\}$. Now, our goal is to guarantee that the perturbation stream works regardless of the clip boundaries chosen by the classifier. Towards this, we need to ensure that any sequential concatenation of partial perturbation clips (the last part of the first clip and the first part of the second clip) results in a valid perturbation. It is easy to see that for this to hold true, we need any *cyclic or circular shift* of the DUP clip to be a valid DUP perturbation too. In other words, we require the perturbation clips $\{p_{16}, p_1, \ldots, p_{15}\}$, $\{p_{15}, p_{16}, \ldots, p_{14}\}$, $\ldots$, all to be valid perturbations. We emphasize here that UP and DUP do not have the cyclic property and thus, a sequential concatenation of parts of two consecutive UP or DUP clips will "not" be a valid perturbation.

To formalize, we define a permutation function $Roll(p, o)$ which yields a cyclic shift of the original DUP perturbation by an offset o. In other words, when using $\{p_1, p_2, \ldots, p_{16}\}$ as input to $Roll(p, o)$, the output is $\{p_{16-o}, p_{16-o+1}, \ldots, p_{16}, p1, \ldots, p_{16-o-1}\}$. Now, for all values of $o \in \{0, 15\}$, we need $p_o = Roll(p, o)$ to be a valid perturbation clip as well. Towards achieving this requirement, we use a post-processor unit which applies the roll function between the generator

and the discriminator. This post processor is captured in the complete architecture as shown in Figure 3.3b.

The details of how the generator and the roll unit operate in conjunction are depicted in Figure 3.8. As before, the 3D generator (G) takes a noise vector as input and outputs a sequence of perturbations (as a perturbation clip). Note that the final layer is followed by a $tanh$ non-linearity which constrains the perturbation generated to the range [-1,1]. The output is then scaled by ξ. Doing so restricts the perturbation's range to $[-\xi, \xi]$. Following the work in [181, 184], the value of ξ is chosen to be 10 towards making the perturbation quasi-imperceptible. The roll unit then "rolls" (cyclically shifts) the perturbation p by an offset in $\{0, 1, 2, \ldots 15\}$. Figure 3.8 depicts the process with an offset equal to 1; the black frame is rolled to the end of the clip. By adding the rolled perturbation clip to the training input, we get the perturbed input. As discussed earlier, the C3D classifier takes the perturbed input and outputs a classification score vector. As before, we want the true class scores to be (a) low for the targeted inputs and (b) high for other (non-targeted) inputs. We now modify our optimization function to incorporate the roll function as follows.

$$\begin{aligned} &\underset{G}{\text{minimize}} \\ &\sum_{o=1,2\cdots w} \{\lambda \times \sum_{x_t \in T} -\log[1 - Q_{c(x_t)}(x_t + Roll(G(z), o))] \\ &+ \sum_{x_s \in S} -\log[Q_{c(x_s)}(x_s + Roll(G(z), o))]\} \end{aligned} \tag{3.3}$$

The equation is essentially the same as Equation 3.2, but we consider all possible cyclic shifts of the perturbation output by the generator.

3.7.4 2D Dual-Purpose Universal Perturbation

We also consider a special case of C-DUP, wherein we impose an additional constraint which is that "the perturbations added to all frames are the same." In other words, we seek to add a single-frame 2D perturbation on each frame which can be seen as a special case of C-DUP with $p_1 = p_2 = \cdots = p_{16}$. We call this kind of C-DUP as 2D-DUP. 2D-DUP allows us to examine the effect of the temporal dimension in generating adversarial perturbations on video inputs. 2D-DUP is light-weight compared to C-DUP in terms of both transmission and storage costs. In addition, 2D-DUP allows other attack possibilities besides the man-in-the-middle case, an example being physically adding transparent foil (to add perturbation) onto the camera lens.

The generator in this case will output a single-frame perturbation instead of a sequence of perturbation frames as shown in Figure 3.9. This is a stronger constraint than the circular constraint, which may cause the attack success rate to decrease (note that the cyclic property still holds).

We denote the above 2D perturbation as p_{2d}. The perturbation clip is then generated by simply creating copies of the perturbation and *tiling* them to compose a clip. The 2D-DUP clip is now $p_{tile} = \{p_{2d}, p_{2d}, \ldots, p_{2d}\}$ (Figure 3.9). Thus, given that the attack objective is the same as before, we simply replace the $Roll(p, o)$ function with a $Tile$ function and our problem formulation now becomes:

$$\begin{aligned} \underset{G_{2D}}{\text{minimize}} \quad \lambda \times \sum_{x_t \in T} & -log[1 - Q_{c(x_t)}(x_t + Tile(G_{2D}(z)))] \\ & + \sum_{x_s \in S} -log[Q_{c(x_s)}(x_s + Tile(G_{2D}(z)))] \end{aligned} \tag{3.4}$$

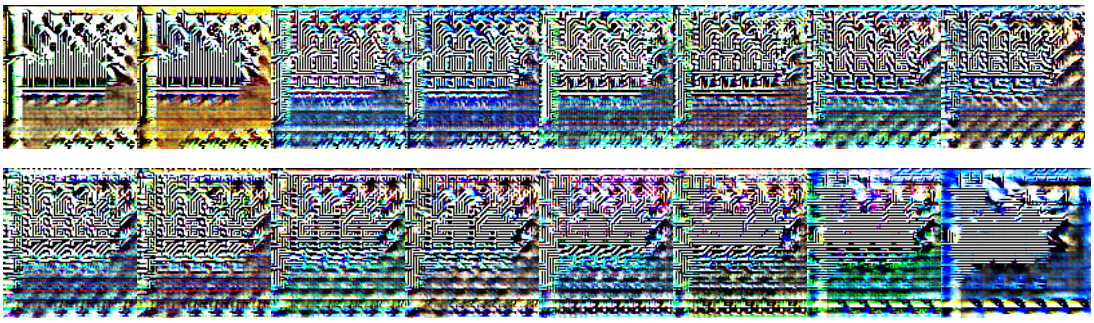


Figure 3.10: DUP on UCF-101

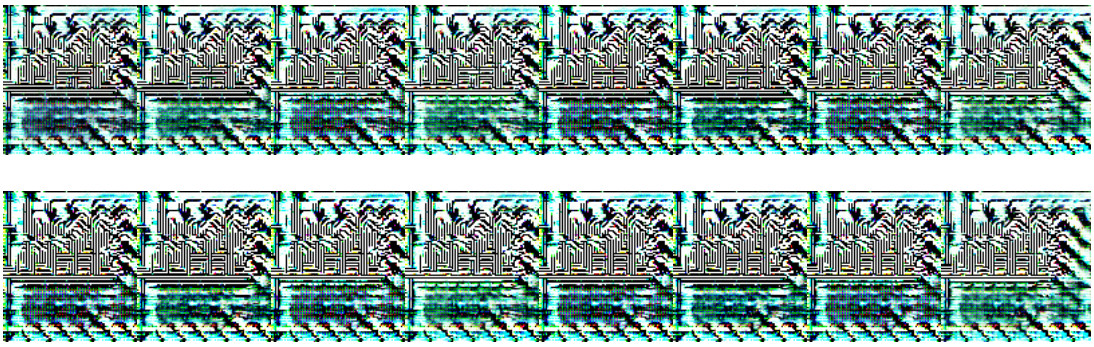


Figure 3.11: C-DUP on UCF-101

3.8 Evaluations

In this section, we showcase the efficacy of the perturbations generated by our proposed approaches on both the UCF-101 and Jester datasets.

3.8.1 Experimental Setup

Discriminator set-up for our experiments: We used the C3D classifier as our discriminator. The discriminator is then used to train our generator. For our experiments on the UCF101 dataset, we use the C3D model available in the Github repository [248]. This pre-trained C3D model achieves an average clip classification accuracy of 91.8% on the UCF101 dataset in benign settings (i.e., no

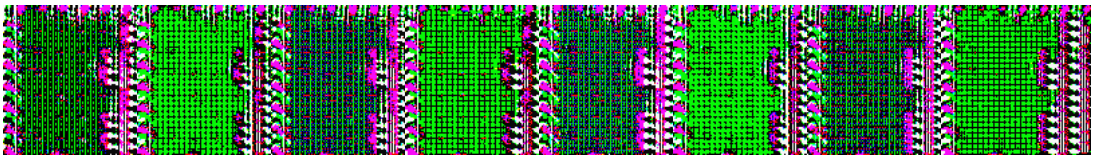


Figure 3.12: C-DUP on Jester for $T_1 = \{\text{slding hand right}\}$

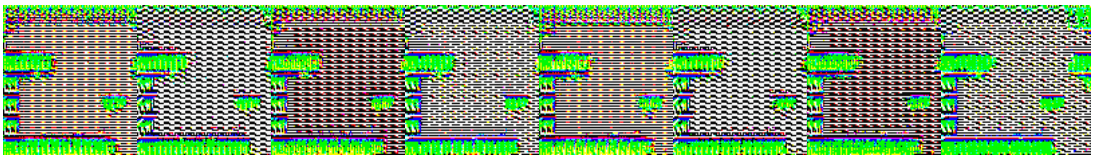


Figure 3.13: C-DUP on Jester for $T_2 = \{\text{shaking hand}\}$

adversarial inputs). For the experiments on the Jester dataset, we fine-tune the C3D model from the Github repository [248]. First, we change the output size of the last fully connected layer to 27, since there are 27 gesture classes in Jester. We use a learning rate with exponential decay [289] to train the model. The starting learning rate for the last fully connected layer is set to be 10^{-3} and 10^{-4} for all the other layers. The decay step is set to 600 and the decay rate is 0.9. The fine-tuning phase is completed in 3 epochs and we achieve a clip classification accuracy of 90.03% in benign settings.

Generator set-up for our experiments: For building our generators, we refer to the generative model used by Vondrik *et al.* [261], which has 3D deconvolution layers.

For generators for both C-DUP and 2D-DUP, we use five 3D de-convolution layers [20]. The first four layers are followed by a batch normalization [118] and a $ReLU$ [188] activation function. The last layer is followed by a $tanh$ [127] layer. The kernel size for all 3D de-convolutions is set to be $3 \times 3 \times 3$. To generate 3D perturbations (i.e., sequence of perturbation frames), we set the kernel stride in the C-DUP generator to 1 in both the spatial and temporal dimensions for the first layer, and 2 in both the spatial and temporal dimensions for the following 4 layers. To generate

a single-frame 2D perturbation, the kernel stride in the temporal dimension is set to 1 (i.e., 2D deconvolution) for all layers in the 2D-DUP generator, and the spatial dimension stride is 1 for the first layer and 2 for the following layers. The numbers of filters are shown in brackets in Figure 3.8 and Figure 3.9. The input noise vector for both generators are sampled from a uniform distribution $U[-1, 1]$ and the dimension of the noise vector is set to be 100. For training both generators, we use a learning rate with exponential decay. The starting learning rate is 0.002. The decay step is 2000 and the decay rate is 0.95. Unless otherwise specified, the weight balancing the two objectives, i.e., λ, is set to 1 to reflect equal importance between misclassifying the target class and retaining the correct classification for all the other (non-target) classes.

Technical Implementation: All the models are implemented in TensorFlow [5] with the Adam optimizer [132]. Training was performed on 16 Tesla K80 GPU cards with the batch size set to 32. The code is available at https://github.com/sli057/Video-Perturbation.git.

Dataset setup for our experiments: On the UCF-101 dataset (denoted UCF-101 for short), different sets of target class T are tested. We use $T = \{\text{apply lipstick}\}$ for presenting the results in the chapter. Experiments using other target sets also yield similar results. UCF-101 has 101 classes of human actions in total. The target set T contains only one class while the "non-target" set $S = X - T$ contains 100 classes. The number of training inputs from the non-target classes is approximately 100 times the number of training inputs from the target class. Directly training with UCF-101 may cause a problem due to the imbalance in the datasets containing the target and non-target classes [166]. Therefore, we under-sample the non-target classes by a factor of 10. Further, when loading a batch of inputs for training, we fetch half the batch of inputs from the target set and the other half from the non-target set in order to balance the inputs.

For the Jester dataset, we also choose different sets of target classes. We use two target sets $T_1 = \{\text{sliding hand right}\}$ and $T_2 = \{\text{shaking hands}\}$ as our representative examples because they are exemplars of two different scenarios. Since we seek to showcase an attack on a video classification system, we care about how the perturbations affect both the appearance information and temporal flow information, especially the latter. For instance, the 'sliding hand right' class has a temporally similar class 'sliding two fingers right;' as a consequence, it may be easier for attackers to cause clips in the former class to be misclassified as the later class (because the temporal information does not need to be perturbed much). On the other hand, 'shaking hands' is not temporally similar to any other class. Comparing the results of these two target sets could provide some empirical evidence on the impact of the temporal flow on our perturbations. Similar to UCF-101, the number of inputs from the non-target classes is around 26 times the number of inputs from the target class (since there are 27 classes in total and we only have one target class in each experiment). So we under-sample the non-target inputs by a factor of 4. We also set up the environment to load half of the inputs from the target set and the other half from the non-target set, in every batch during training.

Metrics of interest: For measuring the efficacy of our perturbations, we consider two metrics. *First,* the perturbations added to the videos should be quasi-imperceptible. *Second*, the attack success rate for the target and the non-target classes should be high. We define attack success rates as follows:

- The attack success rate for the target class is the misclassification rate.
- The attack success rate for the other classes is the correct classification rate.

3.8.2 Stealth with DUP

Recalling the discussion in §3.6, one can expect that UP would cause inputs from the target class to be misclassified, but also significantly affect the correct classification of the other non-target inputs. On the other hand, one would expect that DUP would achieve a stealthy attack, which would not cause much effect on the classification of non-target classes.

By testing on UCF-101 with ”apply lipstick” as the target class, we observe that with UP, ”archery” is misclassified as “swing,” “baby crawling” is misclassified as “cutting in kitchen,” “biking” is misclassified as “golf swing,” and so on. We find that only 45.2% of the video clips from non-target classes are classified correctly, i.e., the attack success rate for non-target inputs is only 45.2%. This violates the stealthiness needed to successfully launch an attack. However, DUP does not affect the classification of non-target inputs much; the non-target attack success rate is 88.03%. At the same time, both UP and DUP work well on target inputs, which means the perturbed target clips are misclassified at high rate. DUP achieves a attack success rate of 84.49 % for target inputs and UP achieves 84.01%. These results are obtained under the assumption that clip boundaries are exactly known while performing the attack. Given the inferior performance of UP on non-target inputs (i.e., in preserving stealth), we do not consider it any further in our evaluations.

3.8.3 Showcasing C-DUP

In this subsection, we discuss the results of the C-DUP perturbation attack. We use DUPs as our baselines.

Experimental Results on UCF101

Visualizing the perturbations: The perturbation clip generated by the DUP model is shown in Figure 3.10 and the perturbation clip generated by C-DUP model is shown in Figure 3.11. The visualizations of all perturbations are scaled from [0,10] to [0,255]. We observe that the perturbation from DUP manifests an obvious disturbance among the frames. With C-DUP, the perturbation frames look similar, which implies that C-DUP does not perturb the temporal information by much, in UCF101.

Impact of misalignment and C-DUP performance: Based on the discussion in §3.7, we expect that DUP would work well only when the perturbation clip is well-aligned with the start point of each input clip to the classifier; and the attack success rate would degrade as the misalignment increases. We expect C-DUP would overcome the misalignment effect and provide a better overall attack performance (even with temporal misalignment).

Case study: We perform a case study to showcase the impact of the misalignment. We consider one "apply lipstick" video clip for our case study. When DUP and C-DUP are added to this clip without any offset (no misalignment) i.e., the clip is in the form $[f_1, f2, \cdots, f_{16}]$, both perturbed clips are misclassified to "apply eye makeup". When there is an offset of 8, meaning that DUP and C-DUP are added to the clip in the form $[f_9, f_{10}, \cdots, f_{16}, f_1, \cdots, f_8]$, DUP fails to misclassify the clip while C-DUP still successfully misclassifies it. In fact, we observe that C-DUP works for all offsets from 0 to 15 while DUP only works when the offset = 0, 1, 2, 15, on this input clip.

Aggregate results: The attack success rates with DUP and C-DUP, on the UCF-101 test set, are shown in Figure 3.14a and Figure 3.14b. The x axis is the misalignment between the perturbation clip and the input clip to the classifier. Figure 3.14a depicts the average attack success rate for inputs from the target class. We observe that when there is no misalignment, the attack success rate with DUP is 84.49%, which is in fact slightly higher than C-DUP. However, the attack success rate with C-DUP is significantly higher when there is misalignment. Furthermore, the average attack success rate across all alignments for the target class with C-DUP is 84%, while with DUP it is only 68.26%. This demonstrates that C-DUP is more robust against misalignment.

Figure 3.14b shows that, with regard to the classification of inputs from the non-target classes, C-DUP also achieves a performance slightly better than DUP when there is mismatch. The average attack success rate (across all alignments) with C-DUP is 87.52% here, while with DUP it is 84.19%.

Experimental Results on Jester

Visualizing the perturbations: Visual representations of the C-DUP perturbations for the two target sets, $T_1 = \{\text{sliding hand right}\}$ and $T_2 = \{\text{shaking hands}\}$ are shown in Figure 3.12 and Figure 3.13. The perturbation clip has 16 frames, and we present a visual representation of the first 8 frames for compactness. We notice that compared to the perturbation generated on UCF-101 (see Figure 3.11). there is a more pronounced evolution with respect to Jester. We conjecture that this is because UCF-101 is a coarse-grained action dataset in which the spatial (appearance) information is dominant. As a consequence, the C3D model does not extract/need much temporal information

to perform well. However, Jester is a fine-grained action dataset where temporal information plays a more important role. Therefore, in line with expectations, we find that in order to attack the C3D model trained on the Jester dataset, more significant evolutions of the perturbations on the frames in a clip are required (i.e., more changes in the temporal dimension).

Attack success rate: To showcase a comparison of the misclassification rates with respect to the target class between the two schemes (DUP and C-DUP), we adjust the weighting factor λ such that the classification accuracy with respect to non-target classes are similar. By choosing λ = 1.5 for DUP and 1 for C-DUP, we are able to achieve this. The attack success rates for the above two target sets are shown in Figure 3.14c and Figure 3.14d, *and* Figure 3.14e and Figure 3.14f, respectively. We see that with respect to $T_1 = \{\text{sliding hand right}\}$, the results are similar to what we observe with UCF101. The attack success rates for C-DUP are a little lower than those for DUP when the offset is 0. This is to be expected since DUP is tailored for this specific offset. However, C-DUP outperforms DUP when there is a misalignment. The average success rate for C-DUP is 85.14% for the target class and 81.03% for the other (non-target) classes. The average success rate for DUP is 52.42% for the target class and 82.36% for the other (non-target) classes.

Next we consider the case with $T_2 = \{\text{shaking hands}\}$. In general, we find that both DUP and C-DUP achieve relatively lower success rates especially with regard to the other (non-target) classes. As discussed in §3.8.1, unlike in the previous case where 'sliding two fingers right' is temporally similar to 'sliding hand right', no other class is temporally similar to 'shaking hand'. Therefore it is harder to achieve misclassification. The attack success rates with the two approaches for the target class are shown in Figure 3.14e. We see that C-DUP significantly outperforms DUP in terms of attack efficacy because of its robustness to temporal misalignment (i.e., the boundary

effect). The average attack success rate for the target class with C-DUP is 79.03% while for DUP it is only 57.78%. Overall, our C-DUP outperforms DUP in being able to achieve a better attack success rate for the target class. We believe that although stealth is affected to some extent, it is still reasonably high.

3.8.4 Effectiveness of 2D-DUP

The visual representations of the perturbations with C-DUP show that perturbations on all the frames are visually similar. Thus, we ask if it is possible to add "the same perturbation" on every frame and still achieve a successful attack. In other words, will the 2D-DUP perturbation attack yield performance similar to the C-DUP attack ?

Experimental Results on the UCF101 Dataset

Visual impact of the perturbation: We present a sequence of original frames and its corresponding perturbed frames in Figure 3.15. Original frames are displayed in the first row and perturbed frames are displayed in the second row. We observe that the perturbation added to the frames is quasi-imperceptible to human eyes (similar results are seen with C-DUP but are omitted in the interest of compactness).

Attack success rate: By adding 2D-DUP on the video clip, we achieve an attack success rate of 87.58% with respect to the target class and an attack success rate of 83.37% for the non-target classes. Recall that the average attack success rates with C-DUP were 87.52% and 84.00%, respectively. Thus, the performance of 2D-DUP seems to be on par with that of C-DUP on the UCF101

dataset. This demonstrates that C3D is vulnerable even if the same 2D perturbation generated by our approach is added to every frame.

Experimental Results on Jester Dataset

Attack success rate: For $T_1 = \{\text{sliding hand right}\}$, the attack success rate for the target class is 84.64% and the attack success rate for the non-target classes is 80.04%. This shows that 2D-DUP is also successful on some target classes in the fine-grained, Jester action dataset.

For the target set T_2, the success rate for the target class drops to 70.92%, while the success rate for non-target class is 54.83%. This is slightly degraded compared to the success rates achieved with C-DUP (79.03% and 57.78% respectively), but is still reasonable. This degradation is due to more significant temporal changes in this case (unlike in the case of T_1) and a single 2D perturbation is less effective in manipulating these changes. In contrast, because the perturbations within C-DUP evolve, they are much more effective in achieving the misclassification of the target class.

3.9 Discussion

Black box attacks: In this work we assumed that the adversary is fully aware of the DNN being deployed (i.e., white box attacks). We argue that this is reasonable given that this is one of the first efforts on generating adversarial perturbations on real-time video classification systems. However, in practice the adversary may need to determine the type of DNN being used in the video classifica-

tion system, and so a black box approach may be needed. Given recent studies on the transferability of adversarial inputs [196], we believe black box attacks are also feasible. We will explore this in our future work.

Context dependency: Second, the approach that we developed does not account for contextual information, i.e., consistency between the misclassified result and the context. While in some cases with a limited set of classes (e.g., actions possible at an elderly care facility), this may be not matter, in some other cases a loss in context may cause a human operator to notice discrepancies. For example, if the context relates to a baseball game, a human overseeing the system may notice an inconsistency when the action of hitting a ball is misclassified into applying makeup. Similarly, because of context, if there is a series of actions that we want to misclassify, inconsistency in the misclassification results (e.g., different actions across the clips) may also raise an alarm. For example, let us consider a case where the actions include running, kicking a ball, and applying make up. While the first two actions can be considered to be *reasonable* with regard to appearing together in a video, the latter two are unlikely. Generating perturbations that are consistent with the context of the video is a line of future work that we will explore and is likely to require new techniques. In fact, looking for consistency in context may be a potential defense, and we will also examine this in depth in the future.

Data Augmentation: We point out here that for both UPs and DUPs, the training set included all possible strides (data augmentation). Unfortunately, the issues relating to the boundary effect cannot be solved by data augmentation. In particular, recall that the misalignment due to the nondeterminism in clip boundaries input to the classifier cause the perturbation clips added by the attacker to be broken up. While UPs are effective on any video clip, concatenations of broken up UPs are no

longer UPs and thus, are not effective.

Defenses: In order to defend against the attacks against video classification systems, one can try some existing defense methods in image area, such as feature squeezing [279, 280] and ensemble adversarial training [252] (although their effectiveness is yet unknown). Considering the properties of video that were discussed, we envision some exclusive defense methods for protecting video classification systems below, which we will explore in future work.

One approach is to examine the consistency between the classification of consecutive frames (considered as images) within a clip, and between consecutive clips in a stream. A sudden change in the classification results could raise an alarm. However, while this defense will work well in cases where the temporal flow is not pronounced (e.g., the UCF101 dataset), it may not work well in cases with pronounced temporal flows. For example, with respect to the Jester dataset, with just an image it may be hard to determine whether the hand is being moved right or left.

The second line of defense may be to identify an object that is present in the video, e.g., a soccer ball in a video clip that depicts a kicking action. We can use an additional classifier to identify such objects in the individual frames that compose the video. Then, we can look for consistency with regard to the action and the object, e.g., a kicking action can be associated with a soccer ball, but cannot be associated with a make up kit. Towards realizing this line of defense, we could use existing image classifiers in conjunction with the video classification system. We will explore this in future work.

3.10 Related Work

There is quite a bit of work [18, 19, 110] on investigating the vulnerability of machine learning systems to adversarial inputs. Researchers have shown that generally, small magnitude perturbations added to input samples, change the predictions made by machine learning models. Most efforts, however, do not consider real-time temporally varying inputs such as video. Unlike these efforts, our study is focused on the generation of adversarial perturbations to fool DNN based real-time video action recognition systems.

The threat of adversarial samples to deep-learning systems has also received considerable attention recently. There are several papers in the literature (e.g., [86, 87, 181, 182, 224]) that have shown that the state-of-the-art DNN based learning systems are also vulnerable to well-designed adversarial perturbations [243]. Szegedy *et al.*show that the addition of hardly perceptible perturbation on an image, can cause a neural network to misclassify the image. Goodfellow *et al.* [87] analyze the potency of adversarial samples available in the physical world, in terms of fooling neural networks. Moosavi-Dezfooli *et al.* [181–183] make a significant contribution by generating image-agnostic perturbations, which they call universal adversarial perturbations. These perturbations can cause all natural images belonging to target classes to be misclassified with high probability.

There are very few recent studies [104, 265] which explore the feasibility of adversarial perturbation on videos. Hosseini et al. [104] attack the Google Cloud Video Intelligence API, which makes decisions only based on the first frame of every second of the video, by inserting images/perturbing frames at the rate of one frame per second. This attack method cannot be generalized to the common case where video classification systems use sequences of consecutive frames to perform activity recognition. In addition, the authors assume that the starting frame used by

the API is known to the attacker, which in real-time applications is not deterministic (and thus, is unknown). Wei et al. [265] attack the video recognition system by adding perturbations only on the first few consecutive frames in a video clip. However, unlike our attack, these attacks do not work on practical real-time video classification systems when the boundaries of video clips are not known.

GANs or generative adversarial networks have been employed by Goodfellow *et al.* [86] and Radford *et al.* [209] in generating natural images. Mopuri *et al.* [184] extend a GAN architecture to train a generator to model universal perturbations for images. Their objective was to explore the space of the distribution of universal adversarial perturbations in the image space. We significantly extend the generative framework introduced by Mopuri *et al.* [184]. In addition, unlike their work which focused on generating adversarial perturbations for images, our study focuses on the generation of effective perturbations to attack videos.

The feasibility of adversarial attacks against other types of learning systems including face-recognition systems [178, 224, 225], voice recognition systems [29] and malware classification systems [90], has been studied. However, these studies do not account for the unique input characteristics that are present in real-time video activity recognition systems.

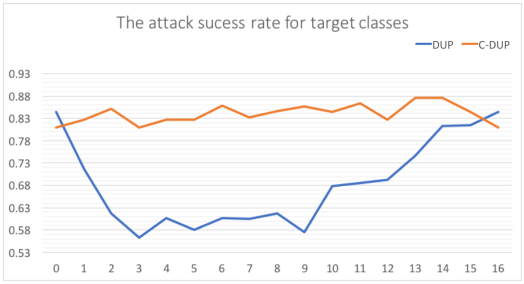
3.11 Conclusions

In this chapter, we investigate the problem of generating adversarial samples for attacking video classification systems. We identify three key challenges that will need to be addressed in order to generate such samples namely, generating perturbations in real-time, making the perturbations stealthy and dealing with the intedeterminism of video clip boundaries that are input

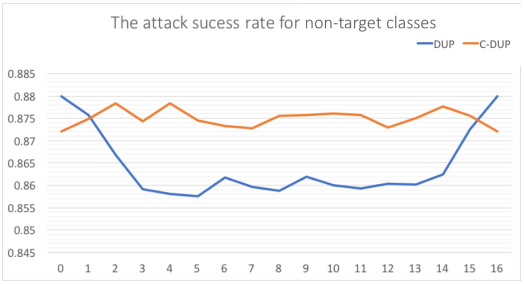
to a real-time video classifier. We exploit recent advances in generative models, extending them significantly to solve these challenges and generate very potent adversarial samples against video classification systems. We perform extensive experiments on two different datasets one of which captures coarse-grained actions (e.g., applying make up) while the other captures fine-grained actions (hand gestures). We demonstrate that our approaches are extremely potent, achieving around 80 % attack success rates in both cases. We also discuss possible defenses that we propose to investigate in future work.

Acknowledgments

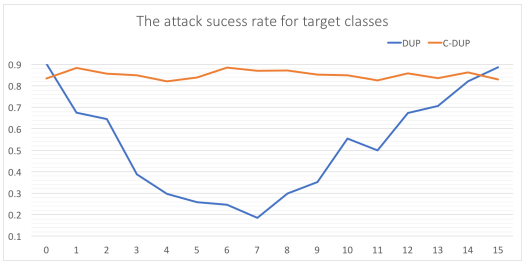
We would like to thank the anonymous reviewers for their valuable feedback on this chapter. This work was partially supported by the U.S. Army Research Laboratory Cyber Security Collaborative Research Alliance under Cooperative Agreement Number W911NF-13-2-0045. The views and conclusions contained in this document are those of the authors, and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to re-produce and distribute reprints for Government purposes, notwithstanding any copyright notation hereon.



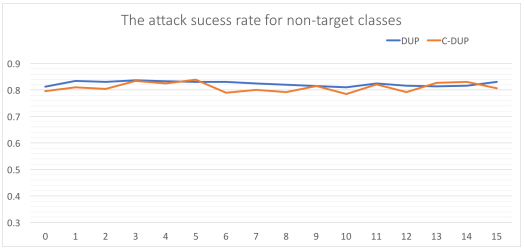
(a) Attack success rate on UCF-101 for target class
'applying lipstick'. The baseline accuracy of attack success rate without perturbation is 4.5%.



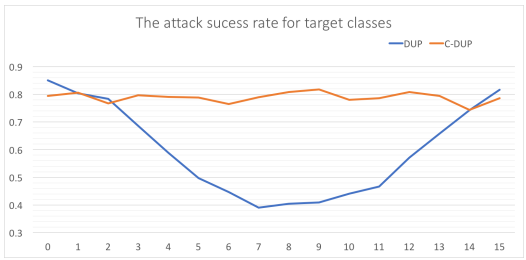
(b) Attack success rate on UCF-101 for other non-target classes (all except 'applying lipstick'). The baseline accuracy of attack success rate without perturbation is 91.8%.



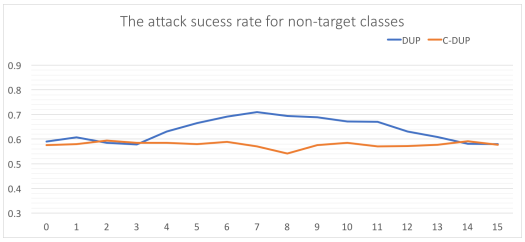
(c) Attack success rate on Jester for target class 'sliding hands right'.The baseline accuracy of attack success rate without perturbation is 12.9%.



(d) Attack success rate on Jester for non-target classes (all excepet 'sliding right'). The baseline accuracy of attack success rate without perturbation is 90.4%.



(e) Attack success rate on Jester for target class 'shaking hand'. The baseline accuracy of attack success rate without perturbation is 6.3%.



(f) Attack success rate on Jester for non-target classes (all except 'shaking hand'). The baseline accuracy of attack success rate without perturbation is 89.9%.

Figure 3.14: Attack success rates for DUP and C-DUP along with the offset of mismatch

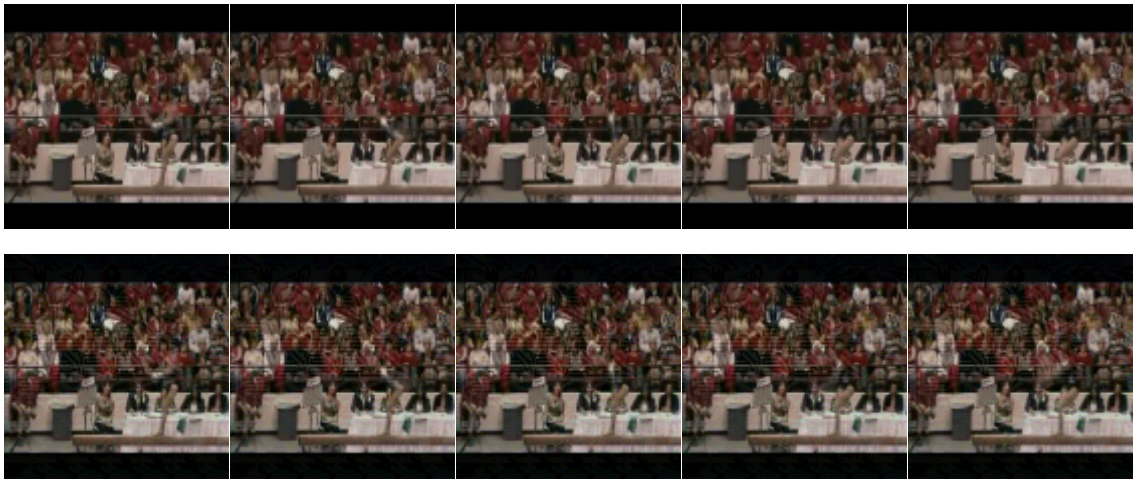


Figure 3.15: Visualizing images after adding 2D dual purpose universal perturbation: Original frames are displayed in the first row and perturbed frames are displayed in the second row. The perturbation added to the frames in the second row is mostly imperceptible to the human eye.

Chapter 4

Geometric Transformations for Effective Black-box Adversarial Attacks on Video Classifiers

4.1 abstract

When compared to the image classification models, black-box adversarial attacks against video classification models have been largely understudied. This could be possibly because, with video, the temporal dimension poses significant additional challenges in gradient estimation. Query-efficient black-box attacks rely on effective estimated gradients towards maximizing the probability of misclassifying the target video. In this work, we demonstrate that such effective gradients can be searched for by parameterizing the temporal structure of the search space with geometric transformations. Specifically, we design a novel iterative algorithm Geometric TRAnsformed Perturbations

(GEO-TRAP), for attacking video classification models. GEO-TRAP employs standard geometric transformation operations to reduce the search space for effective gradients into searching for a small group of parameters that define these operations. This group of parameters describes the geometric progression of gradients, resulting in a reduced and structured search space. Our algorithm inherently leads to successful perturbations with surprisingly few queries. For example, adversarial examples generated from GEO-TRAP have better attack success rates with $\sim 73.55\%$ fewer queries compared to the state-of-the-art method for video adversarial attacks on the widely used Jester dataset. Overall, our algorithm exposes vulnerabilities of diverse video classification models and achieves new state-of-the-art results under black-box settings on two large datasets.

4.2 Introduction

Adversarial attacks are designed to expose vulnerabilities of Deep Neural Networks (DNNs). With real-world applications of video classification based on DNNs emerging [45, 119], a key question that arises is "*what type of adversarial inputs can mislead, and thus render video classification networks vulnerable*?" Designing such adversarial attacks not only helps expose security flaws of DNNs, but can also potentially stimulate the design of more robust video classification models.

Adversarial attacks against image classification models have been studied in both *white-box* [31, 88, 235, 243, 271] and *black-box* [17, 34, 116, 196, 197] settings. In the white-box setting, an adversary has full access to the model under attack, including its parameters and training settings (hyper-parameters, training data, etc.) In the black-box setting, an adversary only has partial information about the victim model, such as the predicted labels of the model. In the case of video classification models, adversarial attacks in both white-box and black-box settings have garnered

some interest [38, 124, 146, 165, 206, 266, 267, 282, 290], although the body of work here is more limited than the case of image classification models.

A common black-box attack paradigm is query-based, wherein the attacker can send queries to the victim model to collect the corresponding predicted labels, and thereby estimate the gradients needed for curating the adversarial examples. Unlike static images, videos naturally include additional information from the temporal dimension. This high dimensionality (i.e., sequence of frames instead of one image) poses challenges to black-box adversarial attacks against video classification models; in particular, significantly more queries are typically needed for estimating the gradients for crafting adversarial samples [124, 267, 282, 290]. [124] reduces the number of queries by adding perturbations on the patch level instead of at the pixel level; [267, 282] propose to add perturbations only on key pixels. [290] considers the intrinsic differences between images and videos (i.e., the temporal dimension), and proposes to use the optical-flow of clean videos as the motion prior for adversarial video generation. Similar to [290], we also explicitly consider the temporal dimension of video. However, rather than fixing the temporal search space using the motion prior of clean videos, we propose to parameterize the temporal structure of the space with geometric transformations. This results in a better structured and reduced search space, which allows us to generate successful attacks with much fewer queries in black-box settings than the state-of-the-art methods, including [290].

Contributions. In this chapter, we propose a novel query-efficient black-box attack algorithm against video classification models. Due to the extra temporal dimension, generating video perturbations by searching for effective gradients remains a challenging task given the exceedingly large search space. These gradients are estimated by searching for 'directions' that maximize the

Table 4.1: Comparison with state-of-the-art. GEO-TRAP, compared to current black-box attack methods for videos, doesn't train a different network to craft perturbations, and parameterizes the temporal dimension of videos in searching for effective perturbation directions.

Method	WITHOUT training a "perturbation" network	CONSIDER temporal dimension?	PARAMETERIZE temporal dimension?
PATCHATTACK [124]	✗	✗	✗
HEURISTICATTACK [267]	✓	✗	✗
SPARSEATTACK [282]	✗	✗	✗
MOTIONSAMPLERATTACK [290]	✓	✓	✗
GEO-TRAP (**Ours**)	✓	✓	✓

probability of the victim model mis-classifying the crafted inputs. Our approach drastically reduces this large search space by defining this space with a small set of parameters that describe the geometric progression of gradients in the temporal dimension, resulting in a reduced and temporally structured search space. Conceptually, this parameterization of the temporal structure of the search space is performed using geometric transformations (e.g. affine transformations). We refer to our algorithm as Geometrically TRAnsformed Perturbations, or GEO-TRAP. Despite this surprisingly simple strategy, GEO-TRAP outperforms existing black-box video adversarial attack methods by significant margins ($\sim$ 1.8% improvement in attack success rate with $\sim$ 73.55% *fewer queries* for targeted attacks in comparison to the state-of-the-art [290] on the Jester dataset [176]).

4.3 Related Works

In this section, we review different black-box adversarial attacks strategies, and categorize our proposed method with respect to state-of-the-art black-box attacks designed for video classifiers.

In most real-world attacks, the adversary only has partial information about the victim models, such as the predicted labels. In such black-box settings, the adversary can first attack a local surrogate model and then transfer these attacks to the target victim model [111, 162], formally called as *transferability-based* black-box attack. Alternatively, they may estimate the adversarial gradient with zero-order optimization methods such as Finite Differences (FD) or Natural Evolution Strategies (NES) by querying the victim model [34, 115, 116], which is called *query-based* black-box attack. GEO-TRAP falls under the category of query-based black-box attacks (designed for videos).

Whilst several white-box attacks have been proposed for video classification models [38, 146, 165, 206, 266], black-box video attacks are relatively under explored. PATCHATTACK (V-BAD) [124] is the first to propose a black-box video attack framework which uses a hybrid attack strategy of first generating initial perturbations for each video frame by attacking a local image classifier, and then updating the perturbations by querying the victim model. Compared to PATCHATTACK [124] , GEO-TRAP does not require training a local classifier. PATCHATTACK [124] crafts video perturbations by treating each frame as a separate image, but reduces the search space of the gradient estimation by morphing the perturbations in patches/partitions. However, its attack performance has been shown to be inferior to that of a more recent approach [290] (discussed below). HEURISTICATTACK [267] uses a query-based attack strategy, and reduces the search space by generating adversarial perturbations only on heuristically selected key frames and salient regions. SPARSEATTACK [282] reduces the search space by adding perturbations only on key frames using a reinforcement learning based framework. MOTIONSAMPLERATTACK [290] proposed a query-based attack strategy that utilizes a motion excited sampler to obtain *motion-aware* perturbation

prior by using the optical-flow of the clean video. This motion-aware prior reduces the search space for gradients resulting in fewer queries. Similar to [290] but different from [124, 267, 282], GEO-TRAP explicitly considers the temporal dimension of video in order to search for effective gradients. However unlike [290], GEO-TRAP does not fix the temporal structure of the search space using a *pre-computed fixed* motion prior, but parameterize it with simple geometric transformations. These black-box video attack methods are summarized in Table 4.1.

4.4 Attacking via Geometrically TRAnsformed Perturbations (GEO-TRAP)

Notation. We denote the tuple of a video clip and its corresponding label as $(\boldsymbol{x}, y)$, which represents a data-point in the distribution $\mathcal{X}$. Each video sample $\boldsymbol{x} \in \mathbb{R}^{T \times H \times W \times C}$ has T frames of H height, W width, and C channels. We denote the victim video classification model as $\boldsymbol{f_\theta} : \mathcal{X} \rightarrow \mathcal{Y}$, where $\boldsymbol{\theta}$ represents the model's parameters learned from the training subset of $\mathcal{X}$, via a mapping to the label space $\mathcal{Y}$. We further assume $\mathcal{X}$ consists of videos from $|\mathcal{Y}| = K$ categories. To make the perturbations imperceptible to humans, we impose the perturbation budget $\rho_{\max}$ with the $\|\cdot\|_p$ norm. Throughout this chapter, we consider $\|\cdot\|_\infty$ norm following [124, 146, 290] (the method can be extended to $p = 1, 2$ norms). To constrain $\|\cdot\|_\infty$ of perturbation below a budget $\rho_{\max}$, we use the $\texttt{clip}(\cdot)$ function to keep the perturbation pixel value in $[-\rho_{\max}, \rho_{\max}]$. The function $\texttt{sign}(\cdot)$ extracts the sign of given input variable. The superscript i, throughout the chapter, denotes the iteration i. The subscript t denotes the frame index. For clarity, we represent vectors/tensors with the bold font and scalars with the regular font.

Problem Statement. We consider the scenario of attacking a standard video classification model using a query-based paradigm under **black-box settings** (assuming no access to $\boldsymbol{\theta}$ nor the training subset of $\mathcal{X}$). Specifically, we aim to craft perturbed videos $\boldsymbol{x}_{\text{adv}}$ with imperceptible differences from $\boldsymbol{x}$, in order to alter the decision of the target model $\boldsymbol{f}_{\boldsymbol{\theta}}$ via multiple queries to guide the gradient estimation. This problem can be mathematically formulated as follows.

$$\underset{\boldsymbol{x}_{\text{adv}}}{\operatorname{argmin}} \quad \mathcal{L}\big(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_{\text{adv}}), y\big) \quad \text{s.t.} \quad \|\boldsymbol{x}_{\text{adv}} - \boldsymbol{x}\|_\infty \leq \rho_{\max} \tag{4.1}$$

$\mathcal{L}\big(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_{\text{adv}}), y\big)$ is the objective function, capturing the similarity between the classifier's output and the ground truth label y, and varies with different attack goals (targeted or untargeted). The challenge is to obtain $\boldsymbol{x}_{\text{adv}}$ with as few queries as possible by estimating gradient $\boldsymbol{g}^\star = \nabla_{\boldsymbol{x}_{\text{adv}}} \mathcal{L}\big(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_{\text{adv}}), y\big)$, which is unknown in the considered black-box setting.

Overview of GEO-TRAP. We propose a novel iterative video perturbation framework that follows the principle of the Basic Iterative Method [138] in order to fool $\boldsymbol{f}_{\boldsymbol{\theta}}$ under $\|\cdot\|_\infty$ norm as follows.

$$\boldsymbol{x}_{\text{adv}}^{(0)} = \boldsymbol{x}, \quad \boldsymbol{x}_{\text{adv}}^{(i)} = \texttt{clip}\big(\boldsymbol{x}_{\text{adv}}^{(i-1)} - h\,\texttt{sign}(\boldsymbol{g}^{(i)})\big) \tag{4.2}$$

where h is a hyperparameter and $\boldsymbol{g}^{(i)}$ is the gradient estimated by querying the black-box victim model at the i^{th} iteration using our proposed GEO-TRAP algorithm. As shown in (4.2), effective perturbations rely on the guidance of the gradient $\boldsymbol{g}^{(i)}$. Therefore, efficiently estimating $\boldsymbol{g}^{(i)}$ is at the core of GEO-TRAP for successfully subverting video classifiers. We execute the following two steps in each iteration to estimate $\boldsymbol{g}^{(i)}$.

1. For any input video $\boldsymbol{x}^{(i)}$, a random noise tensor $\boldsymbol{r}_{\text{frame}} \in \mathbb{R}^{H\times W\times C}$ and a set of geometric transformation parameters $\boldsymbol{\Phi}_{\text{warp}} \in \mathbb{R}^{T\times D}$ are chosen with each element sampled from a

standard normal distribution. D represents the number of parameters needed for the geometric transformation of a single frame (details are provided in Section 4.4.2). In this setup, our search space for estimating $\boldsymbol{g}^{(i)}$ consists of $\boldsymbol{r}_{\text{frame}}$ and $\boldsymbol{\Phi}_{\text{warp}}$.

2. We then warp $\boldsymbol{r}_{\text{frame}}$ with $\boldsymbol{\Phi}_{\text{warp}}$ to get the candidate direction $\boldsymbol{\pi} = [\boldsymbol{r}_1, \boldsymbol{r}_2, \cdots, \boldsymbol{r}_T] \in \mathbb{R}^{T \times H \times W \times C}$ (see Algorithm 4 TRANS-WARP). $\boldsymbol{\pi}$ is then employed to compute a gradient estimator $\boldsymbol{\Delta}$ by querying the black-box victim model with a standard gradient estimation algorithm (see Algorithm 3 GRAD-EST). The gradient estimator $\boldsymbol{\Delta}$ is then used to update $\boldsymbol{g}^{(i)}$.

The overall attack strategy is summarized in Algorithm 2 and pictorially illustrated in Figure 5.2. Since in Step 2 above, the gradient estimation (i.e., GRAD-EST) procedure includes the geometric transformation strategy (i.e., TRANS-WARP), we will next describe GRAD-EST and then move on to TRANS-WARP. For the simplicity of exposition, we drop the superscript i and shorten the loss function to $\mathcal{L}(\boldsymbol{x}_{\text{adv}}, y)$ to $\mathcal{L}$ (since the model parameters $\boldsymbol{\theta}$ remain unchanged) in rest of this section.

4.4.1 GEO-TRAP Gradient Estimation (GRAD-EST)

Let $\boldsymbol{g}^\star = \nabla_{\boldsymbol{x}} \mathcal{L}$ be the ideal value of the gradient of $\mathcal{L}$ at $\boldsymbol{x}$, required to create $\boldsymbol{x}_{\text{adv}}$ in (4.2). To find an efficient estimator $\boldsymbol{g}$ for $\boldsymbol{g}^\star$, a (new) surrogate loss $\ell(\boldsymbol{g}) = -\langle \boldsymbol{g}^\star, \boldsymbol{g} \rangle$ is defined such that the estimator $\boldsymbol{g}$ has a sufficiently large inner product with the actual gradient $\boldsymbol{g}^\star$ ($\boldsymbol{g}$ is normalized to a unit vector; we ignore the normalization operation for ease of explanation). The loss function definition and the algorithm to estimate $\boldsymbol{g}$ follow [116].

As $\boldsymbol{g}^\star$ is unknown in the black-box setting, this surrogate loss function can be estimated

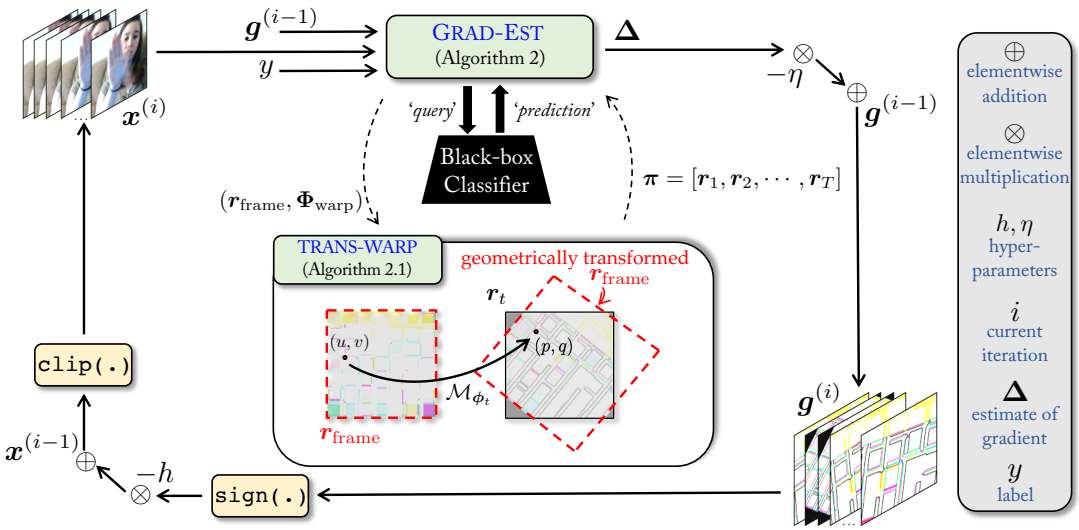


Figure 4.1: Overview of Geo-Trap. Geo-TRAP is a black-box attack algorithm guided by the key observation that strong gradients $\boldsymbol{g}^{(i)}$ can be computed by finding better gradient search direction candidates $\boldsymbol{\pi}$. We propose to search each frame of the directions $\boldsymbol{r}_t$ by warping a randomly sampled $\boldsymbol{r}_{\text{frame}}$ using a geometric transformation $\mathcal{M}_{\phi_t}$; different $\boldsymbol{r}_t$ in $\boldsymbol{\pi}$ are warped by the same $\boldsymbol{r}_{\text{frame}}$, thus have geometric progression among frames.

as

$$\ell(\boldsymbol{g}) = -\langle \boldsymbol{g}^\star, \boldsymbol{g} \rangle = -\langle \nabla_{\boldsymbol{x}} \mathcal{L}, \boldsymbol{g} \rangle \approx -\frac{\mathcal{L}(\boldsymbol{x} + \epsilon \boldsymbol{g}, y) - \mathcal{L}(\boldsymbol{x}, y)}{\epsilon}. \tag{4.3}$$

To iteratively estimate $\boldsymbol{g}$, we need to, in turn, estimate the gradient of $\ell(\boldsymbol{g})$, i.e., $\boldsymbol{\Delta} = \nabla_{\boldsymbol{g}} \ell(\boldsymbol{g})$. With antithetic sampling [215], $\boldsymbol{\Delta}$ can be estimated as

$$\boldsymbol{\Delta} = \frac{\ell(\boldsymbol{g} + \delta \boldsymbol{\pi}) - \ell(\boldsymbol{g} - \delta \boldsymbol{\pi})}{\delta} \boldsymbol{\pi}, \tag{4.4}$$

where δ is a small number adjusting the magnitude of the loss variation and $\boldsymbol{\pi} \in \mathbb{R}^{T \times H \times W \times C}$ is a random candidate direction. Our core contribution lies in the fact that instead of randomly sampling $\boldsymbol{\pi}$ in the search space [116], we reduce the search dimensionality by warping a randomly sampled tensor $\boldsymbol{r}_{\text{frame}} \in \mathbb{R}^{H \times W \times C}$ with another randomly sampled geometric (e.g., affine) transformation parameter tensor $\boldsymbol{\Phi}_{\text{warp}} \in \mathbb{R}^{T \times D}$ to get $\boldsymbol{\pi}$. The search space is then reduced from $T \times H \times W \times C$ to $(H \times W \times C) + (T \times D)$ and D is a relatively small number, $D \ll H \times W \times C$. With $\boldsymbol{w}_1 = \boldsymbol{g} + \delta \boldsymbol{\pi}$

Algorithm 2 GEO-TRAP: Query-based Iterative attack for Video Classifiers

Input : video $\boldsymbol{x}$, corresponding label y, step-size η for updating the gradient, step-size h for updating adversarial video.

Output: adversarial video $\boldsymbol{x}_{\text{adv}}$

Initialize: $\boldsymbol{x}^{(0)} = \boldsymbol{x}$, $\boldsymbol{g}^{(0)} = \mathbf{0}, i = 1$

while $\operatorname{argmax}\left(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})\right) = y$ **do**

  $\boldsymbol{\Delta} = \text{GRAD-EST}\,(\boldsymbol{x}^{(i-1)}, \boldsymbol{g}^{(i-1)}, y)$ /* Gradient Estimation */

  $\boldsymbol{g}^{(i)} \leftarrow \boldsymbol{g}^{(i-1)} - \eta\boldsymbol{\Delta}$

  $\boldsymbol{x}^{(i)} \leftarrow \texttt{clip}\,(\boldsymbol{x}^{(i-1)} - h\texttt{sign}\,(\boldsymbol{g}^{(i)}))$

  $i \leftarrow i + 1$

end

return $\boldsymbol{x}_{\text{adv}} = \boldsymbol{x}^{(i)}$

and $\boldsymbol{w}_2 = \boldsymbol{g} - \delta\boldsymbol{\pi}$ and combining (4.3) with (4.4), we get

$$\boldsymbol{\Delta} = \frac{\mathcal{L}(\boldsymbol{x} + \epsilon\boldsymbol{w}_2, y) - \mathcal{L}(\boldsymbol{x} + \epsilon\boldsymbol{w}_1, y)}{\epsilon\delta}\boldsymbol{\pi}. \tag{4.5}$$

Note that by querying the victim model $\boldsymbol{f}_{\boldsymbol{\theta}}$ with $\boldsymbol{x} + \epsilon\boldsymbol{w}_1$, we are able to retrieve the value of $\mathcal{L}(\boldsymbol{x}+\epsilon\boldsymbol{w}_1, y)$; similarly we can obtain the value of $\mathcal{L}(\boldsymbol{x}+\epsilon\boldsymbol{w}_2, y)$ ($\mathcal{L}(\cdot)$ is defined following [206].). In summary, we estimate $\boldsymbol{\Delta}$ with these two queries to the victim model. The resulting algorithm for estimating gradient of $\nabla_{\boldsymbol{g}}\ell$ or $\boldsymbol{\Delta}$ for consequently estimating $\boldsymbol{g}$ is shown in Algorithm 3. Eventually at every iteration, we use $\boldsymbol{\Delta}$ to update $\boldsymbol{g}$ by applying a one-step gradient descent as $\boldsymbol{g} \leftarrow \boldsymbol{g} - \eta\boldsymbol{\Delta}$, where η is a hyperparameter to update $\boldsymbol{g}$. This updated $\boldsymbol{g}$ is later used to obtain $\boldsymbol{x}_{\text{adv}}$ using (4.2).

4.4.2 Noise Warping using Geometric Transformation (TRANS-WARP)

To tackle the challenge of the high-dimensionality of the search space, we propose to parameterize the search space with a single random noise tensor $\boldsymbol{r}_{\text{frame}} \in \mathbb{R}^{H\times W\times C}$ and a sequence

of geometric transformations $\mathbf{\Phi}_{\text{warp}} \in \mathbb{R}^{T \times D}$. Apart from the reduction of the search space of gradient estimation, our geometric transformation provides a temporal structure to $\boldsymbol{\pi}$, which we discuss next.

At every iteration, $\boldsymbol{\pi} = [\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_T]$ represents the candidate direction for $\mathbf{\Delta}$. These directions $\boldsymbol{r}_t \in \mathbb{R}^{H \times W \times C}$ are used to compute $\mathbf{\Delta}$ in order to update gradient $\boldsymbol{g}$. To obtain $\boldsymbol{\pi}$, we use a sequence of transformation vectors $\mathbf{\Phi}_{\text{warp}} = [\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \ldots, \boldsymbol{\phi}_T]$ where $\boldsymbol{\phi}_t \in \mathbb{R}^D$. The dimensionality D, chosen by the attacker, can vary depending on the transformation type that is populated from $\boldsymbol{\phi}_t$, *e.g.*, $D = 6$ for affine transformation. We take affine transformation as an example to describe the warping process. We start by randomly sampling $\boldsymbol{r}_{\text{frame}}$ and the sequence of $\boldsymbol{\phi}_t$ along with initializing each element in the sequence of $\boldsymbol{r}_t$ with zero in **every** iteration. TRANS-WARP then computes $\boldsymbol{r}_t$ by warping $\boldsymbol{r}_{\text{frame}}$ using the parameters in $\boldsymbol{\phi}_t = [\phi_{11}^t, \phi_{12}^t, \phi_{13}^t, \phi_{21}^t, \phi_{22}^t, \phi_{23}^t] \in \mathbb{R}^6$ of $\mathbf{\Phi}_{\text{warp}} \in \mathbb{R}^{T \times 6}$ as follows. For all C channels, let (p, q) and (u, v) be the target and source coordinates in $\boldsymbol{r}_t$ and $\boldsymbol{r}_{\text{frame}}$, respectively. $\boldsymbol{r}_t$ (for all channels) is computed as

$$\boldsymbol{r}_t(p, q) \leftarrow \boldsymbol{r}_{\text{frame}}(u, v), \quad 1 \leq p, u \leq H,\ 1 \leq q, v \leq W. \tag{4.6}$$

Location (p, q) is computed using the affine transform matrix $\mathcal{M}_{\boldsymbol{\phi}_t}$ created with ϕ_t in homogeneous coordinates [94] as shown below. t is dropped for simplicity.

$$\begin{pmatrix} p \\ q \\ 1 \end{pmatrix} = \mathcal{M}_{\boldsymbol{\phi}} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{bmatrix} \phi_{11} & \phi_{12} & \phi_{13} \\ \phi_{21} & \phi_{22} & \phi_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \tag{4.7}$$

We compactly denote this warping operation in (4.6) and (4.7) with $\boldsymbol{r}_t = \mathcal{T}(\boldsymbol{r}_{\text{frame}}, \boldsymbol{\phi}_t)$. Affine transformation allows translation, rotation, scaling, and skew to be applied to $\boldsymbol{r}_{\text{frame}}$ to get each

Algorithm 3 GRAD-EST($\boldsymbol{x}^{(i-1)}, \boldsymbol{g}^{(i-1)} \in \mathbb{R}^{T\times H\times W\times C}, y$)$\rightarrow$ Estimate $\boldsymbol{\Delta} = \nabla_{\boldsymbol{g}}\ell(\boldsymbol{g}) \in \mathbb{R}^{T\times H\times W\times C}$

Input : video $\boldsymbol{x}^{(i)}$, label y, gradient estimator $\boldsymbol{g}^{(i-1)}$, δ for loss variation, ϵ for approximation.

Output: estimation of $\boldsymbol{\Delta} = \nabla_{\boldsymbol{g}}\ell(\boldsymbol{g})$

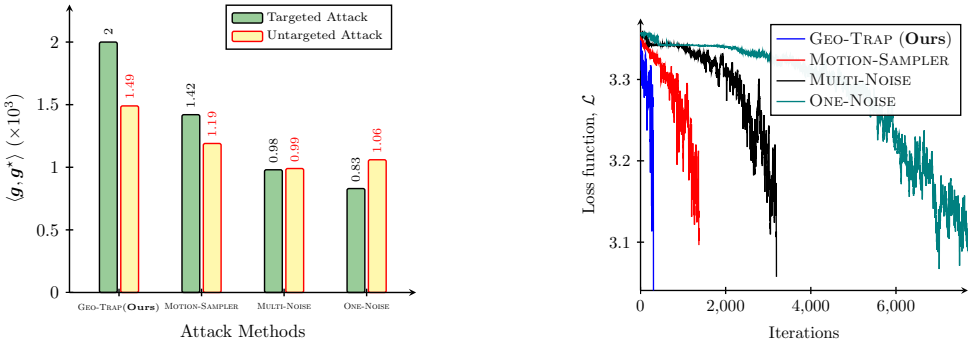
Sample $\boldsymbol{r}_{\text{frame}} \in \mathbb{R}^{H\times W\times C}$, $\boldsymbol{\Phi}_{\text{warp}} \in \mathbb{R}^{T\times D}$ (each element from a normal distribution $\mathcal{N}(0,1)$)
$\boldsymbol{\pi} = \text{TRANS-WARP}\,(\boldsymbol{r}_{frame}, \boldsymbol{\Phi}_{warp})$ /* Use Geometric Transformations */
$\boldsymbol{w}_1 = \boldsymbol{g}^{(i-1)} + \delta\boldsymbol{\pi}$, $\boldsymbol{w}_2 = \boldsymbol{g}^{(i-1)} - \delta\boldsymbol{\pi}$
$L_1 = \mathcal{L}(\boldsymbol{x}^{(i-1)} + \epsilon\boldsymbol{w_2}, y)$, $L_2 = \mathcal{L}(\boldsymbol{x}^{(i-1)} + \epsilon\boldsymbol{w_1}, y)$ /* Query victim model twice */
$\boldsymbol{\Delta} = (L_2 - L_1)\boldsymbol{\pi}/\epsilon\delta$
return $\boldsymbol{\Delta}$

$\boldsymbol{r}_t$. Therefore, the sequence of $\boldsymbol{r}_t$ have affine geometric progression among its temporal dimension. Other examples of geometric transformations may be more constrained, such as the similarity transformation $\mathcal{M}_{\boldsymbol{\phi}}^{\text{S}}$ (that allows translation, dilation (uniform scale) and rotation with $D = 4$) and translation-dilation $\mathcal{M}_{\boldsymbol{\phi}}^{\text{TD}}$ (that allows translation and uniform dilation with $D = 3$) as shown below.

$$[\phi_{11}, \phi_{12}, \phi_{13}, \phi_{23}] \rightarrow \mathcal{M}_{\boldsymbol{\phi}}^{\text{S}} = \begin{bmatrix} \phi_{11} & \phi_{12} & \phi_{13} \\ -\phi_{12} & \phi_{11} & \phi_{23} \\ 0 & 0 & 1 \end{bmatrix}, [\phi_{11}, \phi_{13}, \phi_{23}] \rightarrow \mathcal{M}_{\boldsymbol{\phi}}^{\text{TD}} = \begin{bmatrix} \phi_{11} & 0 & \phi_{13} \\ 0 & \phi_{11} & \phi_{23} \\ 0 & 0 & 1 \end{bmatrix} \tag{4.8}$$

4.5 What Makes GEO-TRAP Effective?

Potent iterative algorithms should rely on few queries for crafting successful perturbations for time efficiency. To minimize the number of queries, iterative algorithms need to find strong gradients in their early iterations. As discussed earlier, videos inherently incur a larger search space due to the temporal dimension and thus, pose challenges in searching for effective gradients. In this section, we provide empirical evidence to show that by parameterizing the temporal dimension, GEO-TRAP finds better gradients, in general, than previous works. We use three baselines in this



(a) **(b)**

Figure 4.2: Gradient Analysis of GEO-TRAP. **(a)** GEO-TRAP's high query-efficiency is a direct implication of good quality gradient estimation (for both targeted and untargeted attack), shown here with higher cosine similarity with $\boldsymbol{g}^\star$ compared to other methods. **(b)** Better quality of estimated gradients by GEO-TRAP results in a successful attack with fewer queries compared to other attacks.

analysis.

- MULTINOISEATTACK [116] which computes search directions $\boldsymbol{r}_t$ separately for each frame by sampling each element of $\boldsymbol{r}_t$ from a standard normal distribution, resulting in a search space dimension of $T \times H \times W \times C$. It does not explicitly consider the temporal dimension; temporal progression in any arbitrary direction is possible between a sequence of perturbation frames.

- ONENOISEATTACK which computes $\boldsymbol{r}_1$ by sampling each element from a standard normal distribution and applies the same $\boldsymbol{r}_1$ across all $\boldsymbol{r}_t (t = 1, 2, \cdots, T)$. ONENOISEATTACK reduces the search space but completely ignores the temporal dimension when generating the perturbation.

- MOTIONSAMPLERATTACK [290] which uses the optical flow of the original video $\boldsymbol{x}$ to warp $\boldsymbol{r}_{\text{frame}}$ to get each $\boldsymbol{r}_t$. It reduces the search space by using the motion prior of $\boldsymbol{x}$ as the temporal progression between perturbation frames. In contrast, rather than fixing the temporal search space using a motion prior, GEO-TRAP parameterizes the temporal structure of the space with $\boldsymbol{\Phi}_{\text{warp}}$.

Algorithm 4 TRANS-WARP($\boldsymbol{r}_{\text{frame}} \in \mathbb{R}^{H\times W\times C}$, $\boldsymbol{\Phi}_{\text{warp}} \in \mathbb{R}^{T\times D}$)$\rightarrow$ Estimate $\boldsymbol{\pi} \in \mathbb{R}^{T\times H\times W\times C}$

Input : noise tensor $\boldsymbol{r}_{\text{frame}}$, warp tensors $\boldsymbol{\Phi}_{\text{warp}}$, transformation operation $\mathcal{T}_{\boldsymbol{\phi}}(\cdot)$.

Output: candidate directions $\boldsymbol{\pi} = [\boldsymbol{r}_1, \boldsymbol{r}_2, \cdots, \boldsymbol{r}_T]$.

Initialize $\boldsymbol{\pi} = \emptyset$

for $t = [1, 2, \cdots, T]$ **do**

  $\boldsymbol{\phi}_t = \boldsymbol{\Phi}_{\text{warp}}[t]$

  $\boldsymbol{r}_t = \mathcal{T}(\boldsymbol{r}_{\text{frame}}, \phi_t)$ /* Warping Operation */

  $\boldsymbol{\pi} \leftarrow$ append $\boldsymbol{r}_t$

end

return $\boldsymbol{\pi}$

We measure the gradient estimation quality by calculating the cosine similarity between the ground truth $\boldsymbol{g}^\star$ and the estimated gradient $\boldsymbol{g}$ following [124] for the aforementioned baselines. For each attack, we average over 1000 randomly selected videos with their cosine similarity values in the first attack iteration. We choose the first iteration because the initial $\boldsymbol{g}^\star$ is the same for the different attack methods, ensuring a fair comparison. As shown in Figure 4.2a, our proposed method for estimating the gradients, yields $\boldsymbol{g}$ of the best quality for both untargeted and targeted attacks among all evaluated approaches. This leads to faster loss convergence / few queries as shown in Figure 4.2b. We validate such trends with different loss functions and more datasets in the Supplementary Material.

The empirical results validate that by carefully considering the temporal dimension and parameterizing the temporal structure of the gradient search space with geometric transformations, GEO-TRAP finds better gradients. GEO-TRAP and MOTIONSAMPLERATTACK [290] are better than the other two; the reason could be that temporally structured perturbations are more likely to disrupt the motion context of videos. However, the gradients estimated by MOTIONSAMPLERATTACK [290] are not as effective as our proposed approach; the reason could be that the motion-prior

of the clean video does not necessarily represent the temporal behavior of effective video perturbations. By allowing flexibility of the temporal progression while maintaining only a minimally sufficient space through its geometric parameterization, GEO-TRAP generates effective temporally structured perturbations. Note that one could use other, potentially better ways to parameterize the temporal progression of the video perturbation; this is regarded as future works.

4.6 Experiments

Datasets. Following previous work like [146], we use the human action recognition dataset UCF-101 [238] and the hand gesture recognition dataset 20BN-JESTER (Jester) [176] to validate our attacks. *UCF-101* includes 13320 videos from 101 human action categories (e.g., applying lipstick, biking, blow drying hair, cutting in the kitchen). Given the diversity it provides, we consider the dataset to validate the feasibility of our attacks on *coarse-grained* actions. *Jester*, on the other hand, includes hand gesture videos that are recorded by crowd-sourced workers performing 27 kinds of gestures (e.g., sliding hand left, sliding two fingers left, zooming in with full hand, zooming out with full hand). The appearance of different hand gestures is similar; it is the motion information that matters in the video classification. We use this dataset to validate our attack with regard to *fine-grained* actions.

Baselines. Among the four state-of-the-art black-box video attack methods [124, 267, 282, 290] described in Section 4.3, we use [282, 290] as baselines for following reasons. Our first baseline is MOTIONSAMPLERATTACK [290], which has been shown to outperform PATCHATTACK [124] , ONE-NOISE and MULTI-NOISE attacks (introduced in Section. 4.5). Our second baseline is HEURISTICATTACK [267]. We note that SPARSEATTACK [282] is not included in our analysis

as we couldn't replicate their results.

Attack Settings. We consider four state-of-the-art video classification models representing diverse methodologies of learning from videos, i.e., C3D [253], SlowFast [75], TPN [283] and I3D [32], as our black-box victim models to attack. More details about the four video models are provided in the Supplementary Materials. For UCF-101, we randomly select one video from each category following the setting in [124,290]. For Jester, since the number of categories is small, we randomly select four videos from each category. All attacked videos are correctly classified by the black-box model. For targeted attack, a random target class is chosen for each video. The maximum noise value $\rho_{\max}$ is 10 pixel values (out of 255) following [146, 181, 185]. We provide more results for different $\rho_{\max}$ in Supplementary Material. Note that since the perturbation generated by HEURISTICATTACK [267] is sparse and thus more imperceptible, we do not impose a perturbation budget on it. We set the maximum query limit to $Q = 60,000$ for untargeted attack and $Q = 200,000$ for targeted attack. The other hyper-parameters, i.e., ϵ, δ, η, and h take the same values as mentioned in [290]. Unless otherwise specified, a transformation-dilation transformation (with $D = 3$) is used for our query-based attack. We provide the implementation of GEO-TRAP in the Supplementary Material as `code.zip`.

Metrics. Following [124, 290], we evaluate GEO-TRAP, in terms of *(a)* Success Rate (SR), i.e., the total success rate in attacking within query and perturbation budgets; and *(b)* Average Number of Queries (ANQ) i.e., the average total queries from attacks for all videos (including failed ones).

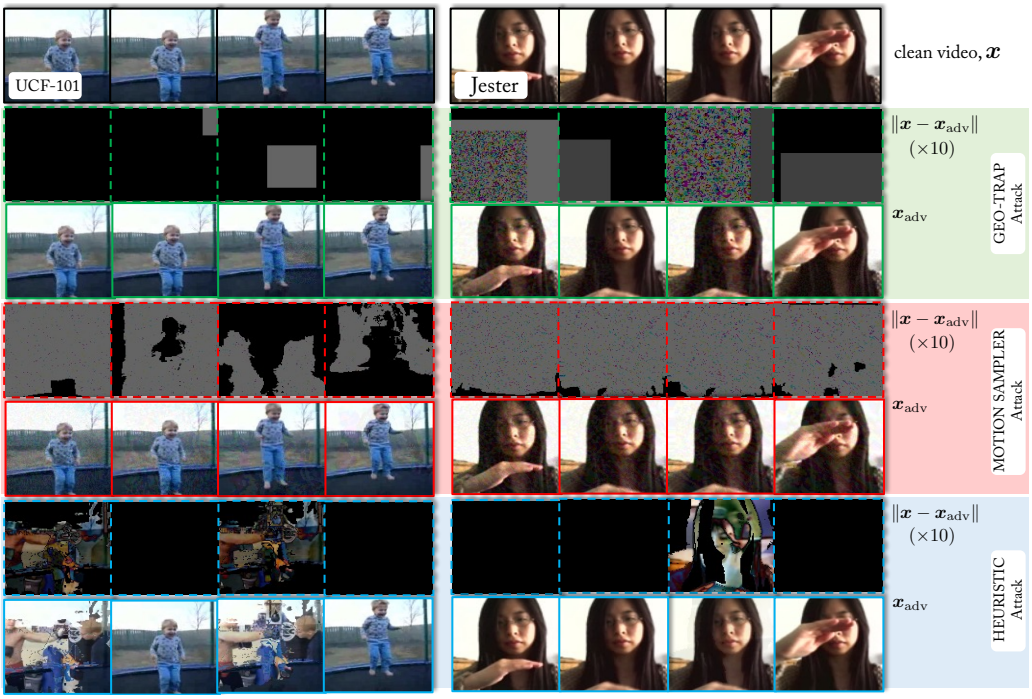


Figure 4.3: Visualization of Perturbations and Perturbed Video. We visualize the generated perturbations and perturbed video for GEO-TRAP and other baselines for UCF-101 (*left*) and Jester (*right*) datasets for untargeted attack against SlowFast classifier with $\rho_{\max} = {}^{10}/_{255}$.

4.6.1 Comparison to State-of-the-Art

Untargeted Attack. We report the untargeted attack performance of our attack method and the baseline methods in Table 4.2. We observe that, in general, GEO-TRAP requires fewer average number of queries when attacking different black-box victim models: on average over 45 % fewer queries than MOTIONSAMPLERATTACK [290]. At the same time, GEO-TRAP yields higher attack success rates: on average about 6% higher than HEURISTICATTACK [267]. When attacking Slow-Fast model on the Jester dataset, GEO-TRAP achieves 100% successful rate with only 521 queries while the baseline methods need at least 1906 queries. We also observe that the TPN model is more robust towards black-box attacks compared with the other three video recognition models.

Visualization. We show two visualizations of adversarial frames on Jester and UCF-101 in Fig. 4.3. We observe that the generated adversarial frames have little difference from the clean ones but can lead to a failed classification. Also, our attack method could lead to *sparse* perturbations in the spatial and temporal dimension as the perturbations are sometimes zoomed out (thus get very small), and sometimes are translated out of the sight with choice of geometric transformation. More examples are in the Supplementary Material.

Targeted Attack. We report the targeted attack performance of our method and the baseline methods in Table. 4.3. We observe that in some cases, HEURISTICATTACK [267] requires fewer number of queries than GEO-TRAP, but its attack success rates are pretty low in those cases. For example, when attacking the TPN model on the Jester dataset, although HEURISTICATTACK [267] requires only 12k average number of queries, its attack success rate is less than half of ours, 44.4% v.s. 92.6%. The reason is that the gradient estimated by HEURISTICATTACK [267] may vanish after a certain number of queries. GEO-TRAP consistently yields higher attack success rates, on average over 30% higher than HEURISTICATTACK [267] and over 8% higher than MOTIONSAMPLERATTACK [290]. In addition, in most cases, GEO-TRAP requires fewer average number of queries than the two baseline attacks, on average over 45 % fewer queries than MOTIONSAMPLERATTACK [290]. The targeted attack performance further validates the effectiveness of our method.

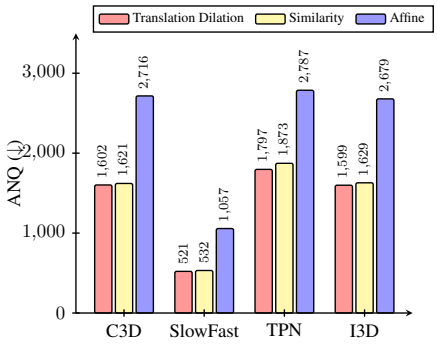


Figure 4.4: Performance with different $\mathcal{M}_{\phi}$. GEO-TRAP results in best performance when $\mathcal{M}_{\phi}$ is set as translation-dilation operation.

4.6.2 Different Geometric Transformations in TRANS-WARP

As discussed in Section 4.4.2, different kinds of geometric transformations could be used in the TRANS-WARP function. In addition to the translation-dilation transformation ($\mathcal{M}_{\phi}^{\text{TD}}$ in (4.8), $D = 3$) employed throughout the chapter, we report the performance of GEO-TRAP with two other different geometric transformations, i.e., similarity transformation ($\mathcal{M}_{\phi}^{\text{S}}$ in (4.8), $D = 4$) and affine transformation ($\mathcal{M}_{\phi}$ in (4.7), $D = 6$). Figure 4.7 shows the untargeted attack performance on Jester with these different geometric transformations (more results are available in the Supplementary Material). We observe that the transformation with fewer degrees of freedom (DOF) (translation-dilation transformation) tends to require fewer queries while having the same or higher attack success rates (the attack success rates are available in the Supplementary Material). We believe that $D = 3$ provides enough temporal flexibility to disrupt the motion context of the videos; additional degrees of freedom seemingly increase the search space unnecessarily, resulting in more queries.

4.7 Conclusion

Black-box adversarial attacks on video classifiers is a challenging problem that has been largely understudied. In this work, we demonstrate that searching for effectual gradients in a reduced but structured search space for crafting perturbations leads to highly successful attacks with fewer queries compared to state-of-the-art attack strategies. In particular, we propose a novel iterative algorithm that employs Geometric transformations to parameterize and reduce the search space, for estimating gradients that maximize the probability of mis-classification of the perturbed video. This simple and novel strategy exposes the vulnerability of widely used video classification models. For instance, GEO-TRAP decreases average query numbers by 64.78%, 72.66% and 47.21% to attack C3D, SlowFast, and I3D, respectively, for close to a 100% success rate in untargeted attacks.

4.8 Broader Impact

In this work, by leveraging geometric transformations for effective gradient estimations, we propose a highly query-efficient adversarial attack on video classification models which demonstrates state-of-the-art results. As more and more safety-critical systems (e.g., perceptual modules in autonomous vehicles) nowadays rely on video models, we are hopeful that our work, in addition to future research, can eventually help build sufficiently robust video models to best avoid malicious sub-versions. On one hand, we believe our algorithm could allow further research in adversarial robustness and data augmentation strategies of deep vision models. It should also give a direction to researchers to design counter defense methodologies. On the other hand, it highlights a key drawback of different video classifiers which will allow adversaries to design more sophisticated attacks, both in white-box and black-box settings. Addressing such fallacies in designing deep neural

networks is of utmost importance before introducing them in real-world scenarios.

4.9 Appendix

In this Supplementary Material, we present details about the victim video classification models used in our experiments. We present more attack performance results with different perturbation budgets to further validate the efficiency and effectiveness of our method, GEO-TRAP. We report the error bars with respect to three sources of randomness during adversarial attacks. We compare different geometric transformations on more datasets and attack goals. We measure the gradient estimation quality with more loss functions and how that GEO-TRAP consistently estimate better gradients compare to the baseline methods. We provide more visualization examples of the generated adversarial examples. Last, we talk about the implementation code of GEO-TRAP.

4.9.1 Victim Video Classifiers: Clean Test Accuracy

We consider four state-of-the-art video classification models, representing diverse methodologies of learning from videos, i.e., C3D [253], SlowFast [75], TPN [283] and I3D [32], as our black-box victim models to perform adversarial attack. The *C3D* model applies 3D convolution to learn spatio-temporal features from videos. *SlowFast* uses a two-pathway architecture where the slow pathway operates at a low frame rate to capture spatial semantics and the fast pathway operates at a high frame rate to capture motion at fine temporal resolution. *TPN* captures actions at various tempos by using a feature-level temporal pyramid network. *I3D* proposes the Inflated 3D ConvNet(I3D) with Inflated 2D filters and pooling kernels of traditional 2D CNNs. All the models are trained using open-source toolbox MMAction2 [49] with their default setups. The test accuracy

of the victim models with clean 16-frame videos on both UCF-101 and Jester datasets are shown in Table 4.4. Note that both datasets do not contain personally identifiable information and offensive contents.

4.9.2 Additional Experiments with Different Perturbation Budgets

We present additional analysis of the attack performance of GEO-TRAP and our two baseline methods, i.e., HEURISTICATTACK [267] and MOTIONSAMPLERATTACK [290] for $\rho_{\max} = 8, 16$ in Table 4.5. Note that for comprehensibility, we also provide the results for $\rho_{\max} = 10$ from the main chapter in Table 4.5. We observe that GEO-TRAP consistently outperforms MOTIONSAMPLERATTACK [290]; GEO-TRAP requires less number of queries while achieves same or higher attack success rates.

4.9.3 Statistical Comparison of Different Attack Methods

We have three sources of randomness in our experiments: *a*) the sampling of $\boldsymbol{r}_{\text{frame}}$ in both GEO-TRAP and MOTIONSAMPLERATTACK [290] and the sampling of Φ_{warp} in GEO-TRAP; *b*) direction initialization sampling in HEURISTICATTACK [267]; *c*) target label sampling in targeted adversarial attacks for all three methods. To account for all these three randomness, we run the targeted attack against I3D model on Jester dataset under perturbation budget $\rho_{\max} = 16$ for the three methods for five times. Using targeted attack strategy allows us to include the randomness of the target label sampling. We choose Jester dataset as it generally takes few queries to attack Jester dataset, thus saving testing time. We choose perturbation budget $\rho_{\max} = 16$ as we observe that the

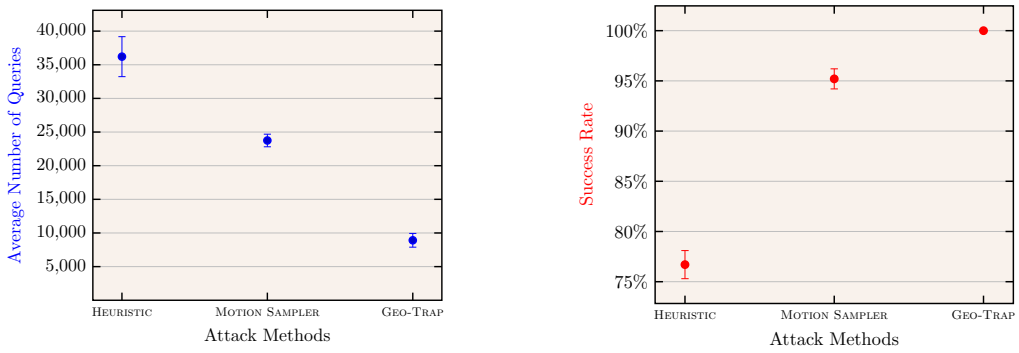


Figure 4.5: Error bar plot to compare the performance (success rate and average number of queries) of different attack methods. We observe that our method outperforms the baseline methods in a statistically significant way. Detailed numbers are presented in Table 4.6

attacks under such budget generally take few queries. We choose I3D model because compared to C3D and SlowFast, the attack success rates against I3D are not always 100%; which is good for measuring the error bars for the attack success rates. In addition, compared to TPN, it generally takes fewer queries to launch the attack against I3D.

We report the mean, standard deviation, and standard error in Table 4.6 and present the error bar plot (with mean and standard error) in Figure 4.5. GEO-TRAP, compared to other methods, requires statistically fewer number of queries while achieving statistically higher attack success rates than the baseline methods.

4.9.4 Additional Experiments with Different Geometric Transformations

GEO-TRAP can employ different kinds of geometric transformations in the TRANS-WARP function. In addition to the translation-dilation transformation ($D = 3$) employed throughout the main chapter, we report the performance of GEO-TRAP with two other different geometric transformations, i.e., similarity transformation ($D = 4$) and affine transformation ($D = 6$).

Recall that untargeted attack performance of GEO-TRAP using these three geometric

transformations on Jester dataset is reported in the main chapter. In this section, we present the a more comprehensive set of results on both targeted and untargeted attacks, for both Jester and UCF-101 datasets in Table 4.7. We observe that the transformation with fewer degrees of freedom, i.e., translation-dilation transformation tends to requires fewer queries while having the same or higher attack success rates on Jester Dataset; this trend is consistent no matter which attack goal is used. On UCF-101 dataset, the transformations with fewer degrees of freedom, i.e., translation-dilation transformation and similarity transformation, require fewer queries while having the same or higher attack success rates compared to the affine transformation.

4.9.5 Additional Experiments on GEO-TRAP with Different Loss Functions

In this section, we further validate that, compared to our three baseline methods (i.e., MULTINOISEATTACK [116], ONENOISEATTACK, MOTIONSAMPLERATTACK [290]), the gradients searched with GEO-TRAP are better. This is demonstrated by the fact that GEO-TRAP's gradients generally have larger cosine similarity with the ground truth gradients. This trend is loss function agnostic, with both untargeted and targeted attacks, as shown in Figure 4.6. We consider four attack loss functions, three untargeted attack loss functions and one targeted attack loss function, described below.

We start with explaining the flicker loss used for untargeted attack and the cross-entropy loss used for targeted attack in the main chapter. Flicker loss is defined with the probability scores of the top-2 labels returned by $\boldsymbol{f_\theta}(\boldsymbol{x})$ following [206]. In particular, if the attack is not successful, the most likely label predicted by $\boldsymbol{f_\theta}(\boldsymbol{x})$ will be the true label y. We denote the probability score associated with this label as $p_y(\boldsymbol{x})$. Similarly, we denote the *second* most likely label predicted

by $\boldsymbol{f_\theta}(\boldsymbol{x})$ as y' and its corresponding probability score as $p_{y'}(\boldsymbol{x})$. The loss function is defined to encourage $p_{y'}(\boldsymbol{x})$ increasing and $p_y(\boldsymbol{x})$ decreasing until $p_{y'}(\boldsymbol{x}) > p_y(\boldsymbol{x})$ and y' becomes the predicted top-1 label. This loss function can be mathematically denoted as follows.

$$\mathcal{L}_{\text{flicker}}(\boldsymbol{x}, y) = \left[\min \left(\frac{1}{m} \mathcal{K}(\boldsymbol{x}, y)^2, \mathcal{K}(\boldsymbol{x}, y) \right) \right]_+ \text{with, } \mathcal{K}(\boldsymbol{x}, y) = p_y(\boldsymbol{x}) - p_{y'}(\boldsymbol{x}) + m \quad (4.9)$$

Here, $[a]_+ = \max(0, a)$ and $m > 0$ is the desired margin of the original class probability below the adversarial class probability. We refer readers to [206] for more detailed explanation of (4.9).

For the targeted attack, the cross-entropy loss is defined as follows.

$$\mathcal{L}(\boldsymbol{x}, y_\top) = -\log \left(p_{y_\top}(\boldsymbol{x}) \right) \quad (4.10)$$

where $p_{y_\top}(\boldsymbol{x})$ is the probability score of the target label returned by $\boldsymbol{f_\theta}(\boldsymbol{x})$.

In addition to the above loss functions, we consider two other untargeted loss functions for gradient analysis of attacks methods. The first one is the untargeted attack loss function defined in [290] based on CW2 loss [31] as shown in the following.

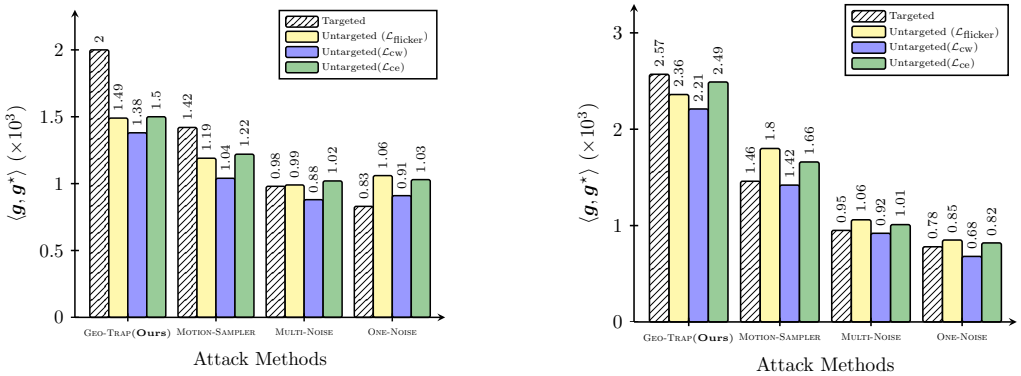
$$\mathcal{L}_{\text{cw}}(\boldsymbol{x}, y) = \left[p_y(\boldsymbol{x}) - p_{y'}(\boldsymbol{x}) \right]_+ \quad (4.11)$$

where, $p_y(\boldsymbol{x})$ is the largest probability score, which should be associated with the true label y, and $p_{y'}(\boldsymbol{x})$ is the second largest probability score, which is associated with the second most confident label y'. The second loss is a cross-entropy loss where a lower $p_y(\boldsymbol{x})$ is encouraged, as shown in the following.

$$\mathcal{L}_{\text{ce}}(\boldsymbol{x}, y) = -\log \left(1 - p_y(\boldsymbol{x}) \right) \quad (4.12)$$

We calculate the average cosine similarity (over 1000 randomly chosen samples) between the ground truth gradients and the estimated gradients for GEO-TRAP and the three baselines. As

shown in Figure 4.6, for all the five different loss functions considered and on both Jester (see Figure 4.6a) and UCF-101 (see Figure 4.6b) dataset, the gradients searched by GEO-TRAP have better quality consistently. This explains why GEO-TRAP requires less number of queries while achieving the same or higher attack success rates.



(a) Jester dataset

(b) UCF-101 dataset

Figure 4.6: Evaluation of gradient estimation quality by calculating the cosine similarity between the ground truth gradient $\boldsymbol{g}^\star$ and the estimated gradient $\boldsymbol{g}$ calculated by different attack methods.

4.9.6 Additional Examples of Adversarial Videos

In this section, we provide additional adversarial examples on both Jester and UCF-101 datasets as shown in Figure 4.7. We observe that the generated adversarial frames have little difference from the clean ones but can lead to a failed classification.


Jester
UCF-101
clean video, $\boldsymbol{x}$
$\|\boldsymbol{x} - \boldsymbol{x}_{\text{adv}}\|$ (×10)
$\boldsymbol{x}_{\text{adv}}$
GEO-TRAP Attack
MOTION SAMPLER Attack
HEURISTIC Attack
Jester
UCF-101
clean video, $\boldsymbol{x}$
$\|\boldsymbol{x} - \boldsymbol{x}_{\text{adv}}\|$ (×10)
$\boldsymbol{x}_{\text{adv}}$
GEO-TRAP Attack
MOTION SAMPLER Attack
HEURISTIC Attack

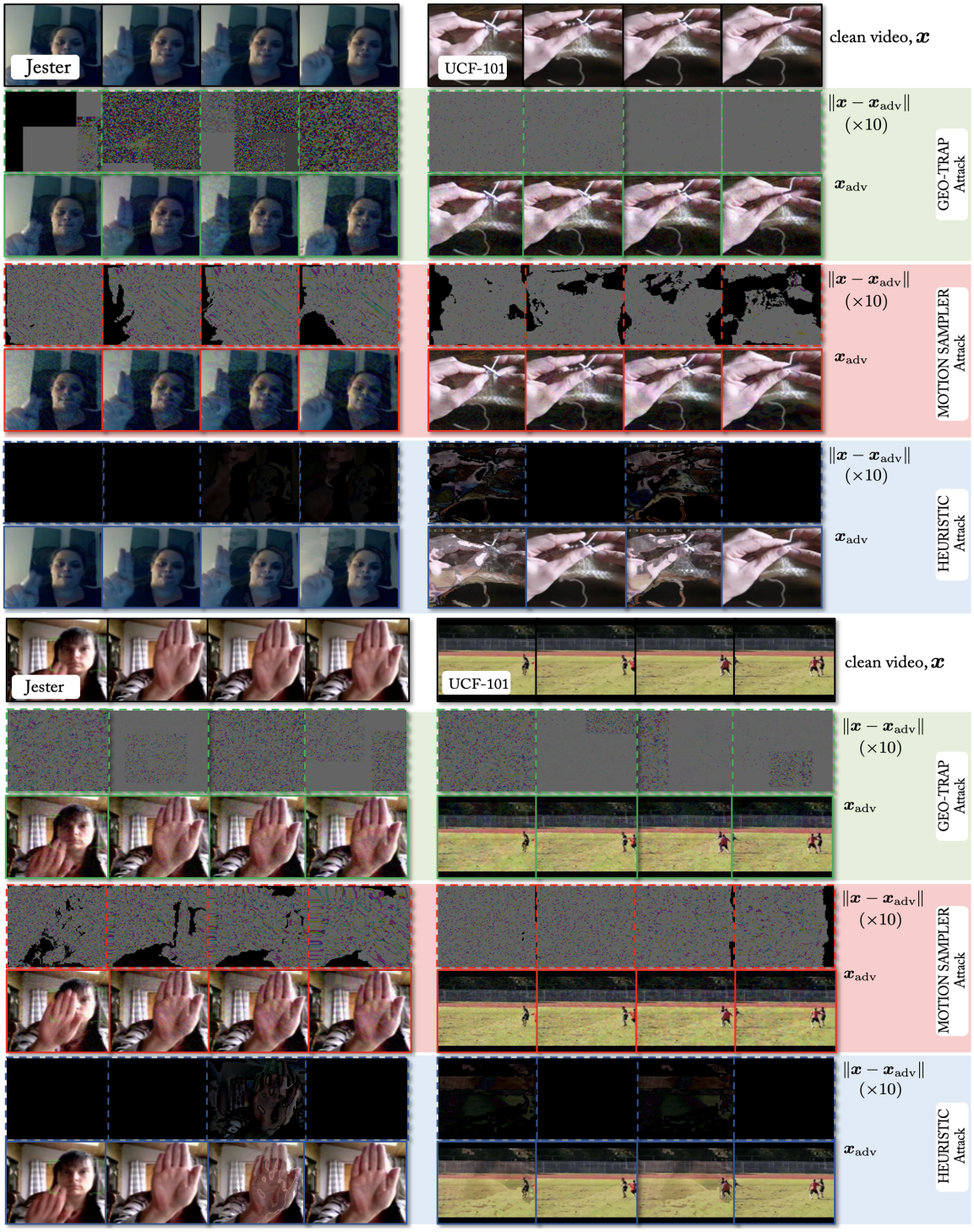


Figure 4.7: The visualization of the perturbation ($\times 10$) and adversarial frames of our methods and the two baseline methods on Jester (left column) and UCF-101 datasets (right column).

Table 4.2: Untargeted Attacks. GEO-TRAP demonstrates highly successful untargeted attacks (high Success Rate (SR)) with fewer queries (low Average Number of Queries (ANQ))

Datasets	Methods	Black-box Video Classifiers							
		C3D		SlowFast		TPN		I3D	
		ANQ (↓)	SR (↑)	ANQ (↓)	SR (↑)	ANQ (↓)	SR (↑)	ANQ (↓)	SR (↑)
Jester	HEURISTICATTACK [267]	4699	99.0%	3572	98.1%	4679	82.0%	4248	98.1%
	MOTIONSAMPLERATTACK [290]	4549	99.0%	1906	100%	6269	91.3%	3029	99.4%
	GEO-TRAP (**Ours**)	**1602**	**100%**	**521**	**100%**	**3315**	**92.4%**	**1599**	**100%**
UCF-101	HEURISTICATTACK [267]	**5206**	70.2%	3507	87.2%	**6539**	71.8%	6949	84.7%
	MOTIONSAMPLERATTACK [290]	14336	81.6%	4673	97.2%	20369	75.8%	7400	94.4%
	GEO-TRAP (**Ours**)	11490	**86.2%**	**1547**	**98.8%**	17716	**76.1%**	**4887**	**97.4%**

Table 4.3: Targeted Attacks. GEO-TRAP demonstrates highly successful targeted attacks (high Success Rate (SR)) with fewer queries (low Average Number of Queries (ANQ))

Datasets	Methods	Black-box Video Classifiers							
		C3D		SlowFast		TPN		I3D	
		ANQ (↓)	SR (↑)	ANQ (↓)	SR (↑)	ANQ (↓)	SR (↑)	ANQ (↓)	SR (↑)
Jester	HEURISTICATTACK [267]	15595	46.3%	30768	98.1%	**12006**	44.4%	31088	77.8%
	MOTIONSAMPLERATTACK [290]	26704	98.2%	33087	100%	63721	80.9%	39037	90.7%
	GEO-TRAP (**Ours**)	**6198**	**100%**	**7788**	**100%**	41294	**92.6%**	**19542**	**98.2%**
UCF-101	HEURISTICATTACK [267]	**26741**	29.0%	22152	61.4%	**71828**	36.4%	92244	43.7%
	MOTIONSAMPLERATTACK [290]	100467	71.1%	57126	86.0%	151409	31.6%	96498	59.6%
	GEO-TRAP (**Ours**)	71820	**85.8%**	**21878**	**95.0%**	141629	**40.0%**	**76708**	**74.6%**

Table 4.4: Clean test Accuracy of the victim classifiers

Datasets	**Black-box Video Classifiers**			
	C3D	**SlowFast**	**TPN**	**I3D**
UCF-101	78.8%	85.4%	74.3%	71.7%
Jester	90.1%	89.5%	90.5%	91.2%

Table 4.5: Additional analysis of attack performance with different perturbation budgets $\rho_{\max}$

Budget	Methods	Black-box Video Classifiers							
		C3D		SlowFast		TPN		I3D	
		ANQ (↓)	SR (↑)	ANQ (↓)	SR (↑)	ANQ (↓)	SR (↑)	ANQ (↓)	SR (↑)
Attack: Untargeted, Dataset: Jester									
$\rho_{\max} = 8$	MOTIONSAMPLERATTACK [290]	7310	96.3%	1926	100%	8056	91.3%	5482	98.1%
	GEO-TRAP (**Ours**)	**2614**	**100%**	**553**	**100%**	**4518**	**92.4%**	**2312**	**100%**
$\rho_{\max} = 10$	MOTIONSAMPLERATTACK [290]	4549	99.0%	1906	100%	6269	91.3%	3029	99.4%
	GEO-TRAP (**Ours**)	**1602**	**100%**	**521**	**100%**	**3315**	**92.4%**	**1599**	**100%**
$\rho_{\max} = 16$	MOTIONSAMPLERATTACK [290]	2201	100%	1421	100%	3786	96.3%	1347	100%
	GEO-TRAP (**Ours**)	**311**	**100%**	**137**	**100%**	**3147**	**96.3%**	**551**	**100%**
Attack: Untargeted, Dataset: UCF-101									
$\rho_{\max} = 8$	MOTIONSAMPLERATTACK [290]	16848	78.0%	5436	95.0%	20687	70.0%	9242	92.0%
	GEO-TRAP (**Ours**)	**12100**	**84.0%**	**2064**	**98.0%**	**18433**	**74.0%**	**6647**	**97.0%**
$\rho_{\max} = 10$	MOTIONSAMPLERATTACK [290]	14336	81.6%	4673	97.2%	20369	75.8%	7400	94.4%
	GEO-TRAP (**Ours**)	**11490**	**86.2%**	**1547**	**98.8%**	**17716**	**76.1%**	**4887**	**97.4%**
$\rho_{\max} = 16$	MOTIONSAMPLERATTACK [290]	11605	82.0%	1944	99.%	18055	75.8%	4437	96.0%
	GEO-TRAP (**Ours**)	**9006**	**86.2%**	**858**	**99.0%**	**15972**	**76.1%**	**2643**	**98.0%**
Attack: Targeted, Dataset: Jester									
$\rho_{\max} = 8$	MOTIONSAMPLERATTACK [290]	42136	92.6%	39833	98.1%	121800	52.2%	48788	85.2%
	GEO-TRAP (**Ours**)	**9333**	**100%**	**11433**	**98.1%**	**51799**	**88.9%**	**25552**	**96.3%**
$\rho_{\max} = 10$	MOTIONSAMPLERATTACK [290]	26704	98.2%	33087	100%	63721	80.9%	39037	90.7%
	GEO-TRAP (**Ours**)	**6198**	**100%**	**7788**	**100%**	**41294**	**92.6%**	**19542**	**98.2%**
$\rho_{\max} = 16$	MOTIONSAMPLERATTACK [290]	8696	100%	18901	100%	40643	90.7%	25308	94.4%
	GEO-TRAP (**Ours**)	**4219**	**100%**	**3855**	**100%**	**16979**	**96.3%**	**9110**	**100%**
Attack: Targeted, Dataset: UCF-101									
$\rho_{\max} = 8$	MOTIONSAMPLERATTACK [290]	136327	51.7%	72807	76.7%	153355	35.0%	107304	51.1%
	GEO-TRAP (**Ours**)	**90401**	**82.5%**	**27306**	**93.0%**	**150052**	**36.8%**	**91773**	**59.3%**
$\rho_{\max} = 10$	MOTIONSAMPLERATTACK [290]	100467	71.1%	57126	86.0%	151409	31.6%	96498	59.6%
	GEO-TRAP (**Ours**)	**71820**	**85.8%**	**21878**	**95.0%**	**141629**	**40.0%**	**76708**	**74.6%**
$\rho_{\max} = 16$	MOTIONSAMPLERATTACK [290]	69344	79.6%	37759	92.8%	143504	45.0%	70707	75.0%
	GEO-TRAP (**Ours**)	**35641**	**98.0%**	**18177**	**95.0%**	**132065**	**45.5%**	**44400**	**86.0%**

Table 4.6: Statistical results with respect to the random seed after running attacks multiple times (*Attack*: Targeted, *victim classifier*: I3D, *Dataset*: Jester, *perturbation budget*: $\rho_{\max} = 16$)

	Methods					
	HEURISTIC		MOTION SAMPLER		GEO-TRAP	
	ANQ (↓)	SR (↑)	ANQ (↓)	SR (↑)	ANQ (↓)	SR (↑)
Run 1	31088	77.9%	25308	94.4%	9110	100%
Run 2	38388	76.0%	20290	96.3%	10110	100%
Run 3	42098	74.1%	23356	94.4%	5758	100%
Run 4	42022	74.0%	24464	96.3%	7799	100%
Run 5	27431	81.5%	25312	94.4%	11782	100%
Mean	36205	76.7%	23746	95.2%	8912	100%
Standard Deviation	6643	3.1%	2092	1.0%	2286	0%
Standard Error	2971	1.4%	936	0.5%	1022	0%

Table 4.7: Additional analysis of attack performance of GEO-TRAP with different geometric transformations $\mathcal{M}_\phi$

Geometric Transformations, $\mathcal{M}_\phi$	**Black-box Video Classifiers**							
	C3D		**SlowFast**		**TPN**		**I3D**	
	ANQ (↓)	SR (↑)	ANQ (↓)	SR (↑)	ANQ (↓)	SR (↑)	ANQ (↓)	SR (↑)
Attack: Untargeted, Dataset: Jester								
Translation Dilation	**1602**	**100%**	**521**	**100%**	**3315**	**92.4%**	**1599**	**100%**
Similarity	1621	100%	532	100%	3746	92.4%	1629	100%
Affine	2716	100%	1057	100%	4579	91.6%	2679	100%
Attack: Targeted, Dataset: Jester								
Translation Dilation	**6198**	**100%**	**7788**	**100%**	**41294**	**92.6%**	**19542**	**98.2%**
Similarity	6431	100%	7939	100%	42594	90.7%	19369	98.2%
Affine	10326	100%	15360	100%	55276	90.7%	32006	94.4%
Attack: Untargeted, Dataset: UCF-101								
Translation Dilation	11490	**86.2%**	1547	**98.9%**	17716	76.1%	**4887**	**97.4%**
Similarity	**10624**	85.8%	**1489**	98.6%	**17492**	**76.7%**	5694	95.0%
Affine	12792	84.8%	3088	98.0%	17773	75.0%	8291	94.0%

Chapter 5

Connecting the Dots: Detecting Adversarial Perturbations Using Context Inconsistency

5.1 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

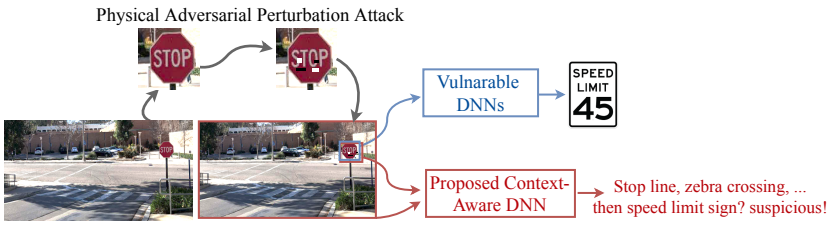


Figure 5.1: An example of how our proposed context-aware defense mechanism works. Previous studies [69, 235] have shown how small alterations (graffiti, patches etc.) to a stop sign make a vulnerable DNN classify it as a speed limit. We posit that a stop sign exists within the wider context of a scene (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.

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.

Key Words: object detection, adversarial perturbation, context

5.2 Introduction

Recent studies have shown that Deep Neural Networks (DNNs), which are the state-of-the-art tools for a wide range of tasks [58, 97, 125, 177, 242], are vulnerable to adversarial perturbation attacks [146, 299]. 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. 5.1; a stop sign often co-exists with a stop line, zebra crossing, street nameplate and other characteristics of a road intersection. 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 [88, 121, 156], 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 [15, 66, 105, 163]. We assume a framework for object detection similar to [216], 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 [68] and Microsoft COCO [154]. 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 [278] that does not use context in detecting adversarial perturbations.

5.3 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 [153, 158, 214, 216]. Faster R-CNN [216] 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 [36, 235, 293] (in addition to those targeting image classifiers [10, 69, 137]). Besides mis-categorization attacks, two new types of attacks have emerged against object detectors: the *hiding attack* and the *appearing attack* [36, 235] (see Section 5.4.1 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 [76, 99, 156]. *Prediction inconsistency based detection methods* process the images and check for consistency between predictions on the original images

and processed versions [151, 278]. *Other methods train a second binary classifier* to distinguish perturbed inputs from clean ones [148, 167, 179]. However many of these are effective only on small and simple datasets like MNIST and CIFAR-10 [30]. 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 5.1 summarizes the differences between our method and the other defense methods; we extend FeatureSqueeze [278], 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 [15, 66, 103, 194, 250]. Earlier works that incorporate context information into DNN-based object detectors [43,77,187] use object relations in post-processing, where the detected objects are re-scored by considering object relations. Some recent works [37, 141] 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 [163] or neural attention models [105] 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, [163] 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.

Detection	Beyond MNIST CIFAR	Do not need perturbed samples for training	Extensibility to object detection
PCAWhiten [99]	✗	✓	✗, PCA is not feasible on large regions
GaussianMix [76]	✗	✗	✗, Fixed-sized inputs are required
Steganalysis [156]	✓	✗	✗, Unsatisfactory performance on small regions
ConvStat [179]	✗	✗	✓
SafeNet [167]	✓	✗	✓
PCAConv [148]	✓	✗	✗, Fixed-sized inputs are required
SimpleNet [84]	✗	✗	✓
AdapDenoise [151]	✓	✗	✓
FeatureSqueeze [278]	✓	✓	✓

Table 5.1: Comparison of existing detection-based defenses; since FeatureSqueeze [278] meets all the basic requirements of our approach, it is used as a baseline in the experimental analysis.

5.4 Methodology

5.4.1 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 Faster R-CNN model

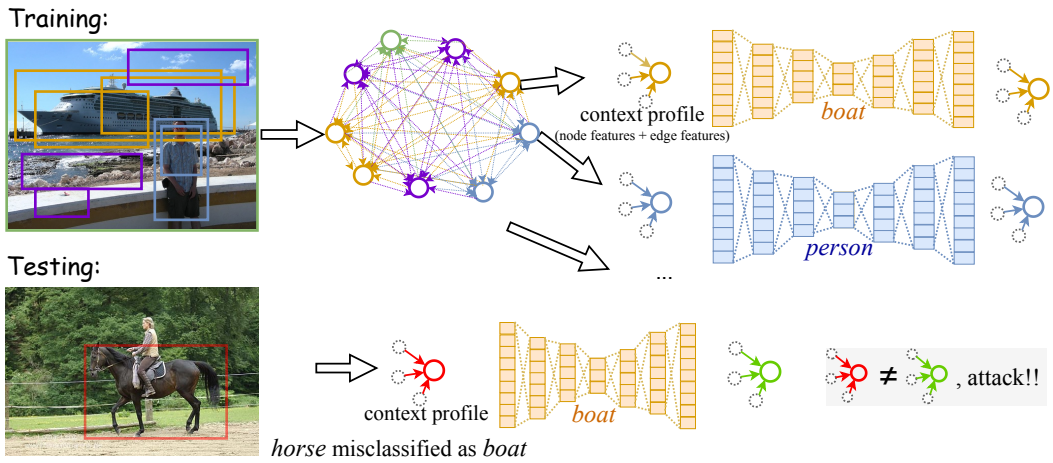


Figure 5.2: Training phase: a fully connected graph is built to connect the regions of the scene image – details in Fig. 5.3; context information relating to each object category is collected and used to train auto-encoders. Testing phase: the context profile is extracted for each region and input to the corresponding auto-encoder to check if it matches with benign distribution.

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 [36, 235, 293]:

- *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. 5.2 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 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.

5.4.2 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

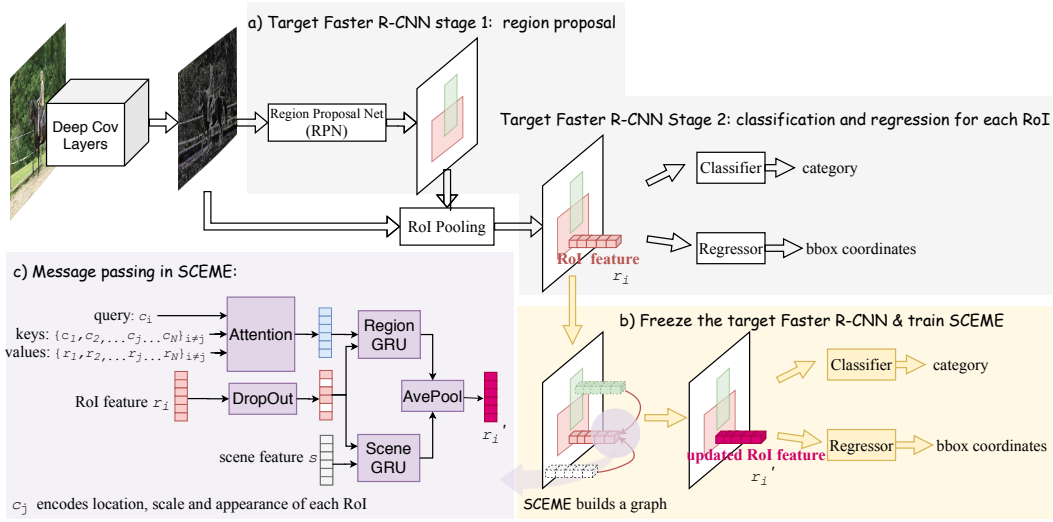


Figure 5.3: (a) The attack target model, the Faster R-CNN, is a two-stage detector. (b) SCEME is built upon the proposed regions from the first stage of the Faster R-CNN, and updates the RoI features by message passing across regions. (c) Zooming in on SCEME shows how it fuses context information into each RoI, by updating RoI features via Region and Scene GRUs.

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. 5.3(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 [101] 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) [41] with attention [11] 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 [163]. 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. 5.3(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'.

5.4.3 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 [41].

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 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 $x = [r, \gamma_{u1}, \gamma_{u2}, \gamma_{r1}, \gamma_{r2}]$.

Algorithm 5 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 $\leftarrow$ 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$[\tilde{c}] \leftarrow$ 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, $\{AutoEncoder_c\}, thresh_{err}$

5.4.4 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 [113, 276], 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 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 5 and 6.

5.5 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

Algorithm 6 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 $PerturbedSet$

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.

5.5.1 Implementation Details

Datasets. We use both PASCAL VOC [68] and MS COCO [154]. PASCAL VOC contains 20 object categories. Each image, on average, has 1.4 categories and 2.3 instances [154]. 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

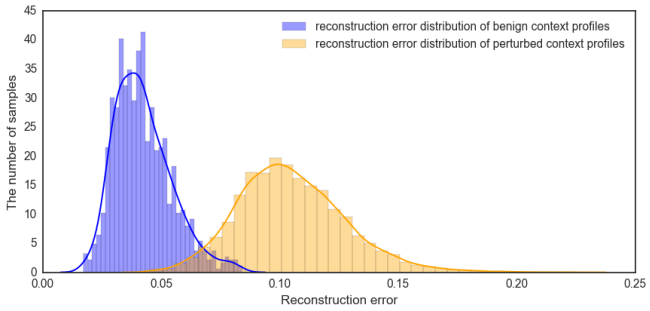
supplementary material.

Attack Implementations. For digital attacks, we use the standard iterative fast gradient sign method (IFGSM) [137] and constrain the perturbation location 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 [69, 235], 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.

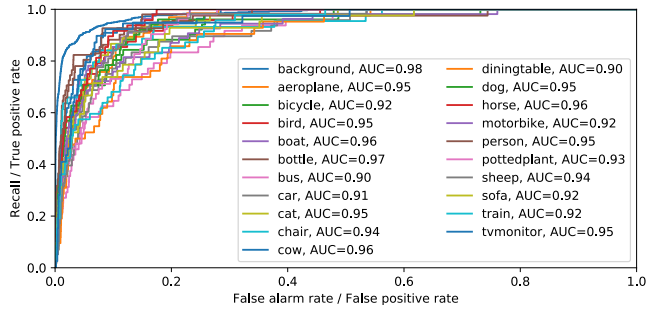
Defense Implementation. Momentum optimizer with momentum 0.9 is used to train `SCEME`. The learning rate is 5e-4 and decays every 80k iterations at a decay rate of 0.1. The training finishes after 250k iterations. Adam optimizer is used to train auto-encoders. The learning rate is 1e-4 and reduced by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

5.5.2 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 per-



(a)



(b)

Figure 5.4: (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.

turbed 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. 5.4(a), the context profiles of perturbed regions do not conform with the benign distribution of aeroplanes' 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. 5.4(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. 5.2. Three baselines are considered.

- *FeatureSqueeze* [278]. As discussed in Tab. 5.1, 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* [14]. 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

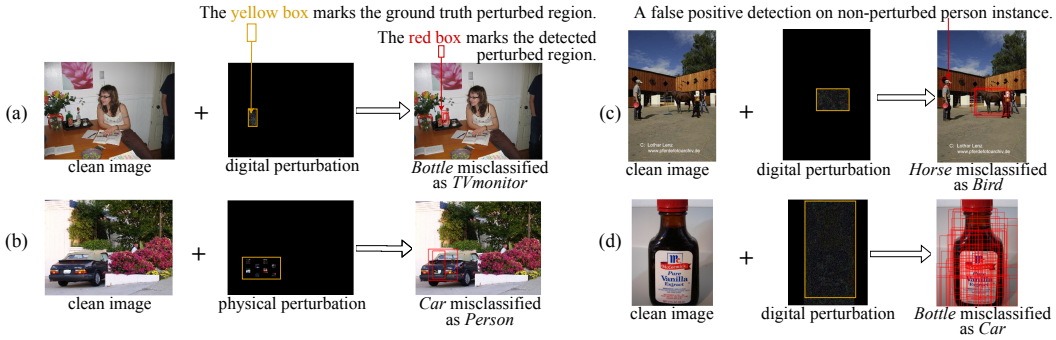


Figure 5.5: A few interesting examples. SCEME successfully detects both digital and physical perturbations as shown in (a) and (b). (c) shows that the horse misclassification affects the context profile of person and leads to false positive detection on the person instance. (d) Appearance information and spatial context are used to successfully detect perturbations.

achieve by explicitly modeling context information with SCEME.

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. 5.5. 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. 5.5(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. 5.5(c). We observe that this rarely happens. In most cases, although

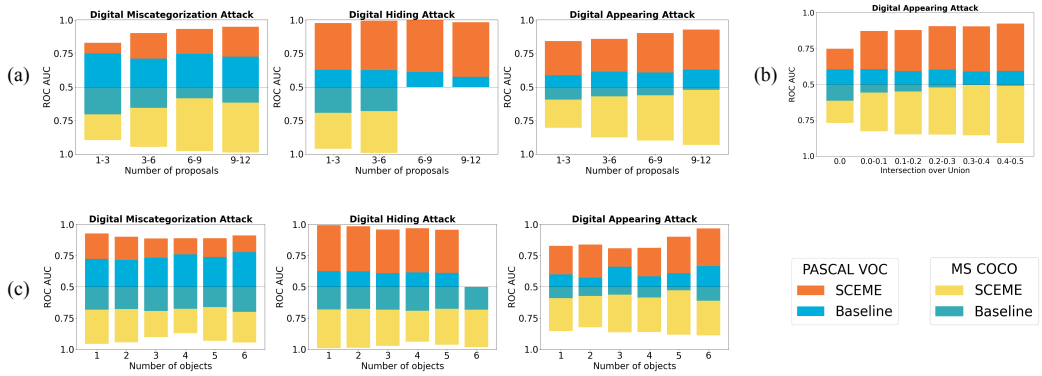


Figure 5.6: Subfigures are diverging bar charts. They start with ROC-AUC = 0.5 and diverge in both upper and lower directions: upper parts are results on PASCAL VOC and lower parts are on MS COCO. For each dataset, we show both the results from the FeatureSqueeze baseline and SCEME, using overlay bars. (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).

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. 5.5(d), appearance information and spatial context are mainly used to detect a perturbation.

5.5.3 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. 5.6(a) shows that the detection performance improves when more 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. 5.6(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. 5.6(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.

5.5.4 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 perturbed 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. 5.3. FeatureSqueeze [278] 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).

5.6 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

Method	Digital Perturbation			Physical Perturbation		
	Miscateg	Hiding	Appearing	Miscateg	Hiding	Appearing
Results on PASCAL VOC:						
FeatureSqueeze [278]	0.724	0.620	0.597	0.779	0.661	0.653
Co-occurGraph [14]	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 [278]	0.681	0.682	0.578	0.699	0.687	0.540
Co-occurGraph [14]	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 5.2: The detection performance (ROC-AUC) against six different attacks on PASCAL VOC and MS COCO dataset

False Positive Rate	0.1%	0.5%	1%	5%	10%
Recall of FeatureSqueeze [278]	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 5.3: Recall for detecting perturbed stop signs at different false positive rate.

5.7 Supplementary Material

In this supplementary material, we provide: 1) numbers used for the plots in this chapter; 2) the architecture of the auto-encoders; 3) how we extend the state-of-the-art adversarial perturbation detection method FeatureSqueeze to defend object detection system; 4) how we apply non-deep Co-occurGraph to defend object detection system using cooccurrence relations inside the scene images; 5) the detection performance of our proposed method against digital perturbations generated by various generation mechanisms; 6) comparing our proposed method with others that use context inconsistency to detect adversarial perturbations.

5.7.1 Values in the Plots

In the chapter, some experimental results have been provided as plots for better visualization. We provide a table for each plot in this supplementary material. Tab. 5.4 and Tab. 5.5 correspond to the upper part and the lower part of Fig. 5.6(a). Tab. 5.6 corresponds to Fig. 5.6(b). Tab. 5.7 and Tab. 5.8 correspond to the upper part and lower part of Fig. 5.6(c). Some entries are missing due to inadequate number of samples. For example, there are no entries for digital hiding attack for images with 6 objects in Tab. 5.7 because there are only 14 hiding-attacked images and the AUC reported would not be accurate. We report AUC when we have at least 50 attacked samples.

#Proposals	Digital Perturbations			Physical Perturbations		
	Miscategorization	Hiding	Appearing	Miscategorization	Hiding	Appearing
FeatureSqueeze [278]:						
1-3	0.751	0.628	0.587	0.762	0.679	0.647
3-6	0.712	0.626	0.614	0.749	0.633	0.653
6-9	0.748	0.612	0.609	0.784	0.654	0.688
9-12	0.727	0.576	0.629	0.767	0.672	0.692
Our method:						
1-3	0.830	0.977	0.843	0.940	0.955	0.950
3-6	0.902	0.995	0.859	0.983	0.982	0.977
6-9	0.933	0.999	0.903	0.993	0.998	0.985
9-12	0.950	0.983	0.929	0.996	1.000	0.991

Table 5.4: The detection performance against different attacks w.r.t. the number of proposals on the perturbed objects in PASCAL VOC dataset.

5.7.2 Architecture of the Auto-encoders

For each category, we use a separate auto-encoder to learn the distribution of its context profile. The architecture of the auto-encoders is identical and is shown in Fig. 5.7. The input to the auto-encoder is the context profile $x = [r, \gamma_{u1}, \gamma_{u2}, \gamma_{r1}, \gamma_{r2}]$. We denote the height and width of the input as H and W. $W = 5$ since there are 5 feature vectors in x and H equals to the dimension of the RoI pooling feature. 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

#Proposals	Digital Attack			Physical Attack		
	Miscategorization	Hiding	Appearing	Miscategorization	Hiding	Appearing
FeatureSqueeze [278]:						
1-3	0.704	0.692	0.594	0.670	0.678	0.502
3-6	0.656	0.679	0.569	0.719	0.692	0.528
6-9	0.584	-	0.562	0.653	0.641	0.552
9-12	0.616	-	0.521	0.682	-	0.556
Our method:						
1-3	0.896	0.961	0.804	0.918	0.938	0.952
3-6	0.947	0.992	0.876	0.985	0.982	0.973
6-9	0.978	-	0.90	0.983	0.999	0.989
9-12	0.988	-	0.932	0.995	-	0.987

Table 5.5: The detection performance against different attacks w.r.t. the number of proposals on the perturbed objects in MS COCO dataset.

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.

IoU	PASCAL VOC		MS COCO	
	Digital Appearing	Physical Appearing	Digital Appearing	Physical Appearing
FeatureSqueeze [278]:				
0.0	0.605	0.653	0.614	0.550
0.0-0.1	0.606	0.605	0.557	0.552
0.1-0.2	0.592	0.642	0.549	0.518
0.2-0.3	0.602	0.752	0.521	0.478
0.3-0.4	0.590	0.640	0.504	0.586
0.4-0.5	0.594	0.644	0.510	0.474
Our method:				
0.0	0.748	0.939	0.769	0.977
0.0-0.1	0.872	0.945	0.827	0.970
0.1-0.2	0.879	0.966	0.849	0.978
0.2-0.3	0.906	0.980	0.850	0.984
0.3-0.4	0.905	0.986	0.855	0.996
0.4-0.5	0.924	0.994	0.910	0.990

Table 5.6: The detection performance against appearing attacks w.r.t. the overlap (IoU) between the perturbed region and some ground truth object in PASCAL VOC and MS COCO

#Objects	Digital Perturbation			Physical Perturbations		
	Miscategorization	Hiding	Appearing	Miscategorization	Hiding	Appearing
FeatureSqueeze [278]:						
1	0.724	0.627	0.600	0.726	0.617	0.657
2	0.715	0.624	0.574	0.806	0.679	0.635
3	0.733	0.610	0.661	0.834	0.716	0.631
4	0.760	0.615	0.584	0.806	0.683	0.578
5	0.740	0.612	0.611	0.879	0.789	0.640
6	0.778	-	0.666	0.825	0.735	0.675
Our method:						
1	0.927	0.994	0.829	0.986	0.987	0.966
2	0.901	0.986	0.838	0.972	0.940	0.979
3	0.888	0.960	0.810	0.913	0.898	0.977
4	0.889	0.969	0.813	0.984	0.976	0.987
5	0.890	0.958	0.902	0.980	1.000	0.986
6	0.912	-	0.968	0.987	0.998	0.995

Table 5.7: The detection performance against different attacks w.r.t. the number of objects in the scene images in PASCAL VOC dataset.

5.7.3 Extending FeatureSqueeze to Region-level Perturbation Detection

FeatureSqueeze

FeatureSqueeze [278] proposes to squeeze the search space available to an adversary, driven by the observation that the feature input spaces are often unnecessarily large, which provides

#Object	Digital Attack			Physical Attack		
	Miscategorization	Hiding	Appearing	Miscategorization	Hiding	Appearing
FeatureSqueeze [278]:						
1	0.683	0.681	0.590	0.674	0.701	0.565
2	0.677	0.676	0.573	0.692	0.688	0.550
3	0.693	0.683	0.562	0.714	0.636	0.539
4	0.676	0.691	0.584	0.707	0.749	0.532
5	0.662	0.676	0.528	0.654	0.596	-
6	0.699	0.683	0.611	0.751	0.621	-
Our method:						
1	0.976	0.991	0.853	0.993	0.957	0.984
2	0.964	0.987	0.824	0.984	0.967	0.975
3	0.922	0.972	0.884	0.982	0.967	0.967
4	0.891	0.938	0.882	0.986	0.984	0.936
5	0.952	0.963	0.903	0.995	0.992	0.995
6	0.965	0.983	0.909	0.991	0.994	0.997

Table 5.8: The detection performance against different attacks w.r.t. the number of objects in the scene images in COCO dataset.

extensive opportunities for an adversary to construct adversarial examples. There are two feature squeezing methods used in their implementation: a) reducing the color bit depth of each pixel; b) spatial smoothing. By comparing a DNN model's prediction on the original input with that on

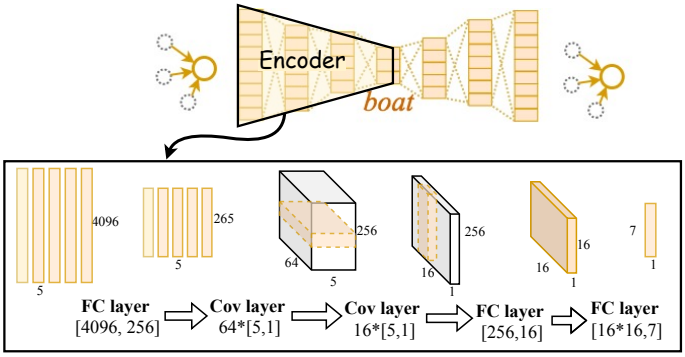


Figure 5.7: Auto-encoder structure. One auto-encoder is learned for each category. The structure of the auto-encoders is identical.

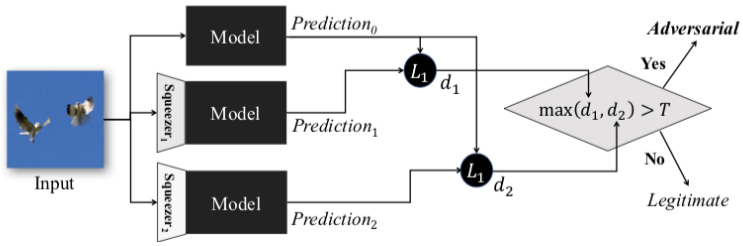


Figure 5.8: This figure is from paper [278]. "The model is evaluated on both the original input and the input after being pre-processed by feature squeezers. If the difference between the models prediction on a squeezed input and its prediction on the original input exceeds a threshold level, the input is identified to be adversarial."

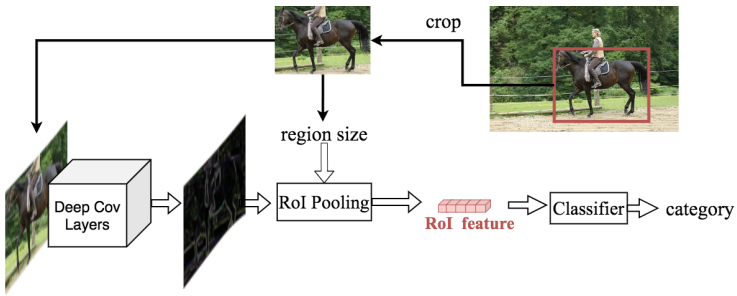


Figure 5.9: Extending the DNN of FeatureSqueeze to region-level classification

squeezed ones, feature squeezing detects adversarial examples with high accuracy and few false positives. The framework of FeatureSqueeze [278] is shown in Fig. 5.8.

Extending to Region-level Detection

To detect perturbed regions inside scene images, the DNN model of FeatureSqueeze is required to operate on region-level. We crop the ground-truth regions, denoted as r, as the input to the DNN model. The output of the DNN model is the predicted category. To deal with region inputs with various size, we use RoI pooling [82] (box size equals to input region size) as the last feature extraction layer as shown in Fig. 5.9. Softmax function [21] is used as the last layer and cross entropy loss [85] is used as the objective loss function.

Implementation Details

We initialize the feature extractor with the weights pretrained on ImageNet. Momentum optimizer with momentum 0.9 is used to train the classifier. The learning rate is 1e-4 and decays every 80k iterations at decay rate 0.1. Training ends after 240k iterations. The final classification accuracy for the 20 categories in PASCAL VOC dataset is 95.6%. The final classification accuracy for the 80 categories in MS COCO dataset is 87.1%. The accuracy is not high because MS COCO is biased among categories, for example, more than 100k person instances v.s. less than 1k hair dryer instances. Even after we balance the number of samples among different categories, the performance is not good because some categories have too few examples, like the hair dryer category. The hyperparameters used for feature squeezing are exactly the same as the authors' GitHub implementation [255].

5.7.4 Co-occurGraph for Misclassification Attack Detection

We consider a non-deep model as baseline where co-occurrence statistics are used to detect misclassification due to adversarial perturbation. This approach uses the inconsistency between prior relational information obtained from the training data and inferred relational information conditioned on misclassified detection to detect the presence of adversarial perturbation. As the co-occurrence statistics of background class cannot be modeled, this approach is not applicable for detecting hiding and appearing attacks.

Prior Relational Information. Same as [14], we use the co-occurrence frequency of different categories of objects in the training data to obtain the prior relational information. Co-occurrence statistics gives an estimate of how likely two object classes will appear together in an image.

Graphical Representation. To encode the relational information of different classes of objects present in an image, we represent each image as an undirected graph $G = (V, E)$. Here, a node in V represents a single proposed region by the region proposal network. The edges $E = \{(i, j)|$ if region v_i and v_j are linked$\}$ represent the relationships between the regions. We formulate a tree structure graph where the region of interest is connected with all other proposed regions. The estimate of class probabilities of each proposed region generated by the object detection model is used as the node potential and the co-occurrence statistics is used as the edge potential.

Detection of Misclassification Attack. For each image instance in test-set, we estimate its class conditional relatedness with other classes by making conditional inference on the representative graph. Conditional inference gives the pairwise conditional distribution of classes for each edge, which we use to obtain the posterior relational information of that image conditioned on the misclassified label. Based on the inconsistency among the prior relational information and posterior

relational information, we detect if there is any misclassification attack.

Implementation Details. We use the Faster R-CNN [216] as the object detection and region proposal generation module. For each image, we consider top 20 proposed regions based on the class confidence score. To formulate the graph and make conditional inference, we use the publicly available UGM Toolbox [220].

5.7.5 Detection performance w.r.t. various perturbation generation mechanisms

Previously, we show our proposed method is effective in detecting six different perturbation attacks, i.e., digital miscategorization attack, digital hiding attack, digital appearing attack, physical miscategorization attack, physical hiding attack and physical appearing attack. These attacks are different in terms of their attack goals and perturbation forms. Other defense papers also evaluate their defense methods w.r.t different perturbation generation mechanisms. Our defense strategy is dependent on the contextual information, and therefore should not rely heavily on the mechanism to generate the perturbation. We validate our hypothesis by testing our method against different perturbation generation mechanisms. The results in Tab. 5.9 show that our method is consistently effective against all the perturbation generation mechanisms.

As stated before, COCO has few examples for certain categories. To make sure we have enough number of context profiles to learn the distribution, out of all the 80 categories, we choose 10 categories with the largest number of context profiles extracted. These 10 categories are “car”, “diningtable”, “chair”, “bowl”, “giraffe”, “person”, “zebra”, “elephant”, “cow”, “cat”. We also choose “stop sign” category because attacks on stop signs have gained long-lasting attentions. In addition to “background”, we have in total 12 categories and learn 12 autoencoders separately. We

Perturbation Generation Mechanism	PASCAL VOC	MS COCO
FeatureSqueeze [278]:		
FGSM [88]	0.788	0.678
BIM [137]	0.724	0.681
Our method:		
FGSM	0.947	0.915
BIM	0.938	0.959

Table 5.9: The detection performance against digital miscategorization attacks w.r.t. different perturbation generation mechanisms on PASCAL VOC and MS COCO

use these 12 autoencoders and evaluate misclassifications to these categories in our experiments.

5.7.6 Comparison with other context inconsistency based adversarial defense methods

The general notion of using context has been used to detect anomalous activities [28, 95, 277, 301]. When it comes to adversarial perturbation detection, spatial context has been used to detect adversarial perturbations against semantic segmentation [271]. Temporal context has been used to detect adversarial perturbation against video classification [120]. Context inconsistency has never been used to detect adversarial examples against objection detection systems. Essentially, our approach utilizes different kinds of context, including the spatial one from these prior works and object-level inter-relationships for the first time, as discussed in Tab. 5.10.

Detection	Temporal	Spatial	Object-object	Object-background	Object-scene	Task
Video [120]	✓					video classification
Seg [271]		✓				semantic segmentation
Our method		✓	✓	✓	✓	object detection

Table 5.10: Comparison with other context inconsistency based adversarial detection methods

Chapter 6

Conclusions

The dissertation presents several novel methods in the direction of adversarial attacks on deep learning in computer vision. Despite the high accuracies of deep neural networks on a wide variety of computer vision tasks, their vulnerability to subtle adversarial perturbations under various attack settings is revealed by the proposed adversarial attack methods. It is demonstrated that currently deep learning can not only be effectively attacked in the white-box setting but also in the black-box setting. From another point of view, adversarial examples are hard to defend against because they require machine learning models to produce good outputs for every possible input. The dissertation makes contributions towards this by proposing an effective adversarial defense strategy where context information in the natural images is learnt and the context violations triggered by adversarial attacks are detected with high detection rates. With deep learning at the heart of the current advances in machine learning and artificial intelligence, this dissertation sheds lights on devising adversarial attacks and their defenses for deep learning.

Bibliography

[1] Apple homekit. https://www.apple.com/ios/home/.

[2] Iot proliferation, the biggest blind spot for companies. https://hotforsecurity.bitdefender.com/blog/iot-proliferation-the-biggest-blind-spot-for-companies-14001.html.

[3] Openface: Free and open source face recognition with deep neural networks. https://cmusatyalab.github.io/openface/.

[4] Samsung smart things. https://www.smartthings.com/.

[5] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016.

[6] Erin L Allwein, Robert E Schapire, and Yoram Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of machine learning research*, 1(Dec):113–141, 2000.

[7] André Anjos and Sébastien Marcel. Counter-measures to photo attacks in face recognition: a public database and a baseline. In *2011 IEEE international joint conference on Biometrics (IJCB)*, pages 1–7, 2011.

[8] Marko Arsenovic, Srdjan Sladojevic, Andras Anderla, and Darko Stefanovic. Facetime–deep learning based face recognition attendance system. In *2017 IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY)*, pages 000053–000058. IEEE, 2017.

[9] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

[10] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. *arXiv preprint arXiv:1707.07397*, 2017.

[11] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[12] Tegala Balaprasad and RVV Krishna. Face recognition based security system using sift algorithm. *IJSEAT*, 3(11):969–973, 2015.

[13] Shumeet Baluja and Ian Fischer. Adversarial transformation networks: Learning to generate adversarial examples. *arXiv preprint arXiv:1703.09387*, 2017.

[14] Jawadul H Bappy, Sujoy Paul, and Amit K Roy-Chowdhury. Online adaptation for joint scene and object classification. In *European Conference on Computer Vision*, pages 227–243. Springer, 2016.

[15] Ehud Barnea and Ohad Ben-Shahar. Exploring the bounds of the utility of context for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7412–7420, 2019.

[16] Arjun Nitin Bhagoji, Daniel Cullina, and Prateek Mittal. Dimensionality reduction as a defense against evasion attacks on machine learning classifiers. *arXiv preprint arXiv:1704.02654*, 2, 2017.

[17] Arjun Nitin Bhagoji, Warren He, Bo Li, and Dawn Song. Practical black-box attacks on deep neural networks using efficient query mechanisms. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 154–169, 2018.

[18] 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.

[19] Battista Biggio, Giorgio Fumera, and Fabio Roli. Pattern recognition systems under attack: Design issues and research challenges. *International Journal of Pattern Recognition and Artificial Intelligence*, 28(07):1460002, 2014.

[20] David SC Biggs. 3d deconvolution microscopy. *Current Protocols in Cytometry*, pages 12–19, 2010.

[21] Christopher M Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[22] Georgescu et.al. Bogdan. Method and system for anatomical object detection using marginal space deep neural networks, June 15 2017. US Patent 9,730,643.

[23] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[24] Matthew R Boutell, Jiebo Luo, Xipeng Shen, and Christopher M Brown. Learning multi-label scene classification. *Pattern recognition*, 37(9):1757–1771, 2004.

[25] Oliver Brdiczka, James L Crowley, and Patrick Reignier. Learning situation models in a smart home. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(1):56–63, 2009.

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

[27] Ignas Budvytis, Patrick Sauer, Thomas Roddick, Kesar Breen, and Roberto Cipolla. Large scale labelled video data augmentation for semantic segmentation in driving scenarios. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 230–237, 2017.

[28] Nan Cao, Chaoguang Lin, Qiuhan Zhu, Yu-Ru Lin, Xian Teng, and Xidao Wen. Voila: Visual anomaly detection and monitoring with streaming spatiotemporal data. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):23–33, 2017.

[29] Nicholas Carlini, Pratyush Mishra, Tavish Vaidya, Yuankai Zhang, Micah Sherr, Clay Shields, David Wagner, and Wenchao Zhou. Hidden voice commands. In *USENIX Security Symposium*, pages 513–530, 2016.

[30] Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 3–14, 2017.

[31] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57. IEEE, 2017.

[32] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4724–4733. IEEE, 2017.

[33] Lingwei Chen, Yanfang Ye, and Thirimachos Bourlai. Adversarial machine learning in malware detection: Arms race between evasion attack and defense. In *2017 European Intelligence and Security Informatics Conference (EISIC)*, pages 99–106. IEEE, 2017.

[34] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM workshop on artificial intelligence and security*, pages 15–26, 2017.

[35] Shang-Fu Chen, Yi-Chen Chen, Chih-Kuan Yeh, and Yu-Chiang Wang. Order-free rnn with visual attention for multi-label classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[36] Shang-Tse Chen, Cory Cornelius, Jason Martin, and Duen Horng Polo Chau. Shapeshifter: Robust physical adversarial attack on faster r-cnn object detector. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 52–68. Springer, 2018.

[37] Xinlei Chen and Abhinav Gupta. Spatial memory for context reasoning in object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4086–4096, 2017.

[38] Zhikai Chen, Lingxi Xie, Shanmin Pang, Yong He, and Qi Tian. Appending adversarial frames for universal video attack. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3199–3208, 2021.

[39] Minhao Cheng, Thong Le, Pin-Yu Chen, Jinfeng Yi, Huan Zhang, and Cho-Jui Hsieh. Query-efficient hard-label black-box attack: An optimization-based approach. *arXiv preprint arXiv:1807.04457*, 2018.

[40] Ivana Chingovska, André Anjos, and Sébastien Marcel. On the effectiveness of local binary patterns in face anti-spoofing. In *2012 BIOSIG-Proceedings of the International Conference of the Biometrics Special Interest Group (BIOSIG)*, pages 1–7. IEEE, 2012.

[41] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[42] Jae Young Choi, Yong Man Ro, and Konstantinos N Plataniotis. Color face recognition for degraded face images. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(5):1217–1230, 2009.

[43] Myung Jin Choi, Antonio Torralba, and Alan S Willsky. A tree-based context model for object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(2):240–252, 2011.

[44] Myung Jin Choi, Antonio Torralba, and Alan S Willsky. Context models and out-of-context objects. *Pattern Recognition Letters*, 33(7):853–862, 2012.

[45] Wongun Choi, Yuanqing Lin, Yu Xiang, and Silvio Savarese. Subcategory-aware convolutional neural networks for object detection, May 8 2018. US Patent 9,965,719.

[46] Moustapha Cisse, Yossi Adi, Natalia Neverova, and Joseph Keshet. Houdini: Fooling deep structured prediction models. *arXiv preprint arXiv:1707.05373*, 2017.

[47] Cmglee. 2d affine transformation matrix.

[48] Annalisa Cocchia. Smart and digital city: A systematic literature review. In *Smart city*, pages 13–43. Springer, 2014.

[49] MMAction2 Contributors. Openmmlab's next generation video understanding toolbox and benchmark. https://github.com/open-mmlab/mmaction2, 2020.

[50] Andrei Costin. Security of cctv and video surveillance systems: threats, vulnerabilities, attacks, and mitigations. In *Proceedings of the 6th International Workshop on Trustworthy Embedded Devices*, pages 45–54. ACM, 2016.

[51] Angel Cruz-Roa, Hannah Gilmore, Ajay Basavanhally, Michael Feldman, Shridar Ganesan, Natalie NC Shih, John Tomaszewski, Fabio A González, and Anant Madabhushi. Accurate and reproducible invasive breast cancer detection in whole-slide images: A deep learning approach for quantifying tumor extent. *Scientific reports*, 7(1):1–14, 2017.

[52] Ali Dabouei, Sobhan Soleymani, Jeremy Dawson, and Nasser Nasrabadi. Fast geometrically-perturbed adversarial faces. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1979–1988. IEEE, 2019.

[53] Tao Dai, Yan Feng, Dongxian Wu, Bin Chen, Jian Lu, Yong Jiang, and Shu-Tao Xia. Dipdefend: Deep image prior driven defense against adversarial examples. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 1404–1412, 2020.

[54] Nilaksh Das, Madhuri Shanbhogue, Shang-Tse Chen, Fred Hohman, Siwei Li, Li Chen, Michael E Kounavis, and Duen Horng Chau. Shield: Fast, practical defense and vaccination for deep learning using jpeg compression. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 196–204, 2018.

[55] Jester Dataset. Humans performing pre-defined hand actions. https://20bn.com/datasets/jester, 2016. [Online; accessed 30-April-2018].

[56] Debayan Deb, Jianbang Zhang, and Anil K Jain. Advfaces: Adversarial face synthesis. *arXiv preprint arXiv:1908.05008*, 2019.

[57] 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.

[58] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[59] Guneet S Dhillon, Kamyar Azizzadenesheli, Zachary C Lipton, Jeremy Bernstein, Jean Kossaifi, Aran Khanna, and Anima Anandkumar. Stochastic activation pruning for robust adversarial defense. *arXiv preprint arXiv:1803.01442*, 2018.

[60] Changxing Ding and Dacheng Tao. A comprehensive survey on pose-invariant face recognition. *ACM Transactions on intelligent systems and technology (TIST)*, 7(3):1–42, 2016.

[61] Santosh K Divvala, Derek Hoiem, James H Hays, Alexei A Efros, and Martial Hebert. An empirical study of context in object detection. In *2009 IEEE Conference on computer vision and Pattern Recognition*, pages 1271–1278. IEEE, 2009.

[62] Xiaoyi Dong, Jiangfan Han, Dongdong Chen, Jiayang Liu, Huanyu Bian, Zehua Ma, Hongsheng Li, Xiaogang Wang, Weiming Zhang, and Nenghai Yu. Robust superpixel-guided attentional adversarial attack. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12895–12904, 2020.

[63] 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.

[64] Xia Du and Chi-Man Pun. Adversarial image attacks using multi-sample and most-likely ensemble methods. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 1634–1642, 2020.

[65] Nguyen Minh Duc and Bui Quang Minh. Your face is not your password face authentication bypassing lenovo–asus–toshiba. *Black Hat Briefings*, 2009.

[66] Nikita Dvornik, Julien Mairal, and Cordelia Schmid. Modeling visual context is key to augmenting object detection datasets. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 364–380, 2018.

[67] Nesli Erdogmus and Sébastien Marcel. Spoofing face recognition with 3d masks. *IEEE Transactions on Information Forensics and Security*, 9(7):1084–1097, 2014.

[68] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[69] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1625–1634, 2018.

[70] Sean Ryan Fanello, Ilaria Gori, Giorgio Metta, and Francesca Odone. One-shot learning for real-time action recognition. In *Iberian Conference on Pattern Recognition and Image Analysis*, pages 31–40. Springer, 2013.

[71] Marcos Faúndez-Zanuy. Are inkless fingerprint sensors suitable for mobile use? *IEEE Aerospace and Electronic Systems Magazine*, 19(4):17–21, 2004.

[72] Marcos Faundez-Zanuy. Privacy issues on biometric systems. *IEEE Aerospace and Electronic Systems Magazine*, 20(2):13–15, 2005.

[73] Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Analysis of Classifiers Robustness to Adversarial Perturbations. *Machine Learning*, pages 481–508, 2018.

[74] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Robustness of classifiers: from adversarial to random noise. *arXiv preprint arXiv:1608.08967*, 2016.

[75] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6202–6211, 2019.

[76] Reuben Feinman, Ryan R Curtin, Saurabh Shintre, and Andrew B Gardner. Detecting adversarial samples from artifacts. *arXiv preprint arXiv:1703.00410*, 2017.

[77] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2009.

[78] Wang Feng, Jiyan Zhou, Chen Dan, Zhou Peiyan, and Zhang Li. Research on mobile commerce payment management based on the face biometric authentication. *International Journal of Mobile Communications*, 15(3):278–305, 2017.

[79] Rainhard D Findling and Rene Mayrhofer. Towards face unlock: on the difficulty of reliably detecting faces on mobile phones. In *Proceedings of the 10th International Conference on Advances in Mobile Computing & Multimedia*, pages 275–280. ACM, 2012.

[80] Homa Foroughi, Baharak Shakeri Aski, and Hamidreza Pourreza. Intelligent video surveillance for monitoring fall detection of elderly in home environments. In *Computer and Information Technology, 2008. ICCIT 2008. 11th International Conference on*, pages 219–224. IEEE, 2008.

[81] Amirhosein Ghaffarianhoseini, Umberto Berardi, Husam AlWaer, Seongju Chang, Edward Halawa, Ali Ghaffarianhoseini, and Derek Clements-Croome. What is an intelligent building? analysis of recent interpretations from an international perspective. *Architectural Science Review*, 59(5):338–357, 2016.

[82] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015.

[83] Akhil Goel, Anirudh Singh, Akshay Agarwal, Mayank Vatsa, and Richa Singh. Smartbox: Benchmarking adversarial detection and mitigation algorithms for face recognition. In *2018 IEEE 9th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, pages 1–7. IEEE, 2018.

[84] Zhitao Gong, Wenlu Wang, and Wei-Shinn Ku. Adversarial and clean data are not twins. *arXiv preprint arXiv:1704.04960*, 2017.

[85] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[86] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

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

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

[89] Gaurav Goswami, Nalini Ratha, Akshay Agarwal, Richa Singh, and Mayank Vatsa. Unravelling robustness of deep learning based face recognition against adversarial attacks. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[90] Kathrin Grosse, Nicolas Papernot, Praveen Manoharan, Michael Backes, and Patrick McDaniel. Adversarial perturbations against deep neural networks for malware classification. *arXiv preprint arXiv:1606.04435*, 2016.

[91] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, et al. Recent advances in convolutional neural networks. *Pattern Recognition*, 2017.

[92] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens Van Der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017.

[93] Jiangfan Han, Xiaoyi Dong, Ruimao Zhang, Dongdong Chen, Weiming Zhang, Nenghai Yu, Ping Luo, and Xiaogang Wang. Once a MAN: Towards Multi-Target Attack via Learning Multi-Target Adversarial Network Once. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5158–5167, 2019.

[94] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[95] Mahmudul Hasan, Jonghyun Choi, Jan Neumann, Amit K Roy-Chowdhury, and Larry S Davis. Learning temporal regularity in video sequences. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 733–742. IEEE, 2016.

[96] Mahmudul Hasan and Amit K Roy-Chowdhury. Context aware active learning of activity recognition models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4543–4551, 2015.

[97] 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.

[98] Zhezhi He, Adnan Siraj Rakin, and Deliang Fan. Parametric noise injection: Trainable randomness to improve deep neural network robustness against adversarial attack. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 588–597, 2019.

[99] Dan Hendrycks and Kevin Gimpel. Early methods for detecting adversarial images. *arXiv preprint arXiv:1608.00530*, 2016.

[100] Samitha Herath, Mehrtash Harandi, and Fatih Porikli. Going deeper into action recognition: A survey. *Image and vision computing*, 60:4–21, 2017.

[101] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[102] Tobias Hinz, Matthew Fisher, Oliver Wang, and Stefan Wermter. Improved techniques for training single-image gans. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1300–1309, 2021.

[103] Andrew Hollingworth. Does consistent scene context facilitate object perception? *Journal of Experimental Psychology: General*, 127(4):398, 1998.

[104] Hossein Hosseini, Baicen Xiao, Andrew Clark, and Radha Poovendran. Attacking automatic video analysis algorithms: A case study of google cloud video intelligence api. In *Proceedings of the 2017 on Multimedia Privacy and Security*, pages 21–32. ACM, 2017.

[105] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3588–3597, 2018.

[106] Weiwei Hu and Ying Tan. Generating adversarial malware examples for black-box attacks based on gan. *arXiv preprint arXiv:1702.05983*, 2017.

[107] Chi-Hsuan Huang, Tsung-Han Lee, Lin-huang Chang, Jhih-Ren Lin, and Gwoboa Horng. Adversarial attacks on sdn-based deep learning ids system. In *International Conference on Mobile and Wireless Technology*, pages 181–191. Springer, 2018.

[108] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely Connected Convolutional Networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[109] Gary B Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, Technical Report 07-49, University of Massachusetts, Amherst, 2007.

[110] Ling Huang, Anthony D Joseph, Blaine Nelson, Benjamin IP Rubinstein, and JD Tygar. Adversarial machine learning. In *Proceedings of the 4th ACM workshop on Security and artificial intelligence*, pages 43–58. ACM, 2011.

[111] Qian Huang, Isay Katsman, Horace He, Zeqi Gu, Serge Belongie, and Ser-Nam Lim. Enhancing adversarial example transferability with an intermediate level attack. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4733–4742, 2019.

[112] Xun Huang, Yixuan Li, Omid Poursaeed, John Hopcroft, and Serge Belongie. Stacked generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, page 4, 2017.

[113] Peter J Huber. Robust estimation of a location parameter. In *Breakthroughs in Statistics*, pages 492–518. Springer, 1992.

[114] Ratnawati Ibrahim and Zalhan Mohd Zin. Study of automated face recognition system for office door access control application. In *2011 IEEE International Conference on Communication Software and Networks (ICCSN)*, pages 132–136. IEEE, 2011.

[115] 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.

[116] Andrew Ilyas, Logan Engstrom, and Aleksander Madry. Prior convictions: Black-box adversarial attacks with bandits and priors. *arXiv preprint arXiv:1807.07978*, 2018.

[117] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, pages 125–136, 2019.

[118] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[119] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231, 2013.

[120] Xiaojun Jia, Xingxing Wei, and Xiaochun Cao. Identifying and resisting adversarial videos using temporal consistency. *arXiv preprint arXiv:1909.04837*, 2019.

[121] Xiaojun Jia, Xingxing Wei, Xiaochun Cao, and Hassan Foroosh. Comdefend: An efficient image compression model to defend adversarial examples. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6084–6092, 2019.

[122] Xiaojun Jia, Xingxing Wei, Xiaochun Cao, and Xiaoguang Han. Adv-watermark: A novel watermark perturbation for adversarial examples. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 1579–1587, 2020.

[123] Zhaoyin et.al. Jia. Object detection neural networks, June 11 2019. US Patent 10,318,827.

[124] Linxi Jiang, Xingjun Ma, Shaoxiang Chen, James Bailey, and Yu-Gang Jiang. Black-box adversarial attacks on video recognition models. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 864–872, 2019.

[125] Di Jin, Shuyang Gao, Jiun-Yu Kao, Tagyoung Chung, and Dilek Hakkani-tur. Mmm: Multi-stage multi-task learning for multi-choice reading comprehension. *arXiv preprint arXiv:1910.00458*, 2019.

[126] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual Losses for Real-Time Style Transfer and Super-Resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016.

[127] Barry L Kalman and Stan C Kwasny. Why tanh: choosing a sigmoidal function. In *Neural Networks, 1992. IJCNN., International Joint Conference on*, volume 4, pages 578–581. IEEE, 1992.

[128] Behnam Karimi. *Comparative analysis of face recognition algorithms and investigation on the significance of color*. PhD thesis, Concordia University, 2006.

[129] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[130] Hirokatsu Kataoka, Yutaka Satoh, Yoshimitsu Aoki, Shoko Oikawa, and Yasuhiro Matsui. Temporal and fine-grained pedestrian action recognition on driving recorder database. *Sensors*, 18(2):627, 2018.

[131] Hirokatsu Kataoka, Teppei Suzuki, Shoko Oikawa, Yasuhiro Matsui, and Yutaka Satoh. Drive video analysis for the detection of traffic near-miss incidents. *arXiv preprint arXiv:1804.02555*, 2018.

[132] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[133] Alexander Klaser, Marcin Marszałek, and Cordelia Schmid. A spatio-temporal descriptor based on 3d-gradients. In *BMVC 2008-19th British Machine Vision Conference*, pages 275–1. British Machine Vision Association, 2008.

[134] Neslihan Kose and Jean-Luc Dugelay. On the vulnerability of face recognition systems to spoofing mask attacks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2357–2361. IEEE, 2013.

[135] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[136] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[137] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

[138] Alexey Kurakin, Ian Goodfellow, Samy Bengio, et al. Adversarial examples in the physical world, 2017.

[139] Kaspersky Lab. Man-in-the-middle attack on video surveillance systems. https://securelist.com/does-cctv-put-the-public-at-risk-of-cyberattack/70008/, Defcon,2014. [Online; accessed 30-April-2018].

[140] Quoc V Le, Will Y Zou, Serena Y Yeung, and Andrew Y Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3361–3368. IEEE, 2011.

[141] Jianan Li, Yunchao Wei, Xiaodan Liang, Jian Dong, Tingfa Xu, Jiashi Feng, and Shuicheng Yan. Attentive contexts for object detection. *IEEE Transactions on Multimedia*, 19(5):944–954, 2016.

[142] Jun Li, Shasha Li, Jiani Hu, and Weihong Deng. Adaptive lpq: An efficient descriptor for blurred face recognition. In *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, volume 1, pages 1–6. IEEE, 2015.

[143] Shasha Li and Weihong Deng. Face recognition based on random feature. In *2015 Visual Communications and Image Processing (VCIP)*, pages 1–4. IEEE, 2015.

[144] Shasha Li, Karim Khalil, Rameswar Panda, Chengyu Song, Srikanth V Krishnamurthy, Amit K Roy-Chowdhury, and Ananthram Swami. Measurement-driven security analysis of imperceptible impersonation attacks. *arXiv preprint arXiv:2008.11772*, 2020.

[145] Shasha Li, Ajaya Neupane, Sujoy Paul, Chengyu Song, Srikanth V Krishnamurthy, Amit K Roy Chowdhury, and Ananthram Swami. Adversarial perturbations against real-time video classification systems. *arXiv preprint arXiv:1807.00458*, 2018.

[146] Shasha Li, Ajaya Neupane, Sujoy Paul, Chengyu Song, Srikanth V Krishnamurthy, Amit K Roy-Chowdhury, and Ananthram Swami. Stealthy adversarial perturbations against real-time video classification systems. In *NDSS*, 2019.

[147] Shasha Li, Shitong Zhu, Sudipta Paul, Amit Roy-Chowdhury, Chengyu Song, Srikanth Krishnamurthy, Ananthram Swami, and Kevin S Chan. Connecting the Dots: Detecting Adversarial Perturbations Using Context Inconsistency. In *European Conference on Computer Vision*, pages 396–413. Springer, 2020.

[148] Xin Li and Fuxin Li. Adversarial examples detection in deep networks with convolutional filter statistics. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5764–5772, 2017.

[149] Yan Li, Ke Xu, Qiang Yan, Yingjiu Li, and Robert H Deng. Understanding osn-based facial disclosure against face authentication systems. In *Proceedings of the 9th ACM symposium on Information, computer and communications security*, pages 413–424. ACM, 2014.

[150] Yuezun Li, Daniel Tian, Ming-Ching Chang, Xiao Bian, and Siwei Lyu. Robust adversarial perturbation on deep proposal-based models. *arXiv preprint arXiv:1809.05962*, 2018.

[151] Bin Liang, Hongcheng Li, Miaoqiang Su, Xirong Li, Wenchang Shi, and Xiaofeng Wang. Detecting adversarial image examples in deep neural networks with adaptive noise reduction. *IEEE Transactions on Dependable and Secure Computing*, 2018.

[152] Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Xiaolin Hu, and Jun Zhu. Defense against adversarial attacks using high-level representation guided denoiser. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1778–1787, 2018.

[153] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2980–2988, 2017.

[154] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[155] Aishan Liu, Xianglong Liu, Jiaxin Fan, Yuqing Ma, Anlan Zhang, Huiyuan Xie, and Dacheng Tao. Perceptual-Sensitive GAN for Generating Adversarial Patches. In *Proceedings of the AAAI conference on artificial intelligence*, pages 1028–1035, 2019.

[156] Jiayang Liu, Weiming Zhang, Yiwei Zhang, Dongdong Hou, Yujia Liu, Hongyue Zha, and Nenghai Yu. Detection based defense against adversarial examples from the steganalysis point of view. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4825–4834, 2019.

[157] Jingtuo Liu, Yafeng Deng, Tao Bai, Zhengping Wei, and Chang Huang. Targeting ultimate accuracy: Face recognition via deep embedding. *arXiv preprint arXiv:1506.07310*, 2015.

[158] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European Conference on Computer Vision*, pages 21–37. Springer, 2016.

[159] Wenqing Liu, Miaojing Shi, Teddy Furon, and Li Li. Defending adversarial examples via dnn bottleneck reinforcement. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 1930–1938, 2020.

[160] Xin Liu, Huanrui Yang, Ziwei Liu, Linghao Song, Hai Li, and Yiran Chen. Dpatch: An adversarial patch attack on object detectors. *arXiv preprint arXiv:1806.02299*, 2018.

[161] Xinhao Liu, Masayuki Tanaka, and Masatoshi Okutomi. Single-image noise level estimation for blind denoising. *IEEE transactions on image processing*, 22(12):5226–5237, 2013.

[162] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *ICLR*, 2017.

[163] Yong Liu, Ruiping Wang, Shiguang Shan, and Xilin Chen. 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*, pages 6985–6994, 2018.

[164] Zihao Liu, Qi Liu, Tao Liu, Nuo Xu, Xue Lin, Yanzhi Wang, and Wujie Wen. Feature distillation: Dnn-oriented jpeg compression against adversarial examples. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 860–868. IEEE, 2019.

[165] Shao-Yuan Lo and Vishal M Patel. Multav: Multiplicative adversarial videos. *arXiv preprint arXiv:2009.08058*, 2020.

[166] Rushi Longadge and Snehalata Dongre. Class imbalance problem in data mining review. *arXiv preprint arXiv:1305.1707*, 2013.

[167] Jiajun Lu, Theerasit Issaranon, and David Forsyth. Safetynet: Detecting and rejecting adversarial examples robustly. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 446–454, 2017.

[168] Jiajun Lu, Hussein Sibai, Evan Fabry, and David Forsyth. No need to worry about adversarial examples in object detection in autonomous vehicles. *arXiv preprint arXiv:1707.03501*, 2017.

[169] Jonathan Lwowski, Prasanna Kolar, Patrick Benavidez, Paul Rad, John J Prevost, and Mo Jamshidi. Pedestrian detection system for smart communities using deep convolutional neural networks. In *2017 12th System of Systems Engineering Conference (SoSE)*, pages 1–6. IEEE, 2017.

[170] Chen Ma, Chenxu Zhao, Hailin Shi, Li Chen, Junhai Yong, and Dan Zeng. Metaadvdet: Towards robust detection of evolving adversarial attacks. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 692–701, 2019.

[171] Chengcheng Ma, Weiliang Meng, Baoyuan Wu, Shibiao Xu, and Xiaopeng Zhang. Efficient joint gradient based attack against sor defense for 3d point cloud classification. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 1819–1827, 2020.

[172] Pingchuan Ma, Stavros Petridis, and Maja Pantic. Detecting adversarial attacks on audio-visual speech recognition. *arXiv preprint arXiv:1912.08639*, 2019.

[173] 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.

[174] R Manjunatha and R Nagaraja. Home security system and door access control based on face recognition. *International Research Journal of Engineering and Technology (IRJET)*, 2017.

[175] Biometrics Market. Industry report 2009-2014. *International Biometric Group*, 2008.

[176] Joanna Materzynska, Guillaume Berger, Ingo Bax, and Roland Memisevic. The jester dataset: A large-scale video dataset of human gestures. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[177] Chris McCool, Tristan Perez, and Ben Upcroft. Mixtures of lightweight deep convolutional neural networks: Applied to agricultural robotics. *IEEE Robotics and Automation Letters*, 2(3):1344–1351, 2017.

[178] Michael McCoyd and David Wagner. Spoofing 2d face detection: Machines see people who aren't there. *arXiv preprint arXiv:1608.02128*, 2016.

[179] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. *arXiv preprint arXiv:1702.04267*, 2017.

[180] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. *arXiv preprint arXiv:1610.08401*, 2016.

[181] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 86–94. IEEE, 2017.

[182] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2574–2582, 2016.

[183] Konda Reddy Mopuri, Utsav Garg, and R Venkatesh Babu. Fast feature fool: A data independent approach to universal adversarial perturbations. *arXiv preprint arXiv:1707.05572*, 2017.

[184] Konda Reddy Mopuri, Utkarsh Ojha, Utsav Garg, and R Venkatesh Babu. Nag: Network for adversary generation. *arXiv preprint arXiv:1712.03390*, 2017.

[185] Konda Reddy Mopuri, Utkarsh Ojha, Utsav Garg, and R Venkatesh Babu. Nag: Network for adversary generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 742–751, 2018.

[186] Konda Reddy Mopuri, Phani Krishna Uppala, and R Venkatesh Babu. Ask, acquire, and attack: Data-free uap generation using class impressions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 19–34, 2018.

[187] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. 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*, pages 891–898, 2014.

[188] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[189] Krishna Kanth Nakka and Mathieu Salzmann. Indirect local attacks for context-aware semantic segmentation networks. In *European Conference on Computer Vision*, pages 611–628. Springer, 2020.

[190] Muzammal Naseer, Salman H Khan, Harris Khan, Fahad Shahbaz Khan, and Fatih Porikli. Cross-Domain Transferability of Adversarial Perturbations. *arXiv preprint arXiv:1905.11736*, 2019.

[191] ZD Net. Surveillance cameras sold on Amazon infected with malware. https://www.zdnet.com/article/amazon-surveillance-cameras-infected-with-malware/, ZD Net,2016. [Online; accessed 30-April-2018].

[192] 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.

[193] Jorge Nocedal. Updating quasi-newton matrices with limited storage. *Mathematics of computation*, 35(151):773–782, 1980.

[194] Aude Oliva, Antonio Torralba, Monica S Castelhano, and John M Henderson. Top-down control of visual attention in object detection. In *Proceedings 2003 International Conference on Image Processing (Cat. No. 03CH37429)*, volume 1, pages I–253. IEEE, 2003.

[195] Nicolas Papernot, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Fartash Faghri, Alexander Matyasko, Karen Hambardzumyan, Yi-Lin Juang, Alexey Kurakin, Ryan Sheatsley, et al. cleverhans v2. 0.0: an adversarial machine learning library. *arXiv preprint arXiv:1610.00768*, 2016.

[196] 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.

[197] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pages 506–519, 2017.

[198] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 372–387. IEEE, 2016.

[199] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 582–597. IEEE, 2016.

[200] Unsang Park, Yiying Tong, and Anil K Jain. Age-invariant face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(5):947–954, 2010.

[201] Omkar M Parkhi, Andrea Vedaldi, and Andrew Zisserman. Deep face recognition. In *BMVC*, volume 1, page 6, 2015.

[202] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, pages 8026–8037, 2019.

[203] Pawan Patidar, Manoj Gupta, Sumit Srivastava, and Ashok Kumar Nagawat. Image denoising by various filters for different noise. *International Journal of Computer Applications*, 9(4), 2010.

[204] Alex Pentland and Tanzeem Choudhury. Face recognition for smart environments. *Computer*, 33(2):50–55, 2000.

[205] Rainer Planinc, Alexandros Chaaraoui, Martin Kampel, and Francisco Flrez-Revuelta. Computer vision for active and assisted living. pages 57–79, 01 2016.

[206] Roi Pony, Itay Naeh, and Shie Mannor. Over-the-air adversarial flickering attacks against video recognition networks. *arXiv preprint arXiv:2002.05123*, 2020.

[207] Omid Poursaeed, Isay Katsman, Bicheng Gao, and Serge Belongie. Generative Adversarial Perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4422–4431, 2018.

[208] Aaditya Prakash, Nick Moran, Solomon Garber, Antonella DiLillo, and James Storer. Deflecting adversarial attacks with pixel deflection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8571–8580, 2018.

[209] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[210] Edward Raff, Jared Sylvester, Steven Forsyth, and Mark McLean. Barrage of random transforms for adversarially robust defense. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6528–6537, 2019.

[211] Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. Classifier chains for multi-label classification. *Machine learning*, page 333, 2011.

[212] Sylvestre-Alvise Rebuffi, Ruth Fong, Xu Ji, and Andrea Vedaldi. There and back again: Revisiting backpropagation saliency methods. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8839–8848, 2020.

[213] Konda Reddy Mopuri, Utkarsh Ojha, Utsav Garg, and R Venkatesh Babu. Nag: Network for adversary generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 742–751, 2018.

[214] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 779–788, 2016.

[215] Hongyu Ren, Shengjia Zhao, and Stefano Ermon. Adaptive antithetic sampling for variance reduction. In *International Conference on Machine Learning*, pages 5420–5428. PMLR, 2019.

[216] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2015.

[217] Mrutyunjaya Sahani, Chiranjiv Nanda, Abhijeet Kumar Sahu, and Biswajeet Pattnaik. Web-based online embedded door access control and home security system based on face recognition. In *Circuit, Power and Computing Technologies (ICCPCT), 2015 International Conference on*, pages 1–6. IEEE, 2015.

[218] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *arXiv preprint arXiv:1606.03498*, 2016.

[219] Raphael Satter. U.s. court: Mass surveillance program exposed by snowden was illegal. *Reuters*, Sep 2020.

[220] Mark Schmidt. UGM: Matlab code for undirected graphical models.

[221] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015.

[222] Paul Scovanner, Saad Ali, and Mubarak Shah. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th ACM international conference on Multimedia*, pages 357–360. ACM, 2007.

[223] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[224] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael 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*, pages 1528–1540. ACM, 2016.

[225] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Adversarial generative nets: Neural network attacks on state-of-the-art face recognition. *arXiv preprint arXiv:1801.00349*, 2017.

[226] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. A general framework for adversarial examples with objectives. *ACM Transactions on Privacy and Security (TOPS)*, 22(3):1–30, 2019.

[227] Peichung Shih and Chengjun Liu. Comparative assessment of content-based face image retrieval in different color spaces. *International Journal of Pattern Recognition and Artificial Intelligence*, 19(07):873–893, 2005.

[228] Eugene Shkolyar, Xiao Jia, Timothy C Chang, Dharati Trivedi, Kathleen E Mach, Max Q-H Meng, Lei Xing, and Joseph C Liao. Augmented bladder tumor detection using deep learning. *European urology*, 76(6):714–718, 2019.

[229] Reza Shokri, Marco Stronati, and Vitaly Shmatikov. Membership inference attacks against machine learning models. *arXiv preprint arXiv:1610.05820*, 2016.

[230] Ramesh Simhambhatla, Kevin Okiah, Shravan Kuchkula, and Robert Slater. Self-driving cars: Evaluation of deep learning techniques for object detection in different driving conditions. *SMU Data Science Review*, 2(1):23, 2019.

[231] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014.

[232] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[233] Krishna Kumar Singh, Dhruv Mahajan, Kristen Grauman, Yong Jae Lee, Matt Feiszli, and Deepti Ghadiyaram. Don't judge an object by its context: Learning to overcome contextual bias. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11070–11078, 2020.

[234] Lukas Smirek, Gottfried Zimmermann, and Michael Beigl. Just a smart home or your smart home–a framework for personalized user interfaces based on eclipse smart home and universal remote console. *Procedia Computer Science*, 98:107–116, 2016.

[235] 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.

[236] Qing Song, Yingqi Wu, and Lu Yang. Attacks on state-of-the-art face recognition using attentional adversarial attack generative network. *arXiv preprint arXiv:1811.12026*, 2018.

[237] Qingquan Song, Haifeng Jin, Xiao Huang, and Xia Hu. Multi-label adversarial perturbations. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 1242–1247. IEEE, 2018.

[238] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[239] Waqas Sultani, Chen Chen, and Mubarak Shah. Real-world anomaly detection in surveillance videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6479–6488, 2018.

[240] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deeply learned face representations are sparse, selective, and robust. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2892–2900, 2015.

[241] Mate Szarvas, Akira Yoshizawa, Munetaka Yamamoto, and Jun Ogata. Pedestrian detection with convolutional neural networks. In *IEEE Proceedings. Intelligent Vehicles Symposium, 2005.*, pages 224–229. IEEE, 2005.

[242] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[243] 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.

[244] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708, 2014.

[245] Kaihua Tang, Hanwang Zhang, Baoyuan Wu, Wenhan Luo, and Wei Liu. Learning to compose dynamic tree structures for visual contexts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6619–6628, 2019.

[246] Guanhong Tao, Shiqing Ma, Yingqi Liu, and Xiangyu Zhang. Attacks meet interpretability: Attribute-steered detection of adversarial samples. In *Advances in Neural Information Processing Systems*, pages 7717–7728, 2018.

[247] Graham W Taylor, Rob Fergus, Yann LeCun, and Christoph Bregler. Convolutional learning of spatio-temporal features. In *European conference on computer vision*, pages 140–153. Springer, 2010.

[248] C3D Tensorflow. C3D Implementation. https://github.com/hx173149/C3D-tensorflow.git, 2016. [Online; accessed 30-April-2018].

[249] 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.

[250] Antonio Torralba. Contextual priming for object detection. *International Journal of Computer Vision*, 53(2):169–191, 2003.

[251] Luis Torres, Jean-Yves Reutter, and Luis Lorente. The importance of the color information in face recognition. In *In 1999 International Conference on Image Processing (ICIP)*, volume 3, pages 627–631. IEEE, 1999.

[252] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

[253] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Computer Vision (ICCV), 2015 IEEE International Conference on*, pages 4489–4497. IEEE, 2015.

[254] Vikas Tripathi, Ankush Mittal, Durgaprasad Gangodkar, and Vishnu Kanth. Real time security framework for detecting abnormal events at atm installations. *Journal of Real-Time Image Processing*, pages 1–11, 2016.

[255] uvasrg. Featuresqueezing. https://github.com/uvasrg/FeatureSqueezing.git, 2018.

[256] Gul Varol, Ivan Laptev, and Cordelia Schmid. Long-term temporal convolutions for action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 2017.

[257] Umboo Computer Vision. Case Study: Elementary Scholl in Taiwai. https://news.umbocv.com/case-study-taiwan-elementary-school-13fa14cdb167.

[258] Umboo Computer Vision. Umbo Customer Case Study NCHU. https://news.umbocv.com/umbo-customer-case-study-nchu-687356292f43.

[259] Umboo Computer Vision. Umbo's Smart City Featured on CBS Sacramento. https://news.umbocv.com/umbos-smart-city-featured-on-cbs-sacramento-26f839415c51.

[260] Umboo Computer Vision. Case Studies. https://news.umbocv.com/case-studies/home, 2016. [Online; accessed 30-April-2018].

[261] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In *Advances In Neural Information Processing Systems*, pages 613–621, 2016.

[262] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 3551–3558. IEEE, 2013.

[263] Lina Wang, Kang Yang, Wenqi Wang, Run Wang, and Aoshuang Ye. Mgaattack: Toward more query-efficient black-box attack by microbial genetic algorithm. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 2229–2236, 2020.

[264] Wenguan Wang, Qiuxia Lai, Huazhu Fu, Jianbing Shen, Haibin Ling, and Ruigang Yang. Salient object detection in the deep learning era: An in-depth survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[265] Xingxing Wei, Jun Zhu, and Hang Su. Sparse adversarial perturbations for videos. *arXiv preprint arXiv:1803.02536*, 2018.

[266] Xingxing Wei, Jun Zhu, Sha Yuan, and Hang Su. Sparse adversarial perturbations for videos. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8973–8980, 2019.

[267] Zhipeng Wei, Jingjing Chen, Xingxing Wei, Linxi Jiang, Tat-Seng Chua, Fengfeng Zhou, and Yu-Gang Jiang. Heuristic black-box adversarial attacks on video recognition models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 12338–12345, 2020.

[268] Lin Wu, Yang Wang, Hongzhi Yin, Meng Wang, and Ling Shao. Few-shot deep adversarial learning for video-based person re-identification. *IEEE Transactions on Image Processing*, 29:1233–1245, 2019.

[269] Zuxuan Wu, Ser-Nam Lim, Larry S Davis, and Tom Goldstein. Making an invisibility cloak: Real world adversarial attacks on object detectors. In *European Conference on Computer Vision*, pages 1–17. Springer, 2020.

[270] Chaowei Xiao, Ruizhi Deng, Bo Li, Taesung Lee, Benjamin Edwards, Jinfeng Yi, Dawn Song, Mingyan Liu, and Ian Molloy. Advit: Adversarial frames identifier based on temporal consistency in videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3968–3977, 2019.

[271] Chaowei Xiao, Ruizhi Deng, Bo Li, Fisher Yu, Mingyan Liu, and Dawn Song. Characterizing adversarial examples based on spatial consistency information for semantic segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 217–234, 2018.

[272] Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating Adversarial Examples with Adversarial Networks. *arXiv preprint arXiv:1801.02610*, 2018.

[273] Chaowei Xiao, Jun Yan Zhu, Bo Li, Warren He, Mingyan Liu, and Dawn Song. Spatially transformed adversarial examples. In *6th International Conference on Learning Representations, ICLR 2018*, 2018.

[274] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. Mitigating adversarial effects through randomization. *arXiv preprint arXiv:1711.01991*, 2017.

[275] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Yuille. Adversarial examples for semantic segmentation and object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1369–1378, 2017.

[276] Jiangjian Xie, Jun Yang, Changqing Ding, and Wenbin Li. High accuracy individual identification model of crested ibis (nipponia nippon) based on autoencoder with self-attention. *IEEE Access*, 8:41062–41070, 2020.

[277] Dan Xu, Rui Song, Xinyu Wu, Nannan Li, Wei Feng, and Huihuan Qian. Video anomaly detection based on a hierarchical activity discovery within spatio-temporal contexts. *Neurocomputing*, 143:144–152, 2014.

[278] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017.

[279] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017.

[280] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing mitigates and detects carlini/wagner adversarial examples. *arXiv preprint arXiv:1705.10686*, 2017.

[281] Yi Xu, True Price, Jan-Michael Frahm, and Fabian Monrose. Virtual u: Defeating face liveness detection by building virtual models from your public photos. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 497–512. USENIX Association, 2016.

[282] Huanqian Yan, Xingxing Wei, and Bo Li. Sparse black-box video attack with reinforcement learning. *arXiv preprint arXiv:2001.03754*, 2020.

[283] Ceyuan Yang, Yinghao Xu, Jianping Shi, Bo Dai, and Bolei Zhou. Temporal pyramid network for action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 591–600, 2020.

[284] Jie-Ci Yang, Chin-Lun Lai, Hsin-Teng Sheu, and Jiann-Jone Chen. An intelligent automated door control system based on a smart camera. *Sensors*, 13(5):5923–5936, 2013.

[285] Zhiyu Yao, Yunbo Wang, Xingqiang Du, Mingsheng Long, and Jianmin Wang. Adversarial pyramid network for video domain generalization. *arXiv preprint arXiv:1912.03716*, 2019.

[286] Andrew Yip and Pawan Sinha. Role of color in face recognition. 2001.

[287] Andrew Yip and Pawan Sinha. Role of color in face recognition. *Journal of Vision*, 2(7):596–596, 2002.

[288] Xiurui Yuan and Shuguang Peng. A research on secure smart home based on the internet of things. In *Information Science and Technology (ICIST), 2012 International Conference on*, pages 737–740. IEEE, 2012.

[289] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[290] Hu Zhang, Linchao Zhu, Yi Zhu, and Yi Yang. Motion-excited sampler: Video adversarial attack with sparked prior. In *European Conference on Computer Vision*. Springer, 2020.

[291] Min-Ling Zhang and Zhi-Hua Zhou. Ml-knn: A lazy learning approach to multi-label learning. *Pattern recognition*, 40(7):2038–2048, 2007.

[292] Zhiwei Zhang, Junjie Yan, Sifei Liu, Zhen Lei, Dong Yi, and Stan Z Li. A face antispoofing database with diverse attacks. In *2012 5th IAPR international conference on Biometrics (ICB)*, pages 26–31. IEEE, 2012.

[293] Yue Zhao, Hong Zhu, Ruigang Liang, Qintao Shen, Shengzhi Zhang, and Kai Chen. 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*, pages 1989–2004, 2019.

[294] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30(11):3212–3232, 2019.

[295] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016.

[296] Nan Zhou, Wenjian Luo, Xin Lin, Peilan Xu, and Zhenya Zhang. Generating multi-label adversarial examples by linear programming. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

[297] Jiajun et.al. Zhu. Method and system for hierarchical human/crowd behavior detection, June 25 2020. US Patent 10,572,717.

[298] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *European Conference on Computer Vision*, pages 597–613. Springer, 2016.

[299] Shitong Zhu, Zhongjie Wang, Xun Chen, Shasha Li, Umar Iqbal, Zhiyun Qian, Kevin S Chan, Srikanth V Krishnamurthy, and Zubair Shafiq. A4: Evading learning-based adblockers. *arXiv preprint arXiv:2001.10999*, 2020.

[300] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9308–9316, 2019.

[301] Yingying Zhu, Nandita M Nayak, and Amit K Roy-Chowdhury. Context-aware activity recognition and anomaly detection in video. *IEEE Journal of Selected Topics in Signal Processing*, 7(1):91–101, 2012.

[302] Fei Zuo and PHN de With. Real-time embedded face recognition for smart home. *IEEE Transactions on Consumer Electronics*, 51(1):183–190, 2005.