Extracting Knowledge For Adversarial Detection and Defense in Deep Learning

Scott Freitas
Georgia Institute of Technology
Atlanta, Georgia
safreita@gatech.edu

Shang-Tse Chen
Georgia Institute of Technology
Atlanta, Georgia
schen351@gatech.edu

Duen Horng (Polo) Chau
Georgia Institute of Technology
Atlanta, Georgia
polo@gatech.edu

ABSTRACT

Deep learning models are being integrated into a wide range of high-impact, security-critical systems, from self-driving cars to biomedical diagnosis. However, recent research has demonstrated that many of these deep learning architectures are highly vulnerable to adversarial attacks—highlighting the vital need for defensive techniques to detect and mitigate these attacks before they occur. To combat these adversarial attacks, we developed UnMask, a knowledge-based adversarial detection and defense framework. The core idea behind UnMask is to protect these models by verifying that an image's predicted class ("bird") contains the expected building blocks (e.g., beak, wings, eyes). For example, if an image is classified as "bird", but the extracted building blocks are *wheel*, *seat* and *frame*, the model may be under attack. UnMask detects such attacks and defends the model by rectifying the misclassification, re-classifying the image based on its extracted building blocks. Our extensive evaluation shows that UnMask (1) *detects* up to 92.9% of attacks, with a false positive rate of 9.67% and (2) *defends* the model by correctly classifying up to 92.24% of adversarial images produced by the current strongest attack, Projected Gradient Descent, in the gray-box setting.

ACM Reference Format:
Scott Freitas, Shang-Tse Chen, and Duen Horng (Polo) Chau. 2019. Extracting Knowledge For Adversarial Detection and Defense in Deep Learning. In *Proceedings of KDD 2019 Workshop on Learning and Mining for Cybersecurity (LEMINCS'19) (LEMINCS @ KDD'19).* ACM, New York, NY, USA, 7 pages.

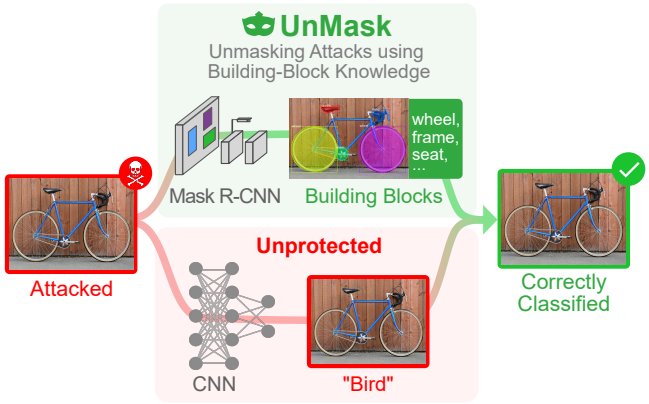

Figure 1: UnMask Framework Overview. UnMask combats adversarial attacks (in red) by extracting *building-block knowledge* (e.g., *wheel*) from the image (top, in green), and comparing them to expected features of the classification ("Bird" at bottom) from the unprotected model. Low feature overlap signals attack. UnMask rectifies misclassification using the image's extracted features. Our approach *detects* 92.9% of gray-box attacks (at 9.67% false positive rate) and *defends* the model by correctly classifying up to 92.24% of adversarial images crafted by the strongest attack, Projected Gradient Descent.

1 INTRODUCTION

In the past few years, deep neural networks (DNNs) have shown significant susceptibility to adversarial perturbation [11, 25]. More recently, a wide range of adversarial attacks have been developed to defeat deep learning systems [3, 6, 15, 19]—in some cases by changing the value of only a few pixels [24]. The ability of these micro perturbations to confuse deep learning architectures highlights a critical issue with modern computer vision systems—that these deep learning systems do not distinguish objects in ways that humans would [4, 14]. For example, when humans see a bicycle, we see its handlebar, frame, wheels, chains, and pedals (Fig. 1, top). Through our visual perception and cognition, we synthesize these detection results with our knowledge to determine that we are actually seeing a bicycle. However, when an image of a bicycle is adversarially perturbed to fool the model into misclassifying it as a bird (by manipulating pixels, as in Fig. 1, bottom), to humans, we still see the bicycle's **building-block features** (e.g., handlebar). On the other hand, the attacked model fails to perceive these building blocks, and is tricked into misclassifying the image. How do we incorporate this intuitive detection capability natural to human beings, into deep learning models to protect them from harm?

There has been a rich body of research studying detection and defense for deep learning, including adversarial training [16, 26], distillation [22] and image pre-processing [2, 8]. However, these approaches have not explicitly considered incorporating the extraction of building-block knowledge from images to protect deep learning models. Furthermore, research has shown that optimization based learning methods often fail to learn representations of objects that strongly align with humans' intuitive perception of those objects [11]. To fill this critical research gap in adversarial machine learning, we propose **UnMask** (Figure 1) — a novel method to protect deep learning models from adversarial perturbations through extracting building-block knowledge from images.

LEMINCS @ KDD'19, August 5th, 2019, Anchorage, Alaska, United States

Symbol	Definition
X	Training images
Y	Training classification labels
S	Training building-block segmentation masks
C	Set of possible classes
V	Class-feature matrix
x	Test image
$\hat{y}$	class prediction from model M
K	Building-block knowledge extraction model
M	Unprotected model
D	UnMask Defense framework
f_e	Extracted building-block features from image, by K
f_a	Expected features of image classified by M
$J(f_e, f_a)$	Jaccard similarity between f_e and f_a
s	similarity score
d	distance score (1-s)
t	Adversarial-benign classification threshold
z	Determination of adversarial or benign
p	Class prediction, by UnMask

Table 1: Symbols and Definition

1.1 Contributions

1. Building-Block Knowledge Extraction. We contribute the major idea that *building-block knowledge extraction* offers a powerful, explainable and practical method of detecting and defending against adversarial perturbations in deep learning models. *Building-block knowledge extraction* extracts higher-level information out of images—extending the core concept of *feature extraction* that is central to numerous successful data mining techniques.

2. UnMask: Detection & Defense Framework. Building on our core concept of building-block knowledge extraction, we propose UnMask as a framework to detect and defeat adversarial image perturbation by quantifying the similarity between the image's extracted features with the expected features of its predicted class. To the best of our knowledge, UnMask is the first framework that utilizes the concept of building-block knowledge extraction to combat adversarial perturbations.

3. Extensive Evaluation. We extensively evaluate UnMask's effectiveness using the large UnMaskDataset, with over 18k images in total. We test multiple factors, including: 3 attacks, including the strongest, *Projected Gradient Descent* (PGD) technique; 2 popular CNN architectures, VGG16 [23] and ResNet50 [13]; and multiple combinations of varying numbers of classes and feature overlap. Experiments demonstrate that our approach *detects* up to 92.9% of gray-box attacks with a false positive rate of 9.67% and (2) *defends* the model by correctly classifying up to 92.24% of adversarial images crafted by PGD. (Section 3)

2 UNMASK: DETECTION AND DEFENSE FRAMEWORK

In this section, we present our building-block knowledge extraction based approach to combating adversarial perturbations (Figure 1). The objective is to defend a **vulnerable deep learning model *M*** (Figure 1, bottom) using our **UnMask defense framework *D***, where the adversary has full access to M but is unaware of the defense strategy D, constituting a *gray-box* attack on the overall classification pipeline [8].

2.1 Intuition: Protection via Building-Block Knowledge Extraction

Our main idea to combat adversarial image perturbations with respect to an input image x, is to extract building-block features f_e using a **building-block knowledge extraction model** K, $f_e = K(x)$. These extracted building blocks, and their collective composition, forges a layer of protection around the model by disrupting the traditional pixel-centric attack [3, 19, 24]. Our building-block defensive layer forces the adversary to now solve a more complex problem of manipulating both the class label and all of the image's constituent parts. For example, in Figure 1 the attacker needs to fool the defensive layer into misclassifying the bike as a bird by, (1) changing the class label and (2) manipulating the bike building-block features (*wheel*, *seat*, *handlebar*) into bird features.

2.2 Overview of UnMask

Leveraging the concept of building-block knowledge extraction, we introduce UnMask as a detection and defense framework (D). Figure 1 summarizes how our method works at the high level. The adversary crafts an *attacked* image by carefully manipulating its pixel values using an adversarial technique (e.g., Projected Gradient Descent [19]). This attacked image then fools the *unprotected* model M (Figure 1, bottom) into misclassifying the image. To guard against this kind of attack on M, we use our UnMask framework D in conjunction with the *building-block knowledge extraction model* K (Figure 1, top). Model K processes the same image, which may be benign or attacked, and extracts the building-block features from the image to compare to the images' expected features. Comparing the set of *expected* features and the actual *extracted* features, UnMask determines the image was attacked, and predicts its class based on the extracted features.

2.3 Technical Walk-Through of UnMask

Now, we detail UnMask's technical operations and algorithm for detection and defense (Algorithm 1). Its major steps are:

1. Classify input. Given an input image x, UnMask obtains its class prediction $\hat{y}$ from (unprotected) model M, i.e., $\hat{y} = M(x)$. At this point, UnMask does not know if image x is adversarial or not.

2. Extract building-block features. UnMask extracts x's features f_e using *building-block knowledge extraction model* K, i.e., $f_e = K(x)$. Armed with these features f_e, UnMask can utilize them to both detect if model M is under attack, and to rectify misclassification caused by the attack. We considered multiple approaches for K, and decided to adopt Mask R-CNN for its ability to leverage image segmentation masks to learn and identify coherent image

regions that closely resemble building-blocks that would appear semantically and visually meaningful to humans [12]. Different from conventional image classification models or object detectors, the annotations used to train our building-block extractor K are *segmented* object parts instead of the whole objects. For example, for the *wheel* feature, an instance of training data would consist of a bike image and a *segmentation mask* indicating which region of that image represents a wheel. Technically, this means K uses only a part of an image, and not the whole image, for training. Furthermore, while an image may consist of multiple image parts, K treats them independently.

3. Detect attack. UnMask measures the similarity between the set of *extracted* features f_e and the set of *expected* features of $\hat{y}$ (obtained through matrix $V[\hat{y}]$), by calculating the Jaccard similarity score $s = J(f_e, f_a)$. If similarity score s is greater than the threshold parameter t, input image x is deemed benign, otherwise adversarial. Adjusting t would allow us to assess the trade-off between sensitivity and specificity, which we describe in detail in Section 3.

4. Defend and rectify. Determining an image to be adversarial also means that model M is under attack and is giving unreliable classification output. Thus, we need to rectify the misclassification. UnMask accomplishes this by comparing the extracted features f_e to every set of class features in V, outputting class $\hat{y}$ that contains the highest feature similarity s, $0 \leq s \leq 1$.

3 EVALUATION

We extensively evaluate UnMask's effectiveness in **defending** and **detecting** adversarial perturbations, using:

Algorithm 1: UnMask

Input: Training images X, labels Y, segmentation masks S, set of possible classes C, attribute matrix V, threshold t, test image x

Result: adversarial prediction $z \in \{0, 1\}$, predicted class p

Train unprotected classification model M***:***

M = NeuralNet(X, Y);

$\hat{y} = M(x)$;

Train building-block extraction model K***:***

K = Mask-RCNN(X, S);

$f_e = K(x)$; (extracted building blocks)

$f_a = V[\hat{y}]$; (expected building blocks)

Detection:

$s = J(f_e, f_a)$; $d = 1 - s$;

$$z = \begin{cases} 0 \text{ (benign)}, & \text{if } d < t \\ 1 \text{ (adversarial)}, & \text{if } d \geq t \end{cases}$$

Defense:

$$p = \begin{cases} \hat{y}, & \text{if } z = 0 \\ \underset{c \in C}{\text{argmin}}\, J(f_e, V[c]), & \text{if } z = 1 \end{cases}$$

return z, p;

Features	Airplane	Bicycle	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Dining Table	Dog	Horse	Motorbike	Person	Potted Plant	Sheep	Sofa	Train	Television
Arm															●					
Beak			●																	
Body	●				●															
Cap					●															
Coach																			●	
Door						●	●													
Engine	●																			
Ear								●		●		●	●		●		●			
Eye			●					●		●		●	●		●		●			
Eyebrow															●					
Foot			●												●					
Front side						●	●													
Hair															●					
Hand															●					
Head			●					●		●		●	●		●		●		●	
Headlight						●	●							●					●	
Hoof													●							
Horn										●							●			
Leg			●					●		●			●		●		●			
License plate						●	●													
Mirror						●	●													
Mouth															●					
Muzzle										●		●	●				●			
Neck			●					●		●		●	●		●		●			
Nose								●				●			●					
Paw								●				●								
Plant																●				
Pot																●				
Saddle		●												●						
Screen																				●
Stern	●																			
Tail	●		●					●		●		●	●				●			
Torso			●					●		●		●	●		●		●			
Vehicle						●	●													
Wheel	●	●				●	●							●						
Window						●	●													
Wing	●		●																	
Class Set (CS)																				
CS3a							●								●				●	
CS3b			●									●			●					
CS5a					●		●					●			●				●	
CS5b			●				●					●			●				●	

Table 2: Class-Feature Matrix. Top: dots mark classes' features. Bottom: four class sets with varying levels of feature overlap. Features *vehicle* and *coach* have sub-features not listed here due to space (see Appendix).

- 3 attacks, including the strongest technique, *Projected Gradient Descent* (PGD);
- 2 popular CNN architectures, VGG16 [23] and ResNet50 [13], as unprotected models M; and
- multiple combinations of varying numbers of classes and feature overlaps.

We begin by describing the experiment setup in Section 3.1. Then we present our results in Section 3.2. To the best of our knowledge, this work proposes the first building-block knowledge extraction to detect and defend against adversarial perturbation for deep learning. We present the first results in this new line of work.

3.1 Experiment Setup

3.1.1 ***Adversarial Attacks.*** We evaluated UnMask against three attacks, where we detail the parameter selection below:

- **DeepFool** (DF) L_2: a non-parametric attack that optimizes the amount of perturbation required to misclassify an image[21]; we set the update steps to 100.
- **Fast Gradient Sign** (FGSM): we set $\epsilon = 8, 16$—two common parameters for this attack [16].
- **Projected Gradient Descent** with Random Start (PGD): PGD is the current strongest first-order attack [19]. Its key parameter ϵ represents how much each pixel may be changed by PGD, e.g., $\epsilon = 4$ means changing up to 4 units of intensity (out of 255). It is common to evaluate up to a value of 16 [8, 16] (as perturbation becomes visible), with a stepsize of 0.01 and 40 iterations.

3.1.2 ***UnMaskDataset.*** We curated the UnMaskDataset for our evaluation, which consists of three component datasets—PASCAL-Part, PASCAL VOC 2010 and a subset of ImageNet—as seen in Tables 3 and 4. The impetus for our curation is to (i) collect all of the data used in our evaluation as a single source to promote ease of reproducibility, and (ii) to increase the number of images available for evaluating the performance of the deep learning models and the UnMask defense framework. We designed multiple *class sets* with varying number of classes and feature overlap (e.g., CS3a, in Table 2, bottom; and Table 5), to study how they would affect detection and defense effectiveness. We further discuss the utilization of the data in Sections 3.1.3 and 3.1.4 below.

3.1.3 ***Training Building-Block Model K.*** As illustrated in Section 2.3, the building-block knowledge extraction model K takes an image as input (e.g., bike) and outputs a set of building-block features (e.g., wheel,...). To train K, we use the PASCAL-Part dataset [7], which consists of 180,423 segmentation masks over 9,323 images across the 44 building-block features. The original dataset contains fine-grained features, such as 18 types of "legs" (e.g., right front lower leg, left back upper leg), while for our purposes we only need the abstraction of "leg". Therefore, we combined these fine-grained features into more generalized ones (rows in Table 2).

We followed a similar procedure described in [1], training K for 40 epochs. We use a ratio of 80/10/10 for training, validating and testing the model respectively (see Table 3). Our work is the first adaptation of Mask R-CNN model for the PASCAL-Part dataset. As such, there are no prior results for comparison. We computed model K's mAP (mean Average Precision), which estimates K's ability to extract features. The model attains an mAP of 0.56, in line with Mask R-CNN on other datasets [12]. Model K processes up to 4 images per second with a single Nvidia Titan X, matching the speeds reported in [1]. As building-block extraction is the most time-intensive process of the UnMask framework, its speed is representative of the overall speed of the framework.

Setup		PASCAL-Part			PASCAL VOC 2010		
Model	Classes	Train	Val	Test	Train	Val	Test
K	44	7,457	930	936	-	-	-
	CS3a	-	-	-	1,750	350	1,400
M	CS3b	-	-	-	2,104	421	1,684
	CS5a	-	-	-	2,264	452	1,812
	CS5b	-	-	-	2,501	500	2,001

Table 3: Number of images used in training models K and M.

	Defense			Detection
Class Set	DF	FGSM	PGD	All Attacks
CS3a	3,485	2,823	3,494	3,494
CS3b	4,749	4,161	4,764	4,764
CS5a	5,827	5,252	5,849	5,849
CS5b	6,728	5,883	6,747	6,747

Table 4: Number of ImageNet images used to evaluate *defense* and *detection* of UnMask. Only the images that can be successfully perturbed by the attack are used, thus the variations in numbers. We report values for PGD and FGSM with ϵ=16. The numbers for ϵ=8 are similar.

Class Set	Classes	Unique Parts	Overlap
CS3a	3	29	6.89%
CS3b	3	18	50.00%
CS5a	5	34	23.53%
CS5b	5	34	29.41%

Table 5: Four *class sets* investigated in our evaluation, with varying number of classes and feature overlap.

3.1.4 ***Training Unprotected Model M.*** As described in Section 2, M is the model under attack, and is what UnMask aims to protect. In practice, the choice of architecture for M and the data it is trained on are determined by the application model developers. Here, our evaluation studies two popular deep learning architectures — VGG16 [23] and ResNet50 [13] — however, UnMask supports other architectures. Training these models from scratch is generally computationally expensive and requires large amount of data. To reduce such need for computation and data, we adopt the approach described in [1], where we leverage a model pre-trained on ImageNet images, and *replace* its dense layers (i.e., the fully connected layers) to enable us to work with various class sets (e.g., CS3a) In detail, the training process for M is as follows:

(1) Load weights from model pre-trained on ImageNet data.
(2) Replace dense layers of the model with new dense layers, allowing us to specify a variable number of classes.

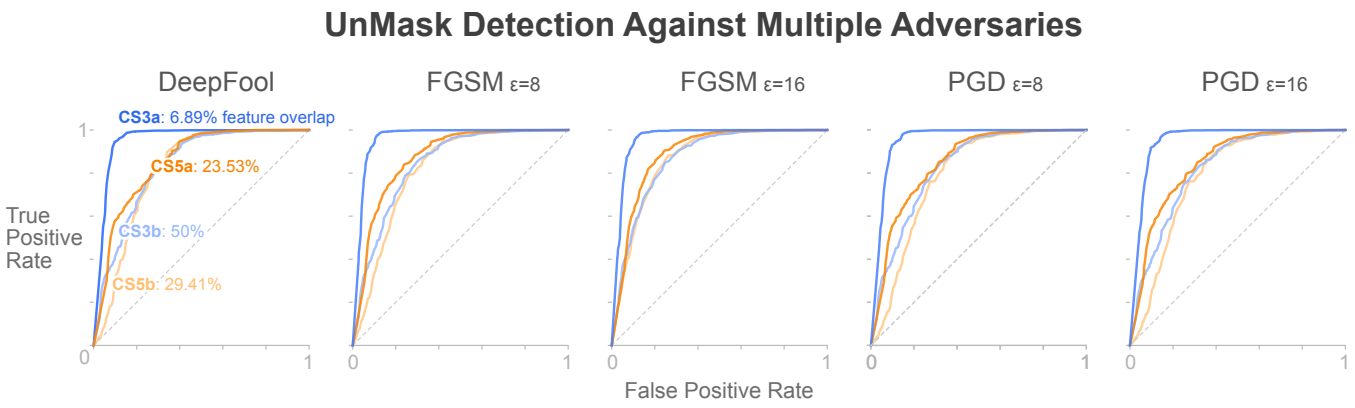

Figure 2: UnMask's effectiveness in detecting detecting three attacks: DeepFool, FGSM (ϵ=8,16), and the strongest, PGD (ϵ=8,16). UnMask's protection may not be affected strictly based on the number of classes. Rather, an important factor is the *feature overlap* among classes. UnMask provides better detection when there are 5 classes (dark orange; 23.53% overlap) than when there are 3 (light blue; 50% overlap). Keeping the number of classes constant and varying their feature overlap also supports our observation about the role of feature overlap (e.g., CS3a at 6.89% vs. CS3b at 50%). Dotted line indicates random guessing.

(3) Freeze all of the model weights except for the newly-added dense layers, allowing us to preserve the ImageNet features contained in the early layers while training the new dense layers on our data.

We chose to train the new dense layers using the PASCAL VOC 2010 dataset [9] for its desirable connection to the Pascal-Part dataset — PASCAL-Part uses the images from PASCAL VOC and adds *segmentation masks* for those images to describe image parts. Thus, we can readily ensure that the classes model M and model K are at parity. In practice, other datasets containing the classes from Table 2 may also be used. Refer to Table 3, for a breakdown of the data used for training and evaluation.

3.2 Evaluating UnMask Defense and Detection

The key research questions that our evaluation aims to address is how effective UnMask can (1) **detect** adversarial images, and (2) **defend** against attacks by rectifying misclassification through inferring the actual class label. Most image datasets containing the classes from Table 2 may be used. However, we use ImageNet data (see Table 4) as it matches our class sets and has a large number of available images. We note that the evaluation is focused on images containing a single-class (i.e., no "person" and "car" in same image) as this allows for a more controlled environment.

3.2.1 ***Evaluating Detection of Attacks.*** To evaluate UnMask's effectiveness in detecting adversarial images against attacks (DF, FGSM, PGD), we use a contamination level of 0.5—meaning half of the images are benign and the other half are adversarial. Figure 2 summarizes UnMask's detection effectiveness, using *receiver operating characteristics* (ROC) curves constructed by varying the adversarial-benign threshold t. The curves show UnMask's performances across operating points as measured by the tradeoff between *true positive* (TP) and *false positive* (FP) rates.

An interesting characteristic of UnMask's protection is that its effectiveness may not be affected strictly based on the number of classes in the dataset as in conventional classification tasks. Rather, an important factor is how much *feature overlap* there is among the classes. The ROC curves in Figure 2 illustrate this phenomenon, where UnMask provides better detection when there are 5 classes (Figure 2, dark orange) than when there are 3 classes (light blue). As shown in Table 5, the 5-class setup (CS5a—dark orange) has a feature overlap of 23.53% across the the 5 classes' 34 unique features, while the 3-class setup (CS3b—light blue) has 50% overlap. Keeping the number of classes constant and varying their feature overlap also supports this observation about the role of feature overlap (e.g., CS3a vs. CS3b in Figure 2). We call each combination of *class count* and *feature overlap* a "class set," abbreviated as "CS." CS3 thus means a class set with 3 classes. CS3a and CS3b have the same number of classes, with different feature overlap. Table 4 details the number of images used in testing detection (and defense) across the four class sets we investigated.

For a given feature overlap level, UnMask performs similarly across attack methods. When examining feature overlap 6.89% (CS3a) on VGG16, UnMask attains an AUC scores of 0.952, 0.96, 0.959, 0.951 and 0.949 on attacks DF, FGSM (ϵ=8,16) and PGD (ϵ=8,16), respectively. This result is significant because it highlights the ability of UnMask to operate against multiple strong attack strategies to achieve high detection success rate. As a representative ROC operating point for the attack vectors, we use PGD (ϵ=8), on feature overlap 6.89%. In this scenario, UnMask is able to detect up to 92.67% of attacks with a false positive rate of 9.67%. We believe that performing well in a low feature overlap environment is all that is required. This is because in many circumstances it is not important to distinguish the exact true class (e.g., dog or cat) of the image, but whether the image is being completely misclassified (e.g., car vs. person). Therefore, in practice, classes can be selected such that feature overlap is minimized.

3.2.2 ***Evaluating Defense and Rectification.*** Detecting an attack is only the first step of UnMask's protection, it also rectifies the misclassification by comparing the extracted features f_e to every set of class features in V, outputting class c that contains the highest feature similarity. As the evaluation focus is on rectifying

Setup				DF		FGSM ($\epsilon = 8$)		FGSM ($\epsilon = 16$)		PGD ($\epsilon = 8$)		PGD ($\epsilon = 16$)	
Model M	Class Set	Overlap	No Attk	No Def	UnMask	No Def	UnMask	No Def	UnMask	No Def	UnMask	No Def	UnMask
VGG16	CS3a	6.89%	87.00	5.13	94.33	0	84.53	0	73.44	0	92.24	0	89.89
	CS3b	50.00%	89.13	3.47	85.62	0	71.64	0	60.11	0	79.49	0	75.19
	CS5a	23.53%	80.35	3.91	91.11	0	79.12	0	65.86	0	86.05	0	82.65
	CS5b	29.41%	81.36	3.04	87.17	0	74.93	0	62.88	0	81.75	0	77.02
ResNet50	CS3a	6.89	86.64	4.51	95.04	0	87.11	0	74.42	0	92.9	0	90.81
	CS3b	50.00	85.75	3.28	86.12	0	76.07	0	66.71	0	82.51	0	78.55
	CS5a	23.53	80.35	3.91	91.11	0	79.12	0	65.86	0	86.05	0	82.65
	CS5b	29.41	79.91	3.33	87.57	0	76.91	0	65.19	0	83.39	0	80.01

Table 6: UnMask's accuracies (in %) in countering three attacks: DeepFool (DF), FGSM, and the strongest, *Projected Gradient Descent* (PGD) technique. We test two popular CNN architectures, VGG16 and ResNet50, as unprotected model M, with four class sets with varying numbers of classes and feature overlap. We show the models' accuracies (1) when not under attack ("No Attk" column); (2) attacked without defense ("No Def"); and (3) attacked and defended by UnMask.

misclassification, our test images have a contamination level of 1—meaning all of the images are adversarial. We evaluate UnMask's rectification capability on:

- 2 neural network models (VGG16, ResNet50)
- 3 attacks (DF, FGSM, PGD)
- 4 class sets (CS3a, CS3b, CS5a, CS5b)

Table 6 shows that UnMask is agnostic to the deep learning model that is being protected, as measured by the ability of UnMask to infer an adversarial images' actual class. This can be seen when comparing the results across each attack on VGG16 and ResNet50.

In addition, we find that the results from Table 6 support our observation that *feature overlap* is the dominant factor in determining the accuracy of the UnMask defense, as opposed to the number of classes. When examining DeepFool (DF) on class set CS3b (3 classes; feature overlap 50%), UnMask is able to determine the underlying class 85.62% of the time. At class set CS5a (5 classes; feature overlap 23.53%) we obtain an accuracy of 91.11%, highlighting the important role that feature overlap plays in UnMask's defense ability.

It is interesting to note that FGSM is more effective at attacking our UnMask defense than the other two attacks. We believe this is due to the single-step attacks' better transferability, which has been reported in prior work [16]. Given this transferability property of FGSM, we believe UnMask provides a significant defense.

We also mention the fact that UnMask's accuracy can be higher than the un-attacked model M due to the fact that, in some instances, model K learned a better representation of the data through the feature masks as opposed to model M, which trained on the images directly. This occurs on multiple occasions in Table 6.

4 RELATED WORK

Many methods have been proposed to combat adversarial image perturbations, which we discuss a few of the primary ones below. However, in our evaluation of the related research, we found none of them take into account the building-block knowledge information that can be extracted from the images themselves to combat adversarial attacks.

Adversarial Training. The objective of adversarial training is to vaccinate deep learning models to adversarial image perturbations by modifying the model's training process to include examples of attacked images [16, 26]. The downside to this technique is that models require large amounts of adversarial data, increasing the model training time [19].

Pre-processing. The goal of pre-processing is to eliminate adversarial perturbation before model inference. There are many proposed techniques, a couple of which include—(i) image compression [8] and dimensionality reduction [2]. The downside of this approach is that most pre-processing techniques have no knowledge of whether the system is actually being attacked.

Adversarial Detection. Instead of performing accurate classifications on adversarial examples, many techniques have been developed to look at the problem of detecting whether the input data is benign or adversarial, using a variety of methods from topological subgraph analysis [10] to various forms of image processing [18, 27, 28] and hidden/output layer activation distribution information [5, 17, 20].

5 CONCLUSION

In this paper, we have introduced a new fundamental concept of *building-block knowledge extraction*, and showed how it protects deep learning models against adversarial attacks through the UnMask detection and defense framework. We draw inspiration from humans' natural ability to make robust classification decisions through the detection and synthesis of contextual building-block knowledge contained in images. We aim to design and develop our UnMask framework to simulate such capability, so it can (1) detect adversarial pixel-centric manipulations targeting a deep learning model, and (2) defend the model against attacks by rectifying the classification. Through extensive evaluation on large-scale real-world image data, we showcase the merits of our ideas through UnMask's ability to *detect* up to 92.9% of attacks with a false positive rate of 9.67% and *defend* deep learning models by correctly classifying up to 92.24% of adversarial images in the gray-box scenario. Our proposed method is fast and architecture-agnostic.

In this work, we direct our efforts to systematically studying the efficacy of UnMask and the concept of building-block knowledge extraction on their own. However, we are currently working on (1) benchmarking UnMask against multiple state-of-the-art adversarial detection and defense techniques; and (2) reducing the dependency of UnMask to expensive data annotation.

REFERENCES

[1] Waleed Abdulla. 2017. Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow. https://github.com/matterport/Mask_RCNN. (2017).

[2] Arjun Nitin Bhagoji, Daniel Cullina, and Prateek Mittal. 2017. Dimensionality Reduction as a Defense against Evasion Attacks on Machine Learning Classifiers. *arXiv preprint arXiv:1704.02654* (2017).

[3] W. Brendel, J. Rauber, and M. Bethge. 2018. Decision-Based Adversarial Attacks: Reliable Attacks Against Black-Box Machine Learning Models. In *International Conference on Learning Representations*.

[4] Angel Cabrera, Fred Hohman, Jason Lin, and Duen Horng Chau. 2018. Interactive Classification for Deep Learning Interpretation. *Demo, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018).

[5] Fabio Carrara, Fabrizio Falchi, Roberto Caldelli, Giuseppe Amato, Roberta Fumarola, and Rudy Becarelli. 2017. Detecting Adversarial Example Attacks to Deep Neural Networks. In *Proceedings of the 15th International Workshop on Content-Based Multimedia Indexing (CBMI '17)*. 38:1–38:7.

[6] Shang-Tse Chen, Cory Cornelius, Jason Martin, and Duen Horng Chau. 2018. ShapeShifter: Robust Physical Adversarial Attack on Faster R-CNN Object Detector. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*.

[7] Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun, and Alan Yuille. 2014. Detect What You Can: Detecting and Representing Objects using Holistic Models and Body Parts. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[8] Nilaksh Das, Madhuri Shanbhogue, Shang-Tse Chen, Fred Hohman, Siwei Li, Li Chen, Michael E Kounavis, and Duen Horng Chau. 2018. Shield: Fast, Practical Defense and Vaccination for Deep Learning using JPEG Compression. In *Proceedings of the 24nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[9] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. [n. d.]. The PASCAL Visual Object Classes Challenge 2010 (VOC2010) Results. http://www.pascal-network.org/challenges/VOC/voc2010/workshop/index.html. ([n. d.]). Accessed: 2019-01-01.

[10] Thomas Gebhart and Paul Schrater. 2017. Adversary Detection in Neural Networks via Persistent Homology. *arXiv preprint arXiv:1711.10056* (2017).

[11] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and Harnessing Adversarial Examples. In *International Conference on Learning Representations*.

[12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask r-cnn. In *IEEE International Conference on Computer Vision*. 2980–2988.

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

[14] Fred Hohman, Nathan Hodas, and Duen Horng Chau. 2017. ShapeShop: Towards Understanding Deep Learning Representations via Interactive Experimentation. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. ACM, 1694–1699.

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

[16] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. 2017. Adversarial Machine Learning at Scale. In *International Conference on Learning Representations*.

[17] Xin Li and Fuxin Li. 2017. Adversarial Examples Detection in Deep Networks with Convolutional Filter Statistics.. In *ICCV*. 5775–5783.

[18] Bin Liang, Hongcheng Li, Miaoqiang Su, Xirong Li, Wenchang Shi, and Xiaofeng Wang. 2017. Detecting Adversarial Examples in Deep Networks with Adaptive Noise Reduction. *arXiv preprint arXiv:1705.08378* (2017).

[19] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *International Conference on Learning Representations*.

[20] Dongyu Meng and Hao Chen. 2017. MagNet: A Two-Pronged Defense Against Adversarial Examples. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS '17)*. 135–147.

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

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

[23] Karen Simonyan and Andrew Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conference on Learning Representations*.

[24] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. 2017. One pixel attack for fooling deep neural networks. *arXiv preprint arXiv:1710.08864* (2017).

[25] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *International Conference on Learning Representations*.

[26] Florian TramÃĺr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. 2018. Ensemble Adversarial Training: Attacks and Defenses. In *International Conference on Learning Representations*.

[27] Jingyi Wang, Jun Sun, Peixin Zhang, and Xinyu Wang. 2018. Detecting Adversarial Samples for Deep Neural Networks through Mutation Testing. *arXiv preprint arXiv:1805.05010* (2018).

[28] Weilin Xu, David Evans, and Yanjun Qi. 2018. Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks. In *25th Annual Network and Distributed System Security Symposium (NDSS)*.