

Fall 2021

Linear Inverse Problems and Neural Networks

Jasjeet Dhaliwal
San Jose State University

Follow this and additional works at: https://scholarworks.sjsu.edu/etd_theses

Recommended Citation

Dhaliwal, Jasjeet, "Linear Inverse Problems and Neural Networks" (2021). *Master's Theses*. 5228.
DOI: <https://doi.org/10.31979/etd.wfjb-2m4y>
https://scholarworks.sjsu.edu/etd_theses/5228

This Thesis is brought to you for free and open access by the Master's Theses and Graduate Research at SJSU ScholarWorks. It has been accepted for inclusion in Master's Theses by an authorized administrator of SJSU ScholarWorks. For more information, please contact scholarworks@sjsu.edu.

LINEAR INVERSE PROBLEMS AND NEURAL NETWORKS

A Thesis

Presented to

The Faculty of the Department of Mathematics and Statistics

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Arts

by

Jasjeet Dhaliwal

December 2021

© 2021

Jasjeet Dhaliwal

ALL RIGHTS RESERVED

The Designated Thesis Committee Approves the Thesis Titled

LINEAR INVERSE PROBLEMS AND NEURAL NETWORKS

by

Jasjeet Dhaliwal

APPROVED FOR THE DEPARTMENT OF MATHEMATICS AND STATISTICS

SAN JOSÉ STATE UNIVERSITY

December 2021

Kyle Hambrook, Ph.D.

Department of Mathematics and Statistics

Wasin So, Ph.D.

Department of Mathematics and Statistics

Tahir Bachar Issa, Ph.D.

Department of Mathematics and Statistics

ABSTRACT

LINEAR INVERSE PROBLEMS AND NEURAL NETWORKS

by Jasjeet Dhaliwal

We investigate two ideas in this thesis. First, we analyze the results of adapting recovery algorithms from linear inverse problems to defend neural networks against adversarial attacks. Second, we analyze the results of substituting sparsity priors with neural network priors in linear inverse problems. For the former, we are able to extend an existing compressive sensing framework to defend neural networks against ℓ_0 , ℓ_2 , and ℓ_∞ norm attacks, and for the latter, we find that our method yields an improvement over reconstruction results of existing neural network based priors.

ACKNOWLEDGMENTS

I am grateful to Dr. Hambrook for his guidance, support, and patience. I am also thankful to Dr. Wasin So and Dr. Tahir Bachar Issa for being a part of my advisory committee.

TABLE OF CONTENTS

List of Tables	viii
List of Figures	ix
1 Introduction.....	1
1.1 Defending Neural Networks Against Adversarial Attacks	1
1.2 Neural Networks as Priors in Linear Inverse Problems	2
1.3 Organization	2
2 Linear Inverse Problems	3
2.1 Recovery Problem	3
2.2 Recovery Algorithms	4
2.2.1 Basis Pursuit	5
2.2.2 Iterative Hard Thresholding	6
3 Neural Networks	8
3.1 Network Training	8
3.2 Denoising Auto Encoder	9
3.3 Adversarial Inputs	9
3.3.1 Attack Algorithms	10
3.3.2 Defenses Against Adversarial Attacks.....	11
4 Defending Neural Networks Against Adversarial Attacks	13
4.1 Introduction	13
4.2 Related Work.....	14
4.3 Compressive Recovery Defense	16
4.3.1 Main Results	18
4.3.2 Proofs	19
4.4 Experiments	25
4.4.1 Defense Against ℓ_2 -norm Attacks.....	26
4.4.2 Defense Against ℓ_∞ -norm Attacks	28
4.4.3 Defense Against ℓ_0 -norm Attacks.....	29
5 Neural Networks as Priors in Linear Inverse Problems	32
5.1 Introduction	32
5.2 Related Work.....	34
5.3 Algorithm and Results.....	36
5.3.1 Algorithm.....	36
5.3.2 Theoretical Results.....	36
5.4 Experiments	40

5.4.1	Setup	40
5.4.2	Compressive Sensing	41
5.4.3	Inpainting	43
5.4.4	Super-resolution	44
6	Conclusions	46
6.1	Defending Neural Networks Against Adversarial Attacks	46
6.2	Neural Networks as Priors in Linear Inverse Problems	46
	Literature Cited	47

LIST OF TABLES

Table 1.	Network Architecture: MNIST and Fashion-MNIST	26
Table 2.	Network Performance: Algorithm 3 for ℓ_2 attacks	28
Table 3.	Network Performance: Algorithm 4 for ℓ_∞ attacks	30
Table 4.	Network Performance: Algorithm 3 for ℓ_0 attacks	31
Table 5.	Network Architecture: CelebA and MNIST	41
Table 6.	DAE-PGD Runtime	44

LIST OF FIGURES

Fig. 1.	Reconstruction quality: Algorithm 3 for ℓ_2 attacks	27
Fig. 2.	Reconstruction quality: Algorithm 4 with and without constraint	29
Fig. 3.	Reconstruction quality: Algorithm 4 for ℓ_0 attacks	30
Fig. 4.	Reconstruction quality: Algorithm 3 for ℓ_0 attacks	30
Fig. 5.	Compressive Sensing: CelebA	42
Fig. 6.	Compressive Sensing: Recovery Error	42
Fig. 7.	Compressive Sensing: MNIST	43
Fig. 8.	Inpainting: CelebA and MNIST	44
Fig. 9.	Super-resolution: CelebA and MNIST	45

1 INTRODUCTION

Consider the linear system $y = Ax + e$ where $A \in \mathbb{C}^{m \times N}$, $x \in \mathbb{C}^N$, $y, e \in \mathbb{C}^m$. Now suppose we are given A and y and asked to recover x . In case $m < N$, the linear system is underdetermined and without additional information, it is impossible to recover x exactly. However, imposing additional constraints on x , such as sparsity, allows us to make progress on this problem. This is the linear inverse problem we study in this thesis.

Separately, consider a function $f : \mathbb{R}^m \mapsto \mathbb{R}^n$, and a finite set of points $X = \{(x_i, f(x_i))\}_{i=1}^l$ with $x_i \in \mathbb{R}^m$ and another set $X' = \{x'_i\}_{i=1}^k$ with $x'_i \in \mathbb{R}^m$ and $\{x_i\}_{i=1}^l \cap \{x'_i\}_{i=1}^k = \emptyset$. Now suppose we are given X and tasked with finding an approximation \hat{f} to f such that $\hat{f}(x'_i) = f(x'_i)$ for $x'_i \in X'$. Assuming l is some large number and without having any additional information on the function f , we would need our search space for \hat{f} to be rich enough, so as to find a good approximation \hat{f} . This is where we turn to neural networks which have been shown to be universal function approximators [1].

We have seen remarkable results on constrained linear inverse problems in recent years [2]–[9]. Simultaneously, there has been tremendous success in applying neural networks to problems in a wide array of problems [10]–[14]. Although progress in each field has been largely independent of the other, researchers have become interested in applying methods from one field to improve results in the other. We continue this line of work and investigate two problems lying at the intersection of both fields.

1.1 Defending Neural Networks Against Adversarial Attacks

We start by considering the problem of defending neural networks against adversarial inputs. In particular, we extend the framework introduced in [15] to defend neural networks against ℓ_2 , ℓ_∞ , and ℓ_0 norm attacks. We call this defense framework Compressive Recovery Defense (CRD) as it utilizes recovery algorithms from the theory of compressive sensing. For defending against ℓ_2 -norm and ℓ_0 -norm attacks, we use Basis

Pursuit (BP) as the recovery algorithm and for the case of ℓ_∞ -norm attacks, we utilize the Dantzig Selector (DS) with a novel constraint. For each recovery algorithm used, we provide rigorous recovery guarantees that do not depend on the noise generating mechanism and can therefore be utilized by CRD against any ℓ_2 , ℓ_∞ , or ℓ_0 norm attacks. Finally, we experimentally demonstrate that CRD is effective in defending neural networks against state of the art ℓ_2 , ℓ_∞ , and ℓ_0 -norm attacks.

1.2 Neural Networks as Priors in Linear Inverse Problems

Generative priors have been shown to provide improved results over sparsity priors in linear inverse problems. However, current state of the art methods suffer from one or more of the following drawbacks: (a) speed of recovery is slow; (b) reconstruction quality is deficient; (c) reconstruction quality is contingent on a computationally expensive process of tuning hyperparameters. In this work, we address these issues by utilizing Denoising Auto Encoders (DAEs) as priors and a projected gradient descent algorithm for recovering the original signal. We provide rigorous theoretical guarantees for our method and experimentally demonstrate its superiority over existing state of the art methods in compressive sensing, inpainting, and super-resolution.

1.3 Organization

We begin by providing the necessary background on linear inverse problems in Section 2. Section 3 provides an introduction to neural networks and adversarial inputs. We investigate the problem of defending neural networks against adversarial attacks in Section 4 which is based on the work done in [16]. Finally, in Section 5 we utilize neural network based priors for solving linear inverse problems which is based on the work done in [17]. The thesis is structured such that Section 4 and Section 5 are self-contained and can be read independently.

2 LINEAR INVERSE PROBLEMS

2.1 Recovery Problem

Linear inverse problems can be formulated mathematically as $y = Ax$ where $y \in \mathbb{C}^m$ is the observed vector, $A \in \mathbb{C}^{m \times N}$ is the measurement process, and $x \in \mathbb{C}^N$ is the original signal. The problem is to recover the signal x , given the observation y and the measurement matrix A .

Classic linear algebra results dictate that in order to recover x , we need $m \geq N$, otherwise, the system is underdetermined and there are zero or infinitely many solutions. However, one may ask whether x can be recovered in case certain constraints are imposed on the structure of x . Surprisingly, it was discovered that an s -sparse vector x can be recovered with high probability provided $m \geq Cs \ln(\frac{N}{s})$, where $C > 0$ is a universal constant (independent of s , m , and N). We first make these notions precise and then introduce recovery algorithms.

Let x be a vector in \mathbb{C}^N and let $S \subseteq \{1, \dots, N\}$ with $\bar{S} = \{1, \dots, N\} \setminus S$

Definition 1. The support of x , denoted by $\text{supp}(x)$, is the set of indices of the non-zero entries of x , that is, $\text{supp}(x) = \{i \in \{1, \dots, N\} : x_i \neq 0\}$.

Definition 2. The ℓ_0 -quasinorm of x , denoted $\|x\|_0$, is defined to be the number of non-zero entries of x , i.e. $\|x\|_0 = \text{card}(\text{supp}(x))$. A vector x is called s -sparse if $\|x\|_0 \leq s$.

We use $x_{h(k)}$ to denote a k -sparse vector in \mathbb{C}^N consisting of the k largest (in absolute value) entries of x with all other entries zero. For example, if $x = [4, 5, -9, 1]^T$ then $x_{h(2)} = [0, 5, -9, 0]^T$. Note that $x_{h(k)}$ may not be uniquely defined. In contexts where a unique meaning for $x_{h(k)}$ is needed, we can choose $x_{h(k)}$ out of all possible candidates according to a predefined rule (such as the lexicographic order). We also define $x_{t(k)} = x - x_{h(k)}$. If $x = [x_1, x_2]^T \in \mathbb{C}^{2n}$ with $x_1, x_2 \in \mathbb{C}^n$, and if x_1 is k -sparse and x_2 is

t -sparse, then x is called (k, t) -sparse. We define $x_{h(k,t)} = [(x_1)_{h(k)}, (x_2)_{h(t)}]^T$, which is a (k, t) -sparse vector in \mathbb{C}^{2n} .

Definition 3. For $p > 0$, the ℓ_p -error of best s -term approximation to a vector $x \in \mathbb{C}^N$ is defined as

$$\sigma_s(x)_p := \inf \{ \|x - z\|_p, z \in \mathbb{C}^N \text{ is } s\text{-sparse} \}$$

Definition 4. The s -th restricted isometry constant $\delta_s = \delta_s(A)$ of a matrix $A \in \mathbb{C}^{m \times N}$ is the smallest $\delta \geq 0$ such that

$$(1 - \delta) \|x\|_2^2 \leq \|Ax\|_2^2 \leq (1 + \delta) \|x\|_2^2$$

for all s -sparse vectors $x \in \mathbb{C}^N$. If $A \in \mathbb{C}^{m \times N}$ is a matrix, we denote by $A_S \in \mathbb{C}^{m \times |S|}$ the column sub-matrix of A consisting of the columns indexed by S . Next, we introduce two primary recovery algorithms used in linear inverse problems.

2.2 Recovery Algorithms

A natural approach to consider when reconstructing an s -sparse vector $x \in \mathbb{C}^N$ from its measurement vector $y \in \mathbb{C}^m$ is to solve ℓ_0 -minimization problem

$$\underset{z \in \mathbb{C}^N}{\text{minimize}} \|z\|_0 \text{ subject to } \|Az - y\|_2 \leq \eta \quad (1)$$

One approach for this problem could be to solve for $A_S z = y$ where the solver iterates over all possible s -size subsets of $[1, 2, \dots, N]$. However, since there are $\binom{N}{s}$ such subsets, this approach is computationally prohibitive. In fact, as the next theorem shows, (1) turns out to be an NP-hard problem.

Theorem 5 (Theorem 2.17 in [18]). *For any $\eta \geq 0$, the ℓ_0 -minimization problem defined in (1) for general $A \in \mathbb{C}^{m \times N}$ and $y \in \mathbb{C}^m$ is NP-hard.*

2.2.1 Basis Pursuit

Since (1) is a non-convex NP-hard problem, the next step to consider is a convex relaxation of the problem that can be solved in practice. The convex relaxation of (1) is an ℓ_1 -minimization problem:

$$\underset{z \in \mathbb{C}^N}{\text{minimize}} \quad \|z\|_1 \quad \text{subject to} \quad \|Az - y\|_2 \leq \eta \quad (2)$$

In fact (2) can be formulated as a Second Order Cone Programming (SOCP) problem and solved using off-the-shell solvers. The algorithm for solving (2) is commonly referred to as Basis Pursuit (BP) and shown in Algorithm 1.

Algorithm 1 Basis Pursuit

Input: The observed vector $y \in \mathbb{C}^m$, where $y = A\hat{x} + e$, the measurement matrix $A \in \mathbb{C}^{m \times N}$, and the norm of the error vector η such that $\|e\|_2 \leq \eta$

Output: $x^\# \in \mathbb{C}^N$

- 1: **procedure** BP(y, A, η)
 - 2: $x^\# \leftarrow \arg \min_{z \in \mathbb{C}^N} \|z\|_1 \quad \text{subject to} \quad \|Az - y\|_2 \leq \eta$
 - 3: **return** $x^\#$
-

As stated previously, reconstruction quality depends on the how sparse x is as well as how close to an isometry A is. The next theorem provides guarantees on recovery with BP based on the RIP constant of A and the sparsity of x .

Theorem 6 (Theorem 6.12 in [18]). *Suppose that the $2s$ -th restricted isometry constant of the matrix $A \in \mathbb{C}^{m \times N}$ satisfies $\delta_{2s} < \frac{4}{\sqrt{41}}$. Then, for any $x \in \mathbb{C}^N$ and $y \in \mathbb{C}^m$ with $\|Ax - y\|_2 \leq \eta$, a solution \hat{x} of*

$$\underset{z \in \mathbb{C}^N}{\text{minimize}} \quad \|z\|_1 \quad \text{subject to} \quad \|Az - y\|_2 \leq \eta$$

approximates the vector x with errors

$$\begin{aligned}\|x - \hat{x}\|_1 &\leq C\sigma_s(x)_1 + D\sqrt{s}\eta \\ \|x - \hat{x}\|_2 &\leq \frac{C}{\sqrt{s}}\sigma_s(x)_1 + D\eta\end{aligned}$$

where the constants $C, D > 0$ depend only on δ_{2s} .

2.2.2 Iterative Hard Thresholding

Another natural approach to consider for solving (2) is using gradient descent methods. In particular, one can consider an iterative gradient descent procedure with the update step defined as: $x_{t+1} = x_t - \gamma A^*(Ax_t - y)$, where γ is the step size. However, since the vector x_{t+1} is not necessarily s -sparse, the sparsity constraint on x may be violated. In order to bypass this issue, we can add a thresholding step that selects the s largest entries of $x_t - \gamma A^*(Ax_t - y)$ which using our notation can be written as $(x_t - \gamma A^*(Ax_t - y))_{h(s)}$. This algorithm is referred to as Iterative Hard Thresholding (IHT) and outlined in Algorithm 2.

Algorithm 2 Iterative Hard Thresholding

Input: Observed vector $y \in \mathbb{C}^n$, measurement matrix $A \in \mathbb{C}^{m \times N}$, and sparsity level s , step size γ , number of iterations T

Output: s -sparse vector $x^{[T+1]}$

```

1: procedure IHT( $y, A, s, T$ )
2:    $x^{[0]} \leftarrow 0$ 
3:   for  $i \in [0, \dots, T]$  do
4:      $z^{[i+1]} \leftarrow x^{[i]} + \gamma A^*(y - Ax^{[i]})$ 
5:      $x^{[i+1]} = (z^{[i+1]})_{h(s)}$ 
6:   return  $x^{[T+1]}$ 
```

Once again we can state recovery guarantees for IHT using the RIP constant of A and the sparsity level of x .

Theorem 7 (Theorem 6.18 in [18]). *Suppose that the $3s$ -th restricted isometry constant of the matrix $A \in \mathbb{C}^{m \times N}$ satisfies $\delta_{3s} < \frac{1}{\sqrt{3}}$. Then, for $x \in \mathbb{C}^N$, $e \in \mathbb{C}^m$, and $S \subset [N]$*

with $\text{card}(S) = s$, the sequence x_n defined by Algorithm 2 with $y = Ax + e$ satisfies, for any $n \geq 0$,

$$\|x^{[n]} - x_S\|_2 \leq \rho^n \|x^{[0]} - x_S\|_2 + \tau \|Ax_S + e\|_2 \quad (3)$$

where $\rho = \sqrt{3}\delta_{3s} < 1$, and $\tau \leq \frac{2.18}{1-\rho}$.

3 NEURAL NETWORKS

Consider the problem of approximating a non-linear function $f : \mathbb{R}^m \mapsto \mathbb{R}^n$, given only a finite set of points $X = \{(x_i, f(x_i))\}_{i=1}^l$ with $x_i \in \mathbb{R}^m$. It seems natural that l would need to be sufficiently large to capture f well. Moreover, if we want a good approximation \hat{f} to f , we would want the search space of functions to be rich enough to represent the complexity of f . For instance, if f is highly non-linear, we would expect our search space of functions to contain such non-linear functions. This is where neural networks become extremely useful. Neural networks are capable of approximating any Borel measurable function with arbitrary accuracy as long as they have enough parameters [1].

A K -layer neural network is a function $F : \mathbb{R}^m \mapsto \mathbb{R}^n$ of the form:

$$F(x) = f_K(W_K(\dots f_2(W_2(f_1(W_1x + b_1)) + b_2)\dots) + b_K)$$

where $W_i \in \mathbb{R}^{m_i \times m_{i-1}}$, $b_i \in \mathbb{R}^{m_i}$, for $1 \leq i \leq K$ with $m_0 = m$ and $m_K = n$. Each f_i for $1 \leq i \leq K$ is a non-linear differentiable function that is applied element wise to its input. The functions f_i are commonly referred to as activation functions and in most cases $f_i = f_j$ for $1 \leq i, j \leq K$. Some examples activation functions are the Sigmoid function: $f(x) = \frac{1}{1+e^{-x}}$ or the Tanh function: $f(x) = \frac{2}{1+e^{-2x}} - 1$. The set of matrices and biases $W = \{(W_i, b_i)\}_{i=1}^K$ comprise the parameters of the neural network. The number K is called the depth of the network and it has been shown that the complexity of functions that the network can represent grows exponentially with K [19]–[21]. We next describe how the values of the network parameters W are learned by solving an optimization problem.

3.1 Network Training

Neural network training refers to the procedure of finding a set of parameters values that "fit" a given dataset. More formally, given a set of data points $X = \{(x_i, y_i)\}_{i=1}^l$ where $x_i \in \mathbb{R}^m, y_i \in \mathbb{R}^n, 1 \leq i \leq l$, the goal of network training is to find values for the parameters W such that $F(x_i) = y_i$. When $y_i \in \{1, 2, \dots, k\}$ for some positive integer k ,

the task is commonly referred to as a classification task and y_i is called the label for x_i . In this case, we can write the neural network function as $F : \mathbb{R}^m \mapsto \{1, 2, \dots, k\}$.

In order solve the above problem, we define an objective function $\mathcal{L}(F, X)$ and search for parameter values that minimize it. This objective function is usually referred to as the loss function. As an example, the Mean Squared Error (MSE) loss is defined as

$$\mathcal{L}_{MSE}(F, X) = \frac{1}{l} \sum_{i=1}^l \|F(x_i) - y_i\|_2^2 \quad (4)$$

Since each f_i is differentiable, iterative gradient based methods can be utilized to minimize loss functions such as (4). We note that in general, neural networks have non-convex loss functions and thus there are no guarantees of finding global minima. For a more thorough introduction to neural networks, we refer the reader to Chapter 6 in [22].

3.2 Denoising Auto Encoder

A DAE is a neural network $F : \mathbb{R}^N \mapsto \mathbb{R}^N$ that can be written as a composition of two neural networks - an encoder neural network $E : \mathbb{R}^N \mapsto \mathbb{R}^k$ where $k < N$ and a decoder neural network $D : \mathbb{R}^k \mapsto \mathbb{R}^N$. Therefore, $F(x) = (D \circ E)(x)$. Given a set of n samples from a domain of interest $\{x_i\}_{i=1}^n$, the training set X is created by adding Gaussian noise to the original samples. That is, $X = \{x'_i\}_{i=1}^n$, where $x'_i = x_i + e_i$ and $e_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$.

The loss function for training F is the MSE loss : $\mathcal{L}_{MSE}(F, X) = \frac{1}{n} \sum_{i=1}^n \|F(x'_i) - x_i\|_2^2$. The training procedure uses gradient descent to minimize $\mathcal{L}_{MSE}(F, X)$.

3.3 Adversarial Inputs

Neural networks have recently been shown to be highly vulnerable to making incorrect classifications with imperceptible changes to their input [23]–[25]. That is, for a neural network F and a point x , it is possible to find δ such that $F(x + \delta) \neq F(x)$ where $\|\delta\|_p$ has a very small (almost zero) p -norm. Formally, let $F : \mathbb{R}^n \mapsto \{1, \dots, k\}$ be a classifier, $x \in \mathbb{R}^n$ be an input sample where $y \in \{1, \dots, k\}$ is the label for x and $F(x) = y$. Then, $x_{adv} = x + \delta$, where $\|\delta\|_p < \varepsilon$ with ε very small, such that $F(x_{adv}) \neq F(x)$.

Such perturbations δ are referred to as adversarial perturbations since they can be used in an adversarial setting to force a neural networks to incorrectly classify an input. The process adding adversarial perturbations to inputs is referred to an adversarial attack. The existence of such adversarial inputs has also been extended into the physical realm, where adversarial inputs remain adversarial even after physical processing [26]–[28]. Given the increasing use of neural networks in real-world applications, it is crucial to design methods capable of detecting and preventing such attacks in order to ensure safe and reliable deployment.

3.3.1 Attack Algorithms

A number of efficient attack algorithms have been developed to create such adversarial inputs [26], [29]–[34]. Goodfellow *et al.* developed the *Fast Gradient Sign Method* (FGSM) that perturbs the input in the direction that maximizes the objective function in a single step [29]. Kurakin *et al.* developed *Basic Iterative Method* (BIM) that enhanced the FGSM attack by performing it iteratively, and taking a small step in the direction maximizing loss at each iteration [26]. While these attacks lead to untargeted mis-classification, Papernot *et al.* introduced the Jacobian Saliency Map Attack (JSMA) to mis-classify an input to a pre-specified target class [30]. Further, Papernot *et al.* showed that such attacks can also be launched when the attacker does not have access to the internal parameters of the neural network [31]. As opposed to gradient-based attacks that use a linear approximation of the loss surface, Carlini and Wagner (C&W) utilized optimization to generate adversarial inputs with minimal perturbation [32]. Moosavi-Dezfooli *et al.* created the *DeepFool* (DF) attack that uses a linear approximation of the decision boundary to find adversarial perturbations even when other attacks fail [33]. Such attacks have been demonstrated against neural networks designed for tasks like image and object recognition [29], [35], speech recognition [36], and malware classification [37].

Adversarial attacks can also be successfully launched in the physical world even though the attacker has less control over the input features. For example, Kurakin *et al.* showed that adversarial perturbations remain adversarial even when they are processed by physical media [26]. Sharif *et al.* [27] designed adversarial eye-frames that an attacker can wear to fool a facial recognition system in a live setting. Eykholt *et al.* were able to create adversarial road signs that were misclassified by the computer vision system present in a moving car [38]. Carlini *et al.* demonstrated the success of adversarial attacks against voice assistants, e.g. Google Now, by adding inaudible perturbations to voice commands [36]. Athalye *et al.* generated a 3-D printed adversarial object which fooled a classifier from a wide-range of view points and angles [28]. Grosse *et al.* crafted adversarial inputs to fool neural networks built to classify malware [37]. The success of these attacks in fooling neural networks presents a major roadblock to their safe and reliable deployment.

Due to the adverse ramifications of such attacks and their implications on the safety of deep learning systems, a number of defenses have been proposed in the literature.

3.3.2 Defenses Against Adversarial Attacks

Several adversarial detectors have been proposed in the literature that rely on differences in statistical properties of adversarial and normal inputs. Hendrycks and Gimpel [39] discovered that adversarial inputs have abnormally large magnitudes of low-ranked principal components and used that to separate them from normal inputs. Bhagoji *et al.* used similar data transformations to prevent adversarial attacks [40]. Xu *et al.* used “feature-squeezing”, a method of reducing the range of values an input can have, to increase the distortion required for adversarial attacks [41]. Papernot and McDaniel used a *k-nearest neighbors* approach to detect adversarial samples [42]. Metzen *et al.* and Grosse *et al.* trained a classifier to detect adversarial inputs [43], [44]. Meng and Chen proposed a two-pronged defense that first used manifold representations to differentiate

normal inputs from adversarial inputs [45]. Then, they created a secondary network to push the adversarial inputs back to the manifold of the normal data. Most of these detectors however rely on statistical properties that are not clearly explainable and may vary from one dataset to another. Further, it was shown that the above detectors can be bypassed by an adversary that can adapt to the defense [46], [47]. He *et al.* further showed that combining a number of weak statistical detectors together also does not lead to a strong detector as an attacker can successfully break an ensemble detector [48].

4 DEFENDING NEURAL NETWORKS AGAINST ADVERSARIAL ATTACKS

4.1 Introduction

Signal measurements are often corrupted by noise. Let us consider the class of machine learning problems where the inputs are compressible (i.e., approximately sparse) in some domain. For instance, images and audio signals are known to be compressible in their frequency domain and machine learning algorithms have been shown to perform exceedingly well on classification tasks that take such signals as input [49], [50].

However, it was found in [51] that neural networks can be easily forced into making incorrect predictions by adding adversarial perturbations to their inputs; see also [29], [35], [52], [53]. Further, the adversarial perturbations that led to incorrect predictions were shown to be imperceptible to human beings. For this class of machine learning tasks, we show how to approximately recover original inputs from adversarial inputs and defend neural networks.

We explain the idea behind the CRD framework in the context of an image classifier. Let $x \in \mathbb{C}^n$ be a (flattened) image vector we wish to classify. However, suppose an adversary perturbs x with a noise vector $e \in \mathbb{C}^n$ such that we observe $y = x + e$, where x and e are unknown to us. Let $F \in \mathbb{C}^{n \times n}$ be the Discrete Fourier Transform (DFT) matrix. The Fourier coefficients of x are $\hat{x} = Fx$. We can therefore write the observed input y as:

$$y = F^{-1}\hat{x} + e \tag{5}$$

It is well-known that natural images are approximately sparse in the frequency domain, so we expect that \hat{x} is approximately sparse (meaning roughly that most of the entries of \hat{x} are very small). If $\|e\|_p \leq \eta$ with η small (as in a ℓ_p -attack), then we can apply an appropriate sparse recovery algorithm, with y and F^{-1} as input, to recover a good approximation $x^\#$ to \hat{x} . Since F is unitary, $F^{-1}x^\#$ will be a good approximation (i.e., reconstruction) of $x = F^{-1}\hat{x}$ as long as $x^\#$ is a good approximation to \hat{x} . If $x^\#$ is indeed a

good approximation to \hat{x} , we can feed $F^{-1}x^\#$ into the classifier and expect to get the same classification as we would have for x .

Note that the same framework can be applied with audio signals or other types of data instead of images. Moreover, the DFT can be replaced by any unitary transformation F for which $\hat{x} = Fx$ is approximately sparse. For example, F may be the Cosine Transform, Sine Transform, Hadamard Transform, or another wavelet transform.

Contributions The authors of [15] introduced this defense framework for the case of ℓ_0 -attacks with Iterative Hard Thresholding (IHT) as the recovery algorithm. We make the following novel extensions to this framework:

- We extend the framework to handle ℓ_2 and ℓ_∞ norm attacks. introduce the Modified Dantzig Selector (MDS) which uses a novel constraint to provide better recovery for ℓ_∞ norm attacks.
- We extend the ℓ_0 -norm defense introduced in [15] to defend against a larger adversarial noise budget.

4.2 Related Work

The authors of [15] introduced the CRD framework which inspired this work. They utilized Iterative Hard Thresholding (IHT) as a recovery algorithm and provided guarantees for ℓ_0 norm attacks. We extend this framework to handle ℓ_2 and ℓ_∞ attacks and a larger attack budget for ℓ_0 norm attacks. We explain this in more detail in Section 4.3.1.

Other works that provide guarantees against adversarial inputs include [54] and [55] where the authors regularize the Lipschitz constant of a network and lower bound the perturbation required to change the classifier decision. The authors of [56] use robust optimization to train the network on adversarial inputs. A similar approach to [56] is [57] in which the authors use robust optimization to lower bound the adversarial perturbations on the training data required to cause misclassification. In [58], the authors use techniques from Differential Privacy [59] and augment the training procedure to improve robustness

to adversarial inputs. The authors of [60] add i.i.d. Gaussian noise to the input and provide guarantees on classifier predictions for ℓ_2 -norm bounded attack vectors.

Most defenses against adversarial inputs do not come with theoretical guarantees. Instead, a large body of research has focused on finding practical ways to improve robustness by either augmenting the training data [29], using adversarial inputs from various networks [61], or by transforming the input [62]. For instance, [63] introduces the Projected Gradient Descent (PGD) defense that uses robust optimization based adversarial training. However, the effectiveness of their approach is determined by the amount and quality of training data available and its similarity to the distribution of the test data. A transformation based approach is [64]. Here, the authors use Generative Adversarial Networks (GANs) to estimate the distribution of the training data and, during inference, use a GAN to reconstruct the input while removing adversarial noise. Other input transformation approaches include [65], where the authors randomly replace coordinates of the input vector with neighboring coordinates. Similarly, [66] use random resizing and padding to remove the effects of adversarial noise.

The field of compressive sensing was essentially initiated with the work of [67] and [2] in which the authors show rigorously how to recover sparse signals using only a small number of measurements with the choice of a random matrix. Some of the earlier work in extending compressive sensing to perform stable recovery with deterministic matrices was done by [7] and [6], where a sufficient condition for recovery was satisfaction of a restricted isometry hypothesis. [68] introduced IHT as an algorithm to recover sparse signals which was later modified in [69] to reduce the search space as long as the sparsity was structured. The standard DS algorithm was introduced in [9] in order to perform stable recovery in the presence of ℓ_∞ noise.

4.3 Compressive Recovery Defense

Since the success of CRD depends on recovering a good approximation to \hat{x} in (5), we select the recovery algorithm that provides the best recovery guarantees based on the type of noise used.¹

For ℓ_2 or ℓ_0 noise, we use Basis Pursuit (Algorithm 3) and for ℓ_∞ noise we use Dantzig Selector with a novel constraint (Algorithm 4).

To motivate the following algorithms, note sparse recovery (without noise) is the optimization $\arg \min_{z \in \mathbb{C}^N} \|z\|_0$ subject to $Az = y$. This is non-convex and NP-hard, and it remains so if $\|z\|_0$ is approximated by $\|z\|_q$ for $0 < q < 1$. Taking $q > 1$ gives a convex, P-time problem, but the solution need not be sparse. However, if $q = 1$, the problem is convex, P-time, and gives sparse solutions. (Details: [3]–[5], [18, p.55-62]).

Algorithm 3 Basis Pursuit: BP(y, A, η)

Input: $y \in \mathbb{C}^m$, where $y = A\hat{x} + e$, $A \in \mathbb{C}^{m \times N}$, and η such that $\|e\|_2 \leq \eta$

Output: $x^\# \leftarrow \arg \min_{z \in \mathbb{C}^N} \|z\|_1$ subject to $\|Az - y\|_2 \leq \eta$

For details, see p.55-62,77 in [18]). The problem becomes convex and tractable for $q = 1$ and has been shown to provide sparse solutions; see [3]–[5].

For ℓ_2 -norm noise, BP is applied with $A = F^{-1}$, a unitary matrix. As unitary matrices are isometries in ℓ_2 norm, BP provides good recovery guarantees for such matrices since they satisfy the robust null space property (Definition 11). Also, since the noise is bounded in ℓ_2 norm and since the solution to BP minimizes the error in ℓ_2 norm, BP proves to be a very good candidate for recovery.

1. An interesting follow up problem is choosing a recovery algorithm when the type of noise is not known a priori. In practice, inputs are normalized to lie within some range $[a, b]$ (for instance $[0, 1]$), thus the the attacker is still bounded in ℓ_2 norm. Thus, Algorithm 3 is a viable candidate for recovery. We leave a deeper analysis for future investigation.

For ℓ_0 -norm noise, where e is t -sparse, the approach is only slightly different. We set $A = [F^{-1}, I]$ and write

$$y = F^{-1}\hat{x}_{h(k)} + e + F^{-1}\hat{x}_{t(k)} = A[\hat{x}_{h(k)}, e]^T + F^{-1}\hat{x}_{t(k)}$$

so that $[\hat{x}_{h(k)}, e]^T$ is a (k, t) -sparse vector that we can recover using Algorithm 3. We utilize BP for ℓ_0 attacks because it provides recovery guarantees for larger values of k and t than IHT which is used in [15]. For instance, in the case of MNIST and Fashion-MNIST, using IHT would allow us to set $k = 4$ and $t = 3$, whereas BP (Theorem 10) allows us to set $k = 8$ and $t = 8$.

Algorithm 4 Modified Dantzig Selector: $\text{MDS}(y, A, \eta)$

Input: $y \in \mathbb{C}^m$, where $y = A\hat{x} + e$, $A \in \mathbb{C}^{m \times N}$, and η such that $\|e\|_\infty \leq \eta$

Output: $x^\# \leftarrow \arg \min_{z \in \mathbb{C}^N} \|z\|_1$ subject to $\|A^*(Az - y)\|_\infty \leq \sqrt{n}\eta$, $\|Az - y\|_\infty \leq \eta$

We utilize MDS for ℓ_∞ norm attacks. The standard Dantzig Selector algorithm does not have the additional constraint $\|Az - y\|_\infty \leq \eta$. MDS includes this constraint for the following reason. In our application, we set $A = F^{-1}$ and we want the reconstruction $Ax^\#$ to be close to the original image x , so that they are classified identically. Thus, we want the search space for $x^\#$ to be restricted to those $z \in \mathbb{C}^N$ such that $\|Az - x\|_\infty$ is small. Note, for any $z \in \mathbb{C}^N$, $\|Az - x\|_\infty \leq \|Az - y\|_\infty + \|x - y\|_\infty$. In an ℓ_∞ -attack, $\|x - y\|_\infty = \|e\|_\infty$ is already small. Thus it suffices to require $\|Az - y\|_\infty$ is small. We experimentally illustrate the improvement in reconstruction due to the additional constraint in Section 4.4.2 (Figure 2, Table 3).

Remarks on Reverse-Engineered Attacks. In the case of Algorithm 3 and Algorithm 4, the minimization problems can be posed as semi-definite programming problems. If solved with interior point methods, one can use random initialization of the central path parameter and add randomness to the stopping criterion. Therefore, in addition to being non-differentiable, recovery is also non-deterministic and we expect that it would be

non-trivial to create a successful reverse-engineered attack. However, we are aware that there are powerful reverse-engineered attacks designed for black-box settings [70] and for defenses relying on non-differentiability and randomness [71]. We do not investigate reverse-engineered attacks against CRD in this work, but intend to do so in future work.

4.3.1 Main Results

We now state the formal recovery guarantees based on the type of noise used by an attacker, i.e. ℓ_2 , ℓ_∞ , and ℓ_0 norm bounded noise.

Theorem 8 (ℓ_2 -norm noise). *If $\|e\|_2 \leq \eta$, then for $x^\# = BP(y, F^{-1}, \eta)$, we have the error bounds*

$$\|x^\# - \hat{x}\|_1 \leq 2 \left(\|\hat{x}_{t(k)}\|_1 + 2\sqrt{k}\eta \right) \quad (6)$$

$$\|x^\# - \hat{x}\|_2 \leq \frac{2}{\sqrt{k}} \|\hat{x}_{t(k)}\|_1 + 6\eta \quad (7)$$

Theorem 9 (ℓ_∞ -norm noise). *If $\|e\|_\infty \leq \eta$, then for $x^\# = MDS(y, F^{-1}, \eta)$, we have the error bounds*

$$\|x^\# - \hat{x}\|_1 \leq 2 \left(\|\hat{x}_{t(k)}\|_1 + 2k\sqrt{n}\eta \right) \quad (8)$$

$$\|x^\# - \hat{x}\|_2 \leq \frac{2}{\sqrt{k}} \|\hat{x}_{t(k)}\|_1 + 6\sqrt{kn}\eta \quad (9)$$

To interpret these results, first note that since F is an isometry, $\|x^\# - \hat{x}\|_2 = \|Fx^\# - F\hat{x}\|_2$. Thus the results of Theorem 8 and Theorem 9 also bound the norm difference of the original image $x = F\hat{x}$ and the reconstructed image $Fx^\#$, where $x^\#$ has no sparsity guarantees. Therefore, the inequalities indicate how confident we should be that the CRD scheme will be able to recover the correct class of the original image, and thus defend the classifier from the adversarial attack.

Note that the recovery guarantees decay with the sparsity of the vector \hat{x} . Theorem 8 allows us to recover sparse vectors with error that depends on the magnitude of its

smallest coefficients $\hat{x}_{t(k)}$. Thus for approximately sparse vectors, CRD provides good recovery guarantees which consequently lead to better classifier performance on the recovered images. Theorem 9 provides similar guarantees when the noise is bounded in ℓ_∞ -norm. Observe also that the results of Theorem 9 incur a factor of \sqrt{n} in the error bounds due to the constraint $\|A^*(Az - y)\|_\infty \leq \sqrt{n}\eta$ in Algorithm 4 which is required to prove the robust null space property (refer to the proof of Theorem 9 for details). This weaker guarantee can also be expected as bounding the ℓ_∞ norm of the noise vector is a very weak constraint.

Finally, we provide a novel result that extends the work of [15] for the case of ℓ_0 norm attacks by providing guarantees for a larger attack budget (i.e. larger values of k and t) than the main theorem of [15].

Theorem 10 (ℓ_0 -norm noise). *Assume $|F_{ij}|^2 \leq \frac{c}{n}$. Define*

$$\delta_{k,t} = \sqrt{\frac{ckt}{n}}, \beta = \sqrt{\frac{\max\{k,t\}c}{n}}, \theta = \frac{\sqrt{k+t}}{(1-\delta_{k,t})}\beta, \tau = \frac{\sqrt{1+\delta_{k,t}}}{1-\delta_{k,t}}.$$

If $0 < \delta_{k,t} < 1$ and $0 < \theta < 1$, then for $x^\# = BP(y, [F^{-1}, I], \|\hat{x}_{t(k)}\|_2)$, we have the error bound

$$\|\hat{x}^\# - \hat{x}_{h(k)}\|_2 \leq \left(\frac{2\tau\sqrt{k+t}}{1-\theta} \left(1 + \frac{\beta}{1-\delta_{k,t}} \right) + 2\tau \right) \|\hat{x}_{t(k)}\|_2 \quad (10)$$

where we write $x^\# = [\hat{x}^\#, e^\#]^T \in \mathbb{C}^{2n}$ with $\hat{x}^\#, e^\# \in \mathbb{C}^n$.

4.3.2 Proofs

Definition 11. The matrix $A \in \mathbb{C}^{m \times N}$ satisfies the ℓ_q robust null space property of order s with constants $0 < \rho < 1$, $\tau > 0$ and norm $\|\cdot\|$ if for every set $S \subseteq [N]$ with $\text{card}(S) \leq s$ and for every $v \in \mathbb{C}^N$ we have

$$\|v_S\|_q \leq \frac{1}{s^{1-1/q}} \rho \|v_{\bar{S}}\|_1 + \tau \|Av\|$$

Note that if $q = 1$ then this is simply the robust null space property.

We now focus on proving Theorem 8. In order to do so, we will need some lemmas that will be used in the main proof.

Lemma 12. *If a matrix $A \in \mathbb{C}^{m \times N}$ satisfies the ℓ_2 robust null space property for $S \subset [N]$, with $\text{card}(S) = s$, then it satisfies the ℓ_1 robust null space property for S with constants $0 < \rho < 1, \tau' := \tau\sqrt{s} > 0$.*

Proof. For any $v \in \mathbb{C}^N$, $\|v_S\|_2 \leq \frac{\rho}{\sqrt{s}}\|v_{\bar{S}}\|_1 + \tau\|Av\|$. Then, using the fact that $\|v_S\|_1 \leq \sqrt{s}\|v_S\|_2$, we get: $\|v_S\|_1 \leq \rho\|v_{\bar{S}}\|_1 + \tau\sqrt{s}\|Av\|$.

□

Lemma 13 (Theorem 4.20 in [18]). *A matrix $A \in \mathbb{C}^{m \times N}$ satisfies the ℓ_1 robust null space property with constants $0 < \rho < 1$ and $\tau > 0$ relative to $S \subset [N]$ if and only if:*

$$\|z - x\|_1 \leq \frac{1 + \rho}{1 - \rho}(\|z\|_1 - \|x\|_1 + 2\|x_{\bar{S}}\|_1) + \frac{2\tau}{1 - \rho}\|A(z - x)\|$$

for all $z, x \in \mathbb{C}^N$.

Lemma 14 (Proposition 2.3 in [18]). *For any $q > p > 0$ and $x \in \mathbb{C}^n$,*

$$\sigma_s(x)_q \leq \frac{1}{s^{\frac{1}{p} - \frac{1}{q}}}\|x\|_p$$

Proof of Theorem 8. Let $0 < \rho < 1$ be arbitrary. Since F^{-1} is a unitary matrix, for any $S \subseteq [n]$ and $v \in \mathbb{C}^n$, we have

$$\|v_S\|_2 \leq \frac{\rho}{\sqrt{k}}\|v_{\bar{S}}\|_1 + \tau\|v\|_2 = \frac{\rho}{\sqrt{k}}\|v_{\bar{S}}\|_1 + \tau\|F^{-1}v\|_2 \quad (11)$$

where $\tau = 1$. Now let $S \subseteq [n]$ such that $\text{card}(S) \leq k$. Then, F^{-1} satisfies the ℓ_2 robust null space property for S . Next, using Lemma 12 we get $\|v_S\|_1 \leq \rho\|v_{\bar{S}}\|_1 + \tau\sqrt{k}\|F^{-1}v\|_2$ for all $v \in \mathbb{C}^n$. Now let $x^\# = \text{BP}(y, F^{-1}, \eta)$. Then we know $\|x^\#\|_1 \leq \|\hat{x}\|_1$. So, by fixing $S \subseteq [n]$ to be the support of $\hat{x}_{h(k)}$ and using Lemma 13 and the fact that $\|F^{-1}(x^\# - \hat{x})\|_2 \leq 2\|e\|_2 \leq 2\eta$, we get:

$$\begin{aligned}
\|x^\# - \hat{x}\|_1 &\leq \frac{1+\rho}{1-\rho} (\|x^\#\|_1 - \|\hat{x}\|_1 + 2\|\hat{x}_{t(k)}\|_1) \\
&\quad + \frac{2\tau\sqrt{k}}{1-\rho} \|F^{-1}(x^\# - \hat{x})\|_2 \\
&\leq \frac{1+\rho}{1-\rho} (2\|\hat{x}_{t(k)}\|_1) + \frac{4\tau\sqrt{k}}{1-\rho} \eta
\end{aligned}$$

Letting $\rho \rightarrow 0$ and recalling that $\tau = 1$ gives (6). Now let S be the support of $(x^\# - \hat{x})_{h(k)}$. Note $\|(x^\# - \hat{x})_{\bar{S}}\|_2 = \sigma_k((x^\# - \hat{x}))_2$. Then, using Lemma 14 and (11), we see that

$$\begin{aligned}
\|x^\# - \hat{x}\|_2 &\leq \|(x^\# - \hat{x})_{\bar{S}}\|_2 + \|(x^\# - \hat{x})_S\|_2 \\
&\leq \frac{1}{\sqrt{k}} \|(x^\# - \hat{x})\|_1 + \frac{\rho}{\sqrt{k}} \|(x^\# - \hat{x})_{\bar{S}}\|_1 \\
&\quad + \tau \|F^{-1}(x^\# - x)\|_2 \\
&\leq \frac{1+\rho}{\sqrt{k}} \|(x^\# - \hat{x})\|_1 + 2\tau\eta \\
&\leq \frac{(1+\rho)^2}{\sqrt{k}(1-\rho)} (2\|\hat{x}_{t(k)}\|_1) + \frac{4\tau(1+\rho)}{(1-\rho)} \eta + 2\tau\eta
\end{aligned}$$

Recalling $\tau = 1$ and letting $\rho \rightarrow 0$ gives the desired result. \square

Proof of Theorem 9. The proof follows the same structure as the proof of Theorem 8. Therefore we provide a sketch and leave out the complete derivation. Let $0 < \rho < 1$ be arbitrary. Since F^{-1} is a unitary matrix, for any $S \subseteq [n]$ and $v \in \mathbb{C}^n$, we have

$$\|v_S\|_2 \leq \frac{\rho}{\sqrt{k}} \|v_{\bar{S}}\|_1 + \|v_S\|_2 \leq \frac{\rho}{\sqrt{k}} \|v_{\bar{S}}\|_1 + \sqrt{k} \|v\|_\infty$$

The rest of the argument is the same as in the proof of Theorem 8. \square

We will establish the restricted isometry property for certain structured matrices. First, we give some definitions.

Definition 15. Let A be a matrix in $\mathbb{C}^{m \times N}$, let $M \subseteq \mathbb{C}^N$, and let $\delta \geq 0$. We say that A satisfies the M -restricted isometry property (or M-RIP) with constant δ if

$$(1 - \delta)\|x\|_2^2 \leq \|Ax\|_2^2 \leq (1 + \delta)\|x\|_2^2$$

for all $x \in M$.

Definition 16. We define M_k to be the set of all k -sparse vectors in \mathbb{C}^N and similarly define $M_{k,t}$ to be the set of (k, t) -sparse vectors in \mathbb{C}^{2n} . In other words,

$$M_{k,t} = \{x = [x_1 \ x_2]^T \in \mathbb{C}^{2n} : \\ x_1 \in \mathbb{C}^n, x_2 \in \mathbb{C}^n, \|x_1\|_0 \leq k, \|x_2\|_0 \leq t\}.$$

We define $S_{k,t}$ to be the following collection of subsets of $\{1, \dots, 2n\}$:

$$S_{k,t} = \{S_1 \cup S_2 : S_1 \subseteq \{1, \dots, n\}, S_2 \subseteq \{n+1, \dots, 2n\}, \\ \text{card}(S_1) \leq k, \text{card}(S_2) \leq t\}$$

Note that $S_{k,t}$ is the collection of supports of vectors in $M_{k,t}$.

Theorem 17. Let $A = [F \ I] \in \mathbb{C}^{n \times 2n}$, where $F \in \mathbb{C}^{n \times n}$ is a unitary matrix with $|F_{ij}|^2 \leq \frac{c}{n}$ and $I \in \mathbb{C}^{n \times n}$ is the identity matrix. Then

$$\left(1 - \sqrt{\frac{ckt}{n}}\right) \|x\|_2^2 \leq \|Ax\|_2^2 \leq \left(1 + \sqrt{\frac{ckt}{n}}\right) \|x\|_2^2 \quad (12)$$

for all $x \in M_{k,t}$. In other words, A satisfies the $M_{k,t}$ -RIP property with constant $\sqrt{\frac{ckt}{n}}$.

Proof. In this proof, if B denotes a matrix in $\mathbb{C}^{n \times n}$, then $\lambda_1(B), \dots, \lambda_n(B)$ denote the eigenvalues of B ordered so that $|\lambda_1(B)| \leq \dots \leq |\lambda_n(B)|$. It suffices to fix an $S = S_1 \cup S_2 \in S_{k,t}$ and prove (12) for all non-zero $x \in \mathbb{C}^S$.

Since $A_S^*A_S$ is normal, there is an orthonormal basis of eigenvectors u_1, \dots, u_{k+t} for $A_S^*A_S$, where u_i corresponds to the eigenvalue $\lambda_i(A_S^*A_S)$. For any non-zero $x \in \mathbb{C}^S$, we have $x = \sum_{i=1}^{k+t} c_i u_i$ for some $c_i \in \mathbb{C}$, so

$$\frac{\|Ax\|_2^2}{\|x\|_2^2} = \frac{\langle A_S^*A_S x, x \rangle}{\langle x, x \rangle} = \frac{\sum_{i=1}^{k+t} \lambda_i(A_S^*A_S) c_i^2}{\sum_{i=1}^{k+t} c_i^2}. \quad (13)$$

Thus it will suffice to prove that $|\lambda_i(A_S^*A_S) - 1| \leq \sqrt{\frac{ckt}{n}}$ for all i . Moreover,

$$\begin{aligned} |\lambda_i(A_S^*A_S) - 1| &= |\lambda_i(A_S^*A_S - I)| \\ &= \sqrt{\lambda_i((A_S^*A_S - I)^*(A_S^*A_S - I))} \end{aligned} \quad (14)$$

where the last equality holds because $A_S^*A_S - I$ is normal. By combining (13) and (14), we see that (12) will hold upon showing that the eigenvalues of $(A_S^*A_S - I)^*(A_S^*A_S - I)$ are bounded by ckt/n .

So far we have not used the structure of A , but now we must. Observe that $(A_S^*A_S - I)^*(A_S^*A_S - I)$ is a block diagonal matrix with two diagonal blocks of the form X^*X and XX^* . Therefore the three matrices $(A_S^*A_S - I)^*(A_S^*A_S - I)$, X^*X , and XX^* have the same non-zero eigenvalues. Moreover, X is simply the matrix F_{S_1} with those rows not indexed by S_2 deleted. The hypotheses on F imply that the entries of X^*X satisfy $|(X^*X)_{ij}| \leq \frac{ct}{n}$. So the Gershgorin disc theorem implies that each eigenvalue λ of X^*X and (hence) of $(A_S^*A_S - I)^*(A_S^*A_S - I)$ satisfies $|\lambda| \leq \frac{ckt}{n}$.

□

Lemma 18. *Let $A \in \mathbb{C}^{n \times 2n}$, if $|\|Ax\|_2^2 - \|x\|_2^2| \leq \delta \|x\|_2^2$ for all $x \in M_{k,t}$, then,*

*$\|A_S^*A_S - I\|_{2 \rightarrow 2} \leq \delta$, for any $S \in S_{k,t}$.*

Proof. Let $S \in S_{k,t}$ be given. Then we have :

$\|A_S x\|_2^2 - \|x\|_2^2 = \langle A_S x, A_S x \rangle - \langle x, x \rangle = \langle (A_S^*A_S - I)x, x \rangle$. Noting that $A_S^*A_S - I$ is

Hermitian, we have:

$$\|A_S^* A_S - I\|_{2 \rightarrow 2} = \max_{x \in \mathbb{C}^S \setminus \{0\}} \left| \frac{\langle (A_S^* A_S - I)x, x \rangle}{\|x\|_2^2} \right| \leq \delta$$

□

Lemma 19 (Proposition A.15 in [18]). *Let $A \in \mathbb{C}^{m \times n}$, with $m \geq n$. If $\|A^* A - I\|_{2 \rightarrow 2} \leq \delta$ for some $\delta \in [0, 1]$, then the largest and smallest singular values of A satisfy*

$$\sigma_{\max}(A) \leq \sqrt{1 + \delta}, \text{ and } \sigma_{\min}(A) \geq \sqrt{1 - \delta}$$

Lemma 20 (Lemma A.12 in [18]). *Suppose that $B \in \mathbb{C}^{n \times n}$ satisfies $\|B - I\|_{2 \rightarrow 2} \leq \eta$ for some $\eta \in [0, 1)$. Then B is invertible and $\|B^{-1}\|_{2 \rightarrow 2} \leq (1 - \eta)^{-1}$.*

Proof of Theorem 10. We will derive (10) by showing that the matrix A satisfies all the hypotheses in Theorem 4.33 in [18] for every vector in $M_{k,t}$.

First note that by Theorem 17, A satisfies the $M_{k,t}$ -RIP property with constant $\delta_{k,t} := \sqrt{\frac{ckt}{n}}$. Therefore, by Lemma 18, for any $S \in S_{k,t}$, we have $\|A_S^* A_S - I\|_{2 \rightarrow 2} \leq \delta_{k,t}$. Since $A_S^* A_S$ is a positive semi-definite matrix, it has only non-negative eigenvalues that lie in the range $[1 - \delta_{k,t}, 1 + \delta_{k,t}]$. Since $\delta_{k,t} < 1$ by assumption, $A_S^* A_S$ is injective. Thus, we can set: $h = A_S(A_S^* A_S)^{-1} \text{sgn}(x_S)$ and get:

$$\begin{aligned} \|h\|_2 &= \|A_S(A_S^* A_S)^{-1} \text{sgn}(x_S)\|_2 \\ &\leq \|A_S\|_{2 \rightarrow 2} \|(A_S^* A_S)^{-1}\|_{2 \rightarrow 2} \|\text{sgn}(x_S)\|_2 \leq \tau \sqrt{k+t} \end{aligned}$$

where $\tau = \frac{\sqrt{1 + \delta_{k,t}}}{1 - \delta_{k,t}}$ and we have used the following facts: since

$\|A_S^* A_S - I\|_{2 \rightarrow 2} \leq \delta_{k,t} < 1$, lemma 20 gives us $\|(A_S^* A_S)^{-1}\|_{2 \rightarrow 2} \leq \frac{1}{1 - \delta_{k,t}}$ and using lemma 19 we have largest singular value of A_S^* is less than $\sqrt{1 + \delta_{k,t}}$. Now let $u = A^* h$, then $\|u_S - \text{sgn}(x_S)\|_2 = 0$. Now we need to bound the value $\|u_{\bar{S}}\|_\infty$. Denoting row j of $A_{\bar{S}}^* A_S$ by the vector v_j , we see that it has at most $\max\{k, t\}$ non-zero entries and that $|(v_j)_t|^2 \leq \frac{c}{n}$

for $l = 1, \dots, (k+t)$. Therefore, for any element $(u_{\bar{S}})_j$, we have:

$$\begin{aligned} |(u_{\bar{S}})_j| &= |\langle (A_S^* A_S)^{-1} \text{sgn}(x_S), (v_j)^* \rangle| \\ &\leq \|(A_S^* A_S)^{-1}\|_{2 \rightarrow 2} \|\text{sgn}(x_S)\|_2 \|v_j\|_2 \\ &\leq \frac{\sqrt{k+t}}{1 - \delta_{k,t}} \sqrt{\frac{\max\{k, t\}c}{n}} \end{aligned}$$

Defining $\beta := \sqrt{\frac{\max\{k, t\}c}{n}}$ and $\theta := \frac{\sqrt{k+t}}{1 - \delta_{k,t}} \beta$, we get $\|u_{\bar{S}}\|_\infty \leq \theta < 1$ and also observe that $\max_{l \in \bar{S}} \|A_S^* a_l\|_2 \leq \beta$. Therefore, all the hypotheses of Theorem 4.33 in [18] have been satisfied. Note that $y = F\hat{x} + e = A[\hat{x}_{h(k)} \ e]^T + F\hat{x}_{t(k)}$, Therefore, setting $x^\# = \text{BP}(y, A, \|\hat{x}_{t(k)}\|_2)$, we use the fact $\|F\hat{x}_{t(k)}\|_2 = \|\hat{x}_{t(k)}\|_2$ combined with the bound in Theorem 4.33 in [18] to get (10):

$$\|x^\# - \hat{x}_{h(k)}\|_2 \leq \left(\frac{2\tau\sqrt{k+t}}{1 - \theta} \left(1 + \frac{\beta}{1 - \delta_{k,t}} \right) + 2\tau \right) \|\hat{x}_{t(k)}\|_2$$

where we write $x^\# = [\hat{x}^\#, e^\#]^T$ with $\hat{x}^\#, e^\# \in \mathbb{C}^n$. □

4.4 Experiments

All of our experiments are conducted on CIFAR-10 [72], MNIST [73], and Fashion-MNIST [74] datasets with pixel values of each image normalized to lie in $[0, 1]$. Each experiment is conducted on a set of 1000 points sampled uniformly at random from the test set of the respective dataset.

For every experiment, we use the Discrete Cosine Transform (DCT) and the Inverse Discrete Cosine Transform (IDCT) denoted by the matrices $F \in \mathbb{R}^{n \times n}$ and $F^T \in \mathbb{R}^{n \times n}$ respectively. That is, for an adversarial image $y \in \mathbb{R}^{\sqrt{n} \times \sqrt{n}}$, such that, $y = x + e$, we let $\hat{x} = Fx$, and $x = F^T \hat{x}$, where $x, \hat{x} \in \mathbb{R}^n$ and $e \in \mathbb{R}^n$ is the noise vector. For an adversarial image $y \in \mathbb{R}^{\sqrt{n} \times \sqrt{n} \times c}$, that contains c channels, we perform recovery on each channel independently by considering $y_m = x_m + e_m$, where $\hat{x}_m = Fx_m, x_m = F^T \hat{x}_m$ for $m = 1, \dots, c$. The value k denotes the number of largest (in absolute value) DCT

coefficients used for reconstruction of each channel. We set $k = 40$ for MNIST and F-MNIST and $k = 500$ for CIFAR-10. We implement Algorithm 3 and Algorithm 4 using the open source library CVXPY [75].

For CIFAR-10, we use the network architecture of [76] while the network architecture for MNIST and Fashion-MNIST datasets is provided in Table 1. We train our networks using the Adam optimizer for CIFAR-10 and the AdaDelta optimizer for MNIST and Fashion-MNIST. In both cases, we use a cross-entropy loss function.

We now describe the training and testing procedure for CRD. For each training image x , we compute $\hat{x}_{h(k)} = (Fx)_{h(k)}$, and then compute the compressed the image $x' = F^{-1}\hat{x}_{h(k)}$. We then add both x and x' to the training set and train the network in the usual way. Given a (potentially adversarial) test image y , we first use a sparse recovery algorithm to compute an approximation $x^\#$ to \hat{x} , then we compute the reconstructed image $y' = F^{-1}x^\#$ and feed it into the network for classification. The code to reproduce our experiments is available [here](#).

4.4.1 Defense Against ℓ_2 -norm Attacks

We use the CW ℓ_2 -norm attack [53] and the Deepfool attack [77] as they are widely considered state of the art. We note that Theorem 8 does not impose any restrictions on k and therefore the guarantees of equations (6) and (7) are applicable for recovery in all experiments of this section.

Table 1: Network Architecture: MNIST and Fashion-MNIST. The first four layers use ReLU activations while the last layer uses a softmax activation.

Layer	Type	Properties
1	Convolution	32 channels, 3×3 Kernel, No padding
2	Convolution	64 channels, 3×3 Kernel, No padding, Dropout with $p = 0.5$
3	Max-pooling	2×2 , Dropout with $p = 0.5$
4	Fully Connected	128 neurons, Dropout with $p = 0.5$
5	Fully Connected	10 neurons

We test the performance of CRD in two ways: a) reconstruction quality, and b) network performance on reconstructed images. To analyze reconstruction quality of Algorithm 3, for each test image, we first create an adversarial image and then use Algorithm 3 to recover its largest k coefficients. We then perform the IDCT on these recovered co-efficients to generate reconstructed images. We illustrate reconstruction on a randomly selected image from the test set in Figure 1.

In order to check whether this high quality reconstruction also leads to improved performance in network accuracy, we test each network on reconstructed images using Algorithm 3. We report the results in Table 2 and note that Algorithm 3 provides a substantial improvement in network accuracy for each dataset and each attack method used. For comparison, we implement the PGD defense framework of [63] for MNIST and Fashion-MNIST and see that CRD outperforms PGD. Due to time and resource constraints we do not report PGD results for CIFAR-10 as we were unable to get the network to converge to an accuracy over 70% for non-adversarial test samples. Since PGD is computationally very expensive [78], [79] and minimizes network loss on adversarial samples, training a network that performs well on adversarial and

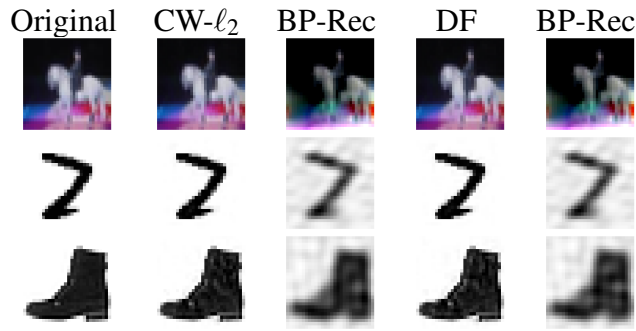


Fig. 1. Reconstruction quality: Algorithm 3 for ℓ_2 attacks. The first column shows the original images, while the adversarial images are shown in the second and fourth column. The reconstructions are shown in columns three and five.

Table 2: Network Performance: Algorithm 3 for ℓ_2 attacks. The $\ell_{2\text{avg}}$ column lists the average ℓ_2 -norm of the attack vector. The Orig. Acc column lists the accuracy of the network on original test inputs, while the Acc. columns under C&W ℓ_2 and DF columns report network accuracy on adversarial inputs. BP Acc. columns lists the accuracy of the network on inputs reconstructed using Algorithm 3. PGD Acc. shows accuracy of the defense in [63].

Data	Orig. Acc.	C&W ℓ_2				Deepfool			
		$\ell_{2\text{avg}}$	Acc.	BP Acc.	PGD Acc.	$\ell_{2\text{avg}}$	Acc.	BP Acc.	PGD Acc.
Cifar10	84.9%	0.12	8.7%	72.3%	-	0.11	7.7%	71.6%	-
Mnist	99.17%	1.35	0.9%	92.4%	83.7%	1.72	1.1	90.7%	6.4%
FMnist	90.3%	0.61	5.4%	78.3%	75.9%	0.63	5.5 %	76.4%	25.6%

non-adversarial samples is highly non-trivial. We note that CRD does not suffer from either of these drawbacks as network training is decoupled from the CRD defense.

4.4.2 Defense Against ℓ_∞ -norm Attacks

For ℓ_∞ -norm bounded attacks, we use the BIM attack [80] as it is state of the art and allows us to control the ℓ_∞ -norm of the attack vector explicitly.²

Therefore, we limit our experimental analysis to the BIM attack. Note that for any attack vector e , $\|e\|_2 \leq \sqrt{n}\|e\|_\infty$ hence allowing ℓ_∞ -norm attacks to create attack vectors with large ℓ_2 -norm. Therefore, we could expect reconstruction quality and network accuracy to be lower when compared to ℓ_2 -norm attacks.

In Figure 2, we compare the reconstruction quality of images reconstructed with Algorithm 4 to those reconstructed using DS without the additional constraint. It can be seen that images reconstructed using DS without the additional constraint may not produce meaningful images. This is also reflected in Table 3, which shows that the accuracy of the network is roughly random on images reconstructed without the additional constraint. We show examples of original images, adversarial images, and their reconstructions using Algorithm 4 in Figure 3. Finally, we report the network

2. We note that while the CW ℓ_∞ -norm attack [53] has the ability to create attack vectors with ℓ_∞ -norm less than or equal to BIM, it is computationally expensive and also does not allow one to pre-specify a value for the ℓ_∞ -norm of an attack vector.

performance on reconstructed inputs using Algorithm 4 in Table 3 and also compare this to the performance on inputs reconstructed using DS without the additional constraint. We also report the results of the PGD defense of [63] and note that PGD outperforms CRD against the BIM attack. This can be expected as PGD training uses a very similar method to BIM in construction adversarial examples used for training.

4.4.3 Defense Against ℓ_0 -norm Attacks

We test CRD against the CW ℓ_0 -norm attack and JSMA. We find that even when t is much larger than the hypotheses of Theorem 10, we find that Algorithm 3 is still able to defend the network. We hypothesize that this may be related to the behavior of the RIP of a matrix for “most” vectors as opposed to the RIP for all vectors, and leave a more rigorous analysis for a follow up work.

Fig 4 shows the reconstruction quality of the images and the improvement in network performance on reconstructed adversarial images using CRD is reported in Table 4. It can also be seen that CRD outperforms PGD for both ℓ_0 attacks.

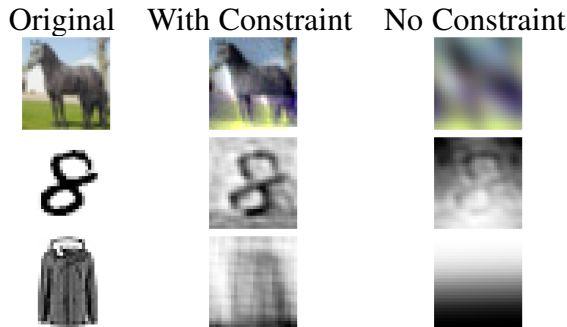


Fig. 2. Reconstruction quality: Algorithm 4 with and without constraint. Comparison of images reconstructed using Algorithm 4 (With Constraint) with images reconstructed using DS without the additional constraint (No Constraint).

Table 3: Network Performance: Algorithm 4 for ℓ_∞ attacks. The $\ell_{\infty\text{avg}}$ column lists the ℓ_∞ -norm of each attack vector, Orig. Acc. and BIM Acc. columns list the accuracy of the network on the original and adversarial inputs respectively, and the MDS Acc. column lists the accuracy of the network on inputs reconstructed using Algorithm 4. We also show accuracy of the network on images reconstructed with DS (without the additional constraint) in the DS Acc. column. PGD Acc. shows accuracy of the defense in [63].

Data	Orig. Acc.	BIM				
		$\ell_{\infty\text{avg}}$	Acc.	MDS Acc.	DS Acc.	PGD Acc.
Cifar10	84.9%	0.015	7.4%	49.4%	17.6%	-
Mnist	99.17%	0.15	4.9%	74.7%	10%	95.8%
FMnist	90.3%	0.15	5.3%	57.5%	11.1%	77.3%

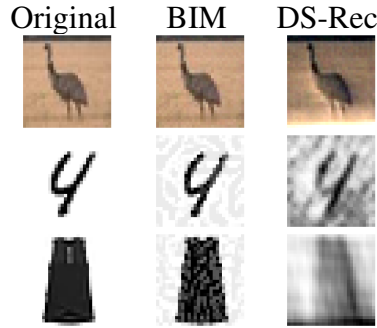


Fig. 3. Reconstruction quality: Algorithm 4 for ℓ_0 attacks. The first column shows the original images, while the second columns shows adversarial images and the third columns shows reconstructions using Algorithm 4 respectively.

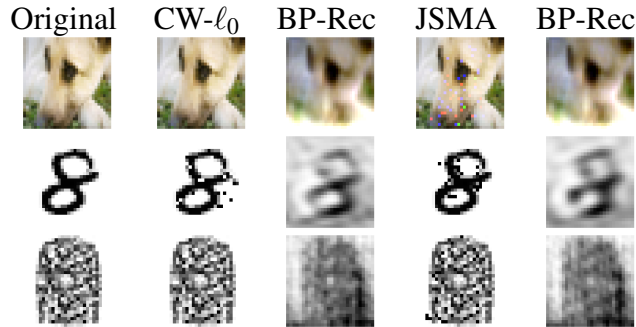


Fig. 4. Reconstruction quality: Algorithm 3 for ℓ_0 attacks. The first column shows randomly selected original images from the test set, while the second and fourth column show the adversarial images. Reconstructions using BP are labeled BP-Rec. We show reconstructions in columns three, and five.

Table 4: Network Performance: Algorithm 3 for ℓ_0 attacks. The t_{avg} column lists the average adversarial noise budget for each attack. The Orig. Acc column lists the accuracy of the network on original test inputs, the Acc. columns under C&W ℓ_0 and JSMA list network accuracy on adversarial inputs. The BP Acc. column lists the accuracy of the network on inputs that have been corrected using BP.PGD Acc. shows accuracy of the defense in [63].

Data	Orig. Acc.	C&W ℓ_0				JSMA			
		t_{avg}	Acc.	BP Acc.	PGD Acc.	t_{avg}	Acc.	BP Acc.	PGD Acc.
Cifar10	84.9%	18	8.7%	67.0%	-	34	2.7%	67.3%	-
Mnist	98.8%	15	0.9%	55.9%	14.1%	17	56.5 %	67.4%	93.5%
FMnist	91.8%	16	5.27%	71.4%	75.1%	17	62.6 %	72.0%	76.6%

5 NEURAL NETWORKS AS PRIORS IN LINEAR INVERSE PROBLEMS

5.1 Introduction

Linear inverse problems can be formulated mathematically as $y = Ax + e$ where $y \in \mathbb{R}^m$ is the observed vector, $A \in \mathbb{R}^{m \times N}$ is the measurement process, $e \in \mathbb{R}^m$ is a noise vector, and $x \in \mathbb{R}^N$ is the original signal. The problem is to recover the signal x , given the observation y and the measurement matrix A . Such problems arise naturally in a wide variety of fields including image processing, seismic and medical tomography, geophysics, and magnetic resonance imaging. In this thesis, we focus on three linear inverse problems encountered in image processing: compressive sensing, inpainting, and super-resolution. We motivate our method using the compressive sensing problem.

Sparsity Prior The problem of compressive sensing assumes the matrix $A \in \mathbb{R}^{m \times N}$ is fat, i.e. $m < N$. Even when no noise is present ($y = Ax$), the system is under determined and the recovery problem is intractable. However, it has been shown that if the matrix A satisfies certain conditions such as the Restricted Isometry Property (RIP) and if x is known to be approximately sparse in some fixed basis, then x can typically be recovered even when $m \ll N$ [2], [6], [81].

Sparsity (or approximate sparsity) is a very restrictive condition to impose on the signal as it limits the applicability of recovery methods to a small subset of input domains. There has been considerable effort in using other forms of structured priors such as structured sparsity [69], sparsity in tree-structured dictionaries [82], and low-rank mixture of Gaussians [83]. Although these efforts improve on the sparsity prior, they do not cater to signals that are not naturally sparse or structured-sparse.

Generative Prior Bora *et al* [84] address this issue by replacing the sparsity prior on x with a generative prior. In particular, the authors first train a generative model $f : \mathbb{R}^k \mapsto \mathbb{R}^N$ with $k < N$ that maps a lower dimensional latent space to the higher dimensional ambient space. This model is referred to as the generator. Next, they impose

the prior that the original signal x lies in (or near) the range of f . Hence, the recovery problem reduces to finding the best approximation to x in $f(\mathbb{R}^k)$.

The quality of the generative prior depends on how well the training set captures the data distribution. Bora *et al* [84] used a Generative Adversarial Network (GAN) as the generator, $G : \mathbb{R}^k \mapsto \mathbb{R}^N$, where $k < N$, to model the distribution of the training data and posed the following non-convex optimization problem

$$\hat{z} = \arg \min_{z \in \mathbb{R}^k} (\|AG(z) - y\|^2 + \lambda \|z\|^2).^3$$

such that $G(\hat{z})$ is treated as the approximation to x . The authors provided recovery guarantees for their methods and validated the efficacy of using generative priors by showing that their method required 5-10x fewer measurements than Lasso (with a sparsity constraint) [81] while yielding the same accuracy in recovery. However, since the problem is non-convex and requires a search over \mathbb{R}^k , it is computationally expensive and the reconstruction quality depends on the initialization vector $z \in \mathbb{R}^k$.

Since then, there have been significant efforts to improve recovery results using neural networks as generative priors [85]–[96]. Shah *et al* [93] extended the work of [84] by training a generator G and using a projected gradient descent algorithm that consists of a gradient descent step $w_t = x_t - \eta A^T(Ax_t - y)$ followed by a projection step $x_{t+1} = G(\arg \min_{z \in \mathbb{R}^k} \|G(z) - w_t\|^2)$. The core idea being that the estimate w_t is improved by projecting it onto the range of G . However, since their method requires solving a non-convex optimization problem at every update step, it also leads to slow recovery.

Raj *et al* [95] enhanced the results of [93] by eliminating the expensive non-convex optimization based projection step with one that is an order of magnitude cheaper. In particular, they trained a GAN G to model the data distribution and also trained a pseudo-inverse GAN G^\ddagger that learned a mapping from the ambient space to the latent space. Next, they used the projection step: $x_{t+1} = G(G^\ddagger(w_t))$. By eliminating the need to

3. We use $\|\cdot\|$ to denote the ℓ_2 -norm throughout the thesis

solve a non-convex optimization problem to update x_{t+1} , they were able to attain a significant speed up in the running time of the recovery algorithm.

However, the recovery algorithm of [95] has two main drawbacks. First, training two networks: G and G^\dagger makes the training process and the projection step unnecessarily convoluted. Second, their recovery guarantees only hold when the learning rate $\eta = \frac{1}{\beta}$, where β is a RIP-style constant of the matrix A . Since it is NP-hard to estimate the constant β [97], it follows that setting $\eta = \frac{1}{\beta}$ is NP-hard as well.⁴

DAE Prior In an effort to address the aforementioned issues, we propose to use a DAE [98] prior in lieu of the generative prior introduced by Bora *et al* [84]. It has previously been shown that DAEs not only capture useful structure of the data distribution [99] but also implicitly capture properties of the data-generating density [100], [101]. Moreover, as DAEs are trained to remove noise from vectors sampled from the input distribution, they integrate naturally with gradient descent algorithms that lead to noisy approximations at each time step.

We replace the generator G used in Bora *et al* [84] with a DAE $F : \mathbb{R}^N \mapsto \mathbb{R}^N$ such that the range of F contains the vectors from the original data generating distribution. We then impose the prior that the original signal x lies in the range of F and utilize Algorithm 5 to recover an approximation to x . We provide theoretical recovery guarantees and find that our framework speeds up recovery by two orders of magnitude (over 100x), improves quality of reconstruction by an order of magnitude (over 10x), and does not require tuning hyperparameters.

5.2 Related Work

Generative Priors Following the lead of [84], there have been significant efforts to improve on previous recovery results using neural networks as generative

4. We observed this problem when trying to reproduce the experimental results of [95]. Specifically, we tried an exhaustive grid-search for η but each value led to poor reconstruction quality.

models [85]–[96]. One line of work [96], [102] extends the efforts of Bora *et al* [84] by fixing a seed z and finding the weights \hat{w} of an untrained neural network G in the optimization problem $\hat{w} = \arg \min_{w \in \mathbb{R}^I} \|AG(w, z) - y\|^2$. However, the optimization problem is highly non-convex and requires a large number of iterations with multiple restarts. Another line of work, [91], [103] trains a neural network to model the transformation $f(y) = \hat{x}$ where \hat{x} is the approximation to the original input x . This approach is limited as a) the inverse mapping is non-trivial to learn and b) will only work for a fixed measurement mechanism.

Denoisers in Linear Inverse Problems Given the success of denoisers in image processing tasks such as image denoising [104]–[106] and image super-resolution [107] to yield good results, Venkatakrishnan *et al* [108] introduced denoisers as plug-and-play (PnP) proximal operators in solving linear inverse problems via alternating directions method of multipliers (ADMM). Ryu *et al* [109] extended this work by investigating convergence properties of ADMM methods and showed that if the denoiser was close to the identity map, then PnP methods are contractive iterations that converge with bounded error.

Chang *et al* [106] showed that neural network based denoisers (such as DAEs) with ADMM could achieve state of the art results for a wide array of linear inverse problems. They also showed that if the gradient of the proximal operator (denoiser) is Lipschitz continuous, ADMM has a fixed point. Xu *et al* [110] analyzed convergence results for minimum mean squared error (MMSE) denoisers used in iterative shrinkage/thresholding algorithm (ISTA). They showed that the iterates produced by ISTA with an MMSE denoiser converge to a stationary point of some global cost function. Meinhardt *et al* [111] demonstrated that using a fixed denoising network as a proximal operator in the primal-dual hybrid gradient (PDHG) method yields state-of-the-art results. Gonzalez *et al* [112] used variational auto encoders (VAEs) as priors defined an optimization method

JPMAP that performs Joint Posterior Maximization using an the VAE prior. They showed theoretical and experimental evidence that the proposed objective function satisfies a weak bi-convexity property which is sufficient to guarantee that the optimization scheme converges to a stationary point.

5.3 Algorithm and Results

5.3.1 Algorithm

Recall that in the linear inverse problem $y = Ax + e$, our goal is to recover an approximation \hat{x} to x such that \hat{x} lies in the range of F . Thus we aim to find \hat{x} such that $\hat{x} = \arg \min_{z \in F(\mathbb{R}^N)} \|Az - y\|^2$. As in [93], [95], we use a projected gradient descent algorithm. Given an estimate x_t at iteration t , we compute a gradient descent step for solving the unrestricted problem: $\min_{z \in \mathbb{R}^N} \|Az - y\|^2$ as: $w_t \leftarrow x_t - \eta A^T(Ax_t - y)$. Next we project w_t onto the range of F to satisfy our prior: $x_{t+1} = F(w_t)$. Note that, compared to [93], [95], the projection step does not require solving a non-convex optimization problem.

Now suppose that the domain of interest is represented by the set $D \subseteq \mathbb{R}^N$. Then, given a vector $x' = x + e$, where $x \in D$, and $e \in \mathbb{R}^N$ is an unknown noise vector, the success of our method depends on how small the error $\|F(x') - x\|$ is. If the training set X captures the domain of interest well and if the training procedure utilizes a diverse enough set of noise vectors $\{e_i\}_{i=1}^N$, then we expect $\|F(x') - x\|$ to be small. Consequently, we expect the projection step of Algorithm 5 to yield vectors in or close to D . We provide the complete algorithm below.

5.3.2 Theoretical Results

We begin by introducing two standard definitions required to provide recovery guarantees.

Algorithm 5 DAE-PGD

Input: $y \in \mathbb{R}^m, A \in \mathbb{R}^{m \times N}, f: \mathbb{R}^N \rightarrow \mathbb{R}^N, T \in \mathbb{Z}_+, \eta \in \mathbb{R}_{>0}$

Output: x_T

```
1:  $t \leftarrow 0, x_0 \leftarrow 0$ 
2: while  $t < T$  do
3:    $w_t \leftarrow x_t - \eta A^T(Ax_t - y)$ 
4:    $x_{t+1} \leftarrow f(w_t)$ 
5: return  $x_T$ 
```

Definition 21 (RIP(S, δ)). Given $S \subseteq \mathbb{R}^N$ and $\delta > 0$, a matrix $A \in \mathbb{R}^{m \times N}$ satisfies the RIP(S, δ) property if

$$(1 - \delta) \|x_1 - x_2\|^2 \leq \|A(x_1 - x_2)\|^2 \leq (1 + \delta) \|x_1 - x_2\|^2$$

for all $x_1, x_2 \in S$.

A variation of the RIP(S, δ) property for sparse vectors was first introduced by Candes *et al* in [7] and has been shown to be a sufficient condition in proving recovery guarantees using ℓ_1 -minimization methods [18]. Next, we define an Approximate Projection (AP) property and provide an interpretation that elucidates its role in the results of Theorem 23.⁵

Definition 22 (AP(S, α)). Let $\alpha \geq 0$. A mapping $f: \mathbb{R}^N \rightarrow S \subseteq \mathbb{R}^N$ satisfies AP(S, α) if

$$\|w - f(w)\|^2 \leq \|w - x\|^2 + \alpha \|f(w) - x\|$$

for every $w \in \mathbb{R}^N$ and $x \in S$.

⁵. Various flavors of the AP(S, α) property have been used in previous works, such as Shah *et al* [93] and Raj *et al* [95].

We now explain the significance of Def. 22. Let $x^* = \arg \min_{z \in S} \|w - z\|$ and observe

$$\|w - f(w)\|^2 \leq (\|w - x^*\| + \|f(w) - x^*\|)^2 \quad (15)$$

Hence, $\alpha \leq \|f(w) - x^*\| + 2\|w - x^*\|$ is needed to ensure the RHS of Def. 22 is bounded by the RHS of (15). In other words, for α to be small, the projection error $\|f(w) - x^*\|$ as well as distance of w to S need to be small. Since the DAE F learns to minimize $\|F(w) - x^*\|^2$ (Section 3.2), we expect a small projection error.

Theorem 23. *Let $f : \mathbb{R}^N \rightarrow S \subseteq \mathbb{R}^N$ satisfy $AP(S, \alpha)$ and let $A \in \mathbb{R}^{m \times N}$ be a matrix with $\|A\|^2 \leq M$ that satisfies $RIP(S, \delta)$. If $y = Ax$ with $x \in S$, the recovery error of Algorithm 5 is bounded as:*

$$\|x_T - x\| \leq (2\gamma)^T \|x_0 - x\| + \alpha \left(\frac{1 - (2\gamma)^T}{1 - (2\gamma)} \right) \quad (16)$$

where $\gamma = \sqrt{\eta^2 M(1 + \delta) + 2\eta(\delta - 1) + 1}$.

Theorem 23 tells us that, if $\gamma < \frac{1}{2}$, then for large T , the recovery error is essentially $\alpha/(1 - 2\gamma)$. Note that the requirement $\gamma < \frac{1}{2}$ is satisfied for a large range of values of η as long as δ is sufficiently small.⁶

Hence, as long as the value of α is small, we expect to see a small recovery error.

We now compare the above results to Theorem 1 of [95], Theorem 2.2 of [93] and Theorem 1 of [113]. As mentioned in Section 5.1, convergence in Theorem 1 of [95] is only guaranteed when $\eta = \frac{1}{\beta}$, which is a much more restrictive condition on η than Theorem 23 provides. In fact, β is a RIP-style constant that is NP-hard to find [97] which makes setting the value of $\eta = \frac{1}{\beta}$ NP-hard as well. The results of Theorem 2.2 from [93] require a less restrictive constraint on η but do require a stricter constraint on $\|A\|^2 \leq \omega$, where ω is a RIP-style constant for A . In contrast, the results of Theorem 23 do not

6. For instance, random Gaussian matrices yield small values for δ with high probability [18]

impose a strict condition on $\|A\|^2$. Finally, the proof of Theorem 1 of [113] relies on the matrix A being Gaussian. We do not impose such a constraint.

Proof of Theorem 23. Using the notation of Algorithm 5 and the fact that f satisfies $AP(S, \alpha)$ we have

$$\|(w_t - x) - (x_{t+1} - x)\|^2 \leq \|w_t - x\|^2 + \alpha \|x_{t+1} - x\|.$$

Noting $\|a - b\|^2 = \|a\|^2 + \|b\|^2 - 2\langle a, b \rangle$ and re-arranging terms we get

$$\|x_{t+1} - x\|^2 \leq 2\langle (w_t - x), (x_{t+1} - x) \rangle + \alpha \|x_{t+1} - x\|.$$

Now we expand the inner product using $w_t = x_t - \eta A^T(Ax_t - y)$ and $y = Ax$ to get

$$\begin{aligned} \|x_{t+1} - x\|^2 &\leq 2\langle (I - \eta A^T A)(x_t - x), (x_{t+1} - x) \rangle \\ &\quad + \alpha \|x_{t+1} - x\|. \end{aligned} \tag{17}$$

Using the Cauchy–Schwarz inequality we have

$$\begin{aligned} &|\langle (I - \eta A^T A)(x_t - x), (x_{t+1} - x) \rangle| \\ &\leq \|(I - \eta A^T A)(x_t - x)\| \|x_{t+1} - x\| \end{aligned} \tag{18}$$

By setting $u = x_t - x$, expanding, and using the $\text{RIP}(S, \alpha)$ property of A , we see that

$$\begin{aligned} \|(I - \eta A^T A)u\|^2 &= \|u\|^2 - 2\eta \|Au\|^2 + \eta^2 \|A^T(Au)\|^2 \\ &\leq \|u\|^2 - 2\eta(1 - \delta)\|u\|^2 \\ &\quad + \eta^2(1 + \delta)M\|u\|^2 \\ &= \gamma^2 \|u\|^2 \end{aligned} \tag{19}$$

We substitute the results of (18) and (19) into (17) and divide both sides by $\|x_{t+1} - x\|$ to get

$$\|x_{t+1} - x\| \leq 2\gamma\|x_t - x\| + \alpha \quad (20)$$

Using induction on (20) gives (16). \square

5.4 Experiments

We provide experimental results for the problems of compressive sensing, inpainting, and super-resolution. We refer to the results of Algorithm 5 as DAE-PGD and compare its results to the methods of Bora *et al* [84] (CSGM), and Shah *et al* [93], (PGD-GAN), and Peng *et al* [113] (P-AE). Although the work of Raj *et al* [95] is the closest to our method, we do not include comparisons to their work as we were unable to reproduce their results.

7

5.4.1 Setup

Datasets Our experiments are conducted on the MNIST [73] and CelebA [114] datasets. The MNIST dataset consists of 28×28 greyscale images of digits with 50,000 training and 10,000 test samples. We report results for a random subset of the test set. The CelebA dataset consists of more than 200,000 celebrity images. We pre-processes each image to a size of $64 \times 64 \times 3$ and use the first 160, 000 images as the training set and a random subset of the remaining 40,000+ images as the test set.

Network Architecture The network architectures for our DAEs are inspired by the Variational Auto Encoder architecture from Fig 2. of [115] with a few key changes. We replace the Leaky Relu activation with Relu, we add the two outputs of the encoder to get the latent representation z , and we alter the kernel sizes as well as the convolution strides of the network as described in Table 5.

7. We used their code, their trained models, their recovery algorithm, and a grid search for η but the reconstructed images were of very poor quality. We also reached out to the authors but they did not have the exact values of η that were used in their experiments.

Table 5: Network Architecture: CelebA and MNIST. C-K, C-S, M-K, and M-S report CelebA Kernel Sizes, CelebA Strides, MNIST Kernel Sizes, and MNIST strides respectively.

Layer	C-K	C-S	M-K	M-S
Conv2D 1	9×9	2	5×5	2
Conv2D 2	7×7	2	5×5	2
Conv2D 3	5×5	2	3×3	2
Conv2D 4	5×5	1	3×3	1
TransConv2d 1	5×5	2	3×3	1
TransConv2d 2	5×5	2	3×3	2
TransConv2d 3	7×7	2	5×5	2
TransConv2d 4	9×9	1	5×5	2

Training We use the Adam optimizer [12] to minimize the MSE loss function with learning rate 0.01 and a batch size of 128. We train the CelebA network for 400 epochs and the MNIST network for 100 epochs.

In an effort to ensure that $\|A(x') - x\|$ defined in Section 5.3.1 is small, we split the training set into 5 equal sized subsets. For each distinct subset, we sample the noise vectors from a Gaussian distribution $F(\mu, \sigma^2)$ with a distinct value for σ for each subset. The five different values for σ that we use are $\{0.25, 0.5, 0.75, 1.0, 1.25\}$.

All of our experiments were conducted on a Tesla M40 GPU with 12 GB of memory using Keras [116] and Tensorflow [117] libraries. The code to reproduce our results is available [here](#).

5.4.2 Compressive Sensing

We consider the problem of compressive sensing without noise: $y = Ax$ and with noise: $y = Ax + e$, with $e \sim \mathcal{N}(0, 0.25)$. We use m to denote the number of observed measurements in our results (i.e. $y \in \mathbb{R}^m$). As done in previous works [84], [93], [95], the matrix $A \in \mathbb{R}^{m \times N}$ is chosen to be a random Gaussian matrix with $A_{ij} \sim \mathcal{N}(0, \frac{1}{m})$. Finally, we set the learning rate of Algorithm 5 as $\eta = 1$. Note that in both (with and w/out noise) cases, we also include recovery results for the Lasso algorithm [81] with a DCT basis (L-DCT) and with a wavelet basis (L-Wavelet).

We begin with CelebA. Figure 5 provides a qualitative comparison of reconstruction results for $m = 1000$. We observe that DAE-PGD provides the best quality reconstructions and is able to reproduce even fine grained details of the original images such as eyes, nose, lips, hair, texture, etc. Indeed the high quality reconstructions support the case that the DAE has a small α as per Def. 22. For a quantitative comparison, we turn to Figure 6 which plots the average squared reconstruction error $\|x - \hat{x}\|^2$ for each algorithm at different values of m . Note that DAE-PGD provides more than 10x improvement in the squared reconstruction error.

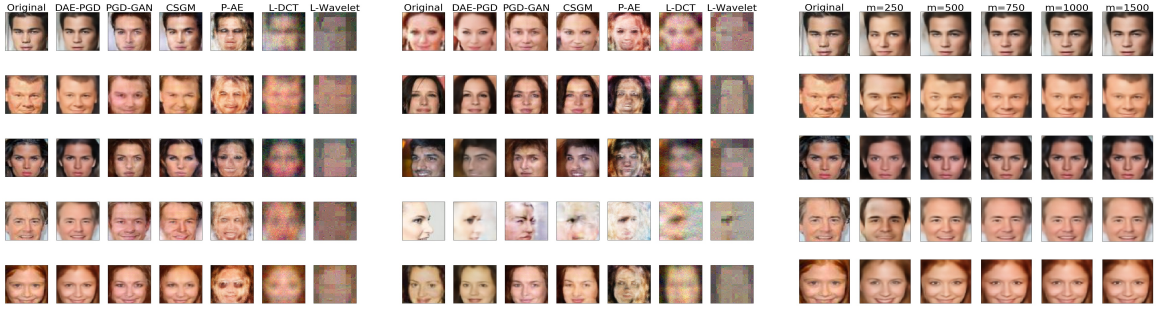


Fig. 5. Compressive Sensing: CelebA. CS on CelebA without noise for $m = 1000$ (left), CS on CelebA with noise for $m = 1000$ (middle), CS on CelebA for various m using DAE-PGD (right). The left and middle images qualitatively capture the 10x improvement in reconstruction error. The right image shows how DAE-PGD reconstructions capture finer grained details as m increases.

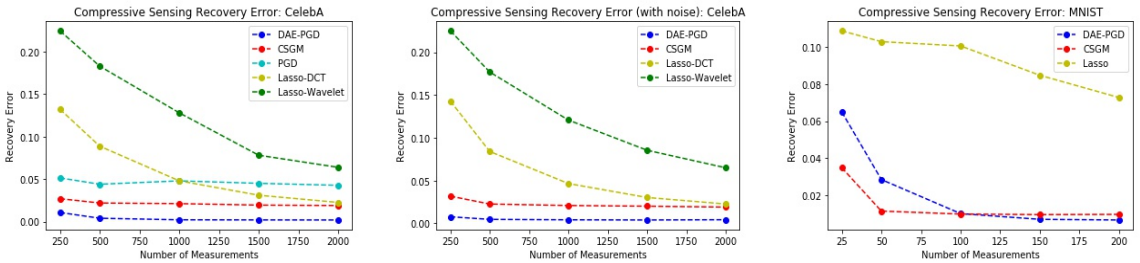


Fig. 6. Compressive Sensing: Recovery Error. Using $\|x - \hat{x}\|^2$. Left: CelebA without noise - DAE-PGD shows over 10x improvement. Middle: CelebA with noise - DAE-PGD shows over 10x improvement. Right: MNIST without noise - DAE-PGD beats CSGM for $m > 100$.

In order to capture how the quality of reconstruction degrades as the number of measurements decrease, we refer to Figure 5, which shows reconstructions for different values of m . We observe that even though reconstructions with a small number of measurements capture the essence of the original images, the fine grained details are captured only as the number of measurements increase. We show a similar comparison for MNIST in Figure 7

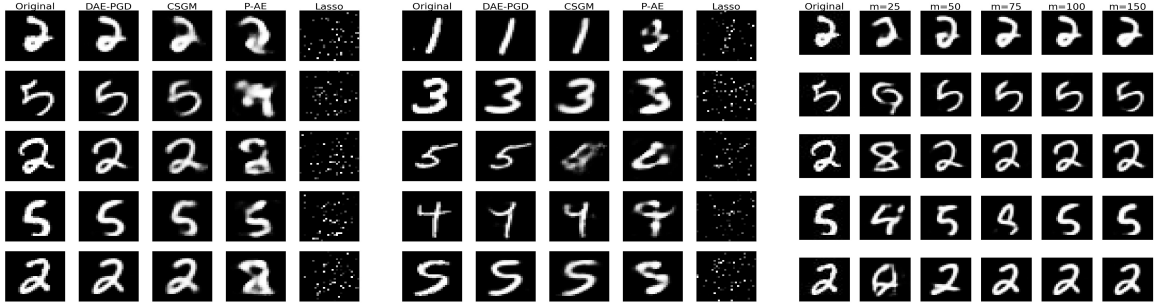


Fig. 7. Compressive Sensing: MNIST. CS on MNIST without noise for $m = 100$ (left), CS on MNIST with noise for $m = 100$ (middle), CS on CelebA for various m using DAE-PGD (right). The left and middle images qualitatively capture the 100x improvement in reconstruction error. The right image shows how DAE-PGD reconstructions capture finer grained details as m increases.

We now turn to the speed of reconstruction. Table 6 shows that our method provides speedups of over 100x as compared to PGD-GAN and CSGM.⁸

5.4.3 Inpainting

Inpainting is the problem of recovering the original image, given an occluded version of it. Specifically, the observed image y consists of occluded (or masked) regions created by applying a pixel-wise mask A to the original image x . We use m to refer to the size of mask that occludes a $m \times m$ region of the original image x .

We present recovery results for CelebA with $m = 10$ in Figure 8 and observe that DAE-PGD is able to recovery a high quality approximation to the original image and

⁸. CSGM is executed for 500 max iterations with 2 restarts and PGD-GAN is executed for 100 max iterations and 1 restart.

Table 6: DAE-PGD Runtime. Average running times (in seconds) for the Compressive Sensing problem (w/out noise) on the CelebA dataset.

m	CGSM	PGD-GAN	DAE-PGD	Speedup
250	53.78	48.40	0.07	692x
500	59.81	48.46	0.09	538x
1000	81.08	48.46	0.11	440x
1500	92.68	48.50	0.14	346x
2000	107.41	48.56	0.21	230x

outperforms CSGM in all cases. Figure 8 also captures how recovery is affected by different mask sizes. As in the compressive sensing problem, we find that DAE-PGD reconstructions capture the fine-grained details of each image. Figure 8 also reports the result for the MNIST dataset. Even though DAE-PGD outperforms CSGM, we see that the recovery quality of DAE-PGD degrades considerably when $m = 15$. We hypothesize this is due to the structure of MNIST images. In particular, since MNIST images are grayscale with most of the pixels being black, putting a 15×15 black patch on the small area displaying the number makes the reconstruction problem considerably more difficult. This causes considerable degradation in reconstruction quality for larger mask sizes.

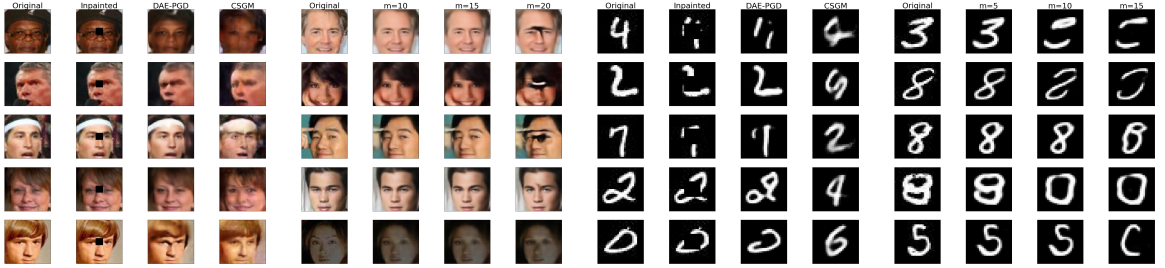


Fig. 8. Inpainting: CelebA and MNIST. Left: CelebA reconstructions for $m = 10$. Middle-Left: DAE-PGD CelebA reconstructions for different m . Middle-Right: MNIST reconstructions for $m = 5$. Right: DAE-PGD MNIST reconstructions for different m .

5.4.4 Super-resolution

Super-resolution is the problem of recovering the original image from a smaller and lower-resolution version. We create this smaller and lower-resolution image by taking the

spatial averages of $f \times f$ pixel values where f is the ratio of downsampling. This results in blurring a $f \times f$ region followed by downsampling the image. We test our algorithm with $f = 2, 3, 4$ corresponding to $4\times, 9\times$, and $16\times$ smaller image sizes, respectively.

The reconstruction results are provided in Figure 9. We see that DAE-PGD provides higher quality reconstruction for $f = 2$ for both CelebA and MNIST. Moreover, reconstruction quality degrades gracefully for CelebA for increasing values of f . However, in the case of MNIST, reconstruction quality degrades considerably when $f = 4$. Noting that $f = 4$ only gives 16 measurements (i.e. $y \in \mathbb{R}^{16}$), we hypothesize that 16 measurements may not contain enough signal to accurately reconstruct the original images.

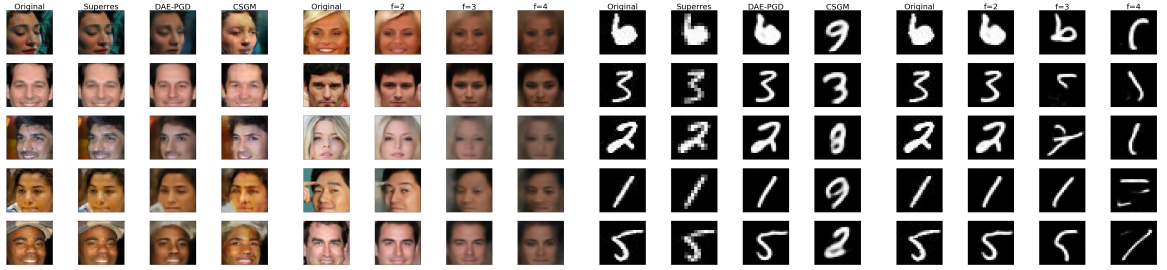


Fig. 9. Super-resolution: CelebA and MNIST. Left: CelebA reconstructions for $f = 2$. Middle-Left: DAE-PGD CelebA reconstructions for different f . Middle-Right: MNIST reconstructions for $f = 2$. Right: DAE-PGD MNIST reconstructions for different f .

6 CONCLUSIONS

6.1 Defending Neural Networks Against Adversarial Attacks

We provided recovery guarantees for corrupted signals in the case of ℓ_2, ℓ_∞ and ℓ_0 -norm bounded noise. We were able to utilize these results in CRD and improve the performance of neural networks substantially in the case of ℓ_2, ℓ_∞ and ℓ_0 norm bounded noise. In particular, we utilized the guarantees of Theorem 8 and Theorem 9 to extend the defense framework of [15] to defend neural networks against ℓ_2, ℓ_∞ and ℓ_0 norm attacks.

6.2 Neural Networks as Priors in Linear Inverse Problems

We utilized DAEs as priors for general linear inverse problems and provided experimental results for the problems of compressive sensing, inpainting, and super-resolution on the CelebA and MNIST datasets. Utilizing a projected gradient descent algorithm for recovery, we provided rigorous theoretical guarantees for our framework and showed that our recovery algorithm does not impose strict constraints on the learning rate and hence eliminates the need to tune hyperparameters. We compared our framework to existing methods experimentally and found that our recovery algorithm provided a speed up of over two orders of magnitude and an order of magnitude improvement in reconstruction quality.

Literature Cited

- [1] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991.
- [2] David L Donoho et al. Compressed sensing. *IEEE Transactions on information theory*, 52(4):1289–1306, 2006.
- [3] David L Donoho and Xiaoming Huo. Uncertainty principles and ideal atomic decomposition. *IEEE transactions on information theory*, 47(7):2845–2862, 2001.
- [4] David L Donoho and Michael Elad. Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ_1 minimization. *Proceedings of the National Academy of Sciences*, 100(5):2197–2202, 2003.
- [5] David L Donoho and Michael Elad. On the stability of the basis pursuit in the presence of noise. *Signal Processing*, 86(3):511–532, 2006.
- [6] 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.
- [7] Emmanuel Candes and Terence Tao. Decoding by linear programming. *arXiv preprint math/0502327*, 2005.
- [8] Emmanuel J Candes and Terence Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE transactions on information theory*, 52(12):5406–5425, 2006.
- [9] Emmanuel Candes, Terence Tao, et al. The dantzig selector: Statistical estimation when p is much larger than n . *The annals of Statistics*, 35(6):2313–2351, 2007.
- [10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.
- [11] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton,

- et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.
- [12] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [13] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99, 2015.
- [14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.
- [15] Mitali Bafna, Jack Murtagh, and Nikhil Vyas. Thwarting adversarial examples: An l_0 -robust sparse fourier transform. In *Advances in Neural Information Processing Systems*, pages 10075–10085, 2018.
- [16] Jasjeet Dhaliwal and Kyle Hambrook. Compressive recovery defense: Defending neural networks against ℓ_2 , ℓ_1 , and ℓ_0 norm attacks. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.
- [17] Jasjeet Dhaliwal and Kyle Hambrook. Daes for linear inverse problems: Improved recovery with provable guarantees. In *VISIGRAPP (5: VISAPP)*, 2022. To Appear.
- [18] Simon Foucart and Holger Rauhut. *A Mathematical Introduction to Compressive Sensing*. 2017.
- [19] Ronen Eldan and Ohad Shamir. The power of depth for feedforward neural networks. In *Conference on Learning Theory*, pages 907–940, 2016.
- [20] Hrushikesh Mhaskar, Qianli Liao, and Tomaso Poggio. Learning functions: when is deep better than shallow. *arXiv preprint arXiv:1603.00988*, 2016.
- [21] Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein. On the expressive power of deep neural networks. *arXiv preprint arXiv:1606.05336*, 2016.
- [22] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

- [23] 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.
- [24] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.
- [25] 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.
- [26] A. Kurakin, I.J. Goodfellow, and S. Bengio. Adversarial examples in the physical world. *International Conference on Learning Representations*, 2017.
- [27] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, CCS '16*, pages 1528–1540, 2016.
- [28] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok. Synthesizing Robust Adversarial Examples. *ArXiv e-prints*, July 2017.
- [29] I.J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.
- [30] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z.B. Celik, and A. Swami. The limitations of deep learning in adversarial settings. In *Proceedings of the 1st IEEE European Symposium on Security and Privacy*, pages 372–387, 2016.
- [31] N. Papernot, P. McDaniel, I. J. Goodfellow, s. Jha, Z. B. Celik, and A. Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, ASIA CCS '17*, pages 506–519, 2017.
- [32] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, pages 39–57, 2017.

- [33] S. M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. Deepfool: A simple and accurate method to fool deep neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2574–2582, 2016.
- [34] S. M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard. Universal adversarial perturbations. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 86–94, 2017.
- [35] C. Szegedy, W. Zaremba, and I. Sutskever. Intriguing properties of neural networks. *International Conference on Learning Representations*, 2014.
- [36] N. Carlini, P. Mishra, T. Vaidya, Y. Zhang, M. Sherr, C. Shields, D. Wagner, and W. Zhou. Hidden voice commands. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 513–530, Austin, TX, 2016. USENIX Association.
- [37] K. Grosse, N. Papernot, P. Manoharan, M. Backes, and P. McDaniel. Adversarial examples for malware detection. In S. N. Foley, D. Gollmann, and E. Sneekenes, editors, *Computer Security – ESORICS 2017*, pages 62–79, Cham, 2017. Springer International Publishing.
- [38] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song. Robust Physical-World Attacks on Deep Learning Visual Classification. In *Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [39] D. Hendrycks and K. Gimpel. Early Methods for Detecting Adversarial Images. *ArXiv e-prints*, August 2016.
- [40] A. Bhagoji, D. Cullina, C. Sitawarin, and P. Mittal. Enhancing Robustness of Machine Learning Systems via Data Transformations. *ArXiv e-prints*, April 2017.
- [41] W. Xu, D. Evans, and Y. Qi. Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks. *ArXiv e-prints*, April 2017.
- [42] N. Papernot and P. McDaniel. Deep k-Nearest Neighbors: Towards Confident, Interpretable and Robust Deep Learning. *ArXiv e-prints*, March 2018.
- [43] J.H. Metzen, T. Genewein, V. Fischer, and B. Bischoff. On detecting adversarial perturbations. *International Conference on Learning Representations*, 2017.

- [44] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. McDaniel. On the (Statistical) Detection of Adversarial Examples. *ArXiv e-prints*, February 2017.
- [45] D. Meng and H. Chen. Magnet: A two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS '17*, pages 135–147, New York, NY, USA, 2017. ACM.
- [46] N. Carlini and D. Wagner. Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods. *ArXiv e-prints*, May 2017.
- [47] N. Carlini and D. Wagner. MagNet and “Efficient Defenses Against Adversarial Attacks” are Not Robust to Adversarial Examples. *ArXiv e-prints*, November 2017.
- [48] W. He, J. Wei, X. Chen, N. Carlini, and D. Song. Adversarial Example Defenses: Ensembles of Weak Defenses are not Strong. *ArXiv e-prints*, June 2017.
- [49] A. Krizhevsky, I. Sutskever, and G. E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [50] I. Sutskever, O. Vinyals, and Q. V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.
- [51] 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.
- [52] 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.
- [53] 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.
- [54] M. Hein and M. Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *Advances in Neural Information*

Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA, pages 2263–2273, 2017.

- [55] Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 854–863. JMLR. org, 2017.
- [56] Aman Sinha, Hongseok Namkoong, and John Duchi. Certifying some distributional robustness with principled adversarial training. *arXiv preprint arXiv:1710.10571*, 2017.
- [57] Eric Wong and J Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. *arXiv preprint arXiv:1711.00851*, 2017.
- [58] Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified robustness to adversarial examples with differential privacy. *arXiv preprint arXiv:1802.03471*, 2018.
- [59] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.
- [60] Bai Li, Changyou Chen, Wenlin Wang, and Lawrence Carin. Second-order adversarial attack and certifiable robustness. *arXiv preprint arXiv:1809.03113*, 2018.
- [61] F. Tramèr, A. Kurakin, N. Papernot, D. Boneh, and P. McDaniel. Ensemble Adversarial Training: Attacks and Defenses. *ArXiv e-prints*, May 2017.
- [62] W. Xu, D. Evans, and Y. Qi. Feature Squeezing Mitigates and Detects Carlini/Wagner Adversarial Examples. *ArXiv e-prints*, May 2017.
- [63] 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.
- [64] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *arXiv preprint arXiv:1805.06605*, 2018.

- [65] 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.
- [66] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. Mitigating adversarial effects through randomization. *arXiv preprint arXiv:1711.01991*, 2017.
- [67] Emmanuel J Candès, Justin Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on information theory*, 52(2):489–509, 2006.
- [68] Thomas Blumensath and Mike E Davies. Iterative hard thresholding for compressed sensing. *Applied and computational harmonic analysis*, 27(3):265–274, 2009.
- [69] Richard G. Baraniuk, Volkan Cevher, Marco F. Duarte, and Chinmay Hedge. Model-based compressive sensing. *IEEE Transactions on Information Theory*, 56(4):1982–2001, 2010.
- [70] 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.
- [71] 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.
- [72] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.
- [73] Yann LeCun. The mnist database of handwritten digits. <http://yann.lecun.com/exdb/mnist/>.
- [74] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.
- [75] Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

- [76] 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.
- [77] 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.
- [78] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In *Advances in Neural Information Processing Systems*, pages 3353–3364, 2019.
- [79] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994*, 2020.
- [80] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.
- [81] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.
- [82] Gabriel Peyre. Best basis compressed sensing. *IEEE Transactions on Signal Processing*, 58(5):2613–2622, 2010.
- [83] Minhua Chen, Jorge Silva, John Paisley, Chunping Wang, David Dunson, and Lawrence Carin. Compressive sensing on manifolds using a nonparametric mixture of factor analyzers: Algorithm and performance bounds. *IEEE Transactions on Signal Processing*, 58(12):6140–6155, 2010.
- [84] Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G Dimakis. Compressed sensing using generative models. *arXiv preprint arXiv:1703.03208*, 2017.
- [85] Jonas Adler and Ozan Öktem. Solving ill-posed inverse problems using iterative deep neural networks. *Inverse Problems*, 33(12):124007, 2017.
- [86] Kai Fan, Qi Wei, Lawrence Carin, and Katherine A Heller. An inner-loop free solution to inverse problems using deep neural networks. *Advances in Neural Information Processing Systems*, 30:2370–2380, 2017.

- [87] Harshit Gupta, Kyong Hwan Jin, Ha Q Nguyen, Michael T McCann, and Michael Unser. Cnn-based projected gradient descent for consistent ct image reconstruction. *IEEE transactions on medical imaging*, 37(6):1440–1453, 2018.
- [88] Ding Liu, Bihan Wen, Xianming Liu, Zhangyang Wang, and Thomas S Huang. When image denoising meets high-level vision tasks: A deep learning approach. *arXiv preprint arXiv:1706.04284*, 2017.
- [89] Morteza Mardani, Qingyun Sun, David Donoho, Vardan Papyan, Hatef Monajemi, Shreyas Vasanawala, and John Pauly. Neural proximal gradient descent for compressive imaging. In *Advances in Neural Information Processing Systems*, pages 9573–9583, 2018.
- [90] Chris Metzler, Ali Mousavi, and Richard Baraniuk. Learned d-amp: Principled neural network based compressive image recovery. In *Advances in Neural Information Processing Systems*, pages 1772–1783, 2017.
- [91] Ali Mousavi, Gautam Dasarathy, and Richard G Baraniuk. Deepcodec: Adaptive sensing and recovery via deep convolutional neural networks. *arXiv preprint arXiv:1707.03386*, 2017.
- [92] JH Rick Chang, Chun-Liang Li, Barnabas Poczos, BVK Vijaya Kumar, and Aswin C Sankaranarayanan. One network to solve them all—solving linear inverse problems using deep projection models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5888–5897, 2017.
- [93] Viraj Shah and Chinmay Hegde. Solving linear inverse problems using gan priors: An algorithm with provable guarantees. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4609–4613. IEEE, 2018.
- [94] Raymond A Yeh, Chen Chen, Teck Yian Lim, Alexander G Schwing, Mark Hasegawa-Johnson, and Minh N Do. Semantic image inpainting with deep generative models. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5485–5493, 2017.
- [95] Ankit Raj, Yuqi Li, and Yoram Bresler. Gan-based projector for faster recovery with convergence guarantees in linear inverse problems. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5602–5611, 2019.

- [96] Reinhard Heckel and Paul Hand. Deep decoder: Concise image representations from untrained non-convolutional networks. *arXiv preprint arXiv:1810.03982*, 2018.
- [97] Afonso S Bandeira, Edgar Dobriban, Dustin G Mixon, and William F Sawin. Certifying the restricted isometry property is hard. *IEEE transactions on information theory*, 59(6):3448–3450, 2013.
- [98] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.
- [99] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12), 2010.
- [100] Guillaume Alain and Yoshua Bengio. What regularized auto-encoders learn from the data-generating distribution. *The Journal of Machine Learning Research*, 15(1):3563–3593, 2014.
- [101] Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. Generalized denoising auto-encoders as generative models. *Advances in neural information processing systems*, 26:899–907, 2013.
- [102] Gauri Jagatap and Chinmay Hegde. Algorithmic guarantees for inverse imaging with untrained network priors. In *Advances in Neural Information Processing Systems*, pages 14832–14842, 2019.
- [103] Ali Mousavi and Richard G Baraniuk. Learning to invert: Signal recovery via deep convolutional networks. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 2272–2276. IEEE, 2017.
- [104] Yankun Wang, Qiegen Liu, Huilin Zhou, and Yuhao Wang. Learning multi-denoising autoencoding priors for image super-resolution. *Journal of Visual Communication and Image Representation*, 57:152–162, 2018.
- [105] Bichuan Guo, Yuxing Han, and Jiangtao Wen. Agem: Solving linear inverse problems via deep priors and sampling. volume 32, pages 547–558, 2019.

- [106] J. H. Rick Chang, Chun-Liang Li, Barnabas Poczos, B. V. K. Vijaya Kumar, and Aswin C. Sankaranarayanan. One network to solve them all – solving linear inverse problems using deep projection models. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.
- [107] Casper Kaae Sønderby, Jose Caballero, Lucas Theis, Wenzhe Shi, and Ferenc Huszár. Amortised map inference for image super-resolution. *arXiv preprint arXiv:1610.04490*, 2016.
- [108] Singanallur V Venkatakrishnan, Charles A Bouman, and Brendt Wohlberg. Plug-and-play priors for model based reconstruction. In *2013 IEEE Global Conference on Signal and Information Processing*, pages 945–948. IEEE, 2013.
- [109] Ernest Ryu, Jialin Liu, Sicheng Wang, Xiaohan Chen, Zhangyang Wang, and Wotao Yin. Plug-and-play methods provably converge with properly trained denoisers. In *International Conference on Machine Learning*, pages 5546–5557. PMLR, 2019.
- [110] Xiaojian Xu, Yu Sun, Jiaming Liu, Brendt Wohlberg, and Ulugbek S Kamilov. Provable convergence of plug-and-play priors with mmse denoisers. *IEEE Signal Processing Letters*, 27:1280–1284, 2020.
- [111] Tim Meinhardt, Michael Moller, Caner Hazirbas, and Daniel Cremers. Learning proximal operators: Using denoising networks for regularizing inverse imaging problems. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1781–1790, 2017.
- [112] Mario González, Andrés Almansa, and Pauline Tan. Solving inverse problems by joint posterior maximization with autoencoding prior. *arXiv preprint arXiv:2103.01648*, 2021.
- [113] Pei Peng, Shirin Jalali, and Xin Yuan. Solving inverse problems via auto-encoders. *IEEE Journal on Selected Areas in Information Theory*, 1(1):312–323, 2020.
- [114] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pages 3730–3738, 2015.

- [115] Xianxu Hou, Linlin Shen, Ke Sun, and Guoping Qiu. Deep feature consistent variational autoencoder. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1133–1141. IEEE, 2017.
- [116] François Chollet. keras. <https://github.com/fchollet/keras>, 2015.
- [117] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.