

ROBUST ADVERSARIAL ℓ_p ATTACK DETECTION USING BLOCK-SPARSE DECOMPOSITION

Emre Acarturk

Department of Electrical and Computer Systems Engineering
Rensselaer Polytechnic Institute
Tory, NY 12180, USA
acarte@rpi.edu

Maruf A. Mridul & Thomas M. Waite

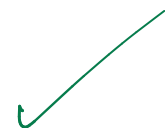
Department of Computer Science
Rensselaer Polytechnic Institute
Tory, NY 12180, USA
{mridum, waitet}@rpi.edu

100

only have to complete the current exp setup

ABSTRACT

Neural networks have consistently outperformed a variety of other models for high-dimensional classification tasks, particularly on image classification. However, with carefully-chosen, visually-imperceptible perturbations to their inputs, adversaries can almost always force the network to misclassify the image. One class of such attacks is the ℓ_p -norm bounded attacks, and while many methods have been developed to minimize the impact of such attack on classification accuracy, fewer approaches have sought to detect and report potentially adversarial examples in a principled way. We take this defensive perspective in developing a robust attack detection and classification algorithm that utilizes block-sparse signal decomposition to detect and report potentially adversarial inputs to a network, thereby providing additional user safety against attackers.



1 INTRODUCTION

Deep neural networks have been widely used in various applications, such as image classification, speech recognition, and natural language processing. However, it has been shown that these models are vulnerable to adversarial attacks, which are carefully crafted inputs designed to deceive the model into producing incorrect outputs (Biggio et al. (2013), Szegedy et al. (2014)). Adversarial attacks can be imperceptible to humans, making them a serious concern for security-sensitive applications.

Several defense methods have been proposed to mitigate the effects of adversarial attacks. However, these defenses have often been shown to be ineffective against new types of attacks. As a result, there is a continuous cycle of developing new attacks (Athalye et al. (2018), Carlini & Wagner (2017), Uesato et al. (2018), Athalye & Carlini (2018)) and new defenses (Madry et al. (2017), Samangouei et al. (2018), Zhang et al. (2019), Papernot et al. (2016), Kurakin et al. (2016), Miyato et al. (2018), Zheng et al. (2016)), which has led to a cat-and-mouse game between attackers and defenders.

1.1 OUR CONTRIBUTIONS

Our work aims to investigate the utility of block-sparse reconstruction of images into clean and perturbed components as a means to create robust detection algorithm for ℓ_p norm bound adversarial attacks. For this, we utilize a framework developed by Thaker et al. (2022) in which the authors claim that signals that have undergone ℓ_p norm bounded attacks can be reverse engineered to recover the clean and perturbed components of the signal. Details for this framework are outlined in Section 3. Starting from this model, however, we contribute the following:

First, we aim to determine whether block-sparse reconstruction can be used to accurately identify the presence and type of attack given a set of potentially perturbed signals. To achieve this we present attack detection and identification algorithms that leverage the block-sparse reconstructed image and a neural network classifier identify potentially attacked images as well as classify the type among ℓ_p attack (ℓ_1 , ℓ_2 , or ℓ_∞) for MNIST images.

Second, we aim to create a more robust detection algorithm, regardless of attack type, that minimizes the number of false positive attack detections. We show that this robust algorithm also finds the majority of attacks that are indeed successful on our network.

to minimize the regardless of conditions under which the above detector and classifier are no longer able to identify adversarial presence in signals. To do this, we examine the assumptions for the detection and attempt to create optimal attacks against our framework and analyze their efficacy, that break our detection algorithm. This will involve designing an attack that is specifically tailored to the framework, and evaluating how well it can bypass the detector.

Next, we present a survey of the related work in the field and summarize theoretical model on which our detection algorithms are based, then we define our algorithms and present the experimental results on the MNIST dataset.

2 RELATED WORK

Structured representations have been successfully utilized in various applications such as image classification (Yang et al. (2009b), Mairal et al. (2008)), action recognition (Yang et al. (2009a), Castrodad & Sapiro (2012)), and speech recognition (Gemmeke et al. (2011), Sainath et al. (2011)) through sparse signal representation. This approach assumes that data from a particular class can be represented by a low-dimensional subspace based on training samples of that class. To classify new data, the correct sparse representation of the signal on the column space of a specific dictionary needs to be recovered. Despite the success of this approach, it fails to consider the challenges posed by adversarially corrupted inputs. The reference study of this project aims to investigate these challenges and their impact on the classification of structured data. (Wright et al. (2010), Mairal et al. (2008)) discuss on more examples of structured representations for data classification.

Recently, denoising-based defense strategies that utilize structured data representations have been proposed in the adversarial learning community. Notable examples of such approaches include the work of (Samangouei et al. (2018)) and (Moosavi-Dezfooli et al. (2018)), which have been extensively surveyed in (Niu et al. (2020)). However, these methods are limited in that they do not perform attack classification. In contrast, the approach in our reference paper jointly recovers the signal and attack, making it unique and novel from a theoretical standpoint. While the approach is not aimed at simply improving defense performance, it can still be compared to other defenses that aim to handle a union of perturbation families simultaneously, such as those developed in (Tramer & Boneh (2019), Maini et al. (2020), Croce & Hein (2019)).

Numerous literature exists on the detection of adversarial attacks. The problem aims to identify whether an example is an adversarial or clean example. Detection methods can be categorized as either unsupervised or supervised, as demonstrated in studies such as (Metzen et al. (2017)) and (Grosse et al. (2017)). (Bulusu et al. (2020)) did an extensive survey on these methods.

3 THEORETICAL BACKGROUND

3.1 REFERENCE PAPER SUMMARY

In this project, we build on the work of (Thaker et al. (2022)) on reverse engineering adversarial attacks. Specifically, given a corrupted signal $\mathbf{x}' = \mathbf{x} + \delta$, where \mathbf{x} is a 'clean' signal and δ is an ℓ_p -norm bounded attack, the goal is to determine the attack type (ℓ_1 , ℓ_2 , or ℓ_∞) as well as the original signal \mathbf{x} . This problem is challenging since there could be many pairs (\mathbf{x}, δ) that yield the same \mathbf{x}' , and the attack vector δ may not be sparse.

To address these challenges, the authors propose to leverage results from the sparse recovery literature, which show that one can perfectly recover a signal \mathbf{x} from a corrupted version $\mathbf{x}' = \mathbf{x} + \delta_0$ when both \mathbf{x} and δ_0 are sparse on a meaningful basis. Specifically, if x is sparse with respect to some signal dictionary D_s , and δ_0 is also sufficiently sparse, then the solution (c^*, δ^*) to the convex problem

$$\min_c \|\mathbf{c}\|_1 + \|\delta\|_1 \text{ s.t. } \mathbf{x}' = D_s \mathbf{c} + \delta \quad (1)$$

is such that $c^* = c_s$ and $\delta^* = \delta_0$. In other words, one can perfectly recover the clean signal as $x = D_s c^* = D_s x_s$ and the corruption δ_0 by solving the convex problem in (1).

However, these classical results from sparse recovery are not directly applicable to the problem of reverse engineering adversarial attacks. One of the challenges is that an attack δ may not be sparse, and while results from sparse recovery can be extended to bounded ℓ_2 errors (Candes et al. (2006)), they only guarantee stable recovery, instead of exact recovery, of sparse vectors close to the solution.

To overcome these challenges, the authors propose a block-sparse recovery problem, where both the signal and the attack are assumed to lie in a union of subspaces that includes one subspace per class and one subspace per attack type. They derive geometric conditions on the subspaces under which any attacked signal can be decomposed as the sum of a clean signal plus an attack. By determining the subspaces that contain the signal and the attack, they can also classify the signal and determine the attack type. Their framework is proposed under three main assumptions :

- They assume that the signal x' to be classified is the sum of a clean signal x plus an ℓ_p - norm bounded adversarial attack δ , i.e., additive attacks.
- They assume that the clean signal x is block sparse with respect to a dictionary of signals D_s , i.e. $x = D_s c_s$, where D_s can be decomposed into multiple blocks, each one corresponding to one class, and c_s is blocksparse, i.e. c_s is only supported on a sparse number of blocks, but not necessarily sparse within those blocks.
- They assume that the ℓ_p - norm bounded adversarial attack also admits a block-sparse representation in the column space of a dictionary D_a , which contains blocks corresponding to different ℓ_p bounded attacks.

The proposed approach is evaluated on digit and face classification tasks, and the experimental results demonstrate its effectiveness in recovering the clean signal and determining the attack type.

3.2 MATHEMATICAL FORMALISM

For this paper, we use the following setting and key optimization problems

Assume there is a clean signal \mathbf{x} and an ℓ_p -bounded adversarial attack δ that generate the observed signal \mathbf{x}' exactly via addition, that is,

$$\mathbf{x}' = \mathbf{x} + \delta.$$

Next, they assume that both \mathbf{x} and δ are *block-sparse*. In order to define this, they start by assuming that these signals are generated via

$$\mathbf{x} = \mathbf{D}_s \mathbf{c}_s \quad \text{and} \quad \delta = \mathbf{D}_a \mathbf{c}_a,$$

where $\mathbf{D}_{s,a}$ are dictionaries that are partitioned into blocks such that \mathbf{D}_s has one block per output class, and \mathbf{D}_a has one block per attack class. The optimization problem they propose is

$$\min_{\mathbf{c}_s, \mathbf{c}_a} \|\mathbf{c}_s\|_{1/2} + \|\mathbf{c}_a\|_{1/2} \quad \text{s.t.} \quad \mathbf{x}' = \mathbf{D}_s \mathbf{c}_s + \mathbf{D}_a \mathbf{c}_a, \quad (2)$$

where $\|\cdot\|_{1/2}$ is an ℓ_1/ℓ_2 norm hybrid that is designed to promote block sparsity on the coefficients $\mathbf{c}_{s,a}$, defined as having non-zero entries in only sparse number of classes while not being necessarily sparse within a class block. They also give a slightly different notation to make the notion of block sparsity more clear:

$$\mathbf{x} = \sum_{i=1}^r \mathbf{D}_s[i] \mathbf{c}_s[i] \quad \text{and} \quad \delta = \sum_{i=1}^r \sum_{j=1}^a \mathbf{D}_a[i][j] \mathbf{c}_a[i][j],$$

where i corresponds to the signal class (of which there are r) and j corresponds to the attack class (of which there are a). Note that attacks are classified via both the attack type and signal class. This is because the authors construct the attack dictionary from the best adversarial attacks on the classifier at the training samples. In this notation, the optimization problem in 2 can be stated only in terms of ℓ_2 norm as

$$\min_{\mathbf{c}_s, \mathbf{c}_a} \sum_{i=1}^r \|\mathbf{c}_s[i]\|_2 + \sum_{i=1}^r \sum_{j=1}^a \|\mathbf{c}_a[i][j]\|_2 \quad \text{s.t.} \quad \mathbf{x}' = \mathbf{D}_s \mathbf{c}_s + \mathbf{D}_a \mathbf{c}_a. \quad (3)$$

They also convert this constrained optimization problem to a regularized noisy version of it, given by

$$\min_{\mathbf{c}_s, \mathbf{c}_a} \|\mathbf{x}' - \mathbf{D}_s \mathbf{c}_s + \mathbf{D}_a \mathbf{c}_a\|_2^2 + \lambda_s \sum_{i=1}^r \|\mathbf{c}_s[i]\|_2 + \lambda_a \sum_{i=1}^r \sum_{j=1}^a \|\mathbf{c}_a[i][j]\|_2. \quad (4)$$

This is the optimization problem we solve to generate the reconstructed image. For justification of its correctness and the algorithm used to solve it, we refer the reader to [Thaker et al. \(2022\)](#). Once solved, however, we are given the optimal \mathbf{c}_s^* , \mathbf{c}_a^* , and we reconstruct the denoised image as follows:

$$\hat{i} = \arg \min_i \|\mathbf{x}' - \mathbf{D}_s[i] \mathbf{c}_s^*[i] - \mathbf{D}_a \mathbf{c}_a^*\|_2 \quad (5)$$

$$\hat{\mathbf{x}} = \mathbf{D}_s[\hat{i}] \mathbf{c}_s^*[\hat{i}] \quad (6)$$

The result of this reconstruction process for an image under all three different attack methods is shown in Figure 1. We will return to this formulation when determining attack classes, but the reconstructed image $\hat{\mathbf{x}}$ will allow us create a naive attack detection algorithm in the next section.

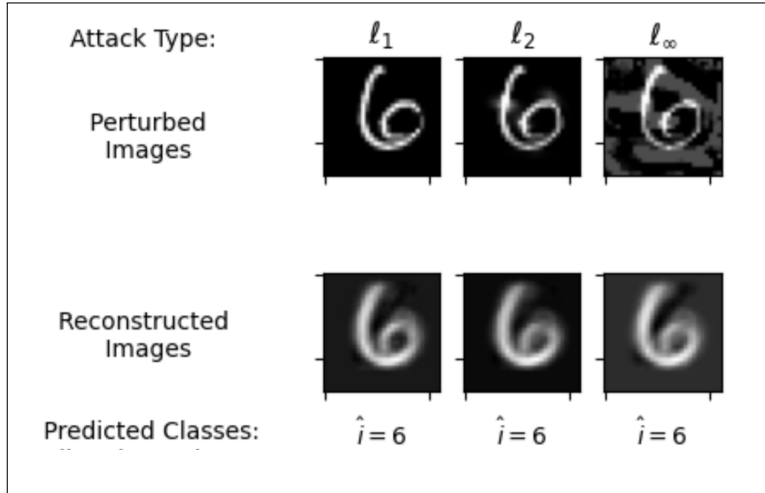


Figure 1: Sample Reconstructed images: $\hat{\mathbf{x}} = \mathbf{D}_s[\hat{i}] \mathbf{c}_s^*[\hat{i}]$ for each attack type.

4 OUR ATTACK DETECTION AND CLASSIFICATION FRAMEWORK

With this ability to reconstruct images into their signal and attack components, we propose the following approaches for attack detection and identification.

4.1 ATTACK DETECTION AND IDENTIFICATION ALGORITHM

Given a set of potentially attacked data to be classified \mathcal{X}_A , for each image \mathbf{x}_A , we first compute its reconstruction $\hat{\mathbf{x}}_A$ according to equation (6). Then use our network classifier to predict both the class of the potentially corrupted image $f(\mathbf{x}_A)$ and the reconstructed image $f(\hat{\mathbf{x}}_A)$. If they do not match, we save the image and its reconstruction to return to the user and we record a potential attack.

In addition simply detecting the presence of an attacked image, we seek to identify the type of attack carried out. Because we have a dictionary for the types of adversarial attacks, we can use it to determine the attack class using the following optimization problem:

$$\hat{j} = \arg \min_j \|\mathbf{x}' - \mathbf{D}_s \mathbf{c}_s^* - \mathbf{D}_a[\hat{i}][j] \mathbf{c}_a^*[\hat{i}][j]\|_2 \quad (7)$$

This finds the attack index $\hat{j} \in \{0, 1, 2\}$ corresponding to the attack types $\ell_p \in \{\ell_1, \ell_2, \ell_\infty\}$ that makes minimizes the difference between the potentially attacked image and our reconstructed image with signal and attack noise components, corresponding to the most likely attack type. The specifics of the algorithm are outlined in Algorithm 1.

Algorithm 1 Attack Detection and Identification Algorithm

Inputs:

Corrupted Test Set: \mathcal{X}_A
 NN Classifier: $f(\mathbf{x})$
 Signal and Adversary Dictionaries: $\mathbf{D}_s, \mathbf{D}_a$

Initialize:

Attacked Image Set: $\mathcal{A} \leftarrow \{\}$

for all $\mathbf{x}_A \in \mathcal{X}_A$ **do**

Compute $\mathbf{c}_s^*, \mathbf{c}_a^*$ with ASHA given $\mathbf{D}_a, \mathbf{D}_s$, and \mathbf{x}_A
 Compute $\hat{i}_A, \hat{\mathbf{x}}_A$, and \hat{j}_A according to Eqns. (5), (6), and (7)

▷ See Appendix fig 3

/* If the predictions don't match, record attack */

if $f(\mathbf{x}_A) \neq f(\hat{\mathbf{x}}_A)$ **then**

$\mathcal{A} \leftarrow \mathcal{A} \cup \{(\mathbf{x}_A, \hat{\mathbf{x}}_A, \hat{i}_A, \hat{j}_A)\}$

end if

end for

return \mathcal{A}

start this
is your
alg

4.2 ROBUST ATTACK DETECTION

The approach above is likely to produce many false positives, given that it must assume that the perturbed image underwent an attack to potentially identify the attack type via the reconstruction. In particular, if a given sample \mathbf{x}' is not an attacked image, then the optimization and reconstruction process in Equations (5), (6) can introduce some additive attack noise from the dictionary of attacks. This effectively amounts to unintentionally adding some adversarial noise to the originally clean image, leading to misclassification by our network and therefore a false positive.

To instead only detect the presence of an attack rather than additionally identify its type and minimize the number of false positives, we reconstruct the image from its sparsity representation assuming it is not attacked. That is, we solve the following unconstrained optimization problem:

$$\mathbf{c}_s^* = \min_{\mathbf{c}_s} \|\mathbf{x}' - \mathbf{D}_s \mathbf{c}_s\|_2^2 + \lambda_s \sum_{i=1}^r \|\mathbf{c}_s[i]\|_2. \quad (8)$$

Then we reconstruct the image as follows:

$$\hat{i} = \arg \min_i \|\mathbf{x}' - \mathbf{D}_s[i] \mathbf{c}_s^*[i]\|_2 \quad (9)$$

$$\hat{\mathbf{x}} = \mathbf{D}_s[\hat{i}] \mathbf{c}_s^*[\hat{i}] \quad (10)$$

5 EXPERIMENTS

Our goal is to leverage the reconstructed, denoised images to robustly detect ℓ_p attacks against a neural network classifier $f(\mathbf{x})$. We present experiments on the MNIST dataset.

5.1 EXPERIMENTAL CONTEXT

Classifier and Attack Generator: For the MNIST dataset, we use a Convolutional Neural Network Classifier with architecture outlined in Table 3. It was trained for 5 epochs with a learning rate of 0.01, momentum of 0.5, and batch size of 128 images. The attack images were generated using projected gradient descent methods and implemented using the Advtorch library. The specific parameters for each type are included in Table 4.

Block Sparse Dictionary Construction: The signal and adversarial dictionaries (\mathbf{D}_s and \mathbf{D}_a , respectively) were such that the columns for each class type are the flattened and normalized MNIST images for 200 sampled images from that class. In particular, to keep as small of set as possible, we implemented \mathbf{D}_s as a $(10 \times 784 \times 200)$ tensor where $\|\mathbf{D}_s[i, :, k]\|_2 = 1 \quad \forall i, k$. Similarly, \mathbf{D}_a is a $(3 \times 10 \times 784 \times 200)$ tensor where the first axis represents each of the three attack types. Sample entries from this dictionary are presented in Figure 2.

Optimization Solver: To solve the optimization problem in Equation (4) that provides the coefficients to reconstruct the clean images, we used the cvxpy package with an SCS solver and 50 maximum iterations.

Metrics for Attack Detection and Identification: To evaluate the efficacy of our detector, we randomly perturbed a subset of the test data with a equal distribution of attack types across classes. In particular, we started with 1000 images drawn from the MNIST test set, then selected 30 images at random from each class to be perturbed. Finally, we assigned 10 of those to be perturbed by ℓ_1 attacks, 10 to be perturbed by ℓ_2 attacks, and 10 to be perturbed by ℓ_∞ attacks.

Given this perturbed set $:= \mathcal{X}_A$, we then run Algorithm (I) and use the returned set \mathcal{A} to determine the following:

- **Overall Attacks Detected:** The percentage of the total real attacks that were detected by our algorithm and are found in the set \mathcal{A} .
- **True positives:** The number of real attacks detected in \mathcal{A} .
- **False positives:** The number of attacks detected in \mathcal{A} that were not real attacks.
- **% ℓ_p Detected:** For each attack type, the percentage of the real ℓ_p type attacks detected. This shows which attack types are easier for our detector to detect.
- **Overall Attacks Correctly Identified:** Of the real attacks we detected correctly, the percent for which we also correctly identified the attack type.
- **% ℓ_p Identified:** Of the real attack we detected correctly, the percent of each attack type we correctly identified. This show which attack types are easier for our detector to identify.

compute the number of attacks detected, as well as the number of false positivethen

Qualitatively, we will also present examples of false positives and false negatives, as well as the attacks that generated them to determine if any trends emerge with across attack type or across classes.

5.2 ATTACK DETECTION AND IDENTIFICATION RESULTS

5.4 OPTIMIZED ATTACK RESULTS

6 PROGRESS & NEXT STEPS

The following section outlines the progress made so far and next steps

6.1 COMPETED SO FAR

1. Drafted intro, background, and theoretical
2. Implemeneted and trained NN in Table 3 on MNIST Data Set

Attack Parameters	Overall Detected	True Positives	False Positives	ℓ_1 Detected	ℓ_2 Detected	ℓ_∞ Detected	Overall Identified	ℓ_1 Identified	ℓ_2 Identified	ℓ_∞ Identified
$\epsilon_{\ell_1} = 5, \epsilon_{\ell_2} = 1, \epsilon_{\ell_\infty} = .15$	44 %	122	172	32 %	24 %	66 %	43 %	59 %	29 %	39 %
$\epsilon_{\ell_1} = 10, \epsilon_{\ell_2} = 2, \epsilon_{\ell_\infty} = .3$	44.2 %	122	172	32 %	24 %	66 %	63 %	59 %	29 %	39 %
$\epsilon_{\ell_1} = 20, \epsilon_{\ell_2} = 4, \epsilon_{\ell_\infty} = .6$	37 %	111	155	22 %	31 %	58 %	63 %	55 %	26 %	86 %

Table 1: Detection and identification results for 300 attacked images of 1000 test images from the MNIST data set with varying attack parameters. 10 images were selected at random from each class to be attack by each attack type.

5.3 ROBUST ATTACK DETECTION RESULTS:

Attack Parameters	Total Attacks	Overall Detected	True Positives	False Positives	Accuracy on True Positives	Accuracy on False Negatives	Accuracy on $\mathcal{X}_A - A$	Raw Accuracy
$\ell_\infty : \epsilon = .3$	300	35 %	105	51	32 %	24 %	66 %	43 %
\vdots								

Table 2: Robust detection results using different attack types

3. Created block sparse dictionaries $\mathbf{D}_{a/s}$ using trained network and adversarial libraries (see Fig 2)
4. Implemented Active Set Homotopy Algorithm from [Thaker et al., (2022)], including debugging errors in original publication.
5. Produced reconstructed images from adversaries. See Fig 1
6. Defined detection and classification algorithms See Section 4.
7. Outlined Experimental data to be collected using above infrastructure.

6.2 NEXT STEPS

1. Run experiments to determine attack detection success and report results in table or graph with data corresponding to the template in Table 5.2 based on the description in Section 5.1.
2. Run experiments to determine attack classification results as described in section 5.1.
3. Design attack paradigm to subvert this detection and classification approach.

4. If above yields results, repeat experiments using this new paradigm. If not, test the detector on other well-known benchmark attacks (C-W, MM).
5. Repeat results for another data set, time permitting.

7 CONCLUSION

The conclusion will summarize the efficacy of our detection algorithm as well as the proposed attack paradigm to subvert it. It will also discuss the limitations and further research directions.

REFERENCES

- Anish Athalye and Nicholas Carlini. On the robustness of the cvpr 2018 white-box adversarial example defenses. *arXiv preprint arXiv:1804.03286*, 2018.
- Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International conference on machine learning*, pp. 274–283. PMLR, 2018.
- Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrđić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part III 13*, pp. 387–402. Springer, 2013.
- Saikiran Bulusu, Bhavya Kailkhura, Bo Li, Pramod K Varshney, and Dawn Song. Anomalous example detection in deep learning: A survey. *IEEE Access*, 8:132330–132347, 2020.
- Emmanuel J Candes, Justin K Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 59(8):1207–1223, 2006.
- 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*, pp. 3–14, 2017.
- Alexey Castrodad and Guillermo Sapiro. Sparse modeling of human actions from motion imagery. *International journal of computer vision*, 100:1–15, 2012.
- Francesco Croce and Matthias Hein. Provable robustness against all adversarial ℓ_p -perturbations for $p \geq 1$. *arXiv preprint arXiv:1905.11213*, 2019.
- Jort F Gemmeke, Tuomas Virtanen, and Antti Hurmalainen. Exemplar-based sparse representations for noise robust automatic speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(7):2067–2080, 2011.
- Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick McDaniel. On the (statistical) detection of adversarial examples. *arXiv preprint arXiv:1702.06280*, 2017.
- Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.
- 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.
- Pratyush Maini, Eric Wong, and Zico Kolter. Adversarial robustness against the union of multiple perturbation models. In *International Conference on Machine Learning*, pp. 6640–6650. PMLR, 2020.
- Julien Mairal, Francis Bach, Jean Ponce, Guillermo Sapiro, and Andrew Zisserman. Discriminative learned dictionaries for local image analysis. In *2008 IEEE conference on computer vision and pattern recognition*, pp. 1–8. IEEE, 2008.

- Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. *arXiv preprint arXiv:1702.04267*, 2017.
- Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993, 2018.
- Seyed-Mohsen Moosavi-Dezfooli, Ashish Shrivastava, and Oncel Tuzel. Divide, denoise, and defend against adversarial attacks. *arXiv preprint arXiv:1802.06806*, 2018.
- Zhonghan Niu, Zhaoxi Chen, Linyi Li, Yubin Yang, Bo Li, and Jinfeng Yi. On the limitations of denoising strategies as adversarial defenses. *arXiv preprint arXiv:2012.09384*, 2020.
- 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)*, pp. 582–597. IEEE, 2016.
- Tara N Sainath, Bhuvana Ramabhadran, Michael Picheny, David Nahamoo, and Dimitri Kanevsky. Exemplar-based sparse representation features: From timit to lvsr. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(8):2598–2613, 2011.
- Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *arXiv preprint arXiv:1805.06605*, 2018.
- Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, IJ Goodfellow, and Rob Fergus. Intriguing properties of neural networks: Proceedings of the international conference on learning representations. 2014.
- Darshan Thaker, Paris Giampouras, and René Vidal. Reverse engineering ℓ_p attacks: A block-sparse optimization approach with recovery guarantees. In *International Conference on Machine Learning*, pp. 21253–21271. PMLR, 2022.
- Florian Tramer and Dan Boneh. Adversarial training and robustness for multiple perturbations. *Advances in neural information processing systems*, 32, 2019.
- Jonathan Uesato, Brendan O’donoghue, Pushmeet Kohli, and Aaron Oord. Adversarial risk and the dangers of evaluating against weak attacks. In *International Conference on Machine Learning*, pp. 5025–5034. PMLR, 2018.
- John Wright, Yi Ma, Julien Mairal, Guillermo Sapiro, Thomas S Huang, and Shuicheng Yan. Sparse representation for computer vision and pattern recognition. *Proceedings of the IEEE*, 98(6):1031–1044, 2010.
- Allen Y Yang, Roozbeh Jafari, S Shankar Sastry, and Ruzena Bajcsy. Distributed recognition of human actions using wearable motion sensor networks. *Journal of Ambient Intelligence and Smart Environments*, 1(2):103–115, 2009a.
- Jianchao Yang, Kai Yu, Yihong Gong, and Thomas Huang. Linear spatial pyramid matching using sparse coding for image classification. In *2009 IEEE Conference on computer vision and pattern recognition*, pp. 1794–1801. IEEE, 2009b.
- Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning*, pp. 7472–7482. PMLR, 2019.
- Stephan Zheng, Yang Song, Thomas Leung, and Ian Goodfellow. Improving the robustness of deep neural networks via stability training. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4480–4488, 2016.

A APPENDIX

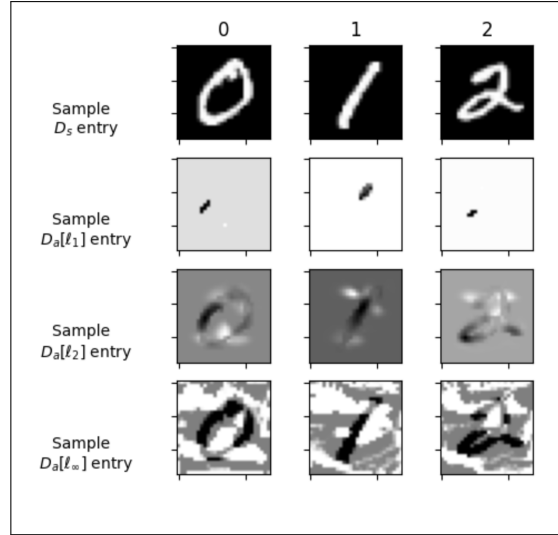


Figure 2: Sample Signal and Adversarial Perturbation Dictionary Entries. Note that images are scaled to be unit size, so the pixel intensity of the perturbed images does not represent the actual size of the generated perturbation. They are included here to demonstrate how the images are organized and the qualitative impact of each attack type.

Table 3: Network Architecture For MNIST Classifier

Layer Type	Size
2x(Conv2d+ReLU)	$3 \times 3 \times 32$, stride = 1
MaxPool2D	2×2 , stride=1
2x(Conv2d+ReLU)	$3 \times 3 \times 64$, stride=1
MaxPool2D	2×2 , stride=1
2x(Linear+ReLU)	200
Linear+ReLU	10

Table 4: Aversarial Attack Parameters

Attack Type	Max Perturbation ϵ	Step Size α	Max Iterationr Num N
ℓ_1	10	0.8	100
ℓ_2	2	0.1	200
ℓ_∞	0.3	0.01	100

Algorithm 1 Active Set Homotopy Algorithm

Results: $\hat{\mathbf{c}}_a^*, \hat{\mathbf{c}}_s^*$
 Initialize : $\hat{\mathbf{c}}_s^0, \hat{\mathbf{c}}_a^0 \leftarrow \mathbf{0}, \mathbf{0}, T_s^0, T_a^0 \leftarrow \emptyset, \emptyset, k \leftarrow 1$
 Set : $\gamma \in (0, 1)$
while $T_s^{k+1} \not\subseteq T_s^k$ and $T_a^{k+1} \not\subseteq T_a^k$ **do**
 $\mathbf{o}^k \leftarrow \mathbf{x}' - \mathbf{D}_s[T_s^k] \hat{\mathbf{c}}_s^k[T_s^k] - \mathbf{D}_a[T_a^k] \hat{\mathbf{c}}_a^k[T_a^k]$
 $\lambda_s^k \leftarrow \gamma \cdot \max_i \|\mathbf{D}_s[i]^T \mathbf{o}^k\|_2$
 $\lambda_a^k \leftarrow \gamma \max_{i,j} \|\mathbf{D}_a[i][j]^T \mathbf{o}^k\|_2$
 $\hat{i}^k \leftarrow \arg \max_i \|\mathbf{D}_a[i]^T \mathbf{o}^k\|_2$
 $\hat{j}^k \leftarrow \arg \max_j \max_i \|\mathbf{D}_a[i][j]^T \mathbf{o}^k\|_2$
 Add \hat{i}^k and \hat{j}^k to T_s^k and T_a^k respectively.
 Solve problem (11) with any solver using $\mathbf{D}_s[T_s^k], \mathbf{D}_a[T_a^k], \lambda_s^k, \lambda_a^k$ and compute $\hat{\mathbf{c}}_s^{k+1}$ and $\hat{\mathbf{c}}_a^{k+1}$.
 $k \leftarrow k + 1$
end while

Figure 3: Algorithm for finding reconstruction coefficients from [Thaker et al. \(2022\)](#)